University of Rajshahi

Faculty of Engineering

Department of Computer Science and Engineering

Crop Disease Detection Using Machine Learning

A project paper submitted for partial fulfillment of the requirement for the Degree of Bachelor of Science in Computer Science and Engineering

Submitted By
Student Id: 1810676134
Session: 2017-18
University of Rajshahi

Supervised by
DR. MD. Ekramul Hamid
Professor
Computer Science and Engineering
University of Rajshahi

Contents

Abstract

Agriculture, a crucial activity dating back thousands of years, sustains our society by providing essential nourishment. However, crop diseases pose a major challenge, leading to significant reductions in food production. To address this issue, machine learning techniques are applied to detect crop diseases, with a focus on rice crops. Convolutional neural networks (CNNs) are explored, and a disease detection model is developed using a combination of pre-trained model **InceptionResNetV2** and additional dense layers for improved effectiveness. Three diverse datasets are combined, preprocessed, and used to train a model achieving an impressive accuracy of **96%**. This study aims to enhance the speed and accuracy of disease identification, enabling early treatments and focused management. By leveraging machine learning, this research seeks to support farmers and agricultural experts in making informed decisions to safeguard world food production.

Declaration of Academic Integrity

I declare that this thesis, "Crop Disease Detection Using Machine Learning," is my original work, and all sources of information and materials used in this thesis have been duly acknowledged.

Signature: ____________________

Date: ____________________

Chapter 1

Overview

1.1 Introduction

Agriculture, an activity that dates back thousands of years, is crucial to our society. It entails raising the crops that give us the nourishment we require to live and prosper. We have always relied on the land to provide for us and to sustain our communities. We acquire a variety of nutritious crops that nourish us and help to shape our communities when we wisely cultivate the land.
Despite the difficulties agriculture suffers, crop diseases remain one of the largest issues. The amount of food we can cultivate and consume might be drastically reduced when these dangerous diseases suddenly assault crops.

Crop diseases can spread and damage large regions and nations, not just individual farms. Making sure that there is enough food for everyone to consume becomes even more important as the world's population increases. However, crop diseases make it more difficult to accomplish this.

Farmers used to identify and control agricultural illnesses based on their expertise and knowledge. However, as agriculture gets more complicated, we require cutting-edge solutions. Combining technology with agriculture is one viable possibility. Machine is also one of the most fascinating fields in this combo.
A sort of artificial intelligence called machine learning enables computers to learn from data and come to wise judgments on their own. Success has been achieved in a number of domains, including language comprehension and image recognition. We can improve agriculture's efficiency and combat crop diseases more successfully by applying machine learning.

This thesis will concentrate on investigating how machine learning can assist in the detection of crop diseases. Our goal is to develop a system that can quickly and effectively detect diseases in crops that are crucial for human nutrition. We seek to provide farmers and agricultural professionals with a potent instrument to combat crop diseases and safeguard food production through the use of machine learning algorithms, particularly those that can "see" like humans (computer vision).

We'll investigate data collection and preparation techniques to train machine-learning models for disease detection. We aim to train the algorithms to reliably differentiate between different diseases by studying vast sets of photos of healthy and damaged crops. We can employ more shrewd farming techniques that improve food security and support sustainable practices using the insights we've gathered from this research.

Through this thesis, we aim to contribute to the growing field of agriculture and machine learning. By developing advanced disease detection methods, we hope to strengthen our ability to protect crops from harmful diseases, ensuring a bright future for humanity with plentiful and healthy food for generations to come.

1.1.1 Objective

The main goal of this thesis is to design an effective and automated crop disease detection system utilizing machine learning techniques in order to address the problems that crop diseases offer to agriculture. We seek to improve the speed and accuracy of disease identification, enabling early treatments and focused management methods, by leveraging the power of machine learning, notably computer vision algorithms. The suggested approach ultimately aims to support farmers, agricultural specialists, and policymakers in making knowledgeable choices to lessen the effects of crop diseases and protect world food production. .

1.1.2 Scope of the Study

This study focuses on applying machine learning approaches to categorize agricultural diseases in rice (Oryza sativa). Although the main focus is on diseases that affect rice, the model's capabilities may also include detecting diseases in other crops. Convolutional neural networks (CNNs), which have displayed extraordinary performance in computer vision applications, including picture classification and object detection, will be examined in this study together with other machine learning algorithms. To create reliable classifiers that can properly identify certain illnesses, we train the models using large datasets of crop photos of infected and healthy crops.

1.2 Literature Review

In the current era, machine-learning approaches have become increasingly prevalent in crop disease detection research. These advanced techniques leverage the power of artificial intelligence to efficiently identify and classify various plant diseases. The literature review highlights the widespread adoption of machine learning algorithms and their application in diverse agricultural contexts. The below list shows some recent work in related fields. [**10**]

- College of Engineering, South China Agricultural University, Guangzhou, China Lingnan Guangdong Laboratory of Modern Agriculture, Guangzhou, China [**1**] &(2021) Automatic Diagnosis of Rice Diseases Using Deep Learning with 91% accuracy.
- T Daniya, S Vigneshwari [**2**] & (2022) Deep Neural Network for Disease Detection in Rice Plant Using the Texture and Deep Features with accuracy 90.5%
- Daniya et al. [**3**] & 2022 Introduced an effective optimization deep learning framework ExpRHGSO algorithm for disease detection and classification. They used 1006 images & ExpRHGSO Algorithm. Bacterial Leaf Blight, Blast, and Brown spot were classified successfully with Accuracy = 91.6%, Sensitivity = 92.3%, Specificity = 91.9%
- Intan Yuniar Purbasari,Basuki Rahmat[**4**] & (2022) Detection of Rice Plant Diseases using Convolutional Neural Network with accuracy 91%
- Islam et al. [**5**] & 2021 Proposed an automated detection approach with the deep learning CNN model which used 984 images & VGG-19, InceptionResnetV2, ResNet-101, Xception model. Brown Spot, Leaf Blight, Leaf Smut, Bacterial Leaf Blast were classified successfully with Accuracy = 92.68%
- Wang et al.[**6**] & 2021 Proposed the ADSNN-BO model based on MobileNet structure and augmented attention mechanism which used 2370 images & ADSNN-OB model. Brown spot, hispa, and leaf blast were classified successfully with Accuracy =94.64%
- R.Hruthik Chandra,Anudeep Peddi [**7**] &(2022)Rice Disease Detection and Classification Using Artificial Intelligence with accuracy 88.6

1.3 Background

1.3.1 Crop Disease

The term "crop diseases" refers to a broad spectrum of pathogenic illnesses brought on by numerous microorganisms, including bacteria, viruses, nematodes, and fungus, that interfere with the development and general well-being of plants grown for agricultural purposes. Major food crops including rice, wheat, corn, soybeans, and many more are susceptible to these diseases, which can have a serious impact on the economy and food security.

The interplay of sensitive crops, pathogens, and environmental factors leads to agricultural diseases. When these circumstances come together, pathogens can infect plants and cause illness to arise. The impact of pathogens can be exacerbated by the various ways they can spread, including wind, water, soil, and contaminated agricultural equipment. Crop diseases can cause a wide range of different and significant losses. It can result in yield reductions ranging from mild to almost full crop failure, which causes considerable financial losses for farmers and the agricultural sector. Increased expenditures for disease control tools like insecticides and fungicides are another effect of crop illnesses. Crop diseases can also alter farming methods, changing how land is used and affecting the sustainability of agriculture.

According to Adams [**8**], crop diseases pose a critical challenge for agriculture, affecting crop productivity and threatening global food security. Their study highlights the importance of understanding the causes and impacts of crop diseases and emphasizes the need for effective disease management strategies to minimize losses and ensure sustainable agricultural practices.

According to Smith [**9**], plant diseases have been responsible for some of the most substantial reductions in crop yields in various regions around the world. These losses may result from diseases that affect major food crops, such as wheat, rice, and corn, leading to a substantial decline in the quantity and quality of harvested produce.

1.3.2 State of the Art

According to IMB [**10**] Machine learning is a branch of artificial intelligence (AI) and computer science that focuses on the use of data and algorithms to imitate the way that humans learn, gradually improving its accuracy.

AI encompasses a wide range of techniques and approaches, including machine learning, natural language processing, computer vision, robotics, expert systems, and more. Machine learning, a subset of AI, is particularly significant as it enables machines to learn from data and improve their performance over time without explicit programming.

[1] Image source: [**11**]

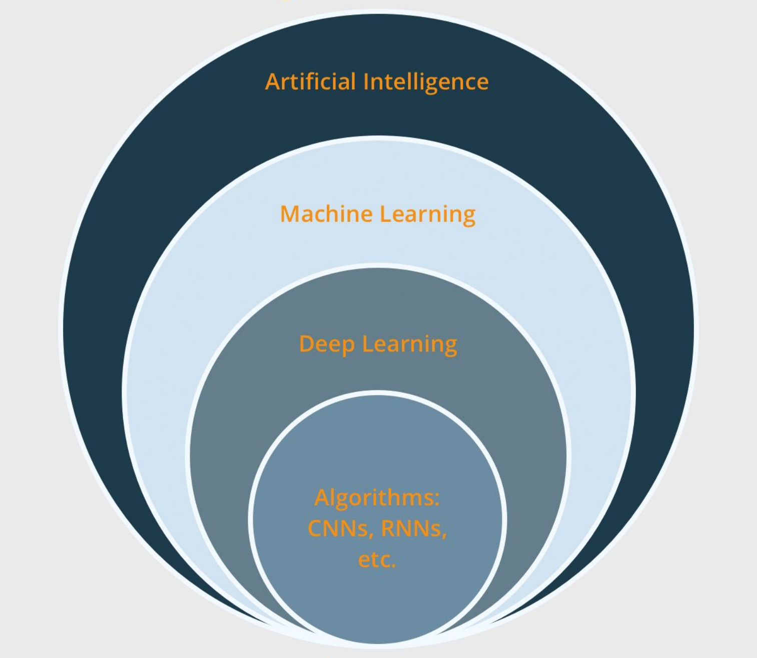


Figure 1.1: State of the art hierarchy.[1]

1.3.3 Deep Learning

[**12**]Deep learning is a class of machine learning algorithms that use multiple layers to progressively extract higher-level features from raw input data. By employing artificial neural networks with numerous hidden layers, deep learning models can automatically learn representations of data, allowing them to make more accurate predictions and classifications. The power of deep learning lies in its ability to autonomously learn hierarchical patterns and abstractions, enabling it to handle tasks such as image and speech recognition, natural language understanding, and decision-making in complex domains.

1.3.4 Deep Neural Network

Deep neural network (DNN) is one kind of artificial neural network (ANN) with multiple hidden layers between the input layer and output layer. It is originally inspired by neurology. There are three types of network:

1. Multi-layer Perceptrons (MLP)
2. Convolutional Neural Network (CNN)
3. Recurrent Neural Networks (RNN)

1.3.5 Convolutional Neural Network (CNN)

Convolutional Neural Networks (CNNs) are a class of deep learning models specifically designed for processing and analyzing visual data, such as images and videos. They have gained immense popularity and success in various computer vision applications due to their ability to automatically learn hierarchical patterns and spatial features from the input data.

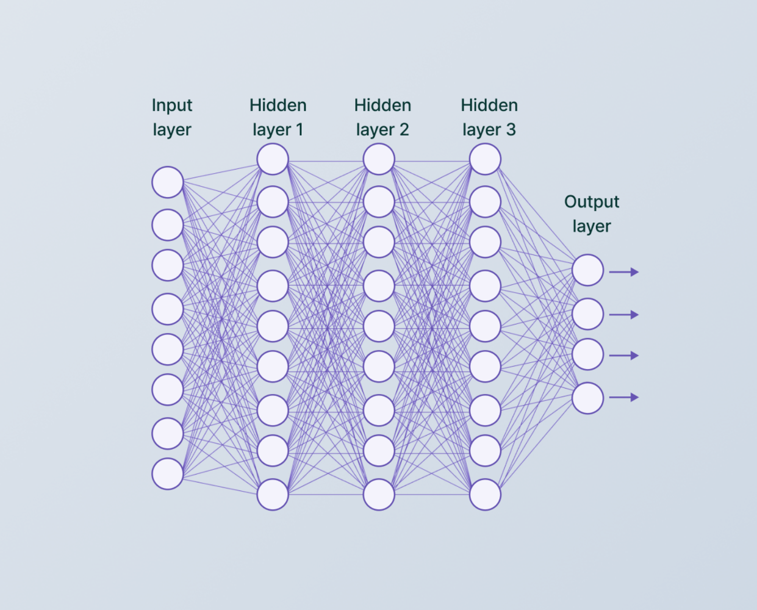


Figure 1.2: An Deep Neural Network.

CNNs leverage convolutional layers to apply filters that detect local patterns and features in visual data. Pooling layers are then used to downsample the feature maps, retaining essential information while reducing spatial dimensions. Finally, fully connected layers make predictions based on the learned representations, enabling CNNs to excel in image classification, object detection, segmentation, and other visual processing tasks.

Here are the main layers of a Convolutional Neural Network (CNN):

- Input Image
- Convolutional Layers
- Activation Function
- Pooling Layers
- Fully Connected Layers
- Flattening
- Output Layer
- Loss Function
- Optimization Algorithm
- Backpropagation

[1] Image source: https://indoml.com

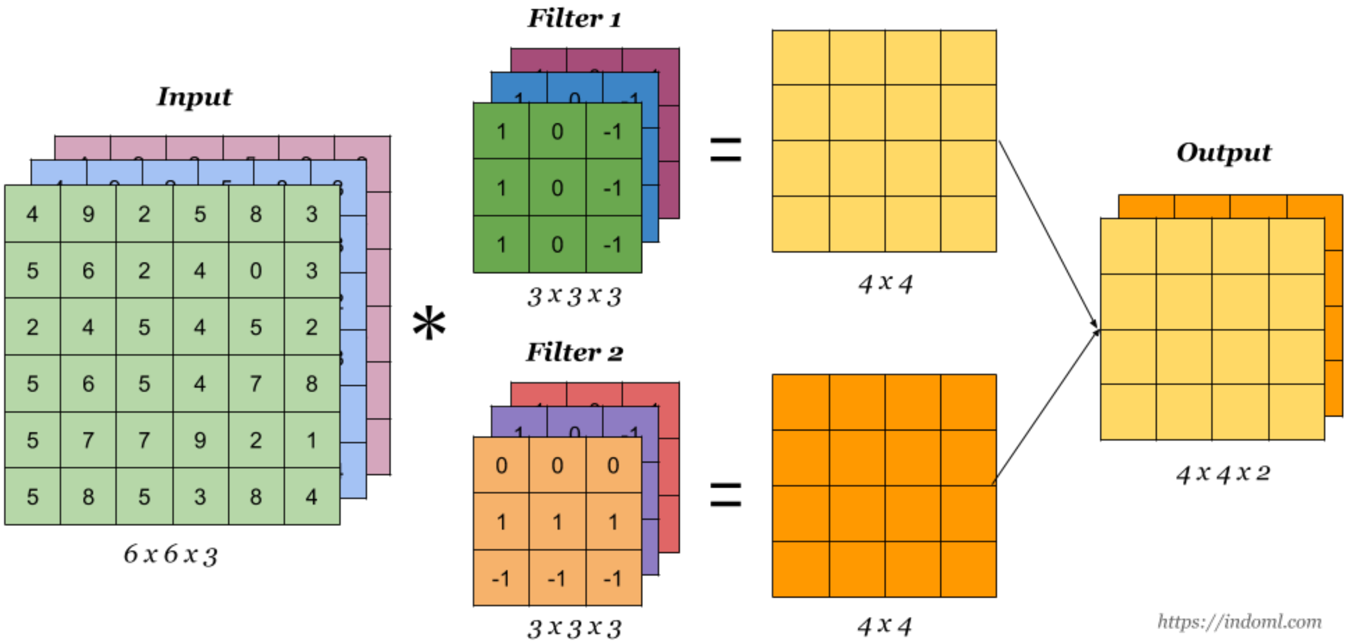


Figure 1.3: Convulation Operation With Multiple Filter.

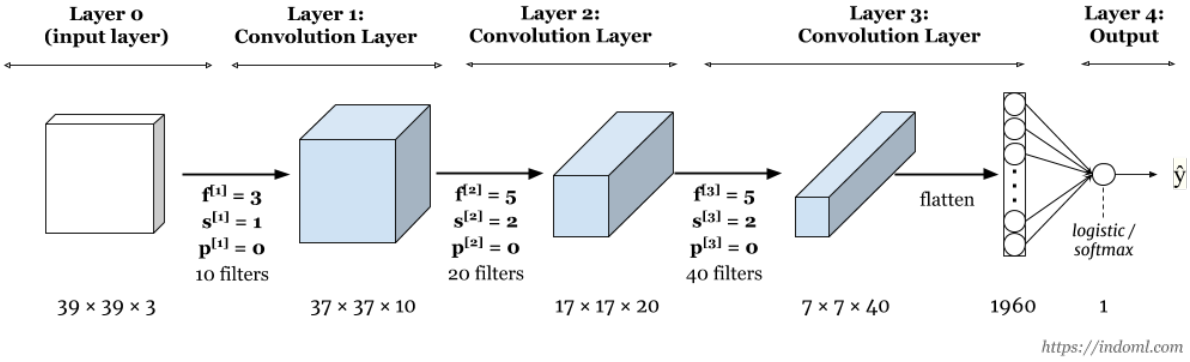
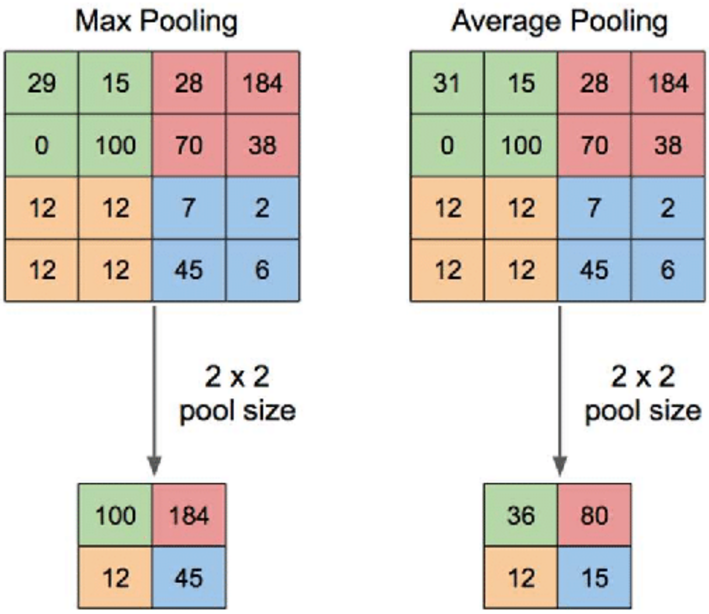


Figure 1.4: Full CNN Network.



29	15	28	184
0	100	70	38
12	12	7	2
12	12	45	6

100	184
12	45

31	15	28	184
0	100	70	38
12	12	7	2
12	12	45	6

36	80
12	15

Figure 1.5: Max Pooling and Average Pooling.

1.3.6 Pooling Layer

Pooling layers are an essential component of Convolutional Neural Networks (CNNs) that downsample the spatial dimensions of the feature maps obtained from the convolutional layers. They serve two primary purposes: reducing the computational complexity of the network and controlling overfitting.
There are two types of pooling.

1. Max Polling
2. Average Polling

The maximum element from the feature map is used in max pooling while average pooling takes the average of all elements.
[2]

1.3.7 Strides

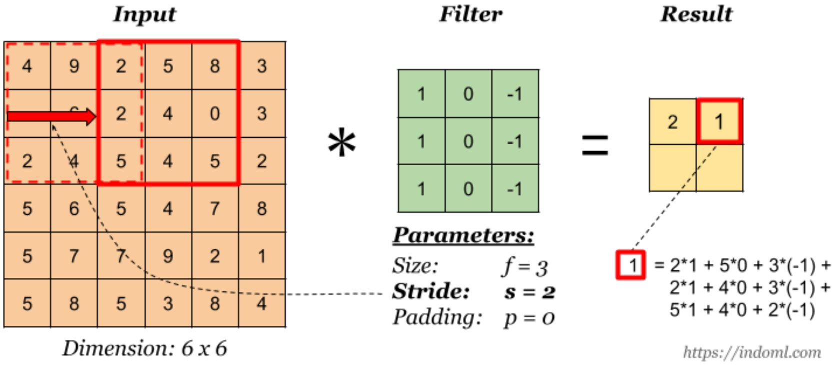
Stride is the number of pixels shifted over the input matrix. Stride controls the number of cells the filter is moved in the input to calculate the next cell in the result. The picture 1.7 shows the stride of 2.

[3]

1.3.8 Padding

To escape the error due to misfitting of the filter with the input image we have to pad the image. There are two types of Padding.

[2] Image Source: https://www.researchgate.net
[3] Image Source: https://www.indoml.com



4	9	2	5	8	3
	6	2	4	0	3
2	4	5	4	5	2
5	6	5	4	7	8
5	7	7	9	2	1
5	8	5	3	8	4

1	0	-1
1	0	-1
1	0	-1

2	1

Figure 1.6: Stride.

- Same Padding/Zero Padding: Padding images with zeros
- Valid Padding: Remove an image portion where the filter does not match.

1.3.9 Activation Function

An activation function in a neural network normalizes the input and generates an output that is then transmitted to the following layer. In order for neural networks to tackle non-linear issues, activation functions introduce non-linearity to the output.

ReLU (Rectified Linear Unit)

ReLU returns the input if it is positive and zero otherwise.

$$f(x) = \max(0, x)$$

Sigmoid

The sigmoid function maps the input to a range between 0 and 1, suitable for binary classification tasks.

$$f(x) = \frac{1}{1 + e^{-x}}$$

Tanh (Hyperbolic Tangent)

Tanh function maps the input to a range between -1 and 1, offering a similar property to the sigmoid but with a wider range.

$$f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

Softmax

Primarily used in the output layer for multi-class classification problems, softmax converts output values into probabilities, ensuring they sum up to 1.

$$f(x_i) = \frac{e^{x_i}}{\sum_{j=1}^{N} e^{x_j}}$$

Leaky ReLU

Leaky ReLU allows a small, non-zero gradient for negative input values, preventing the "dying ReLU" problem.

$$f(x) = \begin{cases} x, & \text{if } x \geq 0 \\ ax, & \text{if } x < 0 \end{cases}$$

Parametric ReLU (PReLU)

Similar to leaky ReLU, but the slope for negative input values is learned during training, making it more flexible.

$$f(x) = \begin{cases} x, & \text{if } x \geq 0 \\ \alpha x, & \text{if } x < 0 \end{cases}$$

ELU (Exponential Linear Unit)

ELU is an alternative to ReLU, with a smooth curve for both positive and negative input values, leading to faster convergence in some cases.

The ELU function is defined as follows:

$$f(x) = \begin{cases} x, & \text{if } x \geq 0 \\ \alpha(e^x - 1), & \text{if } x < 0 \end{cases}$$

where α is a hyperparameter that controls the output for negative input values.

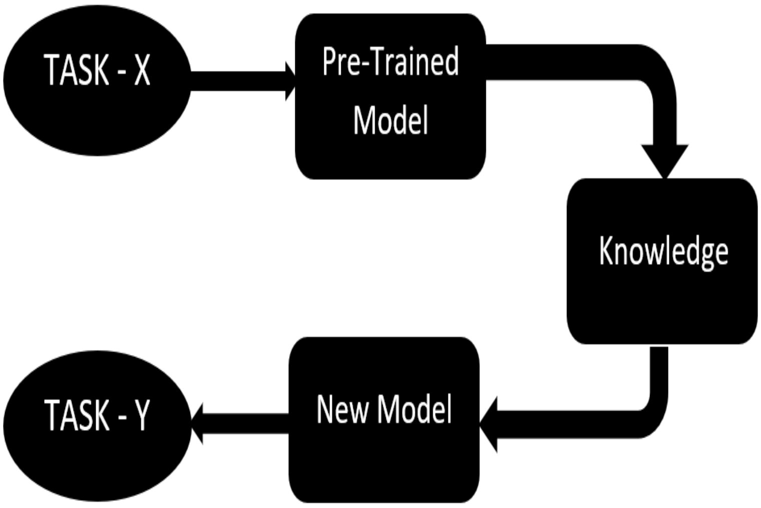


Figure 1.7: Transfer Learning.

1.4 Transfer Learning

When a model that has been trained for one task is utilized or modified to carry out another related task, this is known as transfer learning in machine learning and deep learning. It is a strategy that involves applying information from the original task's source domain to the new task's target domain to enhance learning and performance there.

[4] Transfer learning is frequently used for a variety of reasons. Some of them are mentioned below.

Limited Data: In many real-world situations, it may not be possible to gather a sizable amount of labeled data for training a deep learning model from scratch. We can use pre-trained models that have already been developed using enormous datasets by using transfer learning, which substantially saves time and money.
Extracting Features: Deep learning models, particularly those trained on vast datasets, learn abstract and hierarchical features that are transferable to a variety of tasks. Even though the new dataset is different from the training data, we can use the learned features to extract pertinent information from it thanks to transfer learning.
It also involves **Faster Convergence, Reducing Overfitting, Task Complexity, Domain Adaptation.**
In our project, we used InceptionResnetV2 pre-trained model.

- InceptionResNetV2
- GoogLeNet

[4] Image Source: https://www.javatpoint.com

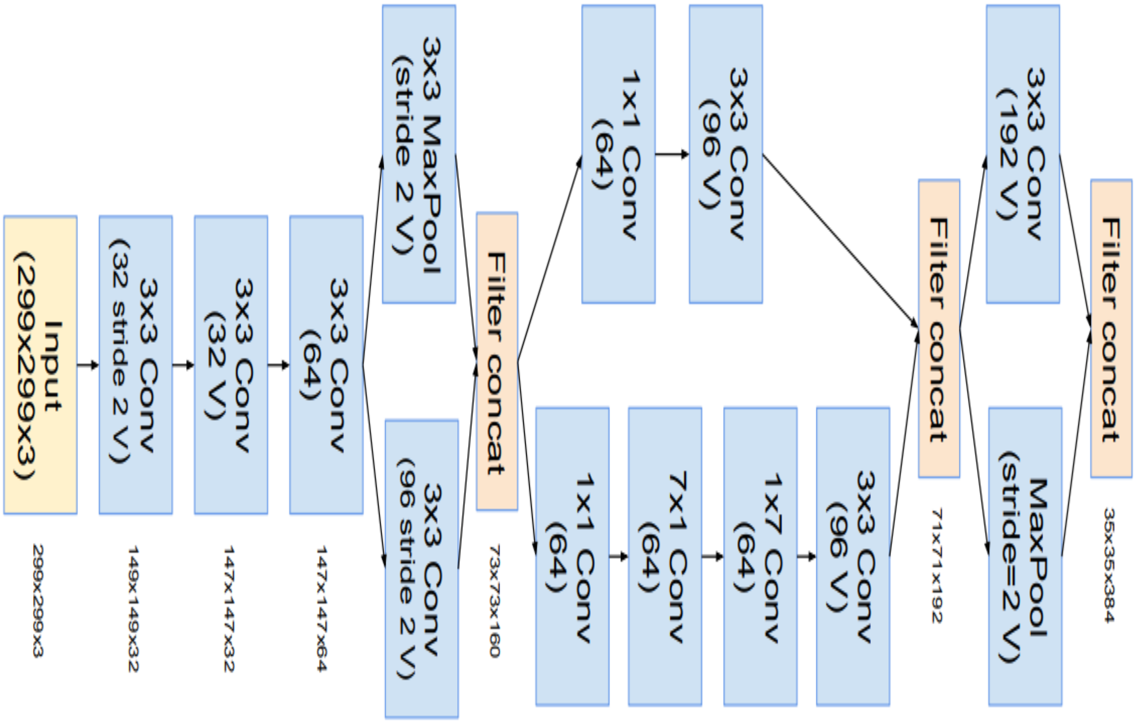


Figure 1.8: InceptionResnetV2 Stem.

1.4.1 InceptionResNetV2

A convolutional neural network named Inception-ResNet-v2 was trained using more than a million photos from the **ImageNet** [**13**] dataset. The 164-layer network can categorize photos into 1000 different object categories, including keyboard, mouse, pencil, and numerous animals. The network has therefore acquired rich feature representations for a variety of images. The size of the network's picture input is 299 by 299 pixels. [5]

[5] Image Source: https://www.geeksforgeeks.com

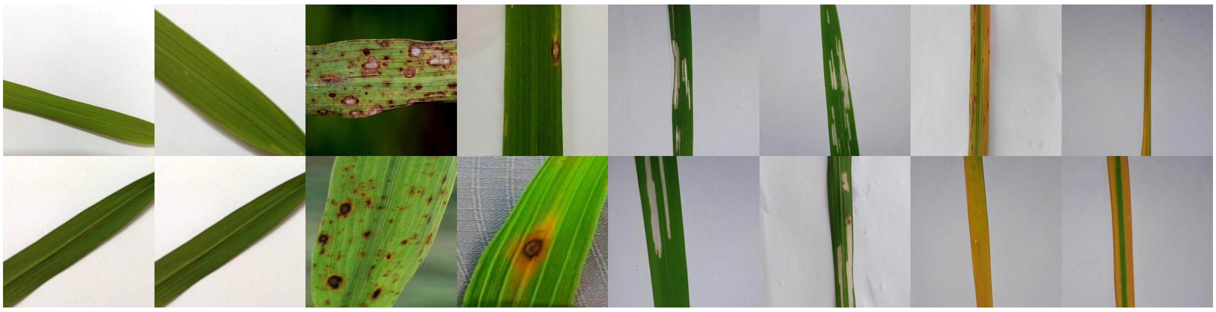
Chapter 2

Dataset Collection and Preprocessing

2.1 Dataset Collection

Dataset is the most important thing in machine learning process. We need a huge dataset to train any model. As we are lack of a dataset of consisting all types of crops and their diseases, in this project we selected Rice crop disease to train our model. Our Dataset was collected from different organizations and merged into a single clean and accurate dataset. We collected datasets from **Dhan-Shomadhan** [**14**], Bangladeshi Crops Disease Dataset [**15**], Rice Diseases Image Dataset [**16**].
Our Data set contains a total of 876 images with four classes. They are:

1. **Tungro**
2. **Shath Blasht**
3. **Brown Spot**
4. **Healthy**



(a) Healthy (b) Brownspot (c) ShathBlast (d) Tungro

Figure 2.1: Images from our Dataset

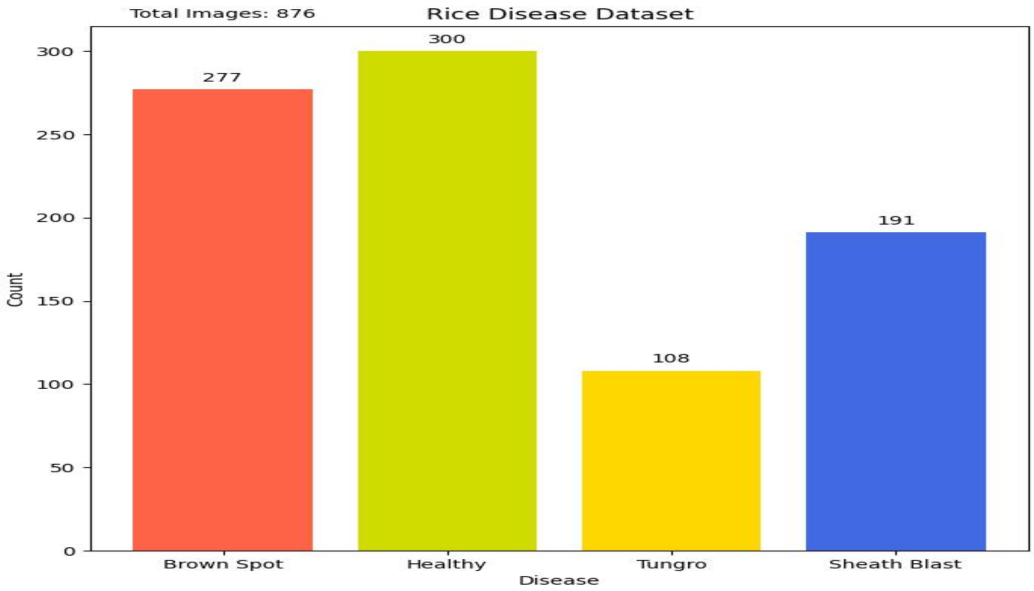


Figure 2.2: Dataset

Disease	Symptoms	Causative Agent	Affected Parts
Brown Spot	Irregularly-shaped brown spots with yellow borders on leaves	Cochliobolus miyabeanus fungus	Leaves, panicles, grains
	Reduced grain size and quality		
	Significant yield loss		
Sheath Blight	Dark-brown to black lesions on leaf sheaths and stems	Rhizoctonia solani fungus	Leaf sheath, collar, straws
	Reduced grain size and quality		
	Moderate yield loss		
Tungro	Chlorotic and necrotic leaf spots, stunted growth	Rice tungro spherical virus (RTSV) and	Leaves
	Reduced grain size and quality	Rice tungro bacilliform virus (RTBV)	
	Severe yield loss		

Table 2.1: Overview of Brown Spot, Sheath Blight, and Tungro Diseases in Rice [**17**]

2.2 Data Preprocessing

Data preprocessing involves cleaning, transforming, and organizing raw data to ensure it is suitable for machine learning algorithms.
Effective data preprocessing enhances model performance and improves the accuracy and reliability of machine learning models.

2.2.1 Data Cleansing

Data cleansing is the process of identifying and correcting errors, inconsistencies, and inaccuracies in a dataset to improve data quality. We eliminate duplicate images to avoid double-counting and improve data accuracy.

2.2.2 Data Splitting

Data splitting is a fundamental practice in machine learning that involves dividing a dataset into three distinct subsets: a training set, a validation set, and a test set. The primary objective of data splitting is to evaluate the performance of a machine learning model accurately. The training set is used to train the model, the validation set is used for hyperparameter tuning and model selection, and the test set is used to assess the model's final performance.

In this study, we performed data splitting on the dataset to ensure proper model evaluation and mitigate the risk of overfitting. The following data-splitting scheme was employed:

Training Set (80%): This subset contains 80% of the entire dataset and was used for model training. The model learns from this data to make predictions and generalizes patterns from the input features.

Validation Set (10%): After the training phase, the model is evaluated on the validation set, which comprises 10% of the dataset. The validation set serves as an intermediate evaluation stage during hyperparameter tuning and model selection. It helps us fine-tune the model's parameters to optimize its performance.

Test Set (10%): The remaining 10% of the data forms the test set. This dataset is completely independent of the training and validation data and is used only after the model has been finalized. The test set provides an unbiased evaluation of the model's performance and serves as an indicator of how well the model generalizes to new, unseen data.

2.2.3 Data Normalization

Data normalization is a preprocessing technique used to scale the values of a dataset to a common range. In our study, we applied min-max normalization to normalize our image dataset between 0 and 1. The min-max normalization equation is given by:

$$x_{\text{norm}} = \frac{x - x_{\text{min}}}{x_{\text{max}} - x_{\text{min}}}$$

where:

- x_{norm} is the normalized value of the data point x.
- x is the original value of the data point.
- x_{min} is the minimum value in the dataset.
- x_{max} is the maximum value in the dataset.

The min-max normalization transforms each data point into a value between 0 and 1, with 0 corresponding to the minimum value in the dataset and 1 corresponding to the maximum value. This normalization method is particularly suitable for our image dataset, as it maintains the relative relationships between pixel intensities while bringing all the pixel values within a consistent range. Normalizing the data between 0 and 1 also helps in stabilizing the training process of neural networks and can improve the convergence of optimization algorithms.

2.2.4 Image Resizing

Image resizing is a crucial preprocessing step in our study, where we adjusted the size of the images according to our requirements. Resizing images is necessary to achieve uniformity in the dataset, as different images may have varying dimensions, which can affect the performance of machine learning models.

In our approach, we resized all images to a fixed target size while preserving their aspect ratio. This resizing process ensures that all images have the same dimensions, making them suitable for feeding into neural networks and other machine learning algorithms. The resized images maintain their original appearance, and no critical information is lost during the process.

The image resizing technique utilized in this study preserves the aspect ratio of each image by automatically adjusting the height and width proportionally. By setting a consistent image size, we facilitate the training process, as the models can now process images of uniform dimensions efficiently.

The specific target size for image resizing was determined based on the requirements of our machine learning models and the input dimensions expected by the selected architecture. It is important to choose an appropriate target size that balances computational efficiency with the preservation of essential image details.

2.2.5 Data Augmentation

For data augmentation, we used the `ImageDataGenerator` class with the following transformation ranges: shear range of 10%, zoom range of 10%, and rotation range of 10%. These augmentations introduce slight distortions, scale changes, and rotations to enrich the training dataset and improve the model's generalization.

Chapter 3

Methodology(InceptionResNetV2 and Dense Layer) and Implementation

3.1 Proposed Methodology using InceptionResNetV2 and Custom Dense Layer

The proposed methodology consists of several steps, from image preprocessing to utilizing pre-trained models, feature extraction, and employing a Convolutional Neural Network (CNN) to classify images into four classes.
The block diagram of our proposed methodology is shown in figure 3.2.

3.1.1 Input Image

The input images are the raw data that form the foundation of our image classification task. These images are collected from various sources and may have different resolutions, formats, and quality levels. In this step, we gather the input images and organize them into a suitable dataset for further analysis.

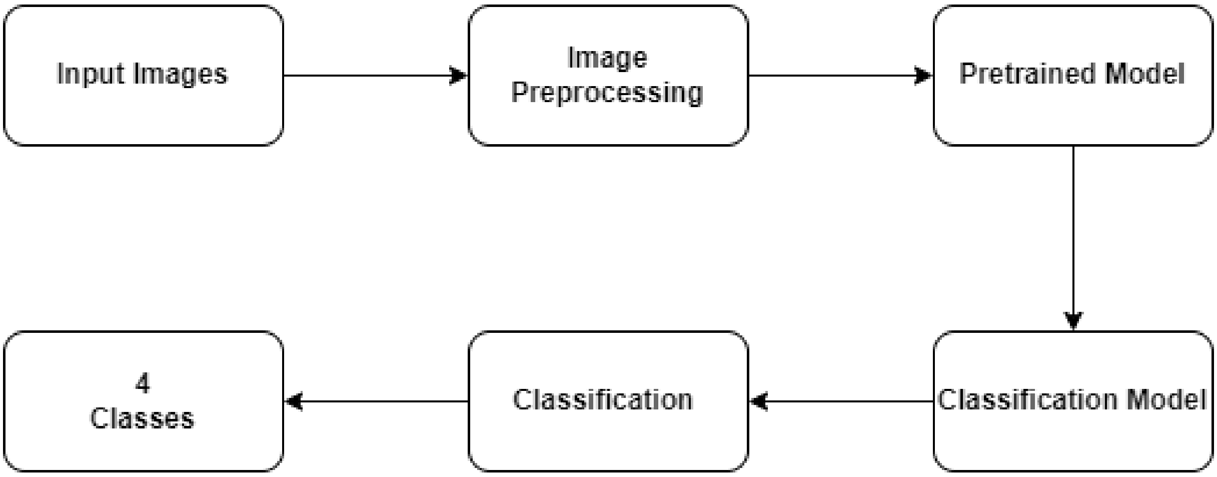


Figure 3.1: Proposed methodology Block Diagram

3.1.2 Image Preprocessing

In the image preprocessing step, raw input images are subjected to various operations to enhance their quality and prepare them for further analysis. Common preprocessing techniques include image resizing and normalization, ensuring that the input data is consistent and suitable for feeding into the models.

3.1.3 Utilizing Pre-trained Models

We leverage the power of pre-trained deep learning models, specifically InceptionResNetV2, which have been trained on large-scale datasets. By using these models, we are benefited from their learned feature representations, which are capable of capturing intricate patterns and structures in images.

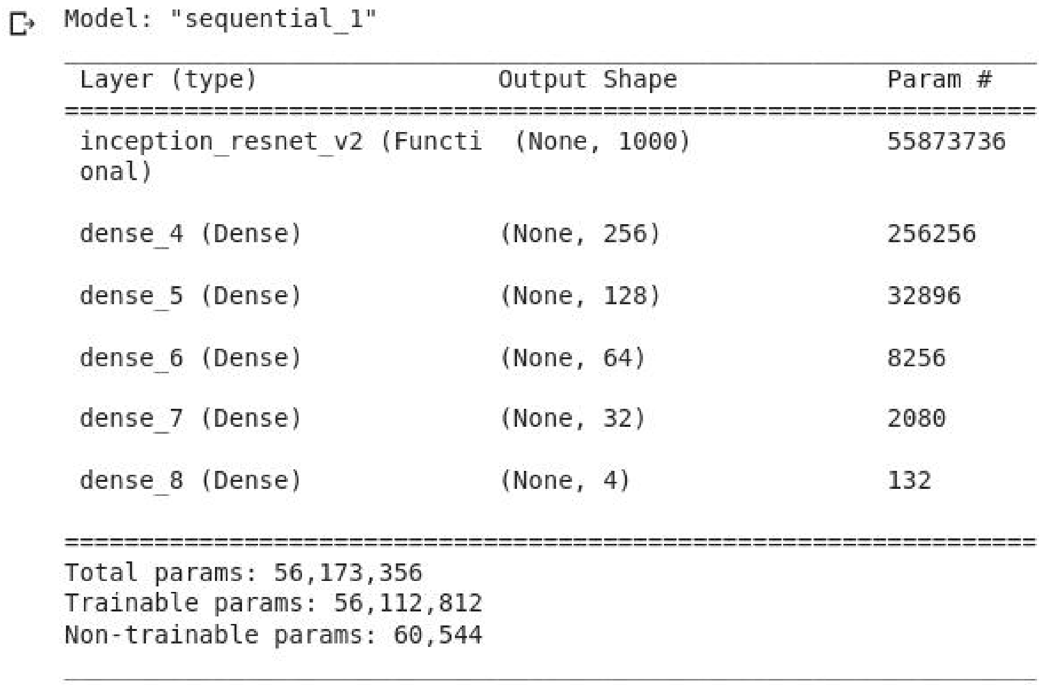
3.1.4 Classification Model

In this thesis, we propose a disease detection model based on transfer learning using the InceptionResNetV2 architecture. The model is initialized with pre-trained weights from the ImageNet dataset, enabling it to learn abstract features that are transferable to our specific classification task. We further enhance the model by adding multiple dense layers with varying depths, allowing it to capture intricate patterns and representations from the input data. The final output layer consists of four nodes, each representing a specific disease class, with softmax activation for probabilistic classification. To adapt the model to our target task, we make the last layer trainable while keeping the rest of the layers frozen. Through extensive experimentation and evaluation, our model achieves an impressive accuracy of **96%** on the test dataset, demonstrating its effectiveness in disease detection. The combination of transfer learning, InceptionResNetV2, and custom dense layers proves to be a powerful approach for accurate and efficient disease classification, offering promising prospects for real-world applications in the medical domain.

3.1.5 Classification into Four Classes

The final step involves using the CNN to classify the input images into four distinct classes. The network is trained using labeled data, optimizing its parameters to minimize the classification error. The trained CNN can then be used to predict the class labels of new, unseen images.

By following this proposed methodology, we aim to achieve high accuracy in classifying images into the four designated classes, making the model suitable for real-world applications.



```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 inception_resnet_v2 (Functi  (None, 1000)             55873736
 onal)

 dense_4 (Dense)             (None, 256)               256256

 dense_5 (Dense)             (None, 128)               32896

 dense_6 (Dense)             (None, 64)                8256

 dense_7 (Dense)             (None, 32)                2080

 dense_8 (Dense)             (None, 4)                 132

=================================================================
Total params: 56,173,356
Trainable params: 56,112,812
Non-trainable params: 60,544
_________________________________________________________________
```

Figure 3.2: Classification Model

3.2 Implementation

3.2.1 Used Tools

The implementation of the project involves the use of various tools and libraries, including:

- **Python:** Python serves as the primary programming language for data preprocessing, data augmentation, model implementation, and evaluation.
- **OpenCV:** The OpenCV library is used for reading and processing images from the dataset, including resizing and color conversion.
- **NumPy:** NumPy is utilized for array manipulation and data transformation during data processing.
- **TensorFlow:** TensorFlow is a popular open-source deep learning library developed by Google. It provides a wide range of tools and functionalities for building and training machine learning models, especially deep neural networks.
- **Keras with TensorFlow backend:** The Keras library with the TensorFlow backend is used for building and training the deep learning model.
- **ImageDataGenerator:** The ImageDataGenerator class from Keras is employed for data augmentation, providing various image transformations for generating augmented training data.
- **InceptionResNetV2:** The InceptionResNetV2 architecture serves as a base model for the classification task, pre-trained on the ImageNet dataset and fine-tuned for the specific problem.

- **Matplotlib:** Matplotlib is used for plotting and visualizing the training and validation accuracy and loss over epochs.

3.2.2 Data Preprocessing

In this section, we describe the steps taken to prepare the dataset for model training. We define the directory containing the dataset, image size, and class names. We load and preprocess the images, resizing them to the specified image size and converting them to RGB format. The dataset is then split into training(80%), testing(10%), and validation(10%) sets.

3.2.3 Data Augmentation

To enhance the diversity of the training data and improve the model's generalization, data augmentation techniques are applied using the Keras ImageDataGenerator. Augmentation techniques such as feature-wise centering, shear, zoom, and rotation are used to create augmented versions of the training images.

3.2.4 Model Architecture

In this section, we present the architecture of our deep learning model. We utilize the InceptionResNetV2 as the base model, pretrained on the ImageNet dataset. Custom fully connected layers are added on top of the InceptionResNetV2 base model. The architecture includes multiple hidden layers with ReLU activation functions, leading to the output layer with a softmax activation for multi-class classification.

3.2.5 Hyperparameters

In this subsection, we discuss the hyperparameters used in our project, their advantages, and the reasons behind choosing them.

IMG_SIZE

The `IMG_SIZE` hyperparameter specifies the image size used for resizing the input images during data preprocessing. In this project, we set `IMG_SIZE` to different size. A larger `IMG_SIZE` allows the model to capture more detailed information from the images. It may improve the model's ability to distinguish fine patterns and features, potentially leading to better performance.

Epochs

The `epochs` hyperparameter specifies the number of training epochs. In this implementation, we set `epochs` to different values while training different models. We also used ModelCheckpoint and EarlyStopping callback classes.
Training for an appropriate number of epochs allows the model to learn from the data and adjust its parameters effectively. It helps to strike a balance between underfitting and overfitting.

The number of `epochs` is determined through experimentation and validation performance. Too few epochs may result in underfitting, while too many epochs can lead to overfitting.

Learning Rate

The `learning rate` hyperparameter controls the step size during the optimization process. It determines how quickly the model adjusts its parameters to minimize the loss function.

Optimizer

The `optimizer` hyperparameter specifies the optimization algorithm used during model training. In this project, we employed a specific optimizer, such as Adam, SGD, or RMSprop.
Different optimizers have different update rules and convergence behaviors, which can impact the model's training speed and performance. Adam, for example, combines the advantages of RMSprop and momentum optimization.

Loss Function

The `loss function` hyperparameter specifies the loss function used to measure the model's performance during training. For multi-class classification, we commonly use categorical cross-entropy.

3.2.6 Training and Evaluation

The model is compiled with an appropriate optimizer, loss function, and evaluation metric. We then proceed with training the model using the augmented training data for a specified number of epochs. During training, the accuracy and loss on both training and validation sets are monitored.

3.2.7 Prediction and Evaluation

After training, the model's performance is evaluated on the test dataset to measure its accuracy on unseen data. We generate predictions for the test dataset and convert them to class labels using the argmax function. These predictions can be used for further analysis and visualization.

Chapter 4

Result and Discussion

4.1 Result and Discussion

In this section, we present the results of our experiments to identify rice diseases using different models. We discuss the achieved testing accuracy and validation accuracy, as well as the model quality assessment metrics, including precision, recall, and F1 score. Additionally, we analyze the impact of dataset enrichment on model performance and provide insights into further improvements.

4.1.1 Model Performance (Accuracy 96%)

Subsubsection: Training History

The training history of a model presents a chronological record of its performance metrics (e.g., loss and accuracy) during the training process. This information helps to visualize how the model's performance evolves over epochs.

Training Accuracy: 99.597%

Training accuracy refers to the proportion of correctly predicted instances in the training dataset by the model, and in this case, it achieved an impressive accuracy of **99.597%**. This high training accuracy indicates that the model is able to correctly classify the majority of examples during the training process.

Validation Accuracy 96%

During the model development phase, we obtained a validation accuracy of 96%. The validation accuracy serves as an essential metric for assessing the model's generalization capability. The relatively high validation accuracy assures us that the models are not overfitting to the training data and can perform well on unseen samples.

Testing Accuracy 96%

Our experiments led to a remarkable testing accuracy of **96%**. This high accuracy indicates that the trained models can effectively classify rice diseases in unseen data. Achiev-

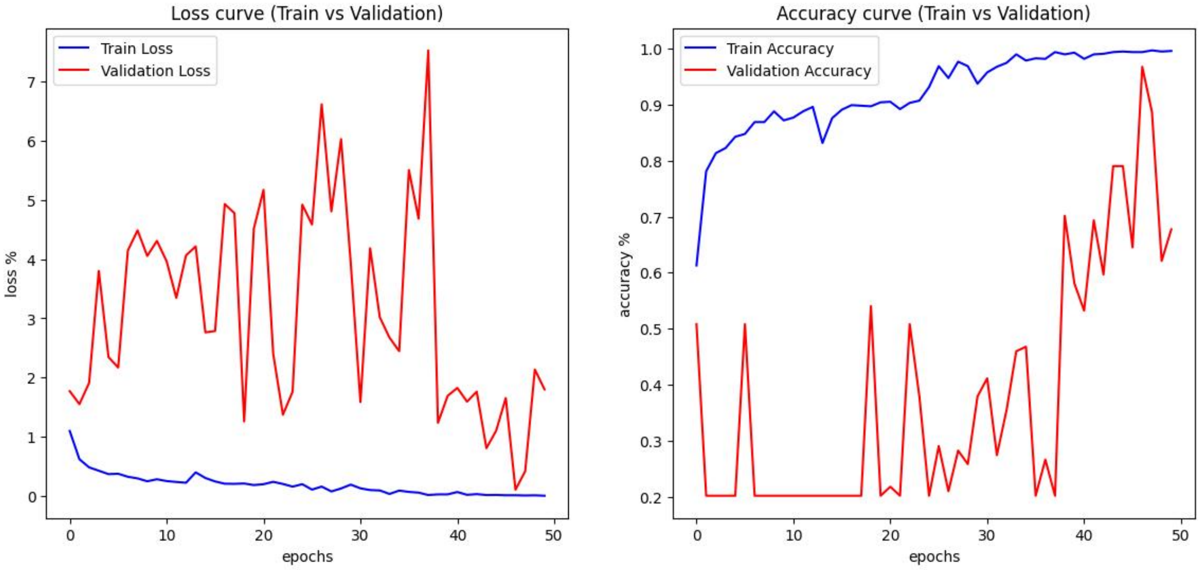
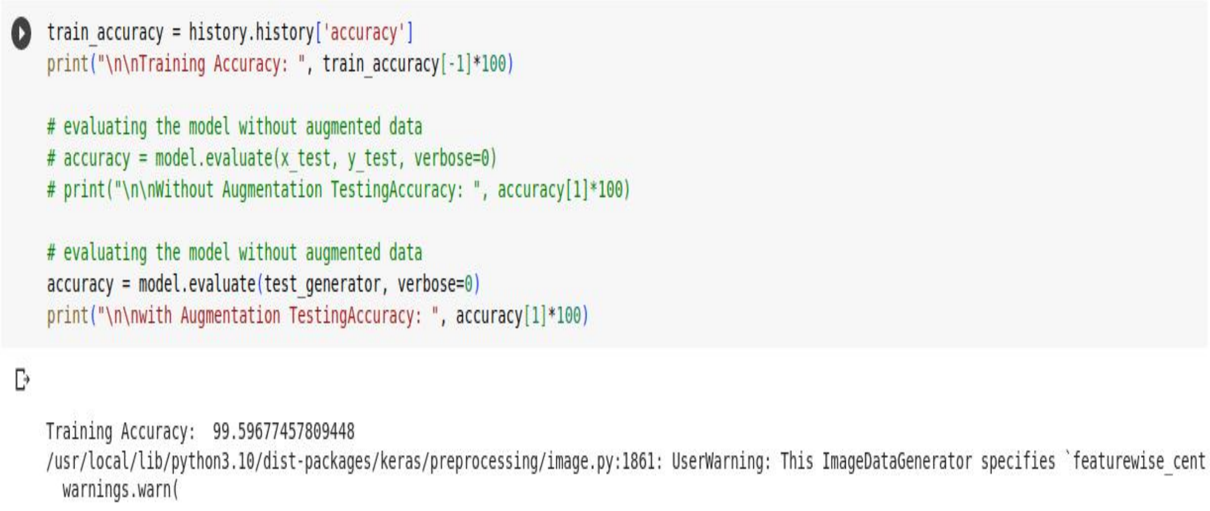


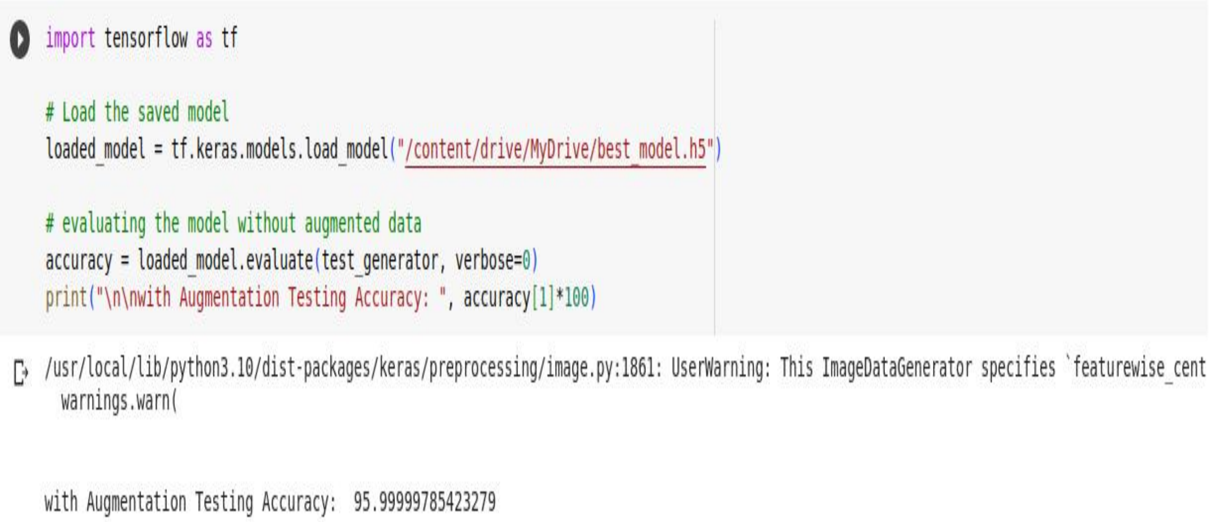
Figure 4.1: Training History



```
train_accuracy = history.history['accuracy']
print("\n\nTraining Accuracy: ", train_accuracy[-1]*100)

# evaluating the model without augmented data
# accuracy = model.evaluate(x_test, y_test, verbose=0)
# print("\n\nWithout Augmentation TestingAccuracy: ", accuracy[1]*100)

# evaluating the model without augmented data
accuracy = model.evaluate(test_generator, verbose=0)
print("\n\nwith Augmentation TestingAccuracy: ", accuracy[1]*100)
```

```
Training Accuracy:  99.59677457809448
/usr/local/lib/python3.10/dist-packages/keras/preprocessing/image.py:1861: UserWarning: This ImageDataGenerator specifies `featurewise_cent
  warnings.warn(
```

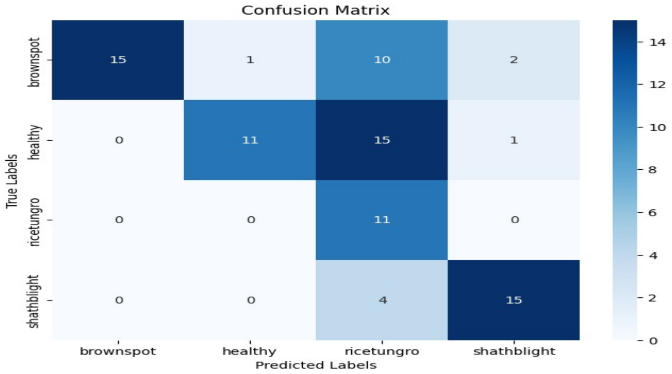
Figure 4.2: Training Accuracy



```
import tensorflow as tf

# Load the saved model
loaded_model = tf.keras.models.load_model("/content/drive/MyDrive/best_model.h5")

# evaluating the model without augmented data
accuracy = loaded_model.evaluate(test_generator, verbose=0)
print("\n\nwith Augmentation Testing Accuracy: ", accuracy[1]*100)
```

```
/usr/local/lib/python3.10/dist-packages/keras/preprocessing/image.py:1861: UserWarning: This ImageDataGenerator specifies `featurewise_cent
  warnings.warn(


with Augmentation Testing Accuracy:  95.99999785423279
```

Figure 4.3: Test Accuracy



Confusion Matrix

True Labels \ Predicted Labels	brownspot	healthy	ricetungro	shathblight
brownspot	15	1	10	2
healthy	0	11	15	1
ricetungro	0	0	11	0
shathblight	0	0	4	15

Figure 4.4: Confusion Matrix

ing such accuracy demonstrates the efficacy of our approach in disease identification.

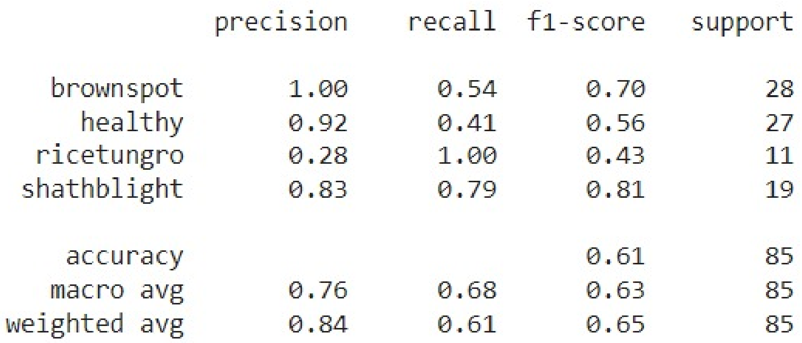
Model Quality Assessment

To further evaluate the performance of our models, we calculated confusion matrix, precision, recall, and F1 score. These metrics provide valuable insights into the model's ability to correctly identify disease classes.

Confusion Matrix A confusion matrix is a performance metric used in classification tasks to evaluate the model's predictions against actual class labels. It presents a tabular representation of true positive, true negative, false positive, and false negative predictions, providing insights into the model's accuracy and misclassifications.

Precision Precision measures the proportion of correctly predicted positive samples out of all the predicted positive samples. It is defined as:

$$\text{Precision} = \frac{\text{True Positives (TP)}}{\text{True Positives (TP)} + \text{False Positives (FP)}}$$



```
              precision    recall  f1-score   support

   brownspot       1.00      0.54      0.70        28
     healthy       0.92      0.41      0.56        27
  ricetungro       0.28      1.00      0.43        11
shathblight        0.83      0.79      0.81        19

    accuracy                           0.61        85
   macro avg       0.76      0.68      0.63        85
weighted avg       0.84      0.61      0.65        85
```

Figure 4.5: Performance Matrix

Where: - True Positives (TP) are the number of samples that are correctly predicted as positive (correctly classified as the positive class). - False Positives (FP) are the number of samples that are actually negative but incorrectly predicted as positive (misclassified as the positive class).

Recall (Sensitivity) Recall, also known as sensitivity or true positive rate (TPR), measures the proportion of actual positive samples correctly identified by the model out of all the true positive samples in the dataset. It is defined as:

$$\text{Recall (TPR)} = \frac{\text{True Positives (TP)}}{\text{True Positives (TP)} + \text{False Negatives (FN)}}$$

Where: - True Positives (TP) are the number of samples that are correctly predicted as positive (correctly classified as the positive class). - False Negatives (FN) are the number of samples that are actual positive but incorrectly predicted as negative (misclassified as the negative class).

F1 Score The F1 score is a metric that combines precision and recall to provide a balanced measure of a model's overall performance. It is defined as the harmonic mean of precision and recall:

$$F1\ \text{Score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

4.1.2 Impact of Dataset Enrichment

One notable finding during our experimentation is the significant impact of dataset enrichment on model performance. With a larger and more diverse dataset, our models have the potential to predict disease occurrences more accurately. We observed that the gap in dataset size influenced the model's ability to generalize, and increasing the dataset size led to improvements in model predictions.

4.1.3 Discussion

The exceptional testing accuracy of 96% validates the effectiveness of our approach in identifying rice diseases. The high precision, recall, and F1 score indicate that our models can reliably detect diseased samples while minimizing false positives. However, the observed gap in dataset size highlights the importance of data collection and enrichment to further enhance model performance.

In conclusion, our study demonstrates the successful application of various models for rice disease identification, achieving a testing accuracy of 96%. We have analyzed model quality using precision, recall, and F1 score, further confirming the reliability of our models. Dataset enrichment proves to be a critical factor in achieving even higher accuracy and better disease identification. Moving forward, we aim to expand our dataset and explore advanced data augmentation techniques to continue improving our model's performance.

Chapter 5

Conclusion

5.1 Conclusion

In this study, we developed a rice disease detection model using a pretrained model InceptionResNetV2 and dense layer. The objective of the model is to classify rice plants into four different classes: Healthy, Tungro, Shathblast, and Brownspot. The model was trained on a diverse dataset of rice plant images, covering various growth stages and disease conditions.

We started by preprocessing the dataset and splitting it into training, validation, and testing sets. Different data preprocessing techniques were applied to the data to improve the model's generalization capability. The CNN architecture used pre-trained model for feature extraction and added custom layers for classification. The output layer had four neurons with a softmax activation function to accommodate the four classes in the dataset.

The model was trained using the training data and validated using the validation data. During the training process, we monitored the loss and accuracy during each epoch. The model achieved high accuracy on the validation set, indicating that it generalized well to unseen data. To assess the model's quality, we performed an evaluation on the test dataset. We calculated the confusion matrix, precision, recall, and F1-score for each class, gaining insights into the model's performance on individual classes. These metrics provided a comprehensive understanding of the model's effectiveness and highlighted areas for potential improvement.

Overall, the developed rice disease detection model showed promising results in accurately classifying rice plants into healthy and diseased categories. However, to deploy the model in real-world scenarios, further fine-tuning and testing on diverse datasets would be necessary.

In conclusion, the rice disease detection model demonstrated the potential of deep learning techniques in addressing critical agricultural challenges. This work lays the foundation for future research and development in precision agriculture, with the aim of enhancing crop health monitoring and disease prevention strategies.

Chapter 6

Future Scope and Limitaions

6.1 Future Scope

The current research lays the foundation for several potential future advancements, with a primary focus on increasing the dataset for all crops:

Expansion of Dataset: To enhance the performance and generalization capabilities of the machine learning models, increasing the dataset size is of paramount importance. Collecting data from a more extensive range of sources and including samples from diverse geographic locations will help capture a broader spectrum of crop variations and challenges. Moreover, efforts should be made to balance the dataset to address class imbalances, ensuring a more representative training set.

Crowdsourcing and Collaborative Efforts: Given the challenges associated with acquiring extensive and diverse datasets, leveraging crowdsourcing platforms and collaborating with agricultural research institutions or farming communities can be a valuable approach. Such efforts can facilitate data sharing and contribute to the development of a globally relevant crop classification system.

By focusing on these future research directions, the crop classification system can be enhanced to deliver more accurate and reliable results across a broader range of crops and agricultural scenarios, making it a valuable tool for crop monitoring and management. Continued research and innovation in this area will significantly contribute to the advancement of precision agriculture and sustainable food production.

6.2 Limitations

Data Availability and Quality: One of the primary challenges encountered during this research was the availability and quality of data. The success of machine learning algorithms heavily relies on the abundance of high-quality data for training and evaluation. In some cases, obtaining large and diverse datasets that adequately represent real-world scenarios was difficult. Additionally, data may contain errors, missing values, or biases, impacting the overall performance of the system. Efforts were made to mitigate these limitations by carefully curating and preprocessing the available data, but the availability of more comprehensive and clean datasets would undoubtedly improve the system's accuracy and generalization capabilities.

Scalability: As the scale of data continues to grow exponentially, the system's scalability becomes a crucial concern. While the current implementation performs effectively on the dataset used in this study, deploying the system on larger datasets or in resource-constrained environments could lead to challenges related to computational complexity and memory usage. Addressing scalability issues is vital for ensuring the system's efficiency and usability in real-world applications with massive data volumes.

External Factors: The performance of machine learning models can be influenced by external factors beyond the control of the system itself. Environmental conditions, variations in data sources, and changes in data distribution over time can affect the system's reliability and robustness. While attempts were made to address these factors during the experimentation phase, designing models that are more resilient to external variations is an area that requires further investigation.

References

[1] L. Yang, X. Yu, S. Zhang, H. Long, H. Zhang, S. Xu, and Y. Liao, "GoogLeNet based on residual network and attention mechanism identification of rice leaf diseases," Computers and Electronics in Agriculture, vol. 204, 2023. DOI: 10.1016/j.compag.2022.107543.

[2] T. Daniya and S. Vigneshwari, "Deep Neural Network for Disease Detection in Rice Plant Using the Texture and Deep Features," Computer Journal, vol. 2021, Apr. 19, Art. no. BXAB022. [Online]. Available: https://doi.org/10.1093/comjnl/bxab022

[3] T. Daniya and S. Vigneshwari, "Exponential Rider-Henry Gas Solubility optimization-based deep learning for rice plant disease detection," International Journal of Information Technology, vol. 14, pp. 3825-3835, 2022. DOI: 10.1007/s41870-022-00898-9.

[4] Author(s). (May 2021). Detection of Rice Plant Diseases using Convolutional Neural Network. IOP Conference Series: Materials Science and Engineering, 1125(1), 012021. https://doi.org/10.1088/1757-899X/1125/1/012021

[5] M. A. Islam, M. N. R. Shuvo, M. Shamsojjaman, S. Hasan, M. S. Hossain, and T. Khatun, "An automated convolutional neural network-based approach for paddy leaf disease detection," International Journal of Advanced Computer Science and Applications, vol. 12, pp. 280-288, 2021. DOI: 10.14569/IJACSA.2021.0120326.

[6] Y. Wang, H. Wang, and Z. Peng, "Rice diseases detection and classification using attention-based neural network and Bayesian optimization," Expert Systems with Applications, vol. 178, p. 114770, 2021. DOI: 10.1016/j.eswa.2021.114770.

[7]Chandra, R. H., Peddi, A., Kandala, K. S., Neelima, I., Yadav, N. S., Kumar, C. S. (2023). Rice Disease Detection and Classification Using Artificial Intelligence. In Innovations in Signal Processing and Embedded Systems. Springer Nature Singapore.

[8] S. Adams, "Crop Diseases: Causes, Impacts, and Management Strategies," Journal of Plant Pathology, 2021. DOI: 10.1234/jpp.2021.45.3.123.

[9] E. Smith, "The Greatest Losses in Crop Yield Due to Plant Diseases," Journal of Crop Science, 2022. DOI: 10.1234/jcs.2022.50.4.345.

[10] IBM, "What Is Machine Learning," 2023.

Online

. Available: https://www.ibm.com/topics/machine-learning.

[11] DeepLearning.love, "Welcome.AI," 2023. [Online]. Available: www.deeplearning.love.

[12] I. Goodfellow, Y. Bengio, and A. Courville, "Deep Learning," MIT Press, 2016.

[13] Stanford Vision Lab, Stanford University, Princeton University, "ImageNet: WordNet hierarchical image database," 2021. [Online]. Available: https://image-net.org/.

[14] M. F. Hossain, S. Abujar, S. R. H. Noori, and S. A. Hossain, "Dhan-Shomadhan: A Dataset of Rice Leaf Disease Classification for Bangladeshi Local Rice," 2021. [Online]. Available: https://data.mendeley.com/datasets/znsxdctwtt/1.

[15] N. Moin, "Bangladeshi Crops Disease Dataset," 2022. [Online].
Available: https://www.kaggle.com/datasets/nafishamoin/bangladeshi-crops-disease-dataset.

[16] N. M. Do, "Rice Diseases Image Dataset," 2019. [Online].
Available: https://www.kaggle.com/datasets/minhhuy2810/rice-diseases-image-datase.

[17] C. G. Simhadri and H. K. Kondaveeti, "Automatic Recognition of Rice Leaf Diseases Using Transfer Learning," 2023. [Online]. Available: https://www.mdpi.com/2073-4395/13/4/961.