Corn Leaf Disease Classification Using Deep-Ensemble-Learning

A PROJECT REPORT

Submitted by

BL.EN.U4AIE19028 K.Vishnu Sainadh

BL.EN.U4AIE19034 K.Satwik

BL.EN.U4AIE19066 V.Ashrith

in partial fulfillment for the award of the degree of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING (ARTIFICIAL INTELLIGENCE)

AMRITA SCHOOL OF COMPUTING, BANGALORE

AMRITA VISHWA VIDYAPEETHAM

BANGALORE 560 035

MAY – 2023

AMRITA VISHWA VIDYAPEETHAM

AMRITA SCHOOL OF COMPUTING, BANGALORE, 560035

BONAFIDE CERTIFICATE

This is to certify that the project report entitled **"Corn Leaf Disease Classification Using Deep-Ensemble-Learning"** submitted by

BL.EN.U4AIE19028	K.Vishnu Sainadh
BL.EN.U4AIE19034	K.Satwik
BL.EN.U4AIE19066	V.Ashrith

in partial fulfillment of the requirements as part of **Bachelor of Technology** in "**COMPUTER SCIENCE AND ENGINEERING (ARTIFICIAL INTELLIGENCE)**" is a bonafide record of the work carried out under my guidance and supervision at Amrita School of Computing, Bangalore.

Mr. Niranjan D K	Dr. Gopalakrishnan E.A.
Lecturer	Professor and Chairperson,
Dept. of CSE	Dept. of CSE

This project report was evaluated by us on ……

EXAMINER1 EXAMINER2

ACKNOWLEDGEMENTS

The satisfaction that accompanies successful completion of any task would be incomplete without mention of people who made it possible, and whose constant encouragement and guidance have been source of inspiration throughout the course of this project work.

We offer our sincere pranams at the lotus feet of **"AMMA," MATA AMRITANANDAMAYI DEVI** who showered her blessing upon us throughout the course of this project work.

We owe our gratitude to **Prof. Manoj P.**, Director, Amrita School of Engineering, Bangalore.

We thank **Dr. Sriram Devanathan**, Principal, Amrita School of Engineering, Bangalore for his support and inspiration.

We also thank **Dr. Gopalakrishnan E.A.,** Chairperson, Dept.of CSE, Amrita School of Computing, Bangalore for his supervision and motivation.

It is a great pleasure to express our gratitude and indebtedness to our project guide **Mr. Niranjan D K,** Lecturer, Department of Computer Science and Engineering, Amrita School of Engineering, Bangalore for her/his valuable guidance, encouragement, moral support, and affection throughout the project work.

We would like to thank express our gratitude to project panel members for their suggestions, encouragement, and moral support during the process of project work and all faculty members for their academic support. Finally, we are forever grateful to our parents, who have loved, supported, and encouraged us in all our endeavors.

ABSTRACT

Global food security is seriously threatened by plant diseases. The quick detection of plant diseases is still difficult and time-consuming, though. Experts are needed to correctly determine the type of illness and whether the plant is healthy or not. In order to automate plant disease diagnosis and aid non-experts in identifying unhealthy plants, deep learning techniques have recently been applied to recognise and diagnose ill plants from digital photos. In this study, image variations were added to the dataset used to train the model using data augmentation techniques. By boosting the diversity and quantity of the photos, the model was able to learn more complicated data situations. The proposed approach extracts deep characteristics from the photos of corn plants using four pre-trained convolutional neural networks (CNNs), Mobilenet, VGG16, Xception, and InceptionV3. At last deep ensemble model is built using the pre trained models. Multiple Ensemble models like Majority voting, averaging, weighted averaging were used. The proposed model is able to achieve a classification accuracy of 97%.

.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER - 1
INTRODUCTION

1.1 INTRODUCTION

This research project is aimed at developing an accurate and reliable system for classifying the various kinds of diseases that can affect corn leaves through the use of deep learning techniques. The project is titled "Corn leaf disease classification using deep ensemble learning." The goal of the project is to increase the accuracy of the system as a whole by employing a deep neural network architecture. This architecture combines the capabilities of several different models.

Corn is widely considered to be one of the most important products in the world, and it serves as a primary source of nutrition in a number of different areas. Corn plants, on the other hand, are susceptible to a wide variety of diseases that can have a negative impact on their development, yield, and quality. It is essential for effective disease management, which can assist farmers in reducing losses and increasing agricultural yields, to have a diagnosis that is accurate and made in a timely manner of these diseases.

Traditional methods of illness diagnosis typically take a significant amount of time, can be quite costly, and call for highly specialized knowledge. Recent developments in deep learning, on the other hand, have led to the emergence of new opportunities for the automated detection of diseases through the application of computer vision methods. The purpose of this project is to make use of the power of deep learning in order to develop a system that is capable of accurately classifying various kinds of corn leaf diseases based on the visual symptoms that they present with.

Training multiple deep neural network models, each of which has a unique architecture and set of hyper parameters, using a large dataset of images of corn leaves is required for the deep ensemble learning approach that has been suggested. After that, the outputs of these models are combined with the results of a weighted voting system in order to arrive at a conclusive

categorization. It has been demonstrated that using an ensemble approach can enhance the deep learning models' overall accuracy as well as their robustness.

The long-term objective of this project is to design a method that is dependable and effective and that can provide assistance to farmers in the detection and diagnosis of diseases that affect the corn leaf, thereby lowering crop losses and increasing crop yields.

1.2 MOTIVATION

In agriculture, the old ways of finding plant diseases by hand require experts to do an eye check and then a more thorough one in labs which takes a lot of time and isn't always available to small farmers. To mitigate the negative effects of these diseases, early and accurate diagnosis is crucial.

1.3 PROBLEM STATEMENT

Developing deep learning models to distinguish between healthy and various unhealthy corn plant leaves. Implement multiple deep learning models and combine them to form a deep ensemble model. In this work, a new model for classifying digital pictures of corn leaves that are affected with gray leaf spot, common rust, northern leaf blight, or are healthy will be suggested.

CHAPTER - 2

LITERATURE SURVEY

2.1 STUDY ON CORN DISEASE IDENTIFICATION BASED ON PCA AND SVM

For the purpose of this particular piece of research, studies on maize rust, corn large spot, corn grey leaf spot, and healthy corn leaves were carried out. Using the Otsu method, OpenCV's morphological operations, and morphological transformations are the first steps in picture background segmentation. These methods are used to define the shape of the item, and then a mask is made to show the area of the picture that has been cut out. This mask corresponds to the area of the image that was segmented. To create a full corn leaf picture, the difference between the corn leaf and the backdrop is used to outline the contour. This difference set captures the differences between the corn leaf and its backdrop to provide a complete depiction. This is done so that the image may be seen in its whole. Image processing uses PCA and SVM. PCA reduces image data dimensionality and extracts the most relevant features, whereas SVM classifies and categorizes pictures based on these characteristics. Image processing algorithms may examine and interpret visual data using these two methods. The accuracy of classification for each of the four distinct diseases is consequently 90.05%, 92.64%, 91.23%, and 95.78% when the penalty parameter C of the SVM is set to 100 and the kernel is linear.

2.2 IDENTIFICATION OF CORN LEAF DISEASE BASED ON IMAGE PROCESSING

In order to reduce the amount of money that is lost due to corn disease, it is essential to make a diagnosis of the illness as quickly and accurately as is humanly possible. A MATLAB-based maize leaf disease image detection system was created due of its importance. After preprocessing a damaged corn leaf picture, the algorithm extracts 11 attributes, including form and texture, to build an ideal subset. The recognition model is built using SVM. To improve data quality for analysis, the system preprocesses maize leaf disease images. The final training set had an

identification success rate of 93.7%, however the test set only had an identification success rate of 89.38%.

2.3 CORN LEAF DISEASES DIAGNOSIS BASED ON K-MEANS CLUSTERING AND DEEP LEARNING

This study identifies rust, grey spot, and leaf spot on maize leaves. Data analysis uses K-means clustering and an upgraded deep learning model. These methods help identify and classify maize leaf diseases. In the process of diagnosing three illnesses, the next step is to input those clusters into an updated version of the deep learning model. This step follows the stage of using the K-means algorithm to cluster sample images, which was the previous step. In this work, a wide range of k values (two, four, eight, sixteen, thirty-two, and sixty-four) and models (VGG-16, ResNet18, Inception v3, VGG-19, and the enhanced deep learning model) are compared and contrasted in terms of their impact on disease identification in maize. The suggested technique has the greatest illness identification effect on 32-means samples, according to experimental results. Leaf spot disease has an 89.24% recall rate, suggesting great accuracy in diagnosis. Rust sickness has 100% recall, indicating good identification. Grey spot illness has a 90.95% recall rate, showing excellent identification. VGG-16 and ResNet18 provide the best 32-means sample diagnostics. ResNet18 has 83.75% diagnostic accuracy, whereas VGG-16 has 84.42%. VGG-16 and ResNet18 outperform other models in illness detection in this scenario. In addition, Inception v3 (83.05%) and VGG-19 (82.63%) obtain the greatest performance on the 64-means samples respectively. This approach offers an average diagnosis accuracy of 93% for the three different maize diseases that are mentioned in this article. Its diagnostic impact is more substantial than those of the other four approaches, and it may be utilized in the agricultural field to protect crops from harm. Moreover, it can be used.

2.4 LEAF DISEASE IMAGE CLASSIFICATION METHOD BASED ON IMPROVED CONVOLUTIONAL NEURAL NETWORK

This research classifies maize illness photos using neural networks. The categorization method uses hundreds of real maize leaf photos with different illnesses. These photographs include maize leaf blight, corn rust, corn grey spot disease, and healthy corn leaves. Classifier neural networks, a prominent machine learning technique, are used to help detect diseased maize leaves. These networks help identify and classify diseases. This page included a comprehensive study of the algorithms that they used, in addition to providing particular information about those algorithms. In this article, a demonstration of the neural network model that the authors advise using, which is based on VGG-16, is shown. The final model has an overall recognition rate that is on average 94.64% higher than other models.

2.5 DETECTION OF CORN GRAY LEAF SPOT SEVERITY LEVELS USING DEEP LEARNING APPROACH

A CNN-based deep learning model has been described in this article for the aim of multi-classification of the disease known as corn grey leaf spot (CGLS) that can be seen on the corn plant. This model takes into account the maize plant's CGLS illness, which can manifest itself in five unique severity levels. Diseases that affect corn leaves, such as corn general Leaf spot, common rust, and leaf blight are widespread and dangerous during maize harvesting. Thus, this work provides a maize plant CGLS (Common Grey Leaf Spot) disease detection technique. A multi-classification deep learning (DL) model identifies. The proposed approach accurately and efficiently classifies maize plants with CGLS disease, offering insights for disease management and mitigation. This method was developed as a result of the findings of the previous study. This model has the highest achievable detection accuracy, which is 95.33% when applied to high-risk severity level pictures.

2.6 THE IDENTIFICATION OF CORN LEAF DISEASES BASED ON TRANSFER LEARNING AND DATA AUGMENTATION

The paper suggests developing maize leaf disease models using a convolutional neural network (CNN) based on data augmentation and transfer learning. Diversifying the input data and developing a CNN model using transfer learning improves generalization and accuracy. The experiment separates Plant Village corn leaf photos into four categories: healthy, corn grey leaf spot, corn common rust, and maize northern leaf blight. Pre-trained GoogLeNet network parameters were modified to improve the model. The optimization model was trained using transfer learning, including GoogLeNet, ResNet18, Vgg16, and Vgg19 networks. The improved model identified maize illnesses such corn common leaf rust, corn northern leaf blight, and healthy leaves with 97.6% accuracy. Each category recognized above 95%. Healthy leaf tissue was recognized at 97.4%. The most accurate GoogLeNet model has 5.9% more precision than the original version. The enhanced model successfully identifies and classifies maize illnesses, outperforming the baseline model.

2.7 COVID-19 AND PNEUMONIA CLASSIFICATION USING ENSEMBLING WITH TRANSFER LEARNING

The diagnosis of COVID-19 is investigated using a chest X-ray in this particular piece of research. Transfer learning was used in conjunction with VGG16, DenseNet, and MobileNet to classify the X-ray images of the chest taken of the patient. The goal of implementing Ensemble Learning is to produce a powerful learner by making use of the aggregation of weak learners in order to ensure that better outcomes are attained. This is accomplished by using the aggregation of weak learners. These models are trained using data from three unique groups of people: COVID-19 patients, persons who have pneumonia, and normal patients. Overall, a degree of correctness of 95.2% was demonstrated by the comprehensive testing outcomes that were accomplished by the use of assembling aggregate.

2.8 MAIZE SMALL LEAF SPOT CLASSIFICATION BASED ON IMPROVED DEEP CONVOLUTIONAL NEURAL NETWORKS WITH A MULTI-SCALE ATTENTION MECHANISM

The authors of this article developed a data set on maize small leaf spot that comprised of 1268 photographs encompassing a spectrum of disease severity as well as leaves that were in good health. This data set was included in the article. According to the findings of comparison trials, the DISE-Net algorithm performed much better than the conventional VGG16 (91.11%), ResNet50 (89.77%), InceptionV3 (90.97%), MobileNetv1 (92.51%), MobileNetv2 (92.17%), and DenseNet121 (94.25%). The accuracy of DISE-Net, as measured by the tests, was 97.12%. In addition, it has been demonstrated that the DISE-Net decision-making process is capable of devoting a larger amount of attention to the important elements by utilizing the network visualization tool known as Grad-Cam.

2.9 CLASSIFICATION OF CORN DISEASES FROM LEAF IMAGES USING DEEP TRANSFER LEARNING

In this particular research project, the authors concentrate on the utilization of deep transfer learning in order to categorize three different maize diseases in addition to healthy plants. Diseases such as Cercospora leaf spot, common rust, and northern leaf blight are examples of these conditions. There was no preprocessing or explicit feature extraction done on the image of a corn that was utilized as the input to models that were constructed using convolutional neural networks. Transfer learning was carried out with the aid of well-established and well-designed deep learning models, and it was thoroughly analyzed with the assistance of a number of alternative scenarios for dividing the data. Both of these processes were carried out with the intention of improving the accuracy of the results. The categorization of the four classes was completed with an accuracy level that averaged out to 98.6%.

2.10 CORN LEAF DISEASE CLASSIFICATION AND DETECTION USING DEEP CONVOLUTIONAL NEURAL NETWORK

The fundamental objective of this study was to determine acceptable architectures that are founded on deep learning and have the capability to classify and pinpoint illnesses that affect maize leaf. They suggested a method for illness classification and localization that was based on convolutional neural networks and incorporated data augmentation, transfer learning, and hyper parameter tuning. This method was based on the idea that diseases may be localized to specific areas. For the purpose of experiments in which it was essential to differentiate healthy leaves from infected leaves and to determine which regions of the leaf are affected, an open-source dataset known as the Corn leaf infection dataset was employed. They solved the classification issue by employing a VGG block model, limiting the number of parameters in order to obtain faster convergence while still preserving a high degree of accuracy, and they kept the number of parameters minimal. A cutting-edge deep learning model known as YOLOv4 was utilized by the researchers so that they could ascertain the precise location of the infection on the corn leaf. When it comes to classification, their model has an accuracy of 99.25%, and when it comes to detection, it has a mean average precision (mAP) of 55.30%.

2.11 MAIZE LEAF DISEASE CLASSIFICATION USING DEEP CONVOLUTIONAL NEURAL NETWORKS

The authors of this specific piece of study built a deep convolutional neural network (CNN)-based architecture, which they called a modified version of the LeNet, for the purpose of classifying maize leaf illnesses. The experiment is carried out with the assistance of pictures of maize leaves that were obtained from the PlantVillage collection. The CNNs that are being considered are educated to distinguish between a total of four unique classes (three illness classes and one healthy class). The learned model is able to obtain an accuracy level of 97.89%.

2.12 SUMMARY

The Summary of the research gap of each literature survey paper is shown in the below Table 2.1.

S.No.	Gaps
1	Dataset is reduced in dimension and has not achieved high accuracy.
2	Used a high-quality image dataset and achieved 89% test accuracy.
3	Used brute force for finding optimal K value which is time taking.
4	Classifying Healthy correctly other classification is poor and average of 93%. Missed Dataset balancing
5	Only Severity Level of Gray Leaf Spot is detected but not other types of disease. Less Data is used to train Model. Data Augmentation is not done. Less accurate nearly 94%.
6	Deep ensemble models are not used and only 500 images were used as part of the experiment
7	The training accuracy and validation accuracy had a gap which may be sign of overfitting and the performance difference was less.
8	This study doesn't look at finding more than one disease in corn.
9	The dataset is of less number of images.
10	The detection model achieves 55.30% accuracy
11	Lesser numbers of pictures are used in the studies.

Table 2.1: Summary of Gaps

CHAPTER – 3

REQUIREMENTS AND ANALYSIS

3.1 SOFTWARE REQUIREMENTS

Python:

Python is a tool for writing that can be used for many different things. It was designed by Guido van Rossum and released in 1991. One of the key features of Python is its emphasis on code readability, which is achieved through the use of whitespace and clear syntax. Python provides a variety of tools and features that allow developers to write efficient and effective code, both for small and large projects.

Windows 11:

Microsoft's Windows OS (Operating System) family of software runs on a range of gadgets, including desktops, laptops, tablets, and smartphones. The newest version is Windows 10, which was made available in July 2015. A graphical user interface, a web browser, a collection of apps, and support for hardware like keyboards, mouse, and gaming controllers are all included in the Windows OS. Additionally, the OS comes with several essential functions including networking, security, and system administration.

3.2 HARDWARE REQUIREMENTS

Windows 10 or higher

I5 processor or higher

Ram 8 GB – Minimum, the meta-model, stacking needs a substantial quantity of data.

CHAPTER – 4

SYSTEM DESIGN

4.1 HIGH LEVEL DESIGN

The High Level design in the figure 4.1 shows that when given an input image of a corn leaf, the image is then sent to four models: VGG16, MobileNet, Xception, and InceptionV3. Each model provides its own prediction or probability for the image.

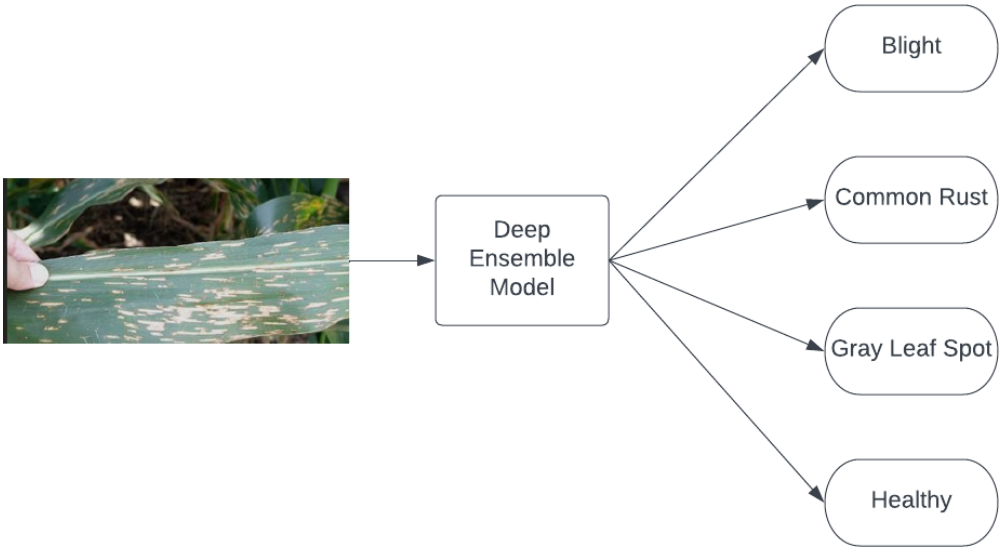

Fig 4.1: High Level Design

To determine the final outcome or class to which the image belongs, an ensemble method such as majority voting, averaging, or weighted averaging is used. Majority voting selects the class with the most votes, averaging calculates the average probability across models, and weighted averaging assigns weights to models and calculates a weighted average probability. These techniques help in obtaining the final prediction for the corn leaf image.

4.2 LOW LEVEL DESIGN

The figure 4.2 shows the Low-Level design of our project. The maize dataset is a collection of photographs of maize plants that need to be categorized according to the properties of the plants themselves. There is a possibility that the dataset has class imbalances, in which certain classes have a significantly higher number of samples than others. Because of this, the performance of the machine learning models may suffer because they have a tendency to be biased toward the classes that have more samples. In order to begin the process of preparing the dataset, the first step is to balance it.

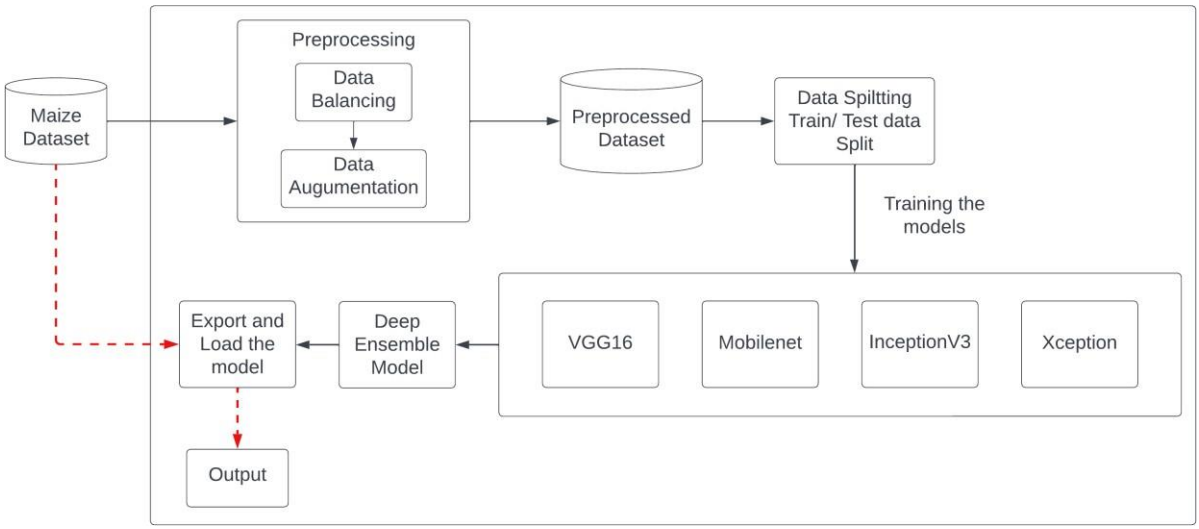

Fig 4.2: Low-Level Architecture

In order to ensure that the dataset is balanced, it is necessary to check that each category contains the same amount of samples. This can be accomplished by systematically deleting samples from classes that initially had a greater number of samples, up until the point at which they had the same number of samples as classes that initially had a smaller number of samples. This guarantees that there is an equal amount of each class's representation in the dataset.

After the information has been cleaned up and reorganized, the size of the dataset can be increased using procedures known as data augmentation. The photos are subjected to a variety of modifications during the process of data augmentation. Some of these transformations include rotation, flipping, zooming, and cropping. This results in the generation of additional samples that are comparable to the first samples but contain minute differences, which enables the machine learning models to more effectively generalize their findings.

Following the completion of the data augmentation step, the previously preprocessed dataset is then subdivided into training and testing sets. While the testing set is used to assess the accuracy of the machine learning models that have been trained using the training set, the training set itself is used to train the models.

Within the context of this scenario, four distinct machine learning models are utilized. These models are VGG16, MobileNet, InceptionV3, and Xception. All of these models are of the type known as convolutional neural networks (CNN), which are frequently applied to problems involving image classification. Each of these models possesses its unique architecture, which governs the manner in which the input photos are processed in order to generate the output categorization.

The machine learning models become adept at recognizing patterns in the input photos that are linked with the various classes as they progress through the training process. They accomplish this by modifying the weights of their internal layers in order to reduce the amount of variance that exists between the outputs that were anticipated and the actual outputs. This is accomplished with the help of a loss function, which determines the amount of variance that exists between the outputs that were predicted and the actual outputs.

The effectiveness of the machine learning models is judged according to the results of the testing set. This is accomplished by feeding the testing set into the models that have been trained and then contrasting the projected outputs with the actual results. The proportion of samples that are correctly identified is used as a measure of how accurate the models are.

The outcomes of the individual models' training and evaluation are then input into a deep ensemble model once the individual models have been trained and evaluated. In order to achieve a higher level of accuracy in the categorization as a whole, the deep ensemble model aggregates the results obtained from the individual models. In order to accomplish this, a weighted average of the probabilities that are predicted by each model is computed. During the training process, the weights are determined by taking into account how well each model performed on the validation set.

The final step involves putting the pre-processed dataset through the deep ensemble model, and the output is the ultimate prediction for each input image. It is dependent on the accuracy of the individual models, the variety of the models, and the weights that are applied to each model in the deep ensemble model for the final forecast to be as accurate as possible.

CHAPTER – 5

SYSTEM IMPLEMENTATION

5.1 DATA PRE-PROCESSING

5.1.1 DATASET

This dataset contains different diseases of corn leaves for identifying whether the plant is healthy or not. Although the dataset has a smaller number of images we use data augmentation to train the model.

The collection includes 1306 pictures of common rust, 574 pictures of grey leaf spot, 1146 pictures of blight, and 1162 pictures of healthy plants. The sample of the dataset is shown in the figure 5.1

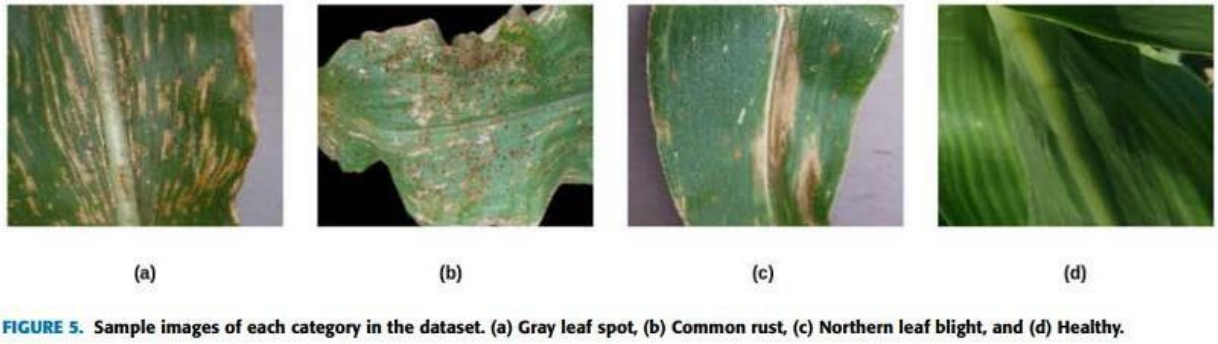

FIGURE 5. Sample images of each category in the dataset. (a) Gray leaf spot, (b) Common rust, (c) Northern leaf blight, and (d) Healthy.

Fig 5.1: Sample of the Dataset

Dataset Link: Corn or Maize Leaf Disease Dataset | Kaggle

5.1.2 DATA BALANCING

The process of altering the distribution of classes or categories within a dataset so that each class has a nearly equal number of samples is referred to as "balancing the data," and it is a part of the process known as "data balancing."

An unbalanced dataset in machine learning or deep learning might result in biased models that are less effective when applied to underrepresented classes of data. For instance, in a problem of binary classification with 90% of the cases being negative and 10% of the examples being positive, a model that successfully predicts all of the negative examples will nevertheless obtain a high accuracy of 90%. However, in applications that take place in the real world, where accurately identifying positive examples is of the utmost importance, this model is virtually useless.

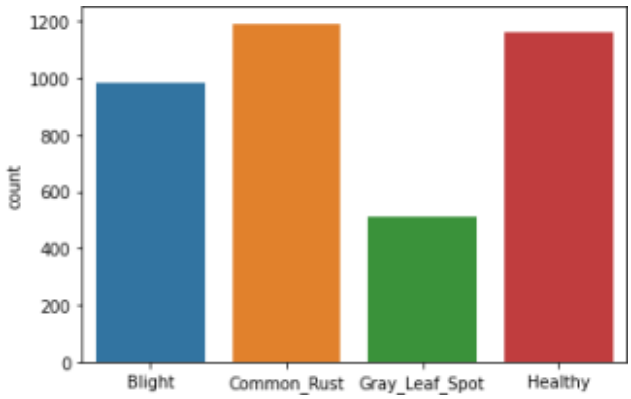

Fig 5.2: Before Balancing

The figure 5.2 shows the dataset distribution before balancing, In order to solve this problem, data balancing strategies can be utilised to make certain that each category receives a same amount of coverage within the dataset. This goal can be reached by either taking too many samples from the minority class or not taking enough samples from the majority class.

Under sampling requires deleting examples from the majority class so as to raise the representation of the minority class in the dataset, whereas oversampling requires reproducing examples from the minority class in order to increase the representation of the minority class in the dataset. In addition, there are hybrid methods that mix oversampling and under sampling in order to produce a dataset that is more evenly distributed.

In situations in which the distribution of classes is substantially uneven, data balancing can improve the performance and generalization of machine learning and deep learning models. This is especially true in situations where the distribution of classes is highly uneven. The figure 5.3 shows the dataset distribution after balancing.

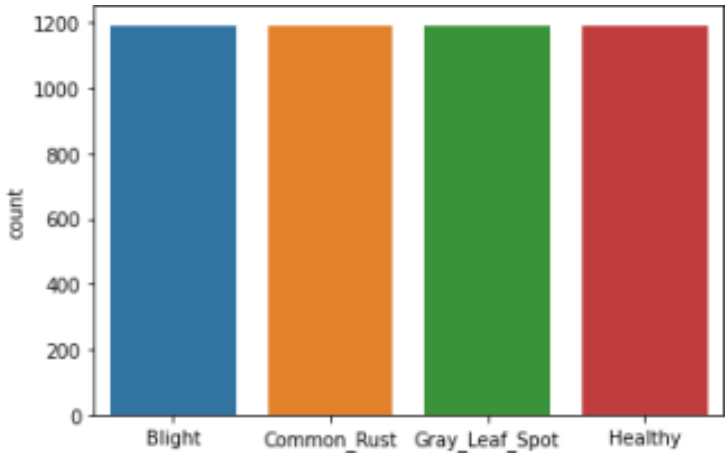

Fig 5.3: After Balancing

5.1.3 DATA AUGMENTATION

Data augmentation is a method used in machine learning and deep learning to make a dataset bigger by making new examples from the ones that are already there. This is accomplished through the process of data augmentation. The purpose of adding more data to a training set is to broaden its scope in order to increase a machine learning model's ability to generalise its findings. This is accomplished through the process of data augmentation.

New data examples can be generated through the process of data augmentation by performing a variety of transformations to the original data. These modifications can include rotating, flipping, scaling, cropping, or adding noise to the photos. These changes have the potential to assist the model in learning invariant features, increase the model's robustness to perturbations in the input data, and decrease overfitting.

When the size of the initial dataset is very small or when the model is prone to overfitting, data augmentation is an especially helpful technique to employ. The prevention of overfitting, the reduction of bias, and the improvement of the model's accuracy can all be assisted by data augmentation, which involves expanding the amount of the dataset. In addition, data augmentation can help to lessen the need for manual data gathering and annotation, both of which are labor-intensive processes that can also add financial burdens.

Types of Augmentation:

- Rotation: Random rotation is a sort of rotation in which the angle of rotation is chosen at random from among a number of different possible angles. As shown in figure 5.4, this may be helpful in increasing the heterogeneity of the dataset and improving the model's capacity to generalise its findings.

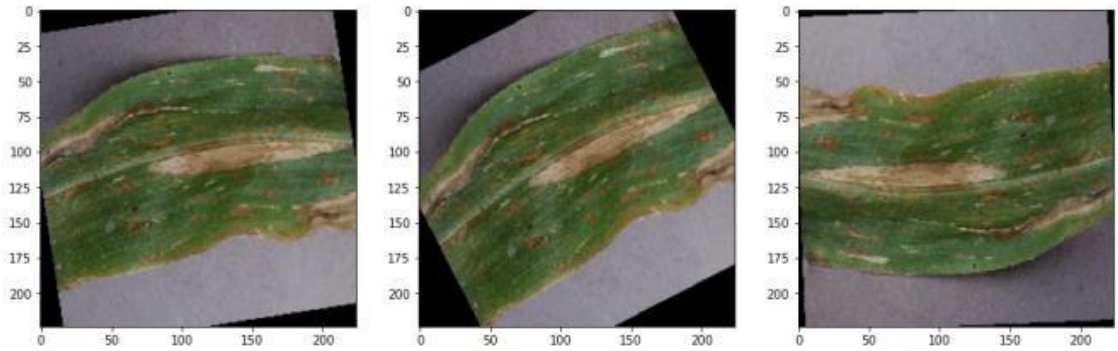

Fig 5.4: Rotation Augmentation Technique

- Width Shift: This method includes moving the picture or data sample horizontally by a predetermined number of pixels or a percentage of the image's width. The width shift technique is also known as the horizontal shift. It is possible for models to learn to recognise items at various places within the picture with the aid of this as shown in figure 5.5

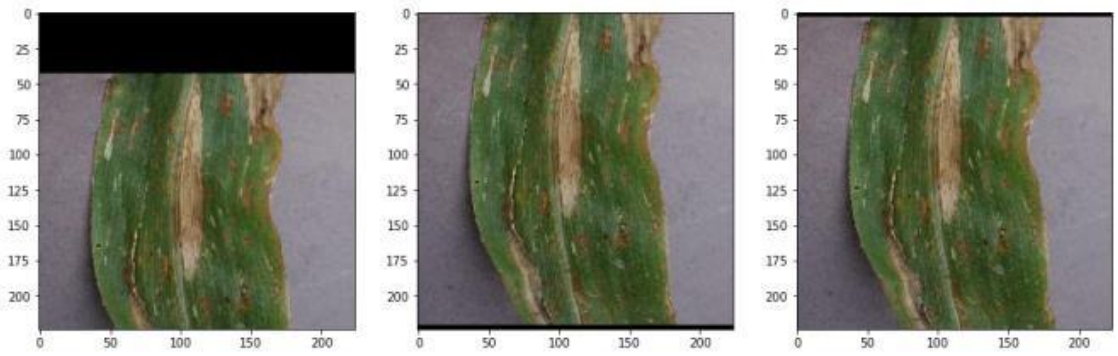

Fig 5.5: Width Shift Augmentation Technique

- Height Shift: Shifting the height of an image or data sample vertically by a certain number of pixels or a percentage of the total height of the picture is what this approach entails. It may teach models to identify things at varying heights inside a picture, which is useful for object recognition as shown in figure 5.6

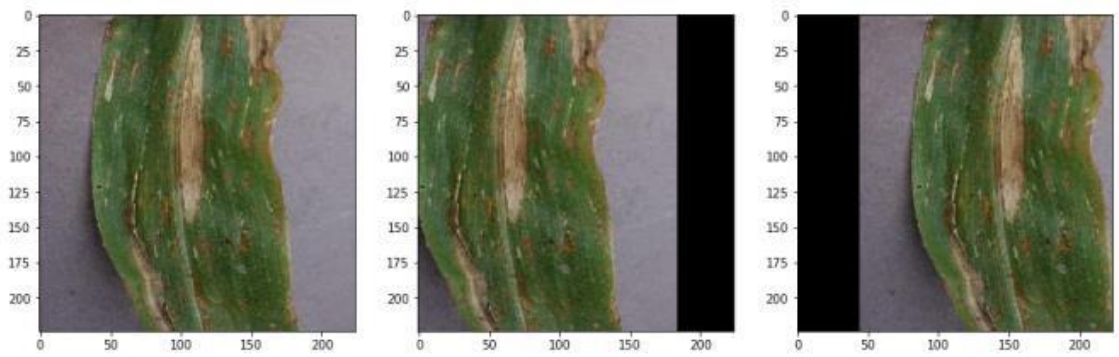

Fig 5.6: Height Shift Augmentation Technique

- Brightness: This method includes increasing the brightness of a picture by either adding or removing a constant value or by multiplying the pixel values by a scaling factor. Brightness may also be adjusted by using a scaling factor. It can assist models in becoming more adept at recognising things under a variety of illumination situations as shown in figure 5.7

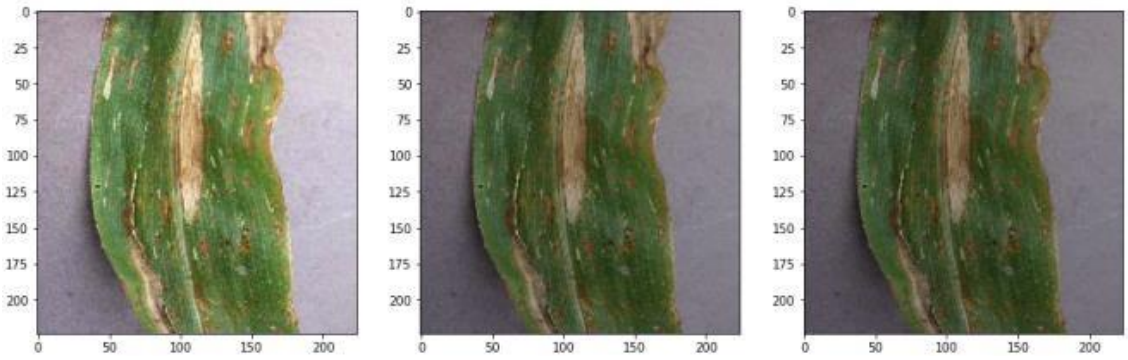

Fig 5.7: Brightness Augmentation Technique

- Zoom: Zoom is a technique that allows a picture to be zoomed in on or out of by cutting off a piece of the image and then resizing it to its original dimensions. It is possible for

models to learn to recognise items at various scales and sizes within the image as a result of using this information as shown in figure 5.8

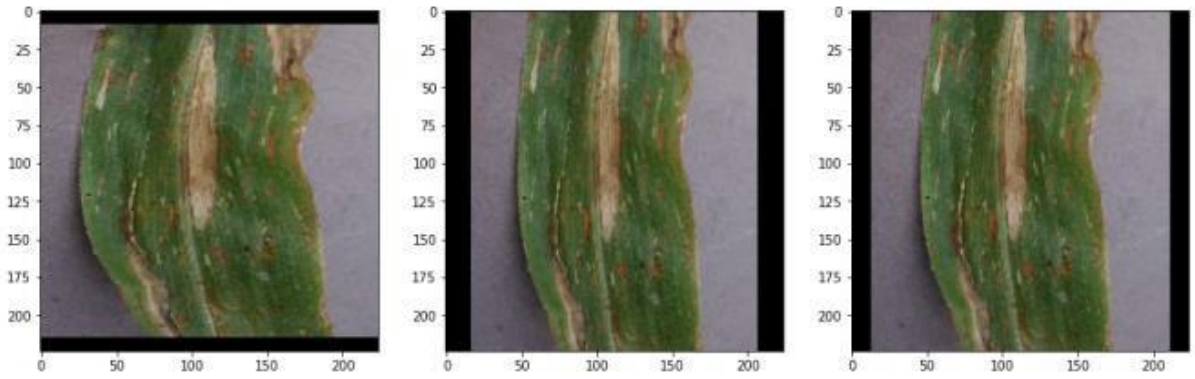

Fig 5.8: Zoom Augmentation Technique

- Horizontal Flip: The picture or data sample is flipped horizontally in this approach, which is referred to as the horizontal flip. It is possible for models to acquire the ability to recognise things that are symmetrical or appear in a variety of orientations with the aid of this as shown in figure 5.9

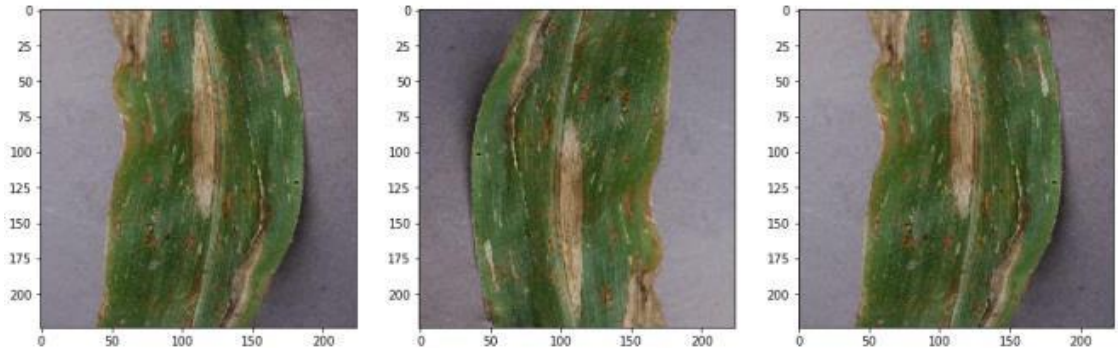

Fig 5.9: Horizontal Flip Augmentation Technique

These data augmentation approaches can assist enhance the performance and generalisation of machine learning and deep learning models, as well as expand the amount and variety of a dataset. The models are able to learn to recognise objects under a variety of settings and become

more resilient to noise and other types of variability in the data when these approaches are applied to the training data and used in the training process.

5.2 MODELS USED

VGG-16:

Image classification is a frequent application of the VGG16 convolutional neural network (CNN) architecture, which is a deep learning model consisting of 16 layers of convolutional and fully connected layers. This architecture was developed by Google.

As shown in figure 5.10, Following the 13 convolutional layers that make up the VGG16 architecture are the three fully linked layers. The first 12 convolutional layers each make use of a 3x3 filter size with a stride of 1, while the final convolutional layer makes use of a 3x3 filter size with a stride of 2, and both make use of the same padding. The pooling layers are implemented with max pooling, utilizing a 2x2 filter size and a stride of 2 for each layer.

Image recognition, object detection, and segmentation are just some of the many computer vision tasks that have been successfully accomplished with the help of VGG16's pre-trained model. Its architecture has also been used as the foundation for the construction of a great number of additional CNN architectures, including VGG19, which is a more in-depth variant of VGG16.

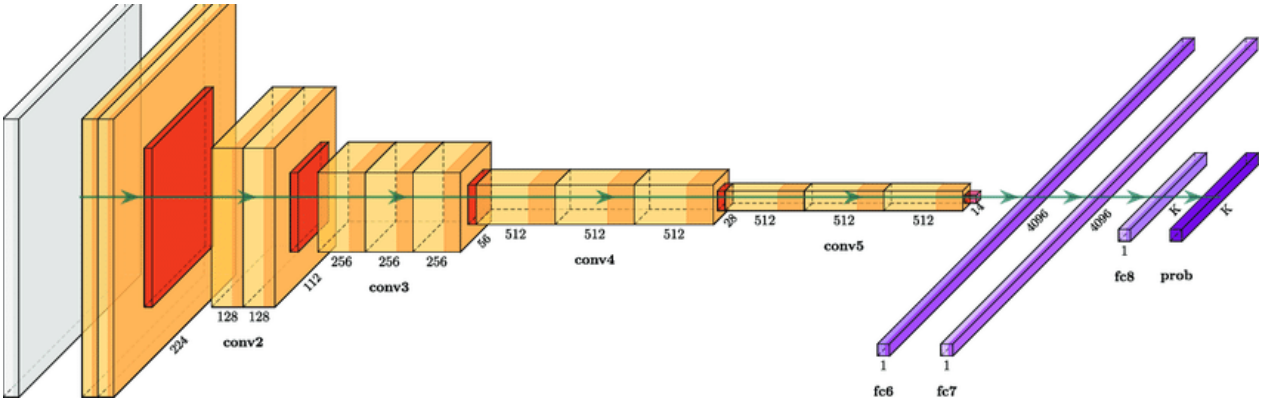

Fig 5.10: VGG-16 Architecture

MobileNet:

MobileNet uses depthwise separable convolution to improve performance by dividing a traditional convolution into two layers: depthwise and pointwise. Depthwise convolution mixes and reduces channels by applying one filter to each input channel. This method captures channel-specific geographical information. Pointwise convolution follows depthwise. A 1x1 filter on the depthwise convolution output mixes and lowers channels. This stage combines spatially improved data from the depthwise convolution and captures more intricate channel interactions. MobileNet balances computational economy and model accuracy by splitting the convolution operation into two layers. Depthwise separable convolution captures relevant characteristics in input data with fewer parameters and calculations as shown in figure 5.11

With the help of this method, the number of computations and parameters that are necessary for the model can be cut significantly while still retaining a respectable degree of precision. Other CNN architectures, such as VGG and ResNet, tend to be significantly larger in size; however, MobileNet models are often substantially smaller and have a much faster inference time.

Several different computer vision tasks, including picture classification, object identification, and segmentation, have been accomplished with the assistance of MobileNet.

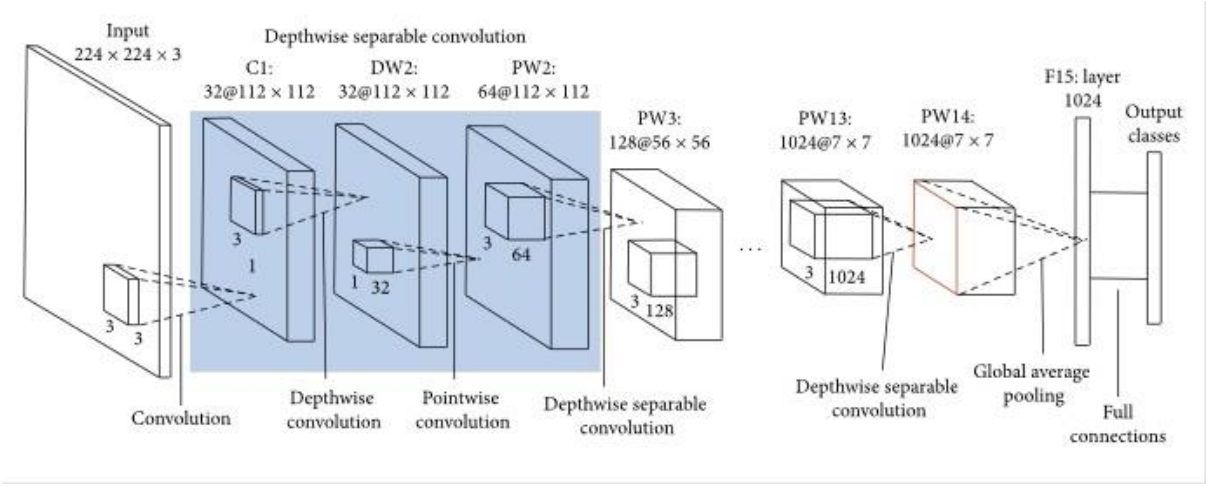

Fig 5.11: MobileNet Architecture

InceptionV3:

A method known as "inception modules," which are convolutional layers with different filter sizes, is utilized by the architecture of InceptionV3. These modules enable the network to capture features at numerous scales and reduce the number of parameters by utilizing 1x1 convolutions to combine the outputs of different filter sizes. As shown in figure 5.12, Additionally, the network is able to reduce the amount of memory required to store the captured features.

InceptionV3 improves generalization using batch normalization and regularization. Batch normalization stabilizes and accelerates convergence by normalizing layer activations. This method makes the model more generalizable and less susceptible to input data fluctuations. L1 or L2 regularization prevents overfitting and improves generalization. Regularization penalizes the loss function, pushing the model to learn simpler, more robust data representations. InceptionV3 has convolutional, pooling, fully connected, and auxiliary classifier layers. These components improve training gradient flow. Convolutional, pooling, and fully linked layers extract characteristics from input data. Auxiliary classifiers at intermediary layers increase gradient propagation and give extra oversight during training. InceptionV3 uses these approaches and architectural components to improve model training, generalization, learning, and feature extraction from input data.

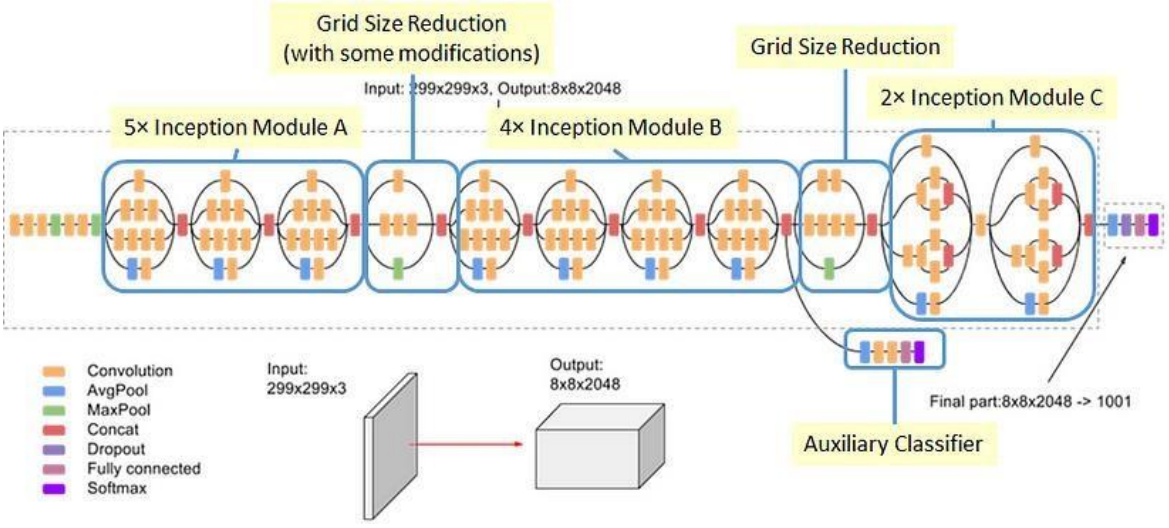

Fig 5.12: InceptionV3 Architecture

Several different computer vision tasks, including picture classification, object recognition, and segmentation, have been accomplished with the assistance of InceptionV3. It has accomplished state-of-the-art performance on a number of benchmark datasets and has served as a foundation for the development of a large number of different CNN designs.

Xception:

The idea of depthwise separable convolution was first presented in the MobileNet design, and this idea serves as the foundation for the Xception architecture. In Xception, depthwise separable convolutions are done in a "cross-channel" manner, which means that each depthwise convolution is applied over all channels rather than just one channel. This is accomplished by using the "cross-channel" modifier. Because of this, the model is able to learn more complicated and fine-grained representations of features while still keeping a relatively low number of parameters.

The Xception architecture is made up of a number of depthwise separable convolutional layers, which are then followed by a number of "shortcut" connections that assist in enhancing the flow of gradients when the model is being trained. As shown in figure 5.13, The generalization performance of the model is improved thanks to the incorporation of batch normalization and regularization techniques within the model.

ImageNet was used to train Xception, and it has now achieved state-of-the-art performance on a number of benchmark datasets for image classification, object detection, and segmentation tasks. The Xception architecture has served as a foundation for the construction of a great number of different CNN architectures and has sparked research into the creation of various kinds of neural network architectures.

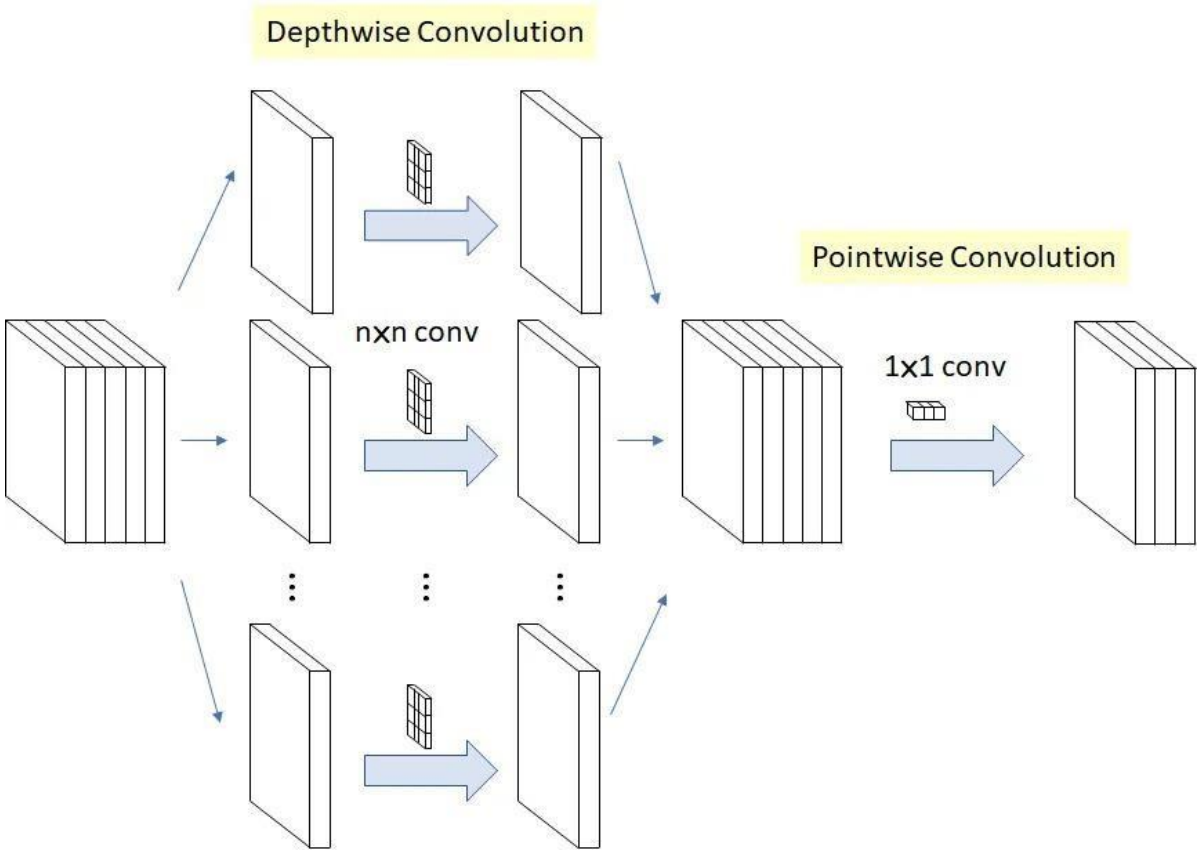

Fig 5.13: Xception Architecture

Ensemble Learning:

Ensemble learning tries to make a model more accurate and reliable by combining several separate models into a single model. This method uses the expectations of many different models to get better results than any single model can get on its own. This is the core concept of ensemble learning.

- **Stacking:**

  Stacking is a common strategy for ensemble learning that entails integrating numerous individual models by way of a meta-model. This meta-model is taught to learn how to integrate the predictions of the individual models into a single forecast.

The concept behind stacking is to train many separate base models on the training data, and then use the predictions from those models as input features for a meta-model that combines the results of all of the base models. After then, the meta-model will have the ability to aggregate the predictions of the underlying models into a single overall forecast. A straightforward linear regression is frequently selected as the meta-model; however, it is possible to make use of alternative models, such as neural networks, gradient boosting, or random forests.

Stacking as a kind of ensemble learning offers a number of advantages to its users. To begin, it makes it possible to combine several models, each of which has its own set of advantages and disadvantages, which often results in a higher level of total performance. Second, it can deal with non-linear correlations between the characteristics that are input and the variable that is produced. Finally, because the meta-model is trained on a different set of data than the basic models, it can contribute to the reduction of overfitting.

Nonetheless, stacking is not without its drawbacks and restrictions. Because it requires training on many models as well as a meta-model, it might be prohibitively costly to compute, which is one of its drawbacks. In addition to this, picking the best possible combination of base models and meta-model can be a challenge in and of itself. Finally, to successfully train both the base models and the meta-model, stacking needs a substantial quantity of data.

We have used all the default values for the hyper parameters which are shown in the below Table 5.1.

MODELS	Batch Size	Epochs	Learning Rate
VGG-16	128	5	0.001
MobileNet	128	5	0.001
InceptionV3	128	5	0.001
Xception	128	5	0.001

Table 5.1: Hyper Parameter values

5.3 MODULES WITH DESCRIPTION

OS:

Python's OS module offers a set of functions that may be used to communicate with various operating systems. The OS module is categorized with Python's other basic utility modules. This module lets you use features that depend on the operating system even when you're not on that system. The os and os.path modules both include a large number of functions that may be used to interface with the file system.

Cv2:

OpenCV is a huge open-source tool for computer vision, machine learning, and picture processing. Modern systems need it to do things in real time. It looks at pictures and videos to find people, items, and handwriting. When paired with NumPy, Python makes it possible to look at the structure of OpenCV's arrays. Python uses vector space to do math on picture shapes to figure out what parts they are made of. OpenCV takes on a wide range of problems, such as finding objects, recognizing faces, making medical images, etc.

Numpy:

Numerical Python, more commonly referred to as NumPy, is a library that comprises of multidimensional array objects as well as a set of functions for processing such arrays. NumPy is also referred to by its other name, Numerical Python. Numerical Python is what is meant to be abbreviated as NumPy. Users of the Python programming language have the ability to perform logical and mathematical operations on arrays thanks to NumPy, NumPy is a library for the Python programming language that focuses on linear algebra, Fourier transforms, matrices, and other related operations. It provides key tools for fast calculation. It is expected that the array object that NumPy offers would be up to 50 times quicker than the regular list data type that Python provides. This is one of the goals of the project.

Matplotlib:

Matplotlib is a data visualization library in Python that is extensively used. It includes a full collection of tools for making plots and charts of a high quality in two dimensions and three

dimensions. Because of its versatility, Matplotlib enables users to generate a wide range of visualizations, such as line plots, scatter plots, bar plots, and histograms, amongst others. Control over colors, typefaces, and axis scales are among of the comprehensive personalization possibilities that are made available by the library when it comes to the production of plots that are suitable for publishing. Because of its integration with Python's data analysis ecosystem, which includes NumPy and Pandas, Matplotlib is a powerful tool that can be used for both the exploration and analysis of data. Matplotlib has quickly become the go-to library for academics, scientists, and data analysts that need to produce high-quality representations for their work. Its thorough documentation and vast feature set are two of the main reasons for its popularity.

Seaborn:

Seaborn is a Python data visualization toolkit that was developed on top of Matplotlib and offers an easier-to-use interface for the creation of complicated and interesting statistical visualizations. Seaborn was created by the Seaborn Data Visualization Team. The higher-level interface of Seaborn makes it easier to create a variety of plots, including scatter plots, line plots, bar plots, histogram plots, and density plots, and has built-in support for statistical annotations. In addition to this, it offers more sophisticated methods of data visualization, such as pair plots, heatmaps, and facet grids, which may be used to investigate complicated relationships in the data. Seaborn was developed to interact effectively with Pandas, a well-known data analysis toolkit written in Python, and is capable of quickly managing enormous datasets. Seaborn has become a popular tool for data scientists, researchers, and analysts who want to quickly build high-quality visualizations to obtain insights from their data. This popularity may be attributed to Seaborn's straightforward application programming interface (API) as well as its complete collection of capabilities.

TensorFlow:

TensorFlow is a well-known open-source software library that is utilized for the construction of machine learning models as well as their subsequent training. It was built by the team at Google Brain, and it has since become one of the machine learning frameworks that is utilized most frequently in the business world. TensorFlow is a platform that allows for the construction of a wide range of machine learning models, ranging from straightforward linear regression models

to intricate deep neural network models. This platform is adaptable and scalable. In addition to support for distributed computing and deployment to production systems, it comes with a variety of built-in tools for data preparation, model development, training, and assessment. Because of its user-friendly interface and comprehensive documentation, TensorFlow is a tool that is often used by machine learning practitioners of all skill levels.

Scikit-Learn:

One of the most popular open-source machine learning packages for Python is called scikit-learn, which is sometimes referred to as sklearn. It is compatible with well-known statistical methods such as linear regression, logistic regression, decision trees, random forests, and support vector machines. It offers tools for the preparation of data, the selection of features and models, as well as assessment. In addition, it offers a range of tools for data preprocessing, feature selection, model selection, and assessment. The clustering and dimensionality reduction tools that are included in Scikit-learn are examples of the unsupervised learning techniques that are included. It is straightforward to use, especially for people who do not have a lot of expertise with machine learning because to its well-designed and user-friendly application programming interface (API). Additionally, NumPy, Pandas, and Matplotlib are all supported by scikit-learn according to its architecture, which makes it an indispensable component of the Python machine learning ecosystem. scikit-learn was developed with this compatibility in mind.

PIL (Python Imaging Library):

PIL, which stands for the Python Imaging Library, is a library in Python that is utilized often for working with various types of pictures. It offers a selection of tools for opening photos in a number of formats, such as JPEG, PNG, and BMP, as well as for altering and saving those images in those formats. PIL contains functionality for cropping, resizing, rotating, and flipping photos, as well as applying filters and other image processing techniques. Other capabilities include the ability to import and export images in a variety of formats. In addition to this, it offers assistance with picture analysis and the extraction of data, including the detection of edges and corners as well as the creation of color histograms. PIL is utilized extensively in a broad variety of applications, including computer vision, web development, and scientific research, to

name a few. Even though PIL has not been actively developed since 2011, its branch, Pillow, is a drop-in replacement for PIL. Pillow also receives active maintenance and offers support for newer Python versions in addition to extra capabilities.

CHAPTER – 6
RESULTS AND ANALYSIS

6.1 RESULTS

At first, the balanced dataset without Augmentation is used to train each of the four algorithms: VGG-16, MobileNet, InceptionV3, and Xception.

The accuracies with the dataset before Augmentation:

- VGG-16

	precision	recall	f1-score	support
Blight	0.85	0.84	0.84	99
Common_Rust	0.96	1.00	0.97	119
Gray_Leaf_Spot	0.80	0.72	0.75	51
Healthy	0.95	1.00	0.97	116
accuracy			0.91	385
macro avg	0.89	0.92	0.88	385
weighted avg	0.91	0.91	0.91	385

Accuracy: 0.91%

Fig 6.1: Classification report of VGG-16 before Augmentation

Figure 6.1 displays the classification report generated by the VGG-16 algorithm after it was trained on the dataset without augmentation.

- MobileNet

	precision	recall	f1-score	support
Blight	0.75	0.79	0.77	99
Common_Rust	0.95	0.94	0.94	119
Gray_Leaf_Spot	0.84	0.72	0.76	51
Healthy	0.93	0.95	0.94	116
accuracy			0.91	385
macro avg	0.87	0.88	0.87	385
weighted avg	0.91	0.91	0.91	385

Accuracy: 0.91%

Fig 6.2: Classification report of MobileNet before Augmentation

Figure 6.2 displays the classification report generated by the MobileNet algorithm after it was trained on the dataset without augmentation.

- InceptionV3

	precision	recall	f1-score	support
Blight	0.81	0.80	0.80	99
Common_Rust	0.98	0.97	0.97	119
Gray_Leaf_Spot	0.75	0.72	0.73	51
Healthy	0.95	0.98	0.96	116
accuracy			0.90	385
macro avg	0.87	0.86	0.86	385
weighted avg	0.90	0.90	0.90	385

Accuracy: 0.90%

Fig 6.3: Classification report of InceptionV3 before Augmentation

Figure 6.3 displays the classification report generated by the InceptionV3 algorithm after it was trained on the dataset without augmentation.

- Xception

	precision	recall	f1-score	support
Blight	0.81	0.97	0.88	99
Common_Rust	0.99	1.00	1.00	119
Gray_Leaf_Spot	0.90	0.51	0.65	51
Healthy	0.99	1.00	1.00	116
accuracy			0.93	385
macro avg	0.92	0.87	0.88	385
weighted avg	0.93	0.93	0.92	385

Accuracy: 0.927%

Fig 6.4: Classification report of Xception before Augmentation

Figure 6.4 displays the classification report generated by the Xception algorithm after it was trained on the dataset without augmentation.

Following the rebalancing of the dataset by the application of a variety of augmentation methods, we trained the dataset once again using each of the four algorithms.

- VGG-16

	precision	recall	f1-score	support
Blight	0.92	0.87	0.90	158
Common_Rust	0.99	0.99	0.99	155
Gray_Leaf_Spot	0.90	0.92	0.91	159
Healthy	0.96	0.98	0.97	159
accuracy			0.94	631
macro avg	0.94	0.94	0.94	631
weighted avg	0.94	0.94	0.94	631

Accuracy: 0.941%

Fig 6.5: Classification report of VGG-16 after Augmentation

Figure 6.5 displays the classification report generated by the VGG-16 algorithm after it was trained on the dataset after augmentation.

- MobileNet

	precision	recall	f1-score	support
Blight	0.90	0.90	0.90	158
Common_Rust	0.98	0.97	0.97	155
Gray_Leaf_Spot	0.94	0.94	0.94	159
Healthy	0.97	0.98	0.97	159
accuracy			0.93	631
macro avg	0.94	0.94	0.94	631
weighted avg	0.93	0.93	0.93	631

Accuracy: 0.933%

Fig 6.6: Classification report of MobileNet after Augmentation

Figure 6.6 displays the classification report generated by the MobileNet algorithm after it was trained on the dataset after augmentation.

- InceptionV3

	precision	recall	f1-score	support
Blight	0.86	0.85	0.85	158
Common_Rust	0.99	0.97	0.98	155
Gray_Leaf_Spot	0.85	0.86	0.86	159
Healthy	0.98	0.99	0.99	159
accuracy			0.92	631
macro avg	0.92	0.92	0.92	631
weighted avg	0.92	0.92	0.92	631

Accuracy: 0.919%

Fig 6.7: Classification report of InceptionV3 after Augmentation

Figure 6.7 displays the classification report generated by the InceptionV3 algorithm after it was trained on the dataset after augmentation.

- Xception

	precision	recall	f1-score	support
Blight	0.91	0.84	0.88	158
Common_Rust	0.99	0.98	0.99	155
Gray_Leaf_Spot	0.85	0.92	0.88	159
Healthy	0.98	0.99	0.98	159
accuracy			0.93	631
macro avg	0.93	0.93	0.93	631
weighted avg	0.93	0.93	0.93	631

Accuracy: 0.932%

Fig 6.8: Classification report of Xception after Augmentation

Figure 6.8 displays the classification report generated by the Xception algorithm after it was trained on the dataset after augmentation.

Comparative analysis of the F1-Score with regard to each algorithm

- VGG-16

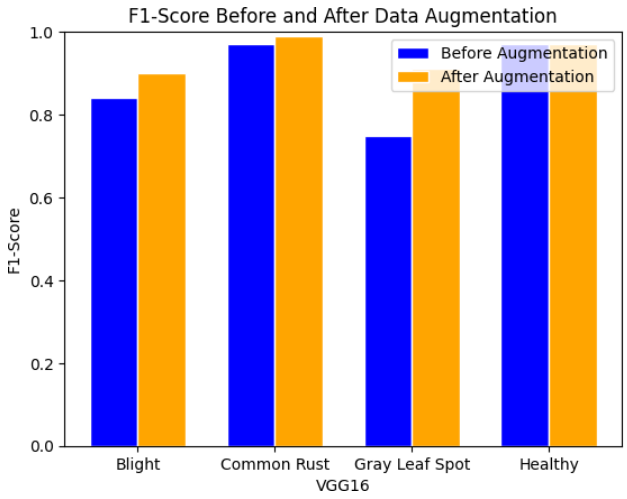

Fig 6.9: F1-Score using VGG-16

The F1-score of VGG-16 model, both before and after training on the augmented dataset, is depicted in the figure 6.9 that can be found above.

- MobileNet

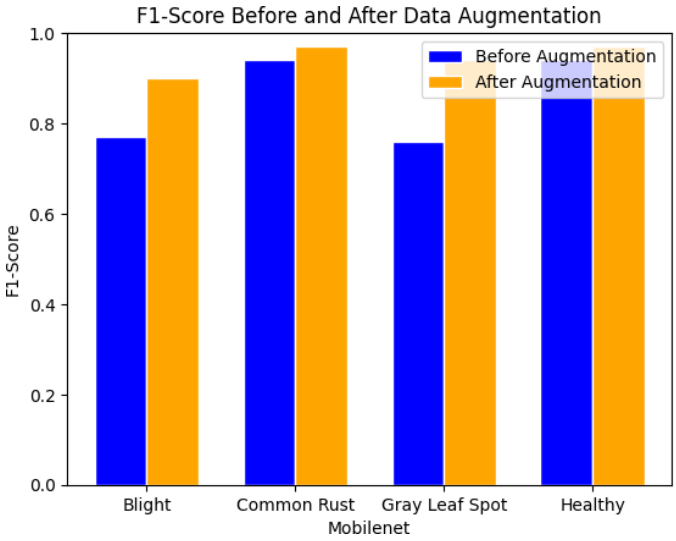

Fig 6.10: F1-Score using MobileNet

The F1-score of MobileNet model, both before and after training on the augmented dataset, is depicted in the figure 6.10 that can be found above.

- InceptionV3

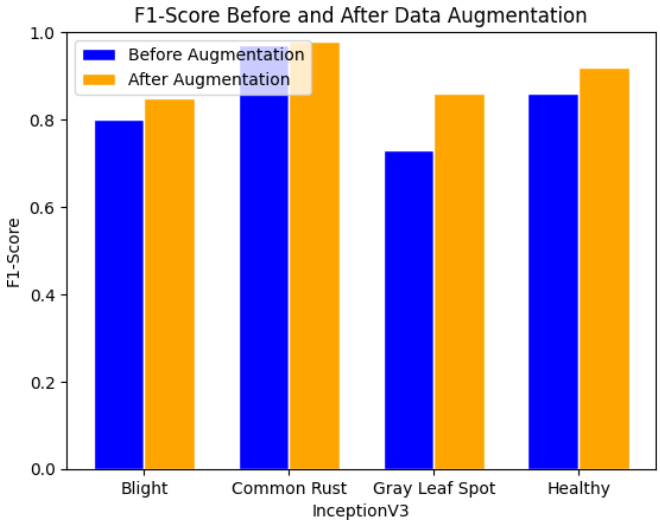

Fig 6.11: F1-Score using InceptionV3

The F1-score of InceptionV3 model, both before and after training on the augmented dataset, is depicted in the figure 6.11 that can be found above.

- Xception

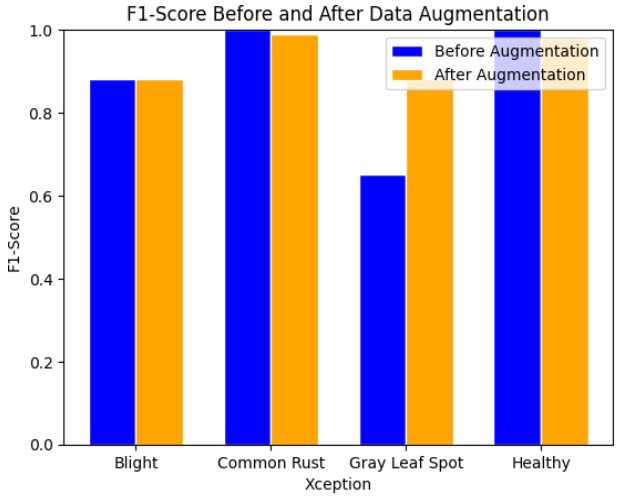

Fig 6.12: F1-Score using Xception

The F1-score of Xception model, both before and after training on the augmented dataset, is depicted in the figure 6.12 that can be found above.

The idea behind ensemble learning is that by combining multiple models that have different strengths and weaknesses, we can create a more robust and accurate system that can better handle a wide range of input data.

The outputs of many models can be combined in a number of straightforward ways, one of which is to simply take the average of their forecasts. This method is also sometimes referred to as "mean ensemble" or "average ensemble." As shown in figure 6.13

	precision	recall	f1-score	support
Blight	0.97	0.88	0.92	228
Common_Rust	1.00	1.00	1.00	239
Gray_Leaf_Spot	0.90	0.97	0.94	239
Healthy	1.00	1.00	1.00	243
accuracy			0.97	949
macro avg	0.97	0.96	0.96	949
weighted avg	0.97	0.97	0.97	949

Fig 6.13: Classification report of average ensemble

Before attempting to use the average of the ensemble, we must first train many models using the same data. After that, we take a specific input and run it through all of the trained models in order to produce a set of predictions. After then, in order to reach the final output, we take the average of all of the projections as shown in figure 6.14

```
Accuracy Score for average ensemble =  0.9652265542676501
```

Fig 6.14: Accuracy of average ensemble

The outputs of numerous models can be combined into a single set by the application of the ensemble learning technique known as weighted average. It is very similar to the average ensemble technique, but instead of taking the basic arithmetic mean of the predictions made by the individual models, we apply weights to each model, and then we compute a weighted average of the predictions made by those models as shown in figure 6.15. This method achieves the same results as the average ensemble technique.

	precision	recall	f1-score	support
Blight	0.96	0.90	0.93	228
Common_Rust	1.00	1.00	1.00	239
Gray_Leaf_Spot	0.91	0.97	0.94	239
Healthy	1.00	1.00	1.00	243
accuracy			0.97	949
macro avg	0.97	0.97	0.97	949
weighted avg	0.97	0.97	0.97	949

Fig 6.15: Classification report of weighted average ensemble

The concept of giving greater emphasis, or "weight," to the models that have superior accuracy or performance on the job at hand is the fundamental premise behind the weighted average ensemble. The performance of each model on a validation set or through cross-validation is often used to decide how the weights should be distributed across the models as shown in figure 6.16

```
Accuracy Score for weighted average ensemble =  0.9673340358271865
```

Fig 6.16: Accuracy of weighted average ensemble

Grid search is an effective method for optimizing the hyperparameters of the various models that make up the ensemble. Grid search can be found here as shown in figure 6.17. We may increase the ensemble's accuracy and robustness by doing a methodical search through a grid of hyperparameters in order to locate the combination that yields the highest performance on the validation set. This will allow us to determine the optimal setting for the ensemble. The figure 6.18 shows the max accuracy from the combinations of the weights.

	precision	recall	f1-score	support
Blight	0.98	0.92	0.95	228
Common_Rust	1.00	1.00	1.00	239
Gray_Leaf_Spot	0.93	0.98	0.96	239
Healthy	1.00	1.00	1.00	243
accuracy			0.98	949
macro avg	0.98	0.98	0.98	949
weighted avg	0.98	0.98	0.98	949

Fig 6.17: Classification report of Grid search

```
Max accuracy of  97.68177028451001  obained with w1= 0.4  w2= 0.3  w3= 0.4  and w4= 0.1
```

Fig 6.18: Accuracy of Grid search

The outputs of numerous models can be combined into a single prediction through the process of majority voting, which is a technique used in ensemble learning. The concept of majority voting proposes that each individual model in the ensemble should make a forecast, and then the output should be determined by the prediction that has the most frequency among all of the models. The figures 6.19 and 6.20 show the classification report and the accuracy of Majority Voting ensemble.

	precision	recall	f1-score	support
Blight	0.91	0.90	0.91	228
Common_Rust	1.00	1.00	1.00	239
Gray_Leaf_Spot	0.91	0.92	0.91	239
Healthy	1.00	1.00	1.00	243
accuracy			0.95	949
macro avg	0.95	0.95	0.95	949
weighted avg	0.95	0.95	0.95	949

Fig 6.19: Classification report of Majority Voting ensemble

```
Accuracy Score for majority voting ensemble =  0.9546891464699684
```

Fig 6.20: Accuracy of Majority Voting ensemble

6.2 CHALLENGES

- The success of the study depends on the dataset's availability and quality, which was utilized to train the models. Finding enough representative data can occasionally be difficult, and the models' performance may be impacted by the data's noise, imbalance, or incompleteness.
- Due to their comparable symptoms, blight as shown in figure 6.21 and gray leaf spot diseases as shown in figure 6.22 can be difficult to recognize precisely. Additionally, some pictures may have common-rust, which makes it more difficult to distinguish between the two diseases. The similar characteristics can be observed in the below images.

Fig 6.21: Blight Image

Fig 6.22: Gray Leaf Spot

- A detailed understanding of the issue domain and the models that are available is necessary for selecting the best deep learning models and setting their hyper-parameters. Overfitting, longer training periods, and poor performance can all be caused by choosing the incorrect models or employing the improper hyper-parameters.
- Another difficult component of the research is using ensembling techniques to mix the outputs of various models. When working with huge and complicated datasets, choosing which technique to employ and how to maximize the weights of each model can be challenging.

- Deep learning model training requires a lot of work, and for certain researchers, finding the necessary computing resources might be very difficult. Training times may be extended or tests may not be completed if high-performance computing infrastructure or hardware is not readily available.

CHAPTER – 7

CONCLUSION AND FUTURE SCOPE

7.1 CONCLUSION

The findings of the trials show, in conclusion, that deep ensemble learning strategies have the potential to considerably increase the accuracy of corn leaf disease classification. In spite of the tiny size of the dataset and the fact that it was imbalanced, the strategy of applying data augmentation in conjunction with balancing the dataset was successful in increasing the performance of the deep learning models. The four deep learning models (VGG16, MobileNet, InceptionV3, and Xception) all exhibited good performance on the dataset, with accuracy ranging from 90% to 92% when using the data before augmentation and 91% to 94% when using the data after augmentation respectively. This underscores the significance of data augmentation as a means of boosting the performance of deep learning models when working with datasets that are both small and imbalanced.

Additionally, the deep ensemble techniques that were utilized in this work (model averaging ensemble and weighted averaging ensemble) considerably increased the classification accuracy, obtaining an accuracy of 96.5% and 97.6%, respectively, in the process. This highlights the potential for deep ensemble learning to enhance the accuracy of classification tasks in areas such as agriculture and other fields where precise classification is essential.

The results of this research are essential for the field of agriculture because an accurate classification of diseases that affect corn leaf can assist farmers in early disease detection and treatment, which in turn leads to increased crop yields and more effective utilization of available resources. It is possible to apply the methodology of applying deep ensemble learning techniques to classify maize leaf diseases to other agricultural domains, which may result in significant gains in both accuracy and efficiency. Overall, this research makes a significant contribution to

the expanding body of literature on the application of deep learning in agriculture and sheds important light on the potential of deep ensemble learning approaches for the accurate categorization of maize leaf diseases.

7.2 FUTURE SCOPE

Extending the scope of the research to include different types of crops and diseases is one possible direction that could be pursued in subsequent research. The method that was created for this project has the potential to be utilized for the early detection and classification of illnesses in other crops, such as wheat, soybeans, or tomatoes. Because of this, farmers might be able to take prompt action to reduce crop losses and make better use of available resources. In addition, the development of a smartphone application that incorporates deep ensemble learning models could assist farmers in making decisions regarding crop management in real time. In addition, researching different strategies for interpretability may assist in comprehending how the models arrive at their conclusions, which, in turn, may assist in enhancing the models' capacity for interpretability as well as their reliability. This may result in a more widespread acceptance of the models within the agriculture business, which in turn may assist farmers in making decisions on crop management that are more informed.

REFERENCES

[1] Z. Liu, Z. Du, Y. Peng, M. Tong, X. Liu and W. Chen, "Study on Corn Disease Identification Based on PCA and SVM," 2020 IEEE 4th Information Technology, Networking, Electronic and Automation Control Conference (ITNEC), Chongqing, China, 2020, pp. 661-664, doi: 10.1109/ITNEC48623.2020.9084689.

[2] T. Ren, Y. Zhang and C. Wang, "Identification of Corn Leaf Disease Based on Image Processing," 2019 2nd International Conference on Information Systems and Computer Aided Education (ICISCAE), Dalian, China, 2019, pp. 165-168, doi: 10.1109/ICISCAE48440.2019.221610.

[3] H. Yu et al., "Corn Leaf Diseases Diagnosis Based on K-Means Clustering and Deep Learning," in IEEE Access, vol. 9, pp. 143824-143835, 2021, doi: 10.1109/ACCESS.2021.3120379.

[4] W. Wenxuan, W. Qianshu, H. Chaofan, S. Xizhe, B. Ruiming and T. T. Toe, "Leaf Disease image classification method based on improved convolutional neural network," 2022 IEEE International Conference on Industry 4.0, Artificial Intelligence, and Communications Technology (IAICT), BALI, Indonesia, 2022, pp. 210-216, doi: 10.1109/IAICT55358.2022.9887392.

[5] A. Baliyan, V. Kukreja, V. Salonki and K. S. Kaswan, "Detection of Corn Gray Leaf Spot Severity Levels using Deep Learning Approach," 2021 9th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO), Noida, India, 2021, pp. 1-5, doi: 10.1109/ICRITO51393.2021.9596540.

[6] Rongjie Hu, Shan Zhang, Peng Wang, Guoming Xu, Daoyong Wang, and Yuqi Qian. 2020. The identification of corn leaf diseases based on transfer learning and data augmentation. In Proceedings of the 3rd International Conference on Computer Science and Software Engineering (CSSE '20). Association for Computing Machinery, New York, NY, USA, 58–65. https://doi.org/10.1145/3403746.3403905

[7] P. Jamadagni, S. Patil, P. E. Grandhi and S. S. Verma, "COVID-19 and Pneumonia classification Using Ensembling with Transfer Learning," 2022 10th International Conference on

Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO), Noida, India, 2022, pp. 1-6, doi: 10.1109/ICRITO56286.2022.9965096.

[8] Yin, Chenghai & Zeng, Tiwei & Zhang, Huiming & Fu, Wei & Wang, Lei & Yao, Siyu. (2022). Maize Small Leaf Spot Classification Based on Improved Deep Convolutional Neural Networks with a Multi-Scale Attention Mechanism. Agronomy. 12. 906. 10.3390/agronomy12040906.

[9] Fraiwan M, Faouri E, Khasawneh N. Classification of Corn Diseases from Leaf Images Using Deep Transfer Learning. Plants. 2022; 11(20):2668. https://doi.org/10.3390/plants11202668

[10] Haque, Md & Adolphs, Julian. (2021). Corn Leaf Disease Classification and Detection using Deep Convolutional Neural Network. 10.13140/RG.2.2.20819.50722.

[11] Ahila Priyadharshini, R., Arivazhagan, S., Arun, M. et al. Maize leaf disease classification using deep convolutional neural networks. Neural Comput & Applic 31, 8887–8895 (2019). https://doi.org/10.1007/s00521-019-04228-3