Published by :
http://www.ijert.org

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 02, February-2025

Application of Deep Transfer Learning in Plant Leaf Diseases Detection

Azlou Jamal
School of Artificial Intelligence
Nanjing University of Information Science and Technology
Nanjing, China

Mehdi Khamir
School of Artificial Intelligence
Nanjing University of Information Science and Technology
Nanjing, China

Bah Ibrahima
School of Computer Science
Nanjing University of Information Science and Technology
Nanjing, China

***Abstract*— Extraction of plant features, identification, and classification of unhealthy plants are critical for food security. This paper describes a Transfer Learning (TL) approach for identifying Plant Diseases (PD) that combines feature extraction and fine-tuning with classification using traditional Machine Learning (ML). Our work mainly focuses on comparing the classification of Plant Diseases using traditional Machine Learning classifiers on top of Deep Learning (DL) extracted features with MobileNet and using deep Transfer Learning fine-tuning techniques to do the classification task. For the traditional ML classifier training, Cross-Validation is used. However, the PlantVillage dataset was downsampled for all of the experimentations, focusing more on the utility of using TL. The augmentation technique has been used in this study to deal with the problem of overfitting. We concluded that applying Transfer Learning in conjunction with data augmentation produces the best results in classifying plant leaf diseases.**

Keywords ; Plant Diseases (PD), Deep Learning (DL), Convolutional Neural Network (CNN), MobileNet, Transfer Learning (TL), Logistic Regression (LR), Cross-Validation (CV).

I. INTRODUCTION

Agriculture is an essential field in human life, forming the base of almost everything because it keeps us alive. It allows humans to provide food for us, but there is an era coming where human society needs to increase food production [1]. It is projected that in 2050, agricultural production needs to be increased by 70% [2] to feed an expected population size that is predicted to be over 9 billion people [3]. Infectious diseases reduce the potential yield by 40% [4], with many farmers in the developing world experiencing yield losses as high as 100%. For that reason, we need to focus our attention on this area and find solutions to the problems that occur and even predict eventual solutions for the upcoming difficulties with the help of Machine learning and AI. In many developing countries, agriculture is an essential sector of their economy. It provides employment, which can be over 50% of the jobs in several countries. Farmers' economic growth is determined by the quality of their produced items, dependent on plant growth and yield. Diseases in plants can be very harmful to the population; it causes economic losses, illnesses, and famine. Therefore, in agriculture, detecting disease in plants plays an instrumental role. This detection is an enormous difficulty because plants have similar properties, such as shape and colour representations. Plant Disease changes a plant's normal state that disrupts or changes its critical processes[5]. Plant Diseases manifest themselves in various plant areas, such as the leaves. Plant Diseases affect all species of plants, both wild and cultivated. However, each species is prone to specific conditions, only a few of them in each situation. Depending on environmental circumstances and pathogen presence, some plants are more prone to disease outbreaks than others. For instance, Plant Diseases such as Wilt; Blight; Canker; Crown gall; Rot; Scab; Anthracnose; Black knot; Clubroot; Damping-off; Dutch elm disease; Ergot; Leaf blister; Mildew; Rust; Smut; Bunt; Mold; Curly top; Mosaic; Psoriasis [6] primarily affect the leaves. With the recent development of Artificial Intelligence, many methodologies have been introduced for PD detection, mainly in Deep Learning. The availability of data on the internet and the labelling data done by many researchers make it easy to train models and use them for future prediction. The Transfer Learning approach is recently gaining more and more specialisation and has proved to be an efficient tool in the computer vision and natural language processing domain. Transfer Learning is the technique of reusing the weights of an already trained model on a new problem

II. PROBLEM STATEMENT

Plant Diseases have become a problem because they can significantly reduce the quality of agricultural goods. In developed countries, experts will classify ailments with their eyes closed by continuously examining plant leaves. Therefore, in developing countries, farmers are unaware of diseases and have no means of contacting experts. They confront many issues in certain circumstances, including time waste and high-cost operations. Automatic detection of plant illness is an important study topic because it might help detect disease symptoms as soon as they occur on plant leaves. It is also more convenient and less expensive than visiting professionals. With the emergence of Artificial Intelligence and the significant expansion of smartphones, the contribution to data collection is helping farmers [8]. Over the years, a variety of ways have been offered. Traditional Machine Learning, such as the Support Vector Machine Algorithm, has been used to detect

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 02, February-2025

and differentiate Plant Diseases [9]. Despite several efforts to use Machine Learning and computer vision to detect Plant Diseases, accurate identification of leaf diseases remains a difficult task. The major challenge is the characteristics and similarity of plant leaves. The lack of sufficient data on diseased plant leaves is a significant barrier to Deep Learning modelling. To deal with these challenges and improve accuracy rates in Plant Disease diagnosis, this research focuses on applying DL in plant classification by utilizing leaf identification. This study also uses Transfer Learning and classical Machine Learning classifiers to overcome the problem of applying DL with small leaf samples.r:

III. PRELATED WORKS:

Plant Disease is a primary factor affecting the production of plant yields. As a result, it is critical to diagnose and classify these disorders appropriately. Plants are essential for life, yet they face several challenges. An early and precise diagnosis reduces the danger of environmental harm [10]. Product quality and quantity deteriorate due to a lack of systematic illness identification. Various methods to automate plant classification using plant leaves have been employed by several researchers [11]. Various scientific methods have been deployed in the extraction of leaf features. Identifying and diagnosing agricultural illnesses is critical as early as possible [12][13]. Chemicals used to prevent diseases, such as fungicides and bactericides, on the other hand, have a detrimental influence on the agricultural ecology. As a result, we require rapid and effective disease classification and detection systems to benefit the agro-ecosystem. Image processing and neural networks, for example, enable the construction of systems capable of early disease detection in plants. As a result of stress, plant yield can be lowered by 50% [13]. The initial stage in detecting illness is inspecting the plant and determining what to work with based on experience [14]. The different innovative ways of identifying and classifying certain diseases have been used extensively. The motivation is to support farmers in identifying diseases early and taking preventive measures. To prevent the effect of low crop yield caused by bacteria, viruses, and fungus, researchers in [14] used Machine Learning methods by comparing the performances of techniques such as K-Nearest Neighbors and Fuzzy classifiers and a Convolutional Neural Network in detecting Plant Diseases. Their findings were that the CNN classifier detects more infections with high accuracy. Researchers in [15] used a back-propagation algorithm to get the weights of the neural network connections and optimised these weights using particle swarm optimization for the Plant Diseases detection. Researchers in [16] trained a deep CNN to recognise 14 crop species and 26 diseases using a publicly accessible dataset of 54305 images of damaged and healthy plant [4] leaves taken under controlled conditions. An approach based on a Recurrent Neural Network (RNN) was developed by [17] to automatically find diseased areas and extract meaningful information for disease classification. Compared to traditional CNN algorithms, their findings attempted to generalise to previously encountered afflicted crop species and diverse Plant Disease domain images. Other researchers have used Deep Learning to identify individual crops such as bananas [18], coffee [19], grape [20], cassava [21], and tomato [22],[10]. The image processing technology and classification algorithm detect and classify plant leaf diseases [23]. Researchers in [24] conducted a review study on the effects of combined abiotic and biotic pressures on plant growth and crop improvement options through physio-morphological features. There have also been investigations on hostile attacks and the detection of Plant Diseases using DL. According to the literature, hostile attacks with a small number of perturbations can significantly reduce the performance of DNN models for Plant Disease detection [25]. Authors of [10] used the Convolutional Neural Network to effectively define and categorise tomato diseases. They experimented on a dataset including 3000 images of tomato leaves infected with nine different diseases and a healthy leaf. The entire procedure is described as follows: The input images are first pre-processed, and the targeted region of images is split from the original images. Second, the images are further processed by altering the CNN model's hyper-parameters. Finally, CNN extracts various features from images like colours, textures, and edges. The results show that the suggested model predictions are 98.49% correct. Furthermore, various Deep Learning techniques outperform classical Machine Learning algorithms in terms of detection performance [26],[27],[28], [29], and [30]. Researchers in [31] suggested a new dataset of 79,265 images to address the limitations and flaws of the existing Plant Disease detection models. Based on the PlantVillage dataset [4], this collection is the largest dataset with leaf images available. On images collected in various weather conditions, at varied angles, and during daytime hours with an uneven background, they applied data augmentation using the traditional and state-of-the-art Generative Adversarial Network (GAN[32]. A novel two-stage neural network design was tested on the newly constructed dataset. The trained model had a 93.67 % accuracy rate. Although the evaluated methods in these publications appear to be effective and seem to learn relevant feature representations of the disease, they unfortunately also learn the irrelevant disease characteristics.

IV. METHODOLOGY

we pre-processed the data, resampling the dataset, showing the power of using Transfer Learning. Data Augmentation is used to deal with the overfitting of the model on the train set. The CNN architecture used is described in this chapter. The Transfer Learning techniques, feature extraction and fine-tuning, and the pre-trained model, are detailed. Logistic Regression is also trained on the extracted features from the pre-trained model.

A. Dataset and Preprocessing

1) PlantVillage Dataset:

The coloured images of the PlantVillage dataset [4] have been used for this work; this dataset contains 38 classes and 54,305 images of 14 different plant species in total, 12 of which are healthy and 26 of which are diseased. All the images are coloured, and the size of each is 256x256. The dataset is available on Kaggle, one of the most popular Machine Learning data repositories. A detailed presentation of the dataset is given in table 1, and image sample visualisation is given in figure 1. In this figure, the title is the number of the class of the corresponding image. This dataset covers many categories of Plant Diseases, though the motivation of using it for this study.

2) Data Pre-processing

The pre-processing technique prepares the data for the training and evaluation of a Machine Learning model. These

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 02, February-2025

TABLE 1: DETAILS OF THE PLANTVILLAGE DATASET

CLASSE	CLASS NAME	NUMBER OF SAMPLES
0	Apple Apple_scab	630
1	Apple Black_rot	621
2	Apple Cedar_apple_rust	275
3	Apple healthy	1645
4	Blueberry healthy	1502
5	Cherry_(including_sour)Powdery_ mildew	1052
6	Cherry_(including_sour)healthy	864
7	Corn_(maize)Cercospora_leaf_spot	513
8	Corn_(maize) Common_rus t_	1192
9	Corn_(maize) healthy	985
10	Corn_(maize)Northern_Leaf_Bligh t	1162
11	Grape Black_rot	1180
12	Grape Esca_(Black_Measles)	1383
13	Grape healthy	1076
14	GrapeLeaf_blight_(Isariopsis_Leaf _Spot)	423
15	OrangeHaunglongbing_(Citrus_gre ening)	5507
16	Peach Bacterial_spot	2297
17	Peach healthy	360
18	Pepper,_bell Bacterial_spot	997
19	Pepper,_bell healthy	1478
20	Potato Early_blight	1000
21	Potato Late_blight	1000
22	Potato healthy	152
23	Raspberry healthy	371
24	Soybean healthy	5090
25	Squash Powdery_mildew	1835
26	Strawberry Leaf_scorch	1109
27	Strawberry healthy	456
28	Tomato Bacterial_spot	2127
39	Tomato Early_blight	1000
30	Tomato Late_blight	1909
31	Tomato Leaf_Mold	952
32	Tomato Septoria_leaf_spot	1771
33	Tomato Spider_mites Two-spotted_spider_mite	1676
34	Tomato Target_Spot	1404
35	TomatoTomato_Yellow_Leaf_Curl _Virus	5357
36	Tomato Tomato_mosaic_viru s	373
37	Tomato healthy	1591

preparations can go from sampling the data to having a balanced one, splitting the data, enhancing the quality of the image, and augmenting the data. For this study, the main focus is the Transfer Learning technique, a challenge in dealing with the problem of having a small number of training samples per category. After downloading the dataset, we only used the colour folder. We downsampled the data, having the same number of samples, 152 for each class. We chose 152 as it's the smallest number in this dataset. The data has been shuffled to ensure we don't have repeated images in the final data. The then obtained images are moved to a new folder.

- Data Preparation for Training, Validation, and Testing:

The training set will be used to train the model, that is, to fit the model's parameters. The validation set will be used to evaluate the model after each step since it is vital to avoid biased evaluation of the fitted model on the training set when tweaking the model hyperparameters. After training, the model will be evaluated using the testing set. Random indexing is utilized for this split, and stratification ensures that each class receives the same number of samples in each split. We gave 80% (4620 images) for the training split, 10% (578 images) for the validation split and 10% (578 images) for the test split. See figure 2.

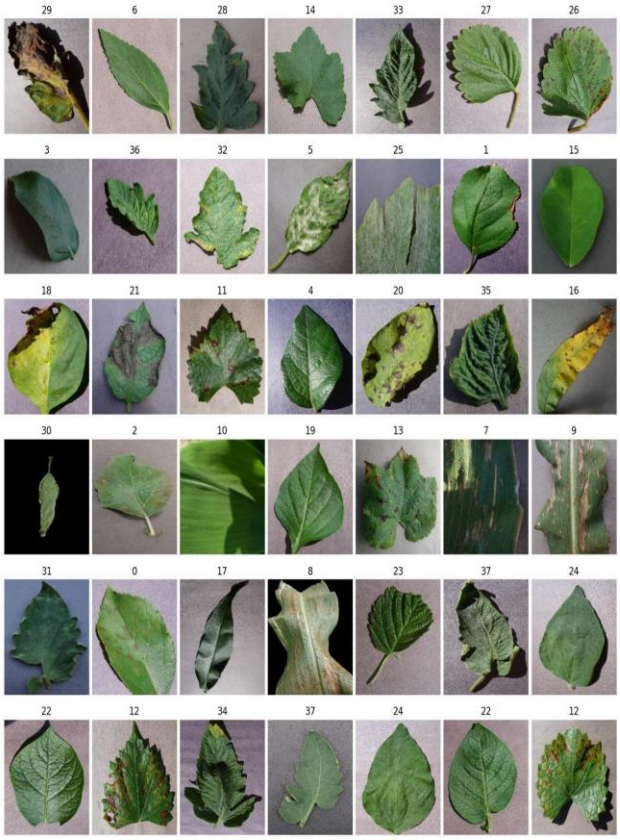

FIGURE 1: SAMPLE IMAGES FROM THE PLANTVILLAGE DATASET

The augmentation technique has been used here to facilitate the image loading and apply some pre-processing techniques to feed the data to the pre-trained models. Image augmentation is a strategy of increasing the size of the data, though encountering the challenge of having insufficient data to train a CNN model. It has been demonstrated that the strategy can effectively overcome overfitting, which is why we found it valuable in our research. When a model is extremely tightly fitted to a tiny amount of input data, this is known as overfitting. However, overfitting can occur when a model is too complicated; therefore, striking the tradeoff between model complexity and the number of input data is critical to avoid this issue.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 02, February-2025

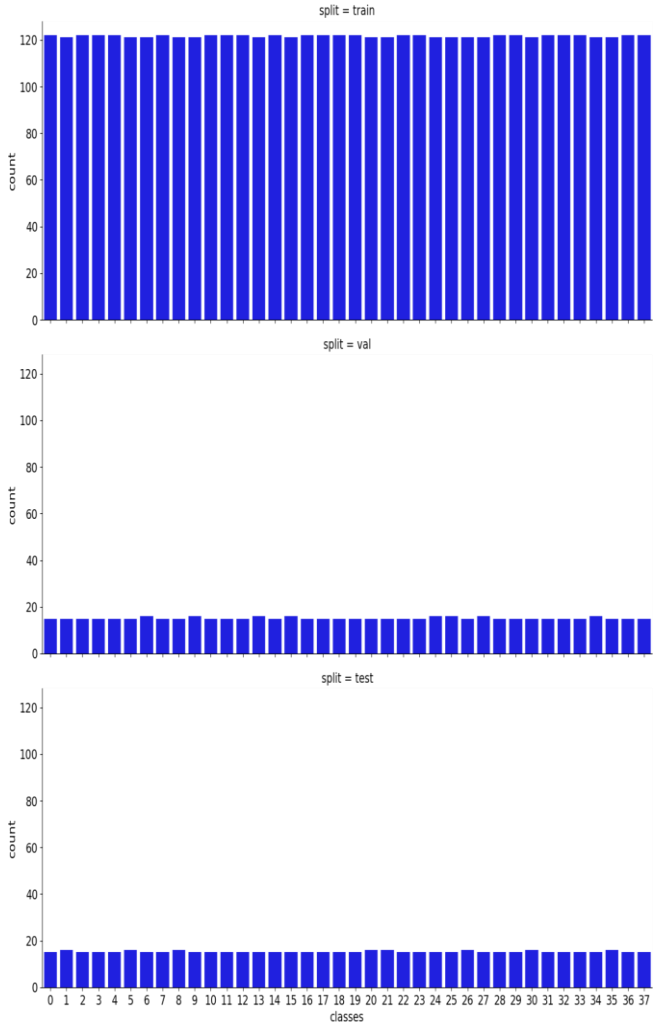

FIGURE 2: TRAINING, VALIDATION, AND TEST SPLITS OF THE DOWNSAMPLED DATASET

Because the goal is to demonstrate the efficacy of using Transfer Learning rather than a shallow CNN, image augmentation is required to make a fair comparison between these two approaches. Those augmentations can be geometric or colour space transformations of the original images, which will create diverse images for the model, allowing for a real-life vision of the data. Geometric transformations are rotation, translation, shifting, flipping, cropping, and zooming. Colour transformation can be colour augmentation, brightness, contrast, and saturation. Besides, we have other transformations, such as adding noise scaling. For our experiment, the only samples obtained for each class after downsampling the PlantVillage dataset are applied the following transformations: scaling, resizing, rotation, horizontal flip, height shift, and width shift to augment the samples' number. See figure 3.

Resizing: The images to the shape (224,224) were an important task, as we are using pre trained images in this shape.

Scaling: In the scaling process, we change the dimension of the image having the pixel range of [0,1] by dividing all the pixels by 255. The loaded images are in RGB channel colour; the highest is 255, corresponding to white, and the lowest is 0, which corresponds to black.

Flipping: We flipped the image horizontally and vertically. The image features are protected during the flipping while only rearranging the pixels.

Rotation: For this work, we rotate each image by 30 degrees.

Shifting: The shifting technique will bring the object to the centre of the image. By shifting the pixels horizontally or vertically, we recentred the object and gave the model a good view of all important features. This study sets the horizontal and vertical shifts to 20% each.

FIGURE 3: GEOMETRIC TRANSFORMATION ON A SAMPLE OF THE DATASET

It's an interesting image pre-processing technique that helps to avoid overfitting the model, and the generalization can be faster. After the data loading, preparation, and preprocessing steps, we prepared the model for training and testing. Details on different models experimented with are given below.

B. Basic Convolutional Neural Network

The first network proposed in this study is a basic CNN. The network consists of layering the convolutional block and classifier section. The feature extraction part has five convolutional layers and four pooling layers. In the beginning, we have two convolutional layers with weights initialized to He Normal, followed by a max-pooling layer. Each convolution is followed by its max-pooling layer for the rest of the feature extraction part. After each convolution, the obtained feature map is activated using the ReLU function. They are then followed by a flattened layer that changes the feature map to a 1D vector and two dense layers, each followed by a dropout layer added as regulariser, the dense layer as the output. We used the softmax activation function to classify the thirty-eight plant leaves in the final output layer. All details expressed above can be observed in our proposed framework architecture below in table 2. We applied a convolution filter (kernel) of size 3x3 for the first two convolution operations. To maintain the same shape as the input image, we used a stride of 1 alongside padding of 1. We started the convolution with 16 filters, the next one has 64, and the final 2 have 128 filters. As we go further into the network, the convolution kernels get bigger, the network learns more features, and the identification accuracy improves. The number of parameters for this network is quite huge (13,361,206 trainable parameters) since the sizes of the images are big (224,224). See table 2.

C. Feature Extraction

In this research, the idea is to use the Keras pre-trained MobileNet [34] to perform two tasks of Transfer Learning:

Feature extraction and fine-tuning the pre-trained model. The architecture is publicly available.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 02, February-2025

TABLE 2: BASIC CNN ARCHITECTURE DETAILS

LAYER TYPE	OUTPUT SHAPE	NUMBER OF PARAMS
Conv2D	(None, 224, 224, 16)	448
Conv2D	(None, 224, 224, 16)	2320
MaxPooling2D	(None, 112, 112, 16)	0
Conv2D	(None, 112, 112, 64)	9280
MaxPooling2D	(None, 56, 56, 64)	0
Conv2D	(None, 56, 56, 128)	73856
MaxPooling2D	(None, 28, 28, 128)	0
Conv2D	(None, 28, 28, 128)	147584
MaxPooling2D	(None, 14, 14, 128)	0
Flatten	(None, 25088)	0
Dense	(None, 512)	12845568
Dropout	(None, 512)	0
Dense	(None, 512)	262656
Dropout	(None, 512)	0
Dense	(None, 38)	19494
Total params: 13,361,206 Trainable params: 13,361,206 Non trainable params: 0		

D. Feature Extraction using MobileNet

MobileNet [34] is a model designed to be used in mobile applications, as the name mentioned. It has been implemented in the Tensorflow framework as the first mobile computer vision model. The architecture won the ILSVRC competition in 2014 with an accuracy of 92.7% [33]. MobileNet [34] uses depth-wise separable convolutions, allowing them to build lightweight deep neural networks. The depth-wise separable convolutions mean that instead of combining all three and flattening them, it conducts a single convolution on each colour channel. The input channels are filtered as a result of this. The architecture of MobileNet [34] consists of 13 depth-wise convolutional layers with a stride of one for some and two for others. It has 14 simple convolutional layers with a stride of two for the first one and one for the rest, before the fully connected layer and the output layer with the softmax activation layer (see Figure 4). This lightweight deep neural network is necessary since it requires less maintenance and operates brilliantly at high speeds. In addition, the most commonly used device in production for taking images and making inferences is a mobile phone. This model will be simple to use and will provide faster predictions. The MobileNet [34] pre-trained model has learnt a robust hierarchy of features invariant in terms of spatial, rotation, and translation compared to a simple CNN as a feature extractor. As a result, the model may be used to extract features from new images suitable for computer vision challenges.

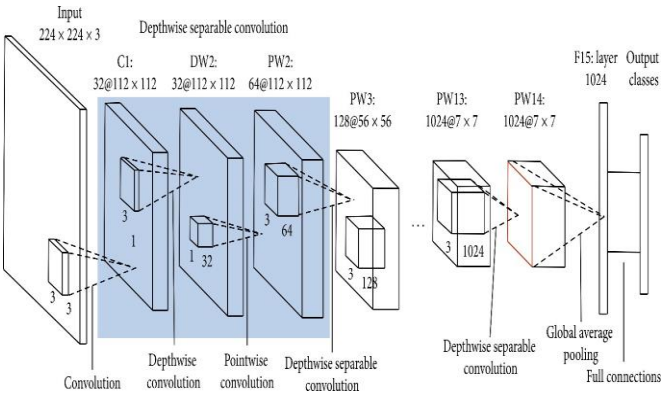

FIGURE 4: MOBILENET ARCHITECTURE

For this study, we don't need the fully connected part; we will be using our classification part to predict the output of the image. The first experiment will use the pre-trained model for feature extraction. We will freeze the feature extraction part and disallow it to update the weights during the training. As a result, we're mostly interested in using the convolution blocks in the MobileNet [34] model and then flattening the final output (from the feature maps) to feed it into our thick layers for our classifier. Before performing the classification, we fixed the feature extraction part and used an average pooling. Experiments proved that the average pooling performs better, which will also considerably reduce the number of features. It will be easier to train the extracted features on traditional Machine Learning classifiers and perform some parameter tuning to find the best model. According to the selected pre-trained model, the final feature dimension extracted will be high. However, we chose to use the MobileNet [34] network. The output feature map will be of size Nx1024. Details are given in table 3.

E. Fine Tuning:

After performing the feature extraction using the pre-trained model for comparison, we did fine-tune the weights of those models and stated the results. The weight of all the blocks, including convolutional and separable convolutional blocks, is set to True. As we pass each batch of data, the weights for these layers will likewise be adjusted by back-propagation in each epoch. In this configuration, the total number of parameters is 5,085,414, where 5,063,526 are all trainable, as described in table 3.

F. Classification using MLP:

The architecture of the classification part of the two above mentioned techniques (feature extraction and fine-tuning) is the same. To reuse the computationally demanding pre-trained network for the feature extraction, we only run the convolutional base over the supplemented dataset, record the output, and feed this data to the standalone densely connected classifier. Because the convolutional base is only performed once for each input image, this technique is quick and inexpensive to execute. The most expensive part of the process is the convolutional base. However, in the fine-tuning approach, we simultaneously used the data generator for the

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 02, February-2025

augmentation during the training, as the network was flowing down to the early layers. For this study, we started with a dense layer with 1024 nodes so that the model would capture more features. Then we added other two dense layers with 512 nodes each. All three dense layers were followed by a dropout layer with a p-value set to 0.3 (see table 2).

TABLE 3: MOBILENET AND MLP ARCHITECTURE DETAILS

LAYER TYPE	OUTPUT SHAPE	NUMBER OF PARAMS
InputLayer	(None, 224,224,3)	0
Conv2D	(None, 112,112,32)	864
BatchNormalization	(None, 112,112,32)	128
Relu	(None, 112,112,32)	0
DepthwiseConv2D	(None, 112,112,32)	288
BatchNormalization	(None, 112,112,32)	128
Relu	(None, 112,112,32)	0
Conv2D	(None, 112,112,64)	2048
BatchNormalization	(None, 112,112,64)	256
Relu	(None, 112,112,64)	0
ZeroPadding2D	(None, 113,113,64)	0
DepthwiseConv2D	(None, 56,56,64)	576
BatchNormalization	(None, 56,56,64)	256
Relu	(None, 56,56,64)	0
Conv2D	(None, 56,56,128)	8192
BatchNormalization	(None, 56,56,128)	512
Relu	(None, 56,56,128)	0
DepthwiseConv2D	(None, 56,56,128)	16384
BatchNormalization	(None, 56,56,128)	512
Relu	(None, 56,56,128)	0
Conv2D	(None, 56,56,128)	32768
BatchNormalization	(None, 56,56,128)	1024
Relu	(None, 56,56,128)	0
…	…	…
BatchNormalization	(None, 14,14,512)	2048
ReLU	(None, 14,14,512)	0
DepthwiseConv2D	(None, 14,14,512)	4608
BatchNormalization	(None, 14,14,512)	2048
ReLU	(None, 14,14,512)	0
DepthwiseConv2D	(None, 14,14,512)	4608
BatchNormalization	(None, 14,14,512)	2048
ReLU	(None, 14,14,512)	0
ZeroPadding2D	(None, 15,15,512)	0
DepthwiseConv2D	(None, 7,7,512)	4608
BatchNormalization	(None, 7,7,512)	2048
ReLU	(None, 7,7,512)	0
Conv2D	(None, 7,7,1024)	524288
BatchNormalization	(None, 7,7,1024)	4096
ReLU	(None, 7,7,1024)	0
DepthwiseConv2D	(None, 7,7,1024)	9216
BatchNormalization	(None, 7,7,1024)	4096
ReLU	(None, 7,7,1024)	0
Conv2D	(None, 7,7,1024)	1048576
BatchNormalization	(None, 7,7,1024)	4096
ReLU	(None, 7,7,1024)	0
GlobalAveragePooling2D	(None, 1024)	0
Dense	(None, 1024)	1049600
Dropout	(None, 1024)	0
Dense	(None, 512)	524800
Dropout	(None, 512)	0
Dense	(None, 512)	262656
Dropout	(None, 512)	0
Dense	(None, 38)	19494
Total params: 5,085,414 Trainable params: 1,856,550 Non trainable params: 3,228,864		

G. Classification using LR:

As mentioned before, the extracted features will be fed into an LR for classification purposes. Deep Learning makes the process easier by automating the extraction or engineering of features fed into Machine Learning models. In previous Machine Learning techniques, input data were only converted into 2D data and fed to the classifier. But with the help of Transfer Learning, we can train better-extracted features, though we get improved performance in terms of accuracy. In this case, we even have smaller computational resources. After the image pre-processing, data augmentation, and feature extraction, we came up with a feature map of Nx1024. All the sets were extracted, including train, validation, and test sets. The train and validation sets were merged to form one bigger train set in the optic to apply Cross-Validation. Cross-Validation (CV) is a way of training Machine Learning models efficiently by interchanging the validation set for each specified number of splits. This study used K-fold CV to split the merged train-val set into five small groups. We performed the training on the k-1 split and evaluated the then obtained model on the remaining one. This process will be altered until all the splits are used for validation. In this process we ensure that the biase is considerably reduced. Because we are not missing information about the which has not been used for the training. In this study we used the StratifiedKfold approach, that consists of equally distributing data for training and evaluations splits. Table 4 summarizes the process.

TABLE 4: CROSS-VALIDATION PROCESS: THE TRAINING IS PERFORMED ON 4 SPLITS AND THE VALIDATION ON THE REMAINING ONE FOR EACH LINE.

TRAINING DATA	CROSS-VALIDATION
(S1, S2, S3, S4)	S5
(S1, S2, S3, S5)	S4
(S1, S2, S4, S5)	S3
(S1, S3, S4, S5)	S2
(S2, S3, S4, S5)	S1

A summary of the methodology is described in the following figure.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 02, February-2025

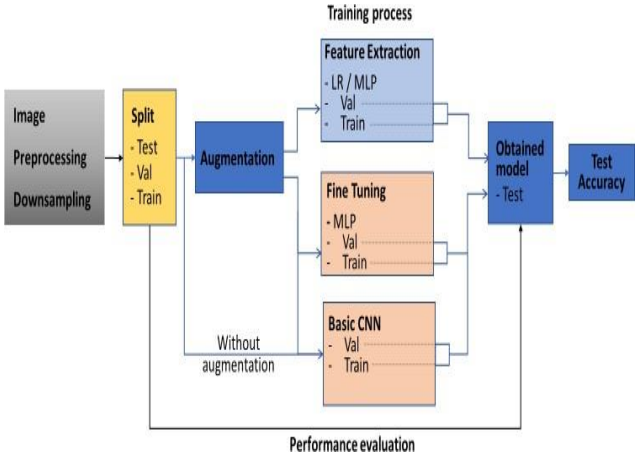

FIGURE 5: THE METHODOLOGY USED IN THIS WORK

H. Performance Evaluation

In the entire study, all the experimentation performances are evaluated using the most popular metrics of ML and DL methods. Accurately evaluation will include precision, recall, f1-score, and accuracy. The confusion matrix is a two-dimensional matrix that shows the actual and predicted classes. All of the assessment metrics are dependent on the different properties used in the confusion matrix.

1) Recall: The recall, also known as sensitivity, defines the true positive rate. The mathematical formula is given as follows:

$$\mathrm{Re} = \frac{TP}{TP + FN}$$

2) Precision: Precision is the metric used to specify the percentage of correctly predicted positives. Its mathematical formula is given as follows:

$$\mathrm{Pr} = \frac{TP}{TP + FP}$$

3) F1-Score: The F1-Score is defined as the harmonic mean of Precision and Recall. In other words, it's a statistical technique for evaluating a system's accuracy while taking Precision and Recall into account. The formula is given by:

$$F1 = 2\frac{\mathrm{Pr} \times \mathrm{Re}}{\mathrm{Pr} + \mathrm{Re}}$$

4) Accuracy: The most commonly used metric for classification tasks is accuracy, which determines what percentage of the complete test set is properly predicted positively and negatively. The mathematical formula is given by:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

In this study, we are dealing with a multilabel classification task. So, the evaluation will be done by the technique of one versus others, meaning for one case, if true is class 0, all the other classes will be set to 1.

V. EXPERIMENTAL RESULTS

A. Results of the Basic CNN

As mentioned in the methodology section, the model's parameters are 13,361,206, all trainable. That was quite quick to train with the huge help of the powerful GPU shortening the training runtime to an average of 42.625 milliseconds per step. For each epoch, there are 144 steps, given that the batch size is 32. On average, the training took 245.52 seconds. At the beginning of the training, the model was largely underfitting with a very high loss and low accuracy both on the training and validation sets due to the random initialisation. As the number of epochs grew, the accuracy started to improve. At epoch six, we had a close value of the training and validation accuracies and losses, respectively 0.7724 and 0.7674 for the accuracies and 0.7128 0.7718 for the losses. Until then, the model was converging well. But after this epoch, we started observing overfitting. Obviously, the model is not converging towards epoch 27 till the end, registering up and downs of the validation loss around 1 and 0.9 while the training loss is still dropping. The training accuracy was 1 while the validation accuracy was 83, and the training loss was 0.01 while the validation loss was 1.03. Details are observed in figure 6.

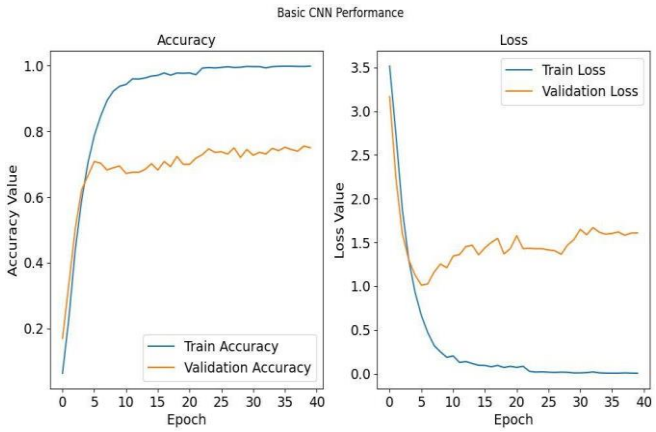

FIGURE 6: ACCURACIES AND LOSSES OF TRAINING AND VALIDATION SPLITS DURING THE TRAINING OF THE BASIC WITHOUT DATA AUGMENTATION

After the training, the freshly obtained model was evaluated with images that it had never seen on the test set. The most important metric, the test accuracy, was 77%; this value was below those registered for the training and validation (100% and 83%). This lower value indicates that our model was slightly overfitting, having fewer samples to leverage features upon (see figure 7). This figure shows that the line decreases, indicating that it's worse in the test set. This model didn't learn enough to generalise well. Having very few samples will drag the model to overlearn and miss important details. Though, a need to have more data to train on. We will see how the data augmentation can deal with overfitting and improve accuracy in the next experiment. The other performance evaluation metrics registered are precision of 78%, a recall of 77%, and an

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 02, February-2025

f1-score of 77%. Those metrics gave the same output alongside the accuracy because we have perfectly balanced data. The slightly low values indicate that the model didn't converge well. A detail of the confusion matrix is given in figure 8. We observe a misclassification of class 7 and class 9, namely (Corn_(maize)__ Cercospora_leaf_spot and Corn_(maize)___ healthy). Those two classes have high similarities patterns, and the model couldn't efficiently distinguish the differences. The lack of sufficient data to draw all the patterns caused this issue.

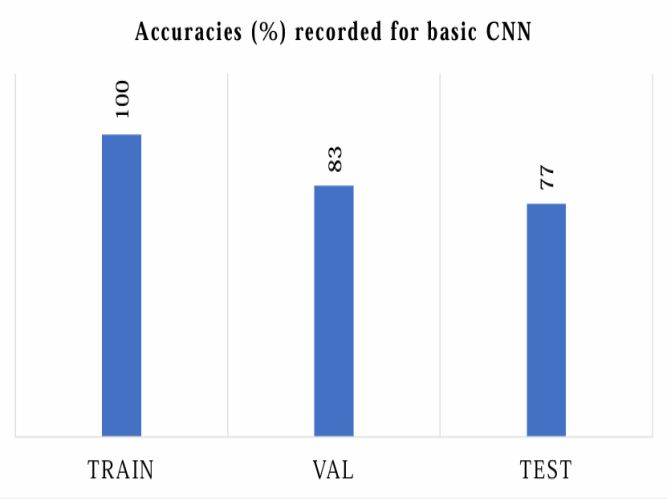

FIGURE 7: ACCURACIES OF TRAINING, VALIDATION, AND TEST SETS RECORDED FOR BASIC CNN

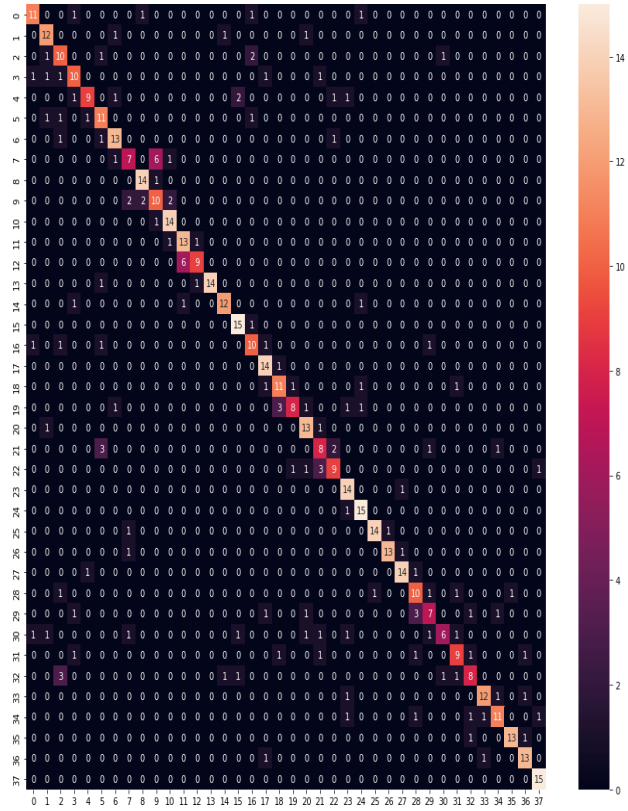

FIGURE 8: CONFUSION MATRIX OF THE BASIC MODEL ON THE TEST SET

B. Results of the Augmented Data on the Basic CNN

The augmentation was performed on the fly for this experiment. A Keras class named ImageDataGenerator was used to perform the image pre-processing and augmentation mentioned in chapter 4. Training the simple CNN using data augmentation took approximately 1173 seconds with 29.3 seconds per epoch. This training took longer than the first one because we had more data and performed the fly's data augmentation. Like the previous training, the model was underfitting at the first epoch. But this time, the accuracy was evolving elongside the number of epochs, and the loss dropped for the training and validation. Towards epoch 30, the accuracy stabilizes at around 83%. At the end of the training, we got an accuracy of 91% on the train set and 88% on the validation set, while the training loss was 0.26 and the validation loss was 0.35. We observe here that there's an improvement in the validation of 5% of accuracy, though, without overfitting, the closeness of the two plots in figure 9 explains it. The loss explains the better generalization of our model. The test accuracy of 83% indicates that the model has learned more features and can generalize better than the previous one due to having more data to train on. A char plot showing the little improvement can be found below in figure 9. In this second experiment, we also observed that the precision, recall and f1-score were close to the accuracy. The registered values are respectively: 83.2%, 82.7%, and 82.1%. The confusion matrix used to find those metrics is given in figure 11.

In this experiment, we can observe that there are not a lot of misclassifications compared to the previous one. Having enough data helped for better convergence.

C. Results of the Extracted Features on MLP:

The chosen MobileNet pre-trained model and added to our MLP network is used for this experiment. This training lasted 1285 seconds, slightly longer than the first two experiments. It lasted longer than the second one because of the feature extraction part. This network has a lot of layers, and the image will have to go through all of them, and only the predicted feature map will be fed to the MLP classifier.

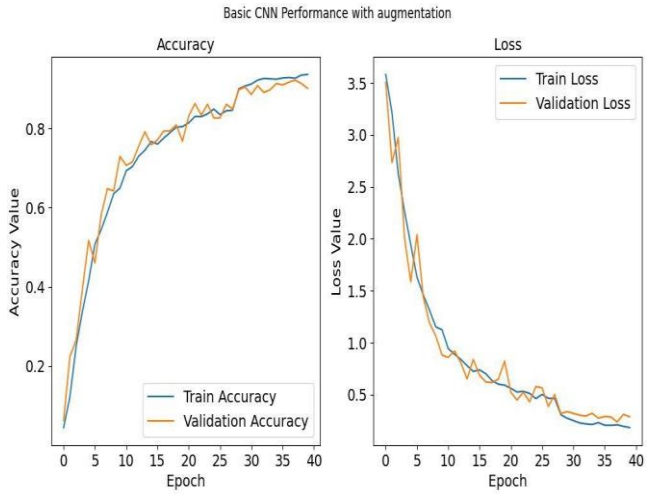

FIGURE 8: ACCURACIES AND LOSSES OF TRAINING AND VALIDATION SPLITS DURING THE TRAINING OF THE BASIC WITH DATA AUGMENTATION

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 02, February-2025

In this experiment, from the beginning of the training, we observe that it generalises well, giving 19% accuracy on the train set and 68% on a validation set; this is due to using the pre trained weights that have learnt relevant features apply them for inference. Finally, we got an overall training accuracy of 98.9% and validation accuracy of 94.3%. Hence, a big improvement is registered at +6.3% without overfitting, as shown in Figure 12.

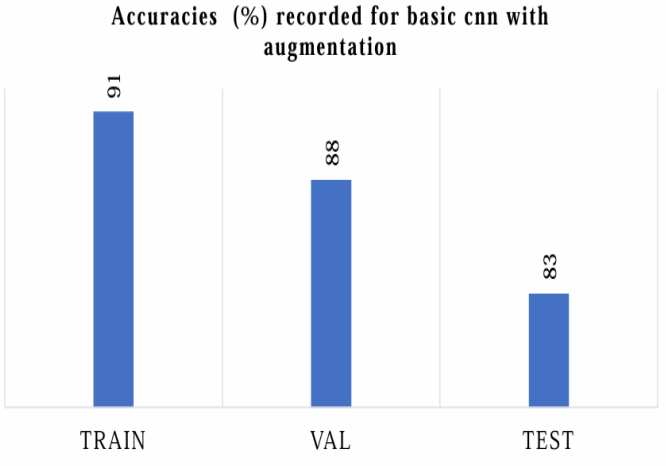

FIGURE 9: ACCURACIES OF TRAINING, VALIDATION AND TEST SETS RECORDED FOR BASIC CNN WITH DATA AUGMENTATION

As for the precision, recall, and f1-score, the registered values are 91.5%, 91.1%, and 91.1%, giving a perfect balance. The accuracy registered on the test set is 91.7%; this shows how good using pre-trained models is to train a DNN.

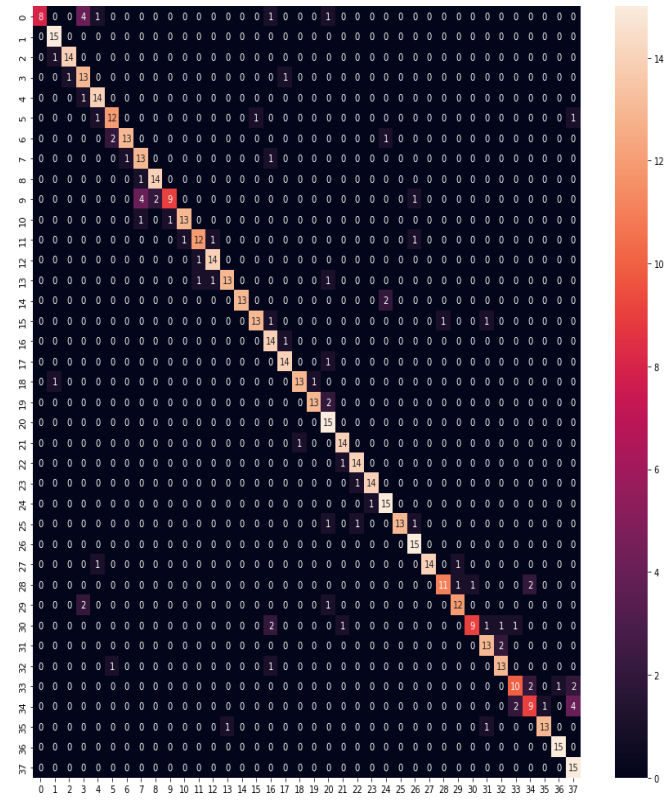

FIGURE 10: CONFUSION MATRIX OF THE BASIC MODEL TRAINED WITH DATA AUGMENTATION ON THE TEST SET

The comparison chart between the train, validation, and test accuracy confirms the argument. Therefore, we observe that there's no big gap between the accuracies, explaining the overfitting is solved. The confusion matrix plotted in figure 15 explains the obtained results. Compared to the two last models' confusion matrix, this one is better in no fatal misclassification (3 is the highest number of misclassified samples. The misclassified images belong to the same category of plants.

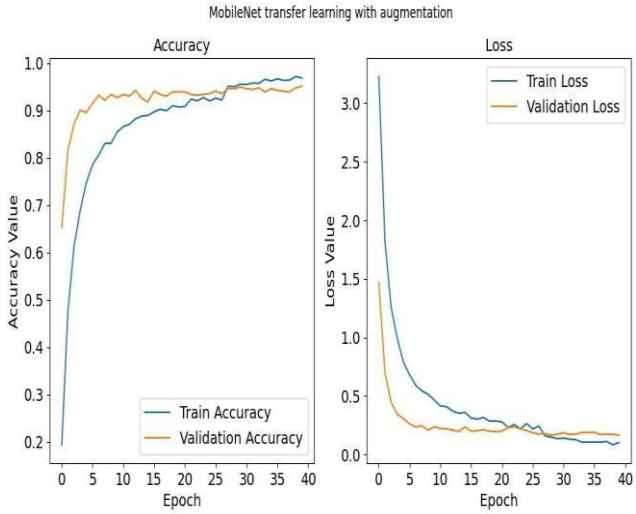

FIGURE 11: ACCURACIES AND LOSSES OF TRAINING AND VALIDATION SPLITS DURING THE TRAINING OF THE MLP ON TOP OF EXTRACTED FEATURES

The following experiment details the results obtained when training a Logistic Regression on top of extracted features. This required a bit more computational time, given that we use CV, but it gave better results in terms of accuracy than the MLP on top of extracted features.

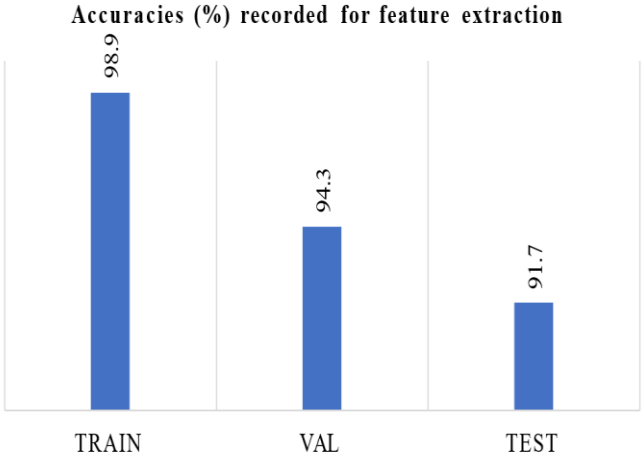

FIGURE 12: ACCURACIES OF TRAINING, VALIDATION, AND TEST SETS RECORDED FOR MLP ON TOP OF EXTRACTED FEATURES

D. Results of the Extracted Features on LR

MobileNet was then used to extract augmented data; the extraction of features for training and evaluation took 2 minutes. An extracted feature map of 23373*1024 (as a global average pooling was used on top of the pre-trained network) to train and validate the LR model. This training lasted for 15 minutes; having a high dimensional feature map and using the CV explains that it took little long to train and convergence.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 02, February-2025

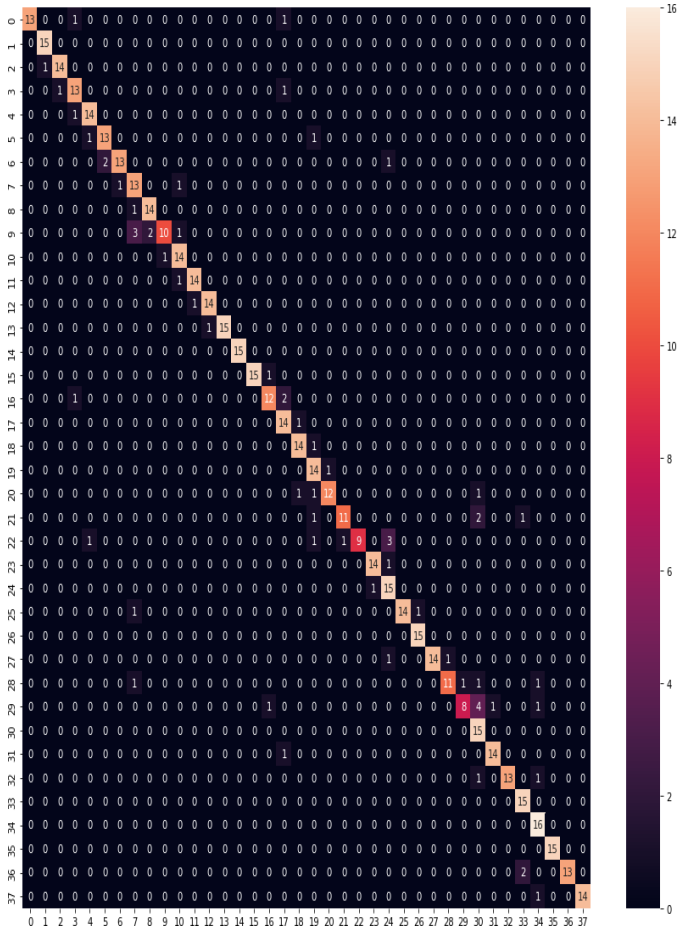

FIGURE 14: CONFUSION MATRIX ON THE TEST SET OF THE EVALUATION OF MLP TRAINED ON TOP OF EXTRACTED FEATURES FROM MOBILENET

A shuffling methodology were used for the Cross-Validation of the 5 splits. The accuracies recorded for each evaluation on the validation split are given in table 5. The lowest value is 94.75%, and the highest value is 96.5%.

TABLE 5:ACCURY SCORES OF THE EVALUATION OF MLP USING CROSS-VALIDATION

TRAINING DATA	CROSS-VALIDATION	SCORES
(S1, S2, S3, S4)	S5	94.75
(S1, S2, S3, S5)	S4	96.50
(S1, S2, S4, S5)	S3	96.30
(S1, S3, S4, S5)	S2	95.70
(S2, S3, S4, S5)	S1	96.20

The training accuracy recorded was 100%, and the average test accuracy was 96% with a standard deviation of 0.00 as we can observe on figure 14. This lower value of the standard deviation indicates that the model performed really well, and though the inference on new data can give be efficient with a high value of confidentiality. The new test accuracy is an improvement of 4.3% compared to the previous model, and it was worth trying. The other metrics used for evaluations were also recorded. The precision output was 95%, the recall gave 95.6%, and the f1-score output was 95.4%. The then confusion matrix is shown in figure 16.

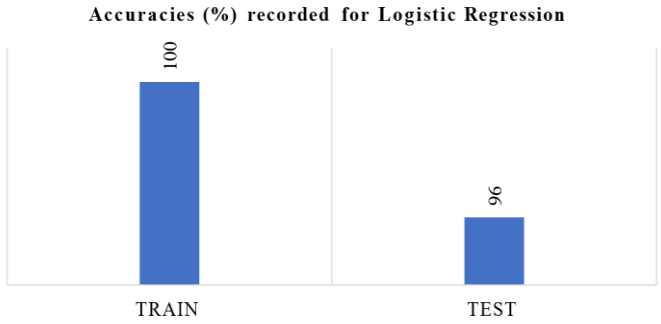

FIGURE 13: ACCURACIES OF TRAINING AND TEST SETS RECORDED FOR LR ON TOP OF MOBILENET EXTRACTED FEATURES

In the next experiment, we pushed our work to improve the results.

E. Results of Fine-Tuning

During this experiment, all layers were unfrozen for the training, allowing the optimizer to update all the weights (including those of the pre-trained network). The same MLP architecture used with the feature extractor is used here. The training lasted for 1100 seconds, a bit faster than the third experiment. This one is a direct improvement of using Transfer Learning for feature extraction.

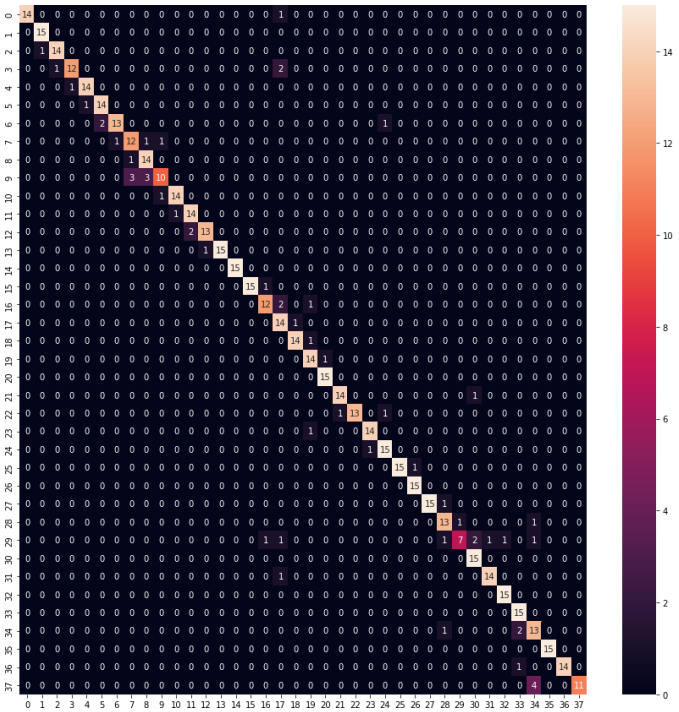

FIGURE 15: CONFUSION MATRIX ON THE TEST SET OF THE EVALUATION OF LR TRAINED ON TOP OF EXTRACTED FEATURES FROM MOBILENET.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 02, February-2025

Since the beginning of the training, the model started to generalise well, and in the end, the training accuracy was 100% and the validation accuracy 99%. An improvement of 4.7% is recorded on the validation accuracy. The loss also dropped to 0.001 for the training set and 0.18 for the validation loss, as shown in figure 17.

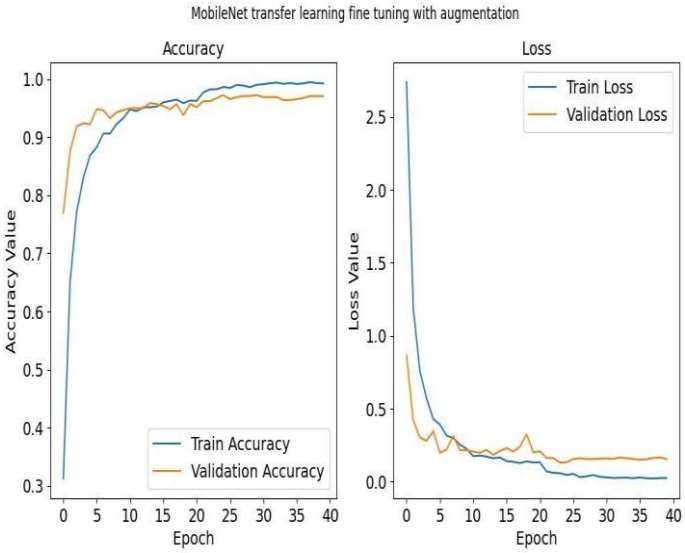

FIGURE 16: ACCURACIES AND LOSSES OF TRAINING AND VALIDATION SPLITS DURING THE TRAINING OF THE MLP WITH FINE-TUNING MOBILENET

The test accuracy explains that we didn't observe any overfitting recorded 98%, though an improvement of 6.3% from the feature extraction using MLP experiment and +1% from the one using LR.

The last and best experiment registered 98% on the validation set and 98% on the test set, confirming that the model has learnt all the relevant features and is ready to give better inference in new unseen data (see figure 18). Other metrics include precision registered at 96%, recall recorded at 95.8%, and f1-score output at 97%.

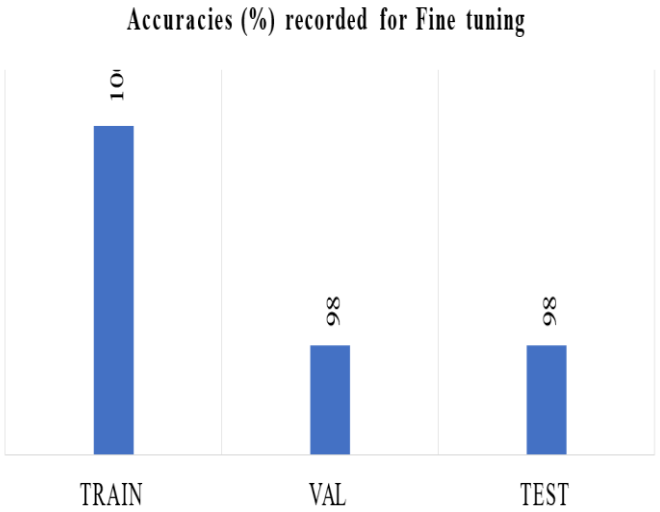

FIGURE 17: ACCURACIES OF TRAINING, VALIDATION AND TEST SETS RECORDED FOR MLP WITH FINE-TUNING MOBILENET

The confusion matrix is shown in figure 19 below. We observe a model that almost perfectly performed on the test set in the confusion matrix, with fewer misclassified samples.

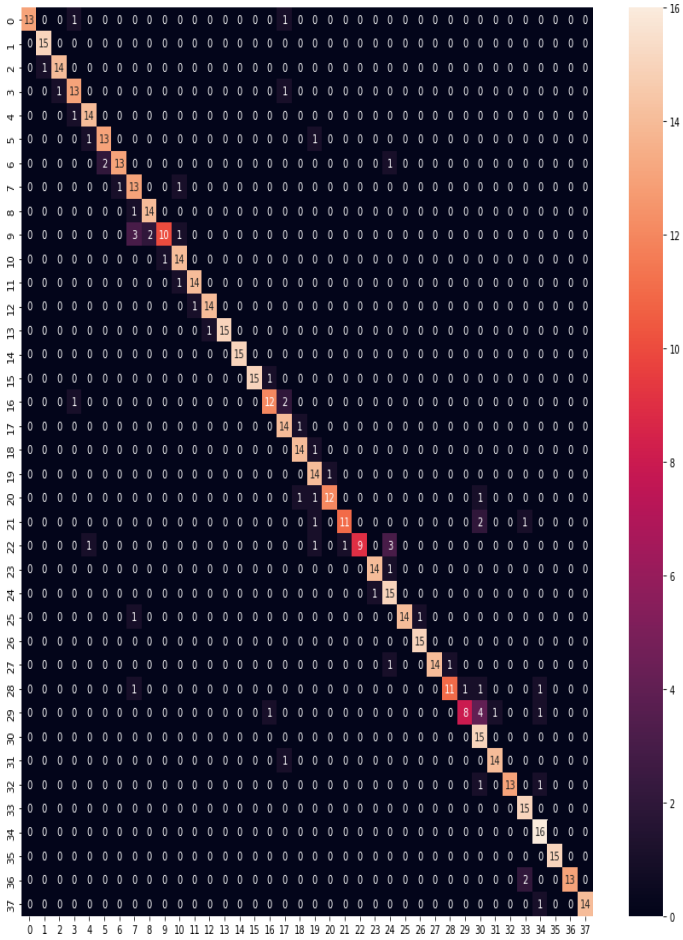

FIGURE 18: CONFUSION MATRIX ON THE TEST SET OF THE EVALUATION OF MLP WITH FINE-TUNING MOBILENET.

VI. SUMMARY

The study focus on identifying diseases in plants through feature extraction, identification, and classification processes by employing the Transfer Learning approach. With the use of this approach in conjunction with data augmentation, several experiments were performed from which the best result for classifying plant leaf diseases was found. Overall, five experiments were performed; we recorded some parameters for each model for comparison purposes. Those parameters are loss, accuracy, precision, recall, f1-score, and runtime. The table 6 below shows the results obtained from the experiments. The corresponding values are the performance evaluation on the test set. Looking at the table, we can definitely see that we are getting interesting results as we are doing experiments. Each new model outperforms the preceding one, which is to be expected given that we used more advanced approaches with each new model. The worst performance is gotten by the simple CNN model, which has accuracy and an f1 score of around 77%. In contrast, our best model is our fine-tuned model with Transfer Learning and image augmenting, which has a model accuracy and F1-score of 98%, which is incredible considering we trained on a resampled dataset.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 02, February-2025

TABLE 6: SUMMARY OF ACCURACY, PRECISION, RECALL, F1-SCORE AND RUNTIME OF ALL THE TRAINED MODELS

MODEL	DATA	ACC (%)	Pr (%	RE (%)	F1	Runtime (min)
Basic CNN	Original	77	78	77	77	4.1
Basic CNN	Augmented	83	83.2	82.7	82.1	19.5
Feature Extraction with MLP	Augmented	91.7	91.5	91.1	91	21.4
Feature Extraction with LR	Augmented	96	95	95.6	95.4	15
Fine Tuning	Augmented	98	96	95.8	97	18.3

Figure 21 below shows a visualization of the fine-tuned model's performance. This figure shows that all 36 except one sample are correctly predicted with very high confidentiality. The lowest observed is 57% confidentiality. This is just an approval that this study is suitable and can directly be used in production. A website or mobile application can deploy the present trained model to make inferences in new images; those samples should fall into the 14 categories of plants and 38 classes of diseased and healthy plants that the model is trained on.

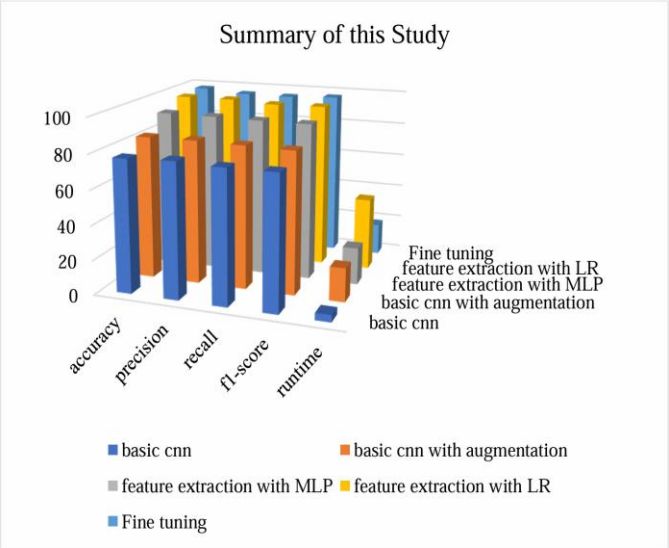

FIGURE 20: FIGURE SUMMARIZING THE RESULTS OBTAINED IN THIS STUDY

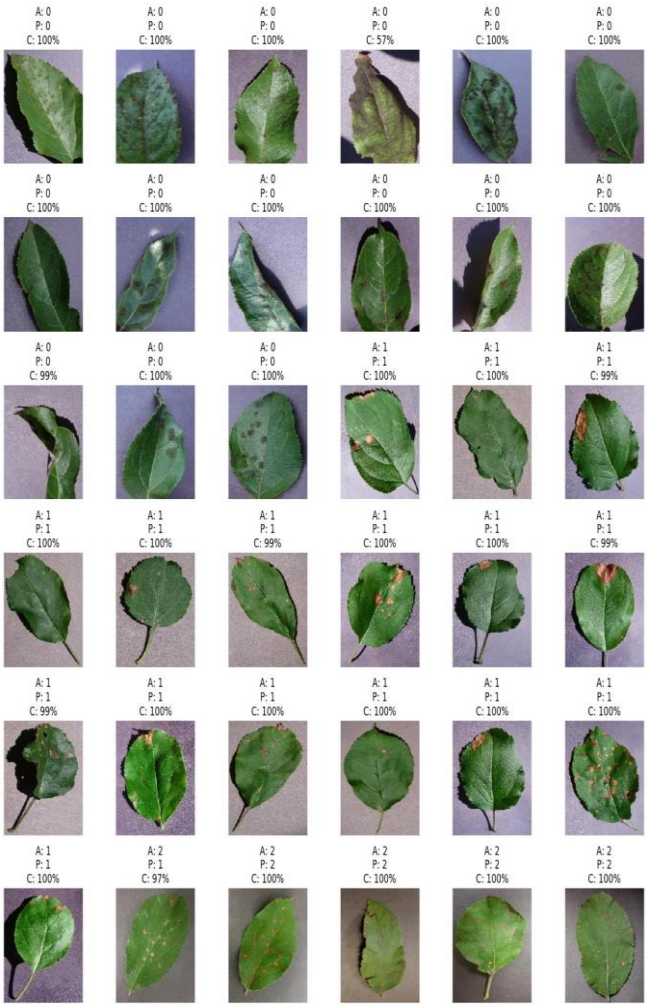

FIGURE 19: PREDICTION ON 36 SAMPLES FROM THE TEST SET

The advantage of this present work is that it is lightweight and faster in inference. The farmer or the user will see the result as soon as he scans the image.

VII. CONCLUSION:

This experiment showed that performance improved when augmentation and Transfer Learning were incorporated. Only 77% accuracy was achieved when the Basic Convolutional Neural Network was trained from scratch. After doing data augmentation, this climbed by up to 83%. The results increased further after using Transfer Learning with the MobileNet pre trained model. A comparison was made between utilizing Logistic Regression and Multilayer Perceptron to classify extracted features from the pre-trained network. It was discovered that the typical Machine Learning classifier was more accurate. MobileNet's fine-tuning had the best performance, with a 98% accuracy level. The next chapter will discuss the conclusion, limitations, and future works

VIII. DISCUSSION:

This study's findings were compared to five earlier results obtained using cutting-edge plant recognition techniques such as Machine Learning and Deep Learning. Even though the circumstances may vary, from the dataset used to the preprocessing stages, the first comparison

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 02, February-2025

Learning approaches offer high recognition accuracy levels. In several trials, Transfer Learning provided high accuracy due to the usage of all available samples in the datasets used, such as in [35]. The F1-score of 94% mentioned in the paper is lower than our results, which is 97%, meaning that our model is more stable.

- Contributions:

This study looked at how to use Transfer Learning to classify plant leaf diseases accurately. The goal was to show how crucial Transfer Learning can be for image classification when just a tiny quantity of data is available. The study also emphasizes the importance of Deep Learning in terms of accuracy computation time compared to current methods. The leaf image was subjected to Deep Learning, which yielded excellent results. This was accomplished by fine-tuning and experimenting with using data augmentation. On the MobileNet pre-trained model, data augmentation was employed with Transfer Learning for feature extraction. Then a typical Machine Learning classifier was trained and assessed on top of the extracted deep features. Though the experiment was worthwhile, the Logistic Regression did better than a simple Multilayer Perceptron network in terms of accuracy. This method of first augmenting the data was to compare the two classifiers fairly and ostensibly save computational resources.

- Limitations and Future Works:

There could be much improvement in the results using appropriate Multilayer Perceptron on top of extracted features. Instead of finding the best suitable network for classification in a hand-crafted manner, an automation algorithm will be assessed in the future. Evolutionary Algorithms like the Genetic Algorithm could be a good asset to automatically find the combination of parameters (number of layers, number of nodes, activation function) that will improve the tradeoff accuracy and computation resources. State-of-the-art models' results show that we can achieve better results with further image preprocessing steps. Combining this step with fine-tuning pre-trained models using Evolutionary computing to retrain the weights could achieve the best-expected results with few image samples.

IX. REFERENCES:

[1] M. Berners-Lee et al., "Current global food production is sufficient to meet human nutritional needs in 2050 provided there is radical societal adaptation," Elem. Sci. Anthr., vol. 6, 2018.

[2] C. Le Mouël and A. Forslund, "How can we feed the world in 2050? A review of the responses from global scenario studies," European Review of Agricultural Economics, vol. 44, no. 4. Oxford University Press, pp. 541–591, Sep. 01, 2017. doi: 10.1093/erae/jbx006.

[3] H. B. D. Sorensen et al., "Comparison of automated methods for REM sleep without atonia detection," J. Sleep Res., vol. 27, no. Supplement 1, p. 138, 2018.

[4] D. P. Hughes and M. Salathé, "An open access repository of images on plant health to enable the development mobile disease diagnostics." [Online]. Available: http://www.fao.org/fileadmin/templates/wsfs/docs/expert_paper/How_to_Feed_the_World_in_2050.pdf! !!

[5] T. S. Schubert, L. L. Breman, and S. E. Walker, "Basic Concepts of Plant Disease and How to Collect a Sample for Disease Diagnosis," Plant Pathol. Circ., vol. 1988, no. 307, pp. 1–8, 1999.

[6] R. K. Horst, "Host Plants and Their Diseases." in Host Plants and Their Diseases. In: Westcott's Plant Disease Handbook. Springer, Boston, MA., pp. 516–517, 2013. doi: 10.1007/978-94-007-6076-9_2.

[7] P. J. Freire, D. Abode, J. E. Prilepsky, and S. K. Turitsyn, "Power and modulation format transfer learning for neural network equalizers in coherent optical transmission systems," Opt. InfoBase Conf. Pap., vol. 39, no. 21, pp. 6733–6745, 2021, doi: 10.1364/sppcom.2021.spm5c.6.

[8] K. Dehnen-Schmutz, G. L. Foster, L. Owen, and S. Persello, "Exploring the role of smartphone technology for citizen science in agriculture," Agron. Sustain. Dev., vol. 36, no. 2, Jun. 2016, doi: 10.1007/s13593 016-0359-9.

[9] A. M. Abdu, M. M. Mokji, and U. U. Sheikh, "Machine learning for plant disease detection: An investigative comparison between support vector machine and deep learning," IAES Int. J. Artif. Intell., vol. 9, no. 4, pp. 670–683, Dec. 2020, doi: 10.11591/ijai.v9.i4.pp670-683.

[10] N. K. Trivedi et al., "Early detection and classification of tomato leaf disease using high-performance deep neural network," Sensors, vol. 21, no. 23, 2021, doi: 10.3390/s21237987.

[11] A. Camargo and J. S. Smith, "An image-processing based algorithm to automatically identify plant disease visual symptoms," Biosyst. Eng., 10.1016/j.biosystemseng.2008.09.030. vol. 102, no. 1, pp. 9–21, 2009, doi:

[12] S. Sakhamuri and V. S. Kompalli, "An Overview on Prediction of Plant Leaves Disease using Image Processing Techniques," IOP Conf. Ser. Mater. Sci. Eng., vol. 981, no. 2, pp. 10–16, 2020, doi: 10.1088/1757-899X/981/2/022024.

[13] P. Schreinemachers, E. B. Simmons, and M. C. S. Wopereis, "Tapping the economic and nutritional power of vegetables," Glob. Food Sec., vol. 16, no. September, pp. 36–45, 2018, doi: 10.1016/j.gfs.2017.09.005.

[14] R. Wang, M. Lammers, Y. Tikunov, A. G. Bovy, G. C. Angenent, and R. A. de Maagd, "The rin, nor and Cnr spontaneous mutations inhibit tomato fruit ripening in additive and epistatic manners," Plant Sci., vol. 294, no. January, p. 110436, 2020, doi: 10.1016/j.plantsci.2020.110436.

[15] M. Chanda and M. Biswas, "Plant disease identification and classification using back-propagation neural network with particle swarm optimization," in Proceedings of the International Conference on Trends in Electronics and Informatics, ICOEI 2019, 2019, vol. 2019-April, no. Icoei, pp. 1029–1036. doi: 10.1109/icoei.2019.8862552.

[16] S. P. Mohanty, D. P. Hughes, and M. Salathé, "Using deep learning for image-based plant disease detection," Front. Plant Sci., vol. 7, no. September, pp. 1–10, 2016, doi: 10.3389/fpls.2016.01419.

[17] S. H. Lee, H. Goëau, P. Bonnet, and A. Joly, "Attention-Based Recurrent Neural Network for Plant Disease Classification," Front. Plant Sci., vol. 11, no. December, pp. 1–8, 2020, doi: 10.3389/fpls.2020.601250.

[18] M. G. Selvaraj et al., "AI-powered banana diseases and pest detection," Plant Methods, vol. 15, no. 1, pp. 1–11, 2019, doi: 10.1186/s13007-019-0475-z.

[19] M. Kumar, P. Gupta, P. Madhav, and Sachin, "Disease detection in coffee plants using convolutional neural network," in Proceedings of the 5th International Conference on Communication and Electronics Systems, ICCES 2020, 2020, no. Icces, pp. 755–760. doi: 10.1109/ICCES48766.2020.09138000.

[20] B. Liu, Z. Ding, L. Tian, D. He, S. Li, and H. Wang, "Grape Leaf Disease Identification Using Improved Deep Convolutional Neural Networks," Front. Plant Sci., vol. 11, no. July, pp. 1–14, 2020, doi: 10.3389/fpls.2020.01082.

[21] A. Ramcharan, K. Baranowski, P. McCloskey, B. Ahmed, J. Legg, and D. P. Hughes, "Deep learning for image-based cassava disease detection," Front. Plant Sci., vol. 8, no. October, pp. 1–7, 2017, doi: 10.3389/fpls.2017.01852.

[22] A. Fuentes, S. Yoon, S. C. Kim, and D. S. Park, "A robust deep-learning-based detector for real-time tomato plant diseases and pests recognition," Sensors (Switzerland), vol. 17, no. 9, 2017, doi: 10.3390/s17092022.

[23] S. Ishak, "Jurnal Teknologi LEAF DISEASE CLASSIFICATION ARTIFICIAL NEURAL NETWORK," J. Teknol., vol. 17, pp. 109–114, 2015.

[24] P. Pandey, V. Irulappan, M. V. Bagavathiannan, and M. Senthil-Kumar, "Impact of combined abiotic and biotic stresses on plant growth and avenues for crop improvement by exploiting physio-morphological traits," Front. Plant Sci., vol. 8, no. April, pp. 1–15, 2017, doi: 10.3389/fpls.2017.00537.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 02, February-2025

[25] Z. Luo, Q. Li, and J. Zheng, “A study of adversarial attacks and detection on deep learning-based plant disease identification,” Appl. Sci., vol. 11, no. 4, pp. 1–16, 2021, doi: 10.3390/app11041878.

[26] J. Chen, J. Chen, D. Zhang, Y. Sun, and Y. A. Nanehkaran, “Using deep transfer learning for image-based plant disease identification,” Comput. Electron. Agric., vol. 173, no. March, p. 105393, 2020, doi: 10.1016/j.compag.2020.105393.

[27] K. P. Ferentinos, “Deep learning models for plant disease detection and diagnosis,” Comput. Electron. Agric., vol. 145, no. September 2017, pp. 311–318, 2018, doi: 10.1016/j.compag.2018.01.009.

[28] N. Q. K. Le, “Fertility-GRU: Identifying Fertility-Related Proteins by Incorporating Deep-Gated Recurrent Units and Original Position-Specific Scoring Matrix Profiles,” J. Proteome Res., vol. 18, no. 9, pp. 3503–3511, 2019, doi: 10.1021/acs.jproteome.9b00411.

[29] N. Q. K. Le, D. T. Do, T. N. K. Hung, L. H. T. Lam, T. T. Huynh, and N. T. K. Nguyen, “A computational framework based on ensemble deep neural networks for essential genes identification,” Int. J. Mol. Sci., vol. 21, no. 23, pp. 1–16, 2020, doi: 10.3390/ijms21239070.

[30] E. C. Too, L. Yujian, S. Njuki, and L. Yingchun, “A comparative study of fine-tuning deep learning models for plant disease identification,” Comput. Electron. Agric., vol. 161, no. October 2017, pp. 272–279, 2019, doi: 10.1016/j.compag.2018.03.032.

[31] M. Arsenovic, M. Karanovic, S. Sladojevic, A. Anderla, and D. Stefanovic, “Solving current limitations of deep learning based approaches for plant disease detection,” Symmetry (Basel)., vol. 11, no. 7, 2019, doi: 10.3390/sym11070939.

[32] I. Goodfellow et al., “Generative adversarial networks,” Commun. ACM, vol. 63, no. 11, pp. 139–144, 2020, doi: 10.1145/3422622.

[33] A. Khan, A. Sohail, U. Zahoora, and A. S. Qureshi, “A survey of the recent architectures of deep convolutional neural networks,” Artif. Intell. Rev., vol. 53, no. 8, pp. 5455–5516, Dec. 2020, doi: 10.1007/s10462-020-09825-6.

[34] A. G. Howard and W. Wang, “MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications,” 2012.

[35] T. Aboneh, A. Rorissa, R. Srinivasagan, and A. Gemechu, “Computer Vision Framework for Wheat Disease Identification and Classification Using Jetson GPU Infrastructure,” Technologies, vol. 9, no. 3, p. 47, 2021, doi: 10.3390/technologies9030047.