Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Plant Disease Management: A Fine-Tuned Enhanced CNN Approach with Mobile App Integration for Early Detection and Classification

Mudassir Iftikhar
Sindh Madressstual Islam university
Irfan Ali Kandhro
Sindh Madressstual Islam university
Neha Kausar
Sindh Madressstual Islam university
Asadullah Kehar
Institute of Computer Science, Shah Abdul Latif University, Khairpur
Mueen Uddin
University of Doha for Science and Technology UDST
Abdulhalim Dandoush
abdulhalim.dandoush@udst.edu.qa
University of Doha for Science and Technology UDST

Research Article

Keywords: Convolutional neural network, Plant Disease Detection, ANN

Posted Date: December 22nd, 2023

DOI: https://doi.org/10.21203/rs.3.rs-3782911/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

Version of Record: A version of this preprint was published at Artificial Intelligence Review on June 6th, 2024. See the published version at https://doi.org/10.1007/s10462-024-10809-z.

Plant Disease Management: A Fine-Tuned Enhanced CNN Approach with Mobile App Integration for Early Detection and Classification

Mudassir Iftikhar[1], Irfan Ali Kandhro[1], Neha Kausar[1], Asadullah Kehar[3], Mueen Uddin[2], Abdulhalim Dandoush[2,4]
[1] Department of CS, Sindh Madressstual Islam university, Pakistan
[2] CCIT college, University of Doha for Science and Technology UDST, Qatar
[3] Institute of Computer Science, Shah Abdul Latif University, Khairpur
[4] Corresponding Author: abdulhalim.dandoush@udst.edu.qa

Note: This work is supported by the QNL, Qatar.

Abstract

Farmers face a daunting challenge in meeting the escalating demands of a rapidly growing population for agricultural products, while plant diseases continue to exact a devastating toll on food production. Despite investing billions of dollars in disease management, agriculturists often struggle to achieve effective disease control without the support of advanced technology. The article explores a deep learning-based approach for disease detection. Specifically, it employs a Convolutional Neural Network (CNN) architecture for the detection. For the automated detection of plant disease, using plant images. This paper presents a new model for the early detection of plant detection based on processing plant images. And compare the in-depth performance analysis of hyper parameters in the context of plant disease detection by focusing on three distinct crops: (Apple, Corn, and Potato). Moreover, the data augmentation impact is analyzed. To enhance accessibility for farmers, our model is seamlessly integrated with a mobile application. The experimental results show the efficiency of our fine-tuned enhanced CNN model (E-CNN) achieving 98.17% accuracy on fungal classes. This research endeavors to pave the way for more effective plant disease management and ultimately to improve agricultural productivity in the face of mounting global challenges.

Keywords Convolutional neural network · Plant Disease Detection · ANN · Machine Learning

Abbreviations

CNN	Convolutional Neural Network
E-CNN	Enhanced Convolutional Neural Network
KNN	k-Nearest Neighbors
SVM	Support Vector Machine
ANN	Artificial Neural Network
SVM	Support Vector Machine
ML	Machine Learning
DL	Deep Learning
TF Lite	Tensor Flow Lite.

1 Introduction

Plant diseases are now a major worldwide issue, causing food shortages in numerous nations. Identifying these diseases accurately is essential to increasing agricultural yields. Convolutional Neural Networks (CNNs) have become increasingly popular in today's artificial intelligence era as efficient instruments for accurate disease identification through picture analysis. As a result, CNNs have become a prevalent choice for the early detection of crop diseases, leading to improved accuracy [1], [2], [3] and the expansion of technology's role in precision agriculture. Detecting diseases in maize leaves typically requires continuous crop monitoring According to professionals, these methods can come with high expenses, consume a significant amount of time, and inconsistencies. To address these challenges, CNNs are employed to automate the detection and classification of maize leaf diseases swiftly and accurately [4]. Numerous researchers have embarked on developing predictive models capable of determining whether maize leaves will succumb to disease or remain healthy [5]. The Figure 1 likely shows a visual representation that distinguishes between plant diseases caused by infections (such as pathogens like bacteria, viruses, fungi, or pests) and diseases that result from non-infectious factors (such as nutrient deficiencies, environmental stress, or genetic factors).

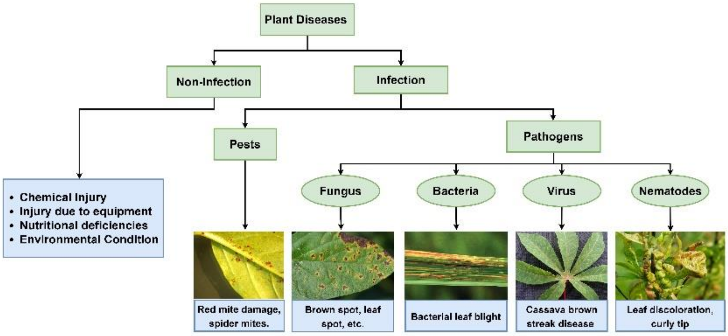

Figure 1. Types of Plant Diseases

This research focuses on CNNs and explores hyper parameter optimization to address farmers' concerns regarding disease detection and classification. The study concentrates on different kind of diseases and makes four key contributions:

(i) Exploration of pre-trained Convolutional Neural Networks (CNNs) alongside a well-tuned set of hyper parameters, resulting in the development of a suite of classifiers.

(ii) Machine learning models are employed for plant disease detection.

(iii) Optimizing CNN hyper parameters, like learning rate and architecture, improves performance using methods such as grid search and proper dataset splitting.

(iv) Integration of Best model in Flutter mobile Application.

As the global population is projected to reach 10 billion in the next 25 years, enhancing agricultural systems is imperative. Leveraging artificial intelligence, particularly CNNs, can help farmers increase food production, mitigating the looming threat of food scarcity, a potential leading cause of mortality.

There are still many problems in the previous work such as, K. Sathya [6] proposed their model RDA-CNN, inspired by CNN architecture and their accuracy is 96.59% while they did not test their model by integrating their model in mobile application. Md. Tariqul Islam [7] their proposed model has 94.29% accuracy while they also not created any system to test their model. Shradha Verma [8] presented the CNN model by tuning hyperparameters but their results are not that accurate also they did not create any system to test their model.

We will propose a model which will be more optimized, fast, and accurate than previous ones. We will experiment with different hyperparameters of CNN architecture. You can see the basic architecture of the CNN model in fig. 2. We will increase the number of layers and will use the best performer hyperparameters such as optimizer, activation function, dropout, learning rate, epoch, batch size and many more. It will be called Enhanced Convolutional Neural Network (E-CNN). We will compare this model with pretrained models, and machine learning models. And in the end, we will convert our trained model into TF Lite file then we will integrate our model in mobile based application.

The discoveries in this paper contribute valuable support for plant disease detection and classification. The subsequent sections are organized as follows: Section 2 presents relevant previous studies. The formulation of the problem and the fundamental idea underlying the proposed method are detailed in Section 3. Section 4 provides an overview of the experimental results, and the paper concludes in the conclusions section, summarizing key insights and findings.

2 Related Works

Numerous plant diseases and research papers addressing different ailments were previously examined. In our research, we concentrated on Potato, Apple, and Corn diseases, specifically investigating Healthy Potato leaves, Early Blight Potato leaves, and Late Blight potato leaves, and diseases of apple leaves and corn. Convolutional Neural Networks (CNNs) were harnessed as a tool for accurate disease detection. In this context, we explored various hyperparameters such as optimizers, activation functions, and dropout rates, seeking optimal configurations that would maximize accuracy [9]. It is worth noting that previous research primarily employed Grid Search for hyperparameter optimization, which is an exhaustive search method. Our literature review revealed a wealth of research papers dedicated to plant disease detection using various techniques [10], [11], [12], [13] This section of our paper underscored the significance of these valuable studies within the context of CNN models. The subsequent section provided an overview of different CNN architectures applied to plant disease diagnosis. CNNs, typically adopting the Lenet-5 architecture proposed by Yann LeCun in 1998 [14], served as the common architecture. Convolution layers were frequently employed for feature extraction, followed by pooling techniques like max, min, or average pooling [9]. The final layer comprised a fully connected layer for disease classification. This study employed CNN-based deep learning techniques for plant disease identification. The process typically involved dataset collection, which included high-quality images of both healthy and diseased leaves, which were later split into training (80%), validation (10%), and testing (10%) sets. The study also featured a discussion on an innovative model for leaf prediction and identification, as well as the utilization of AlexNet and other deep learning models for forecasting leaf diseases [15]. These models were optimized using the NAG Algorithm and demonstrated impressive accuracy in identifying disease characteristics from images. Furthermore, we examined research that applied CNNs for precise disease identification, including pre-processing techniques and optimizers [16]. The study also delved into image enhancement methods, feature extraction, and various classification methods like, FUZZY, ANN and SVM [17]. In contemporary image analysis research, the selection of image datasets significantly influenced research outcomes. Colored images offered a rich visual representation but posed computational challenges due to their size and complexity. Grayscale images were more computationally efficient but lacked color-related information. Segmented images were useful for precise object localization but required labor-intensive annotation [18]. Researchers had to carefully consider their dataset choice in line with their study objectives, available resources, and the balance between data

richness and complexity. Table 1 depicts the dataset used, the used ML/DL model, and the detection use-cases for recent studies in the field. In particular, Durmus et el. [19] used method of deep learning to detect disease in tomato leaves. In this method two different deep learning models are used: (i) AlexNet and (ii) SqueezeNet. The training was done with the help of ten different classes of apple, potato and corn and the accuracy was 95.65% and 94.30 % for AlexNet and SqueezeNet respectively. In [15], the authors applied the detection on Apple leaves using CNN's pre-trained model which is AlexNet. The model is trained to recognize four major diseases of apple leaves and the model achieved up to 97.62% of accuracy. Another study by DeChant et al. [20] conducted automatic identification of Northern leaf blight of Maize leaf through computation pipeline of CNN that shows the challenging part of the limited data. The system got accuracy up to 96.7%. Furthermore, AlexNet and GoogLeNet architectures for deep learning were used by Mohanty et al. [24] to produce models for categorizing tomato leaf diseases. By combining learning methods and different training and testing splits, their system got accuracy of 99.35% by using PlantVillage [21] dataset.
Ahmed et el. [22] used four different Pre-trained CNN network VGG-16, ResNet50, Inception V3 and VGG-19, for identification and classification of tomato leaf disease where the model got 93% accuracy. The next study is Rumpf et al. in 2010 when he discussed about the differences between healthy and un-healthy leaves of sugar beet, he trained his model in such a way that without disease particular symptoms visibility on leaves the model identify the disease by using SVM and their model got the accuracy of 97%.
In [23], Nachtigall et el. used CNN in detection and grouping of leaf disease also nutritional deficiencies and herbicides damage on apple leaves their system got the accuracy of 97.3% and the dataset was up to 2539 images. Recently, Swapnil Dadabhau Daphal et. al. [24] collected their own dataset of up to 2569 images with five different categories. The author used well-known deep learning technique MobileNet-V2 proposing the system for better generalization and the system got accuracy of 84%. The primary contribution of this study lies in the comprehensive analysis of hyperparameters in the context of plant disease detection, emphasizing three specific classes while demonstrating the effectiveness of the fine-tuned model. Our study provides valuable insights by evaluating the different CNN-based networks used in the field, offering a foundation for future research in plant disease detection.

Table 1 List of previously implemented deep-learning techniques:

Year	Author	Datasets	Method	Research Areas
2016	J. G. A. Barbedo [25]	Own dataset	Digital Image Processing	**Multiple leaves disease**
2016	Iqbaldeep Kaur et al. [26]	Own dataset	SVM and Ant Colony Algorithm	**Multiple leaves disease**
2016	Mohanty et al. [27]	PlantVillage	AlexNet and GoogLeNet	**Multiple plant leaves**
2016	Nachtigall et al. [23]	PlantVillage	AlexNet	**Apple leaf diseases**
2016	Pranjali B. Padol et al. [28]	Own dataset	SVM	**Grape leaf diseases**
2017	Durmus et al. [19]	PlantVillage	AlexNet and SqeezeNet	**Tomato leaf diseases**
2017	DeChant et al. [20]	PlantVillage	Layers of CNN architecture	**Maize leaf diseases**
2017	Lu et al. [29]	PlantVillage	Multistage CNN	**Rice leaf diseases**
2018	Liu et al. [15]	PlantVillage	AlexNet with inception layer	**Apple leaf diseases**

2018	Shradha Verma [30]	PlantVillage	Multiple techniques	**Tomato leaves diseases**
2019	Adedoja et al. [31]	PlantVillage	NASNet	**Multiple leaves diseases**
2019	Ozguven et al. [32]	Own dataset	Faster R-CNN	**Beet leaf diseases**
2019	Gensheng et al. [33]	Own dataset	Modified Cifer10	**Tea Leaf diseases**
2020	Li et al. [34]	PlantVillage	Shallow CNN with SVM, RF	**Maize leaf diseases**
2020	Chen et al. [35]	PlantVillage	INC-VGGN	**Rice and Maize leaves diseases**
2020	Ahmed et al. [22]	PlantVillage	VGG16, VGG19, ResNet, InceptionV3	**Tomato leaf disease**
2021	Atila et al. [36]	PlantVillage	EfficientNet	**Multiple leaves disease**
2021	Tuncer et al. [37]	PlantVillage	Hybrid CNN	**Multiple leaves disease**
2022	R. S. Sandhya Devi [38]	PlantVillage	EfficientNet V2, Inception V3	**Cassava and PlantVillage-mix-up leaves disease**
2023	Swapnil Dadabhau Daphal [24]	Own dataset	MobileNet V2	**Sugarcane leaf diseases**

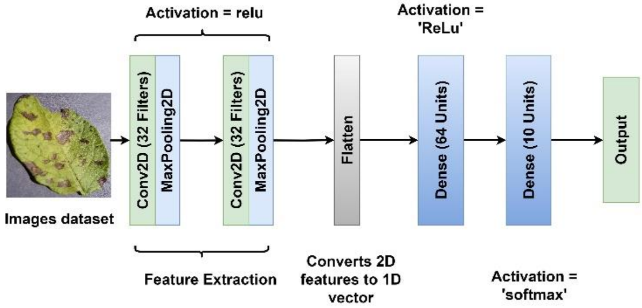

Figure 2. Architecture of Basic CNN Model

3 Methodology

In this section, the architecture of the CNN model and our proposed model E-CNN for detecting and classifying plant diseases is elucidated comprehensively.

3.1 The Architecture of CNN

Convolutional Neural Networks (CNNs) stand as a fundamental framework in deep learning. They are designed to process images by determining the significance of various elements within the image through adjustable biases and weights. This process enables CNNs to distinguish between different visual aspects. The initial design of a CNN involves the manual creation of filters in its training phase, which are then refined to recognize various features in the training data.

The structural design of CNNs closely mirrors the neuronal connectivity patterns observed in the human brain, particularly drawing inspiration from the organization of the Visual Cortex. In this biological parallel, individual neuron are responsive to specific stimuli present within a constrained zone of the visual field, known as the receptive field. To comprehensively analyze the entire visual area, these receptive fields overlap, creating a collective coverage. Similarly, in CNNs, this concept is emulated to ensure a detailed and holistic interpretation of the visual input. [22, 23].

3.2 Enhanced (E-CNN) Model for Plant Disease Detection and Classification

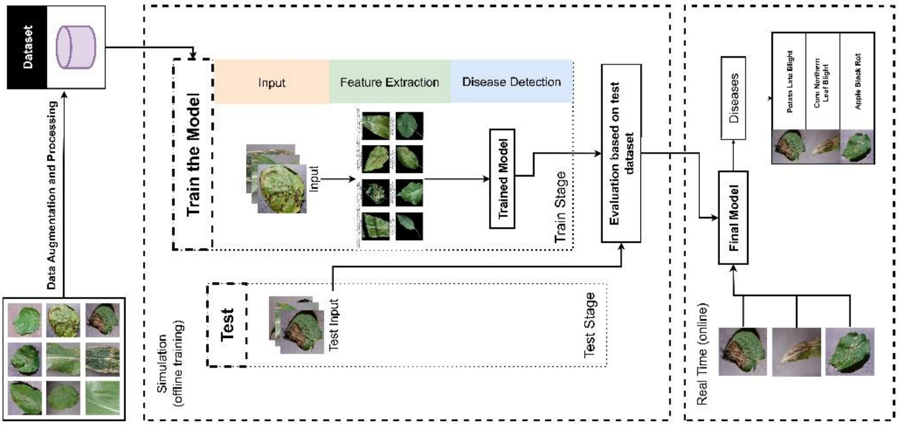

Figure 3. Illustration of Proposed Framework

Fig. 3 visually represents the entire workflow of our enhanced E-CNN Disease Detection Model. It offers a comprehensive overview of the model's functionality and the distinct stages it undergoes for efficient disease detection. In the next subsections we delve into a detailed breakdown of the components and steps as illustrated.

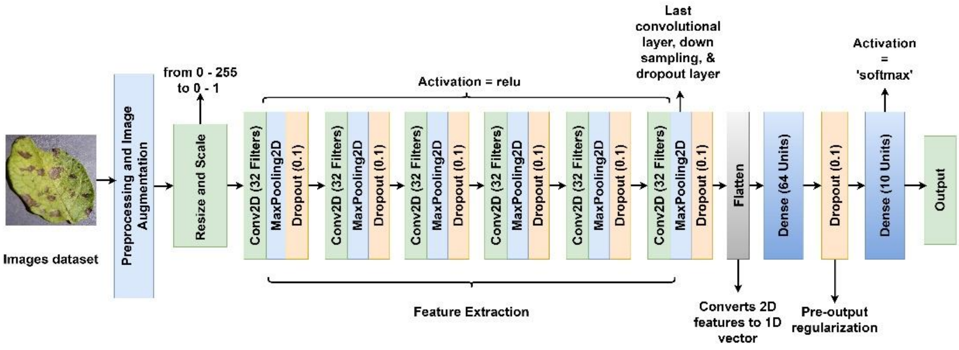

Figure 4. Architecture of E-CNN Model

Fig. 2 illustrates the structure of the Basic Convolutional Neural Network (CNN), whereas fig. 4 showcases the E-CNN model's design. CNNs, inspired by the human visual system, are extensively used in image processing and analysis [39]. Their strength lies in classifying unprocessed input data by autonomously determining the most appropriate filters for this task. This inherent ability of CNNs to extract and recognize features reduces the computational demands typically encountered in other machine learning methods for these functions [40]. Key elements of a CNN include convolutional, pooling, and fully connected layers, as outlined in summary table 2. In contrast, Artificial Neural Networks (ANNs) usually have three distinct layers: Input, Hidden, and Output. Neurons in the Hidden layer are characterized by specific bias values and weights. These are multiplied with input values and passed through an activation function. If the output value surpasses a certain threshold, it gets forwarded to the next layer in the network; if not, no data is transmitted. ANNs function in a feed-forward manner, meaning data moves in sequence through the layers. The main goal in adjusting the model is to lower the cost function for each input. The primary objective during model adjustment is to minimize the cost function for each input. CNNs represent a type of neural network with one or more layers designed to extract dependencies from inputs, such as text and images. A key feature of CNNs is the convolution operation performed across multiple intermediate layers. Convolution involves the dot-product of input bundles with a grid structure and a set of weights. CNNs are particularly popular in image processing and recognition. CNN architectures have seen significant advances, with LeNet-5 in 1998 being a notable milestone. Earlier computer vision techniques relied on feature identification, demanding substantial expertise in image processing. However, CNNs have transformed image processing by automating feature extraction. They are compatible with matrices, RGB color images, and even tensors, enabling image classification, segmentation, face detection, and object identification. CNNs have found successful applications in diverse fields, including healthcare, web services, mail, and natural language processing. A CNN comprises stacked layers, including convolution, pooling, ReLU activation, and fully connected layers as mentioned in Fig. 4. These layers process each input image through stages of filtering, correction, and reduction before being transformed into a vector. The convolution layer is instrumental in teaching CNNs to recognize specific features, such as object detection. Multiple convolution layers can be employed for added efficiency. The pooling layer further enhances efficiency by down-sampling, significantly reducing computational demands.

Table 2: The Configuration of E- CNN Architecture

Layer	Type	Input Shape	Output Shape	Kernal	Stride
1	Conv2D	(32, 254, 254, 32)	(32, 254, 254, 32)	(3,3)	(1,1)
2	MaxPooling2D	(32, 127, 254, 32)	(32, 127, 254, 32)	(2,2)	-
3	Conv2D	(32, 125, 125, 64)	(32, 125, 125, 64)	(3,3)	(1,1)
4	MaxPooling2D	(32, 62, 62, 64)	(32, 62, 62, 64)	(2,2)	-
5	Conv2D	(32, 60, 60, 64)	(32, 60, 60, 64)	(3,3)	(1,1)
6	MaxPooling2D	(32, 30, 30, 64)	(32, 30, 30, 64)	(2,2)	-
7	Conv2D	(32, 28, 28, 64)	(32, 28, 28, 64)	(3,3)	(1,1)
8	MaxPooling2D	(32, 14, 14, 64)	(32, 14, 14, 64)	(2,2)	-
9	Conv2D	(32, 12, 12, 64)	(32, 12, 12, 64)	(3,3)	(1,1)
10	MaxPooling2D	(32, 6, 6, 64)	(32, 6, 6, 64)	(2,2)	-
11	Conv2D	(32, 4, 4, 64)	(32, 4, 4, 64)	(3,3)	(1,1)
12	MaxPooling2D	(32, 2, 2, 64)	(32, 2, 2, 64)	(2,2)	-
13	Flatten	(32, 256)	1D tensor	-	-
14	Dense	(32, 64)	(64)	-	-
15	Dense	(32, 3)	(10)	-	-

A CNN architecture using the Keras Sequential model, designed for image classification. The initial operation, assumed to be rescaling, adjusts pixel values from the typical 0-255 range to a normalized scale of 0-1. The network then applies Conv2D layers with 32, 64, and 64 filters, sequentially, to detect increasingly complex features in the input images. After each Conv2D layer, MaxPooling2D is employed for spatial down sampling, reducing the spatial dimensions of the feature maps. To prevent overfitting, Dropout layers with a dropout rate of 0.1 (deactivating 10% of neurons) are strategically inserted after each MaxPooling2D layer for regularization.

This process of convolution, pooling, and dropout is repeated for several layers, progressively extracting detailed hierarchical features from the input images. The final part of the network involves flattening the 2D feature maps into a 1D vector, followed by a densely connected layer with 64 units and ReLu activation. Another dropout layer is applied before the output layer, which consists of 10 units, representing the number of classes in the classification task, and utilizes softmax activation for multi-class classification. The architecture is capped off with a final dropout layer before the output, enhancing model generalization. Overall, this CNN structure is well-suited for discerning intricate patterns in image data, contributing to accurate classification into one of the specified classes.

Hyper parameters	Description
Learning Rate	0.001
Batch Size	32
Epochs	100
Dropouts	0.1, 0.2, 0.3, 0.4
Optimizers	Adagrad, Adam, Adamax, Nadam
Activation Function	softmax, softplus, sigmoid, relu

Table 3: Hyperparameters

Table 3 shows several important hyper parameters which set up to affect the learning process during the training phase of a machine learning model. Training is done for this model over a period of 100 epochs, where an epoch is one full iteration over the dataset. The model's weights are updated using the ADAMAX optimizer, and the loss function selection is set to sparse categorical cross-entropy, which denotes a situation in which the target variable is categorical, and the classes are incompatible. To maximize computing effectiveness and memory utilization, the training data is handled in batches of 32 samples at a time. The dataset's images are standardized to 256 pixels in size, with three channels signifying the representation of RGB color.

3.3 Image Processing and Augmentation

We show in Fig. 5, three types of samples used for training and testing. Furthermore, as shown in Fig. 6, Data augmentation was employed during training, encompassing positional techniques like scaling, cropping, flipping, and rotation. Additionally, adjustments in brightness, contrast, and saturation were applied to enhance color quality [41]. Data augmentation also featured random rotations and distortions, along with horizontal flips. This process generated eight enhanced images for each original image. All images, both original and enhanced, were initially normalized by dividing each pixel's value by 255. Furthermore, the images were resized to match the input requirements of various model architectures. Due to hardware constraints, the input size for all EfficientNet architecture models was set at 132x132, ensuring uniform comparisons among models as done in [42]. Table 4 and Fig. 5 report information about different types of diseases affecting plants.

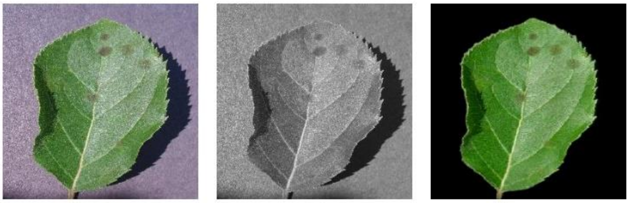

Fig. (a) RGB Dataset Sample Fig. (b) Greyscale Dataset Sample Fig. (c) Segmented Dataset Sample

Figure 5. Types of Samples for training and testing

Table 4: Diseases classification.

Plant Name	Disease Name	Scientific Name	Type of Disease	No. of Images
Potato	-	-	-	152
Potato	Late Blight	Phytophthora infestans (Montagne)	Fungal	1000
Potato	Early Blight	Alternaria solani	Fungal	1000
Apple	Apple Scab	Venturia Inaequalis	Fungal	630
Apple	Black Rot	Botryosphaeria Obtusa	Fungal	621
Apple	Cedar Apple Rust	Gymnosporangium juniperi-virginianae Schwein		275
Apple	-	-	-	1645
Corn (Maize)	-	-	-	1162
Corn (Maize)	Northern Leaf Blight	Exserohilum turcicum Pleosporales	fungal meningitis	985
Corn (Maize)	Cercospora Leaf Spot Gray_leaf_spot	Cercospora zeae-maydis	fungal pathogen	513

Original Image Rotated Image Horizontal Flip Image Vertical Flip Image Salt & pepper Image

Types of Noises we used

Figure 6. Data Augmentation.

Figure 7. Classes Distribution

Table 6: Algorithm of Proposed Enhanced CNN Model

Input: image from dataset D
Output: the disease recognized label
1: Initialize the enhanced CNN parameters.
2: Data augmentation and preprocessing on input image
3: for each di Eshuffle (D) do
4: Extract complicated features with the enhanced CNN
5: Input all extracted features into a global average pooling layer
6: Classify an image in Dense + Softmax layer
7: end for

Table 6 shows the algorithm of our proposed E-CNN model where you can see the each step clearly.

3.4 Feature Extraction and Selection

Feature Extraction: First, basic properties like statistical characteristics of colors (like RGB, HSV, and CIELab, min, max, mean, standard deviation, bias, and kurtosis) will be extracted. Textural and morphological elements like Haralick or Local Binary Patterns can be extracted if necessary.

Feature Selection: We can determine the salient characteristics of all diseases plus the salient characteristics of nutritious vegetables with the aid of feature selection. To find hidden patterns in the data, we employed filter techniques like Pearson Correlation Coefficient (PCC), Variance, ANOVA and Principal Component Analysis (PCA).

Pearson Correlation Coefficient (PCC): PCC is a statistical method used in measuring the correlation which is linear in nature, between two variables x and y. The range for this variable is in between +1 to -1, Positive 1 refers positive association between variables and negative 1 means there is negative association between variables.

Variance: Variance is a measure of how sensitive the model is to variation in the training set. It is a metric for the extent that the model's predictions change depending on the subsets of the training set it is trained on. Moreover, it can be defined as how much the predicted values are away from each other. A high variance model runs the risk of being overfitting because it is excessively complicated and captures both the random noise and the deeper trends in the data. Because the model has basically remembered the training data instead of generalizing from it, overfitting happens if a model performs outstanding over trained data but badly on unknown data.

ANOVA: ANOVA refers to the Analysis of Variance. It is used to show the difference between two or more than two means. It is generally used to identify which features are most important in predicting outputs. It is an important feature of feature selection.

Principal Component Analysis (PCA): PCA is used to overcome overfitting errors in model through different views, it is a statistical procedure that uses orthogonal alteration that fluctuates set of correlated variables to set uncorrelated variables.

Statistical Analysis serves as a pivotal scientific instrument, facilitating the examination and collection of extensive datasets. Its primary purpose is to discern prevalent patterns and trends within these large volumes of data, thereby converting them into insightful and valuable information.

We will use only one way of Analysis of Variance (ANOVA) and we assist t-student test with in the most important feature to determine the statistical difference between diseased and healthy leaves.

3.5 Mobile App Integration

The combination of TFLite, Firebase, Flutter, and Dart [43] presents a comprehensive and powerful stack for developing machine learning-powered mobile applications. TFLite, or TensorFlow Lite, facilitates the deployment of machine learning models on mobile and embedded devices, ensuring efficient and optimized inference. Firebase integration adds a robust backend to the application, offering features like real-time database, authentication, and cloud functions, enhancing the app's functionality and scalability. Flutter, a UI toolkit developed by Google, along with Dart as its programming language, provides a cross-platform framework for building aesthetically pleasing and high-performance mobile applications. The inclusion of Dart for background processing ensures smooth and responsive app behavior even when running tasks in the background. Diagrams, presumably used for system architecture or workflow visualization, contribute to the clarity and understanding of the overall application structure. Fig. 8 represents the users interaction with our mobile application in which we integrated our proposed E-CNN model.

There are two options, one to capture an image of leaf at instant while second option is to upload captured image from gallery. After uploading or capturing, the image will pass through our proposed E-CNN model and then user can see the label of result on their screen. Users can also see the details, causes, and treatment of that observed disease. In our experiment we used Redmi 13C mobile to test this E-CNN based Mobile App. After testing we observed that our application took 25 seconds on average to detect disease from the image, almost 15 seconds required for uploading or capturing image while only 10 seconds on average required for processing E-CNN model and in showing result. In the whole testing we used 4G network. Tests results are shown in fig. 16.

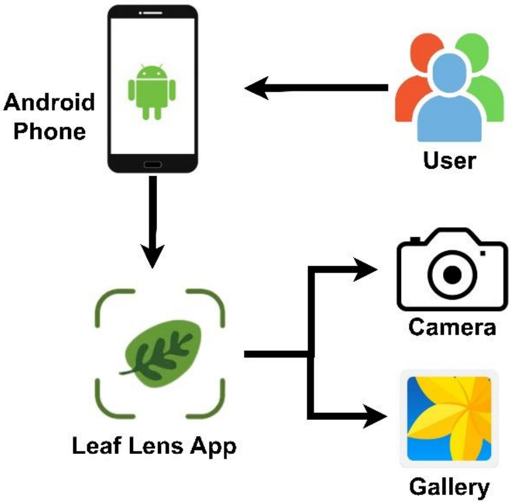

Figure 8. User Interaction with Android App

4 Results and Discussion

4.1 Implementation of the Model

This section describes how the proposed method of E-CNN, was taken into consideration for comparison and evaluation, was implemented using the Scikit-Learn library and Keras, a high-level API based on TensorFlow Lite, in Python programming language. We used Jupyter Notebook which is an interactive web-based platform for machine learning training and testing. The dataset in the above experiment was taken from kaggle, a dataset provider named PlantVillage contained 54,000 images of different plants. We used 10% of dataset as validation, 10% of dataset as testing and remaining 80% as training. During training the batch size was set to 32, and epochs was set to 100. Other hyper parameters such as learning rate was set as 0.0001. During training we compared ADAM, ADAMAX, ADAGRAD & NADAM, we selected ADAMAX as optimizer because of its better accuracy among all.

4.2 Dataset

By 2050, human society will need to raise food production by approximately 70 percent in order to feed the world's estimated 9 billion inhabitants. Currently, infectious illnesses limit potential yield by 40% on average, with yield losses reaching 100% for many farmers in developing nations. For evaluation of the proposed method of E-CNN, The dataset in the above experiment was taken from kaggle, a dataset provider named Plant Village contained 54,000 images of healthy and infected leaves of different crops.

4.3 Evaluation Metrics

It has been noted that the PlantVillage dataset was used in most of the research. As a result, the classifications' nature and outcome are extremely similar. Additionally, some researchers have classified bean crop diseases using deep learning algorithms. To prevent findings from repeating, with the help of three deep learning models which are utilizing different optimization techniques the dataset of leaf disease has been used. After the DL model and optimization technique were shown to be the most efficient duo, the collection of photos of bean leaves underwent a varied class illnesses classification. To distinguish diseases, we collected photos of diseased leaves from PlantVillage dataset.

$$Accuracy = \frac{T_P + T_N}{T_P + F_N + F_P + T_N} \quad (1)$$

$$Sensitivity\ or\ Recall(R) = \frac{T_P}{T_P + F_N} \quad (2)$$

$$Specificity = \frac{T_N}{F_P + T_N} \quad (3)$$

$$Precision\ (P) = \frac{T_P}{T_P + F_P} \quad (4)$$

$$F-measure = \frac{2 \times P \times R}{P + R} \quad (5)$$

$$G - Mean = \sqrt{Sensitivity \times Specificity} \quad (6)$$

Where T_N and T_P are the number of accurate guesses made when the actual class is False or True, respectively. Additionally, the numbers F_N and F_P indicate how many wrong predictions there were when the actual class was True or false, accordingly. These metrics are combined to create a graphic known as the receiver's operating characteristic (ROC) curve. This curve illustrates the trade-off between a model's classification mistakes and F_N and F_P rates. Furthermore, using the ROC curve, the AUC graph can be generated. Specifically, the level of separability is represented by AUC, which is led by ROC, a probability curve. AUC indicates a model's capacity to perform well in a classification test. High separability metrics indicate a superior model with an AUC close to one. Conversely, a very poor separability measure is indicated by an AUC close to zero. Assuming that sensitivity & 1-specificity correspond with the probability of T_P and F_P respectively, the following is an estimate of the AUC.

Table 7 shows how confusion matrix represent values.

Table 7 Confusion matrix

Confusion matrix		Predicted class	
		Positive	Negative
Actual class	Positive	T_P	F_N
	Negative	F_P	T_N

$$\mathrm{AUC} = \sum i\{[Sensitivity.\Delta(1 - Specificity)] + \frac{1}{2}[\Delta Sensitivity.\Delta(1 - Specificity)]\}, \quad (7)$$

in which

Δ(1-Specificity) = (1-Specificity)i - (1-Specificity)i-1 (8)

And

ΔSensitivity = Sensitivityi – Sensitivityi-1, where i is used as index. (9)

Table 8: Features Extraction

Feature Name	Real Feature	Obtained Feature Values
mean_c1_rgb	mean of red	1.3442651765346868e-300
mean_c2_rgb	mean of green	0.0
mean_c3_rgb	mean of blue	8.939919952500531e-225
std_c1_rgb	standard deviation of red	4.6115436039578915e-125
std_c2_rgb	Standard deviation of green	2.3159068791968023e-123
std_c3_rgb	standard deviation of blue	8.677760022335703e-144
min_c1_rgb	minimum of red	1.0800379855618637e-225
min_c2_rgb	minimum of green	4.15e-322
min_c3_rgb	minimum of blue	3.184826853782704e-206
max_c1_rgb	maximum of red	7.204390205067731e-220
max_c2_rgb	maximum of green	3.3134991119925323e-249
max_c3_rgb	maximum of blue	1.5844288235712915e-207
mean_c1_lab	mean of L	4.5420900929257516e-237
mean_c2_lab	mean of a*	0.0
mean_c3_lab	mean of b*	0.0
std_c1_lab	standard L	5.242290141124147e-140
std_c2_lab	standard a*	0.0
std_c3_lab	standard b*	3.990956620006536e-282
min_c1_lab	minimum of L	7.469894518070457e-301
min_c2_lab	minimum of a*	1.3269063332187237e-213
min_c3_lab	minimum of b*	0.0
max_c1_lab	maximum of L	1.7317817297094814e-231
max_c2_lab	maximum of a*	0.0
max_c3_lab	maximum of b*	7.871720232890082e-238

The following table 8 represents the feature name and its real features such as mean_c1_rgb having the mean of red color, mean_c2_rgb having the mean of green color, and mean_c3_rgb having the mean of blue color. While std_c1_rgb refers to the standard deviation of red color, std_c2_rgb refers to the standard deviation of green color and std_c3_rgb refers to the standard deviation of blue color while min_c1_rgb is minimum of red, min_c2_rgb is minimum of green and min_c3_rgb is minimum of blue and same as for max. Max_c1_rgb is maximum of red, max_c2_rgb is maximum of green and max_c3_rgb is maximum of blue. Mean_c1_lab represents mean of L, mean_c2_lab represents mean of a, mean_c3_lab represents the mean of b same as std_c1_lab showing the feature of standard L, std_c2_lab showing the feature of standard a and std_c3_lab showing the features of standard b. at the end min_c1_lab, min_c2_lab and min_c3_lab represents minimum L, a and b and max_c1_lab represents maximum L, a and b.

Table 9: Logistic Regression on Extracted Features

	Precision	Recall	F1-Score	Support
Apple Black Rot	0.88	0.88	0.88	200
Apple Cedar Rust	0.85	0.85	0.85	200
Apple Healthy	0.90	0.91	0.90	200
Apple Scab	0.85	0.82	0.84	200
Corn Cercosporin Leaf	0.64	0.67	0.65	200
Corn Healthy	1.00	0.99	0.99	200
Corn Northern Leaf	0.65	0.61	0.63	200
Potato Early	0.93	0.97	0.95	200
Potato Healthy	0.76	0.75	0.76	152
Potato Late Blight	0.74	0.78	0.76	200
Accuracy	0.82	0.82	0.82	0.82
Macro avg	0.82	0.82	0.82	1952
Weighted avg	0.82	0.82	0.82	1952

We have predicted these values and their classification report with the help of logistic regression and Pearson Correlation Coefficient, which is generally known as PCC, we have performed this classification using k-fold cross validation predict, which is initially put as value of 10 and we got the accuracy of up to 82% of logistic regression.

4.4 Performance Analysis

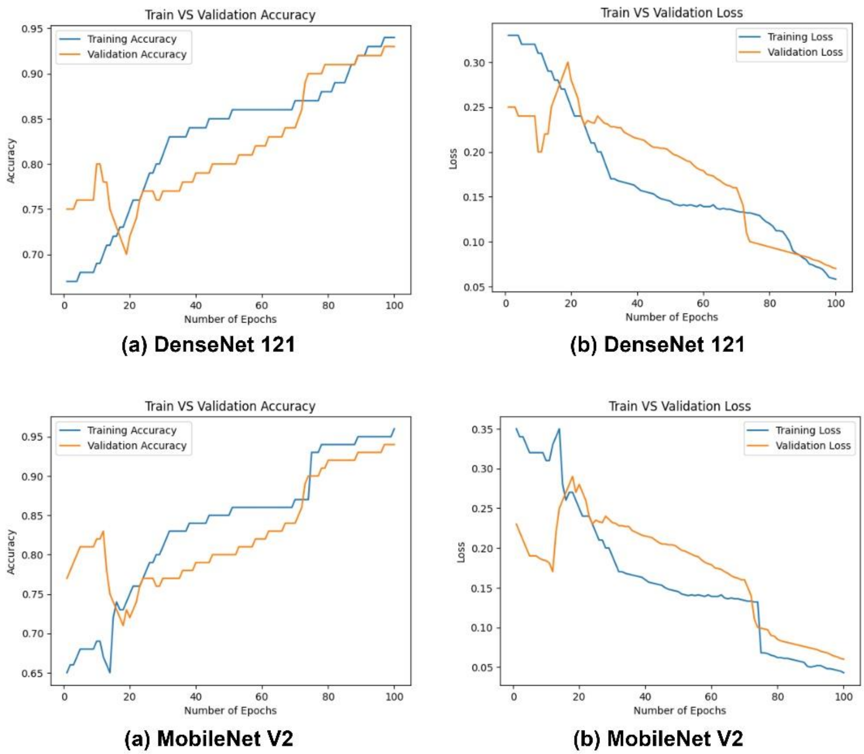

Figure 9. Training and Validation of MobileNet and DenseNet.

Table 10. Result comparisons of Pretrained models on plant disease detection

Pre-Trained Models	**No. of Layers**	**Parameter (million)**	**Size**	**Accuracy**	**Disease Types**	**Dataset**
ResNet50	50	23	98 MB	98%	Fungal	10 Classes (PlantVillage)
DenseNet 121	121	8.06	33 MB	94.20%	Fungal	10 Classes (PlantVillage)
AlexNet	8	60	240 MB	87.56%	Fungal	10 Classes (PlantVillage)
VGG16	23	138	528 MB	91%	Fungal	10 Classes (PlantVillage)
MobileNet V2	88	3.37	14 MB	95.69%	Fungal	10 Classes (PlantVillage)

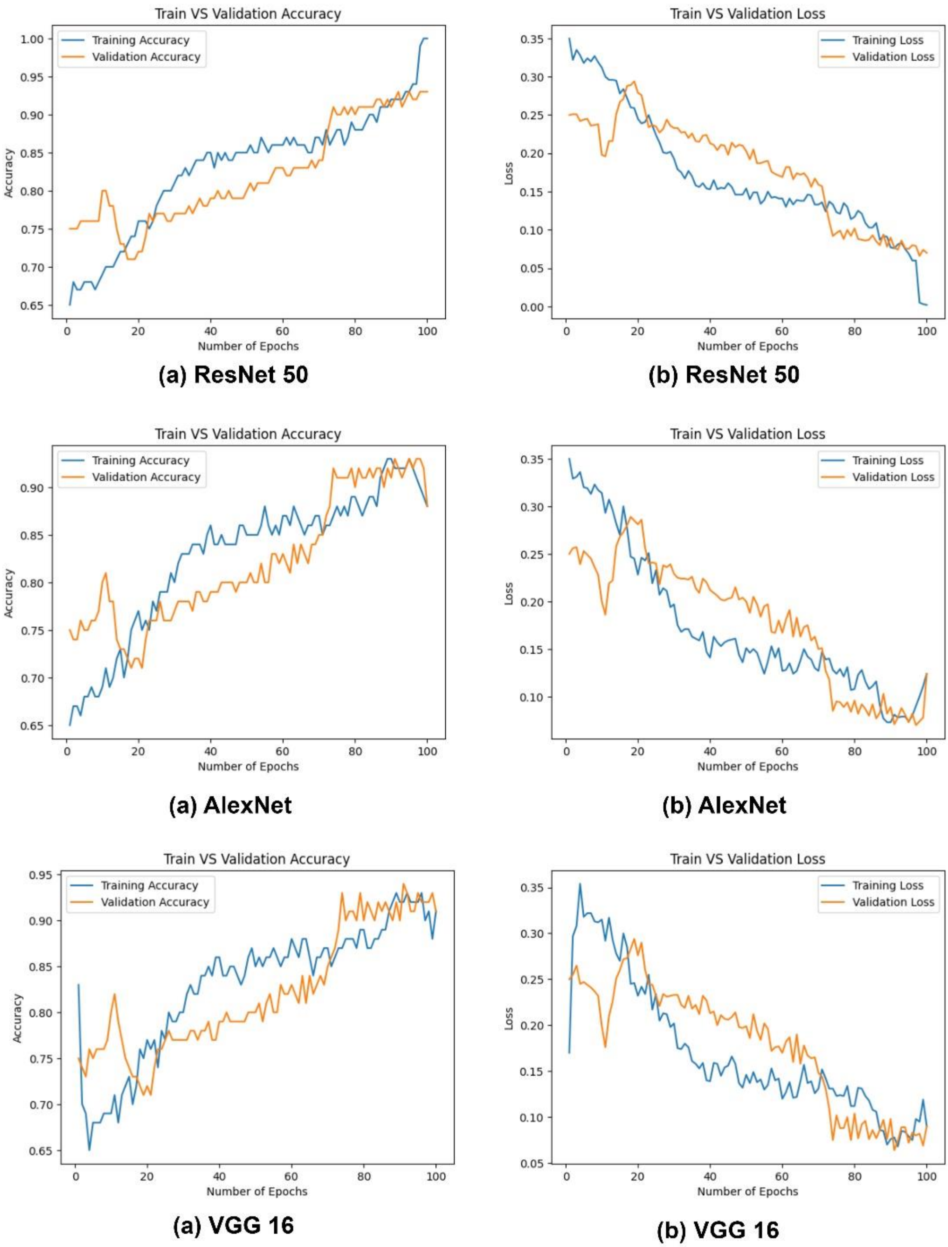

Figure 10. Training and Validation of ResNet 50, AlexNet and VGG16.

Figure 11 presents the classification results for different machine learning models on a given task. The Support Vector Machine (SVM) and ResNet50 achieved accuracy, with a score of 96% and 97.80. The Decision Tree model performed well with an 80% accuracy, indicating its ability to create a hierarchical decision structure for classification. Lastly, the K-Nearest Neighbor (KNN) model achieved a lower accuracy of 71%, implying that its performance was not as strong in this task, potentially due to its sensitivity to local variations in the dataset. While our E-CNN model achieved

98.2% accuracy.

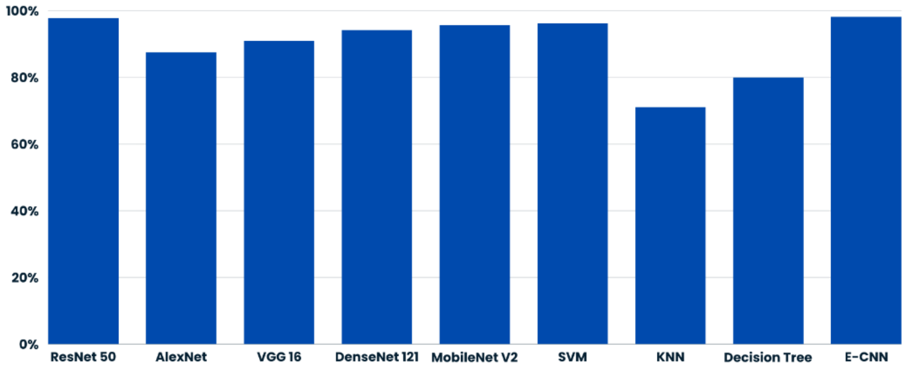

Figure 11: Accuracy Bar Chart of Pretrained Models and Machine Learning Models

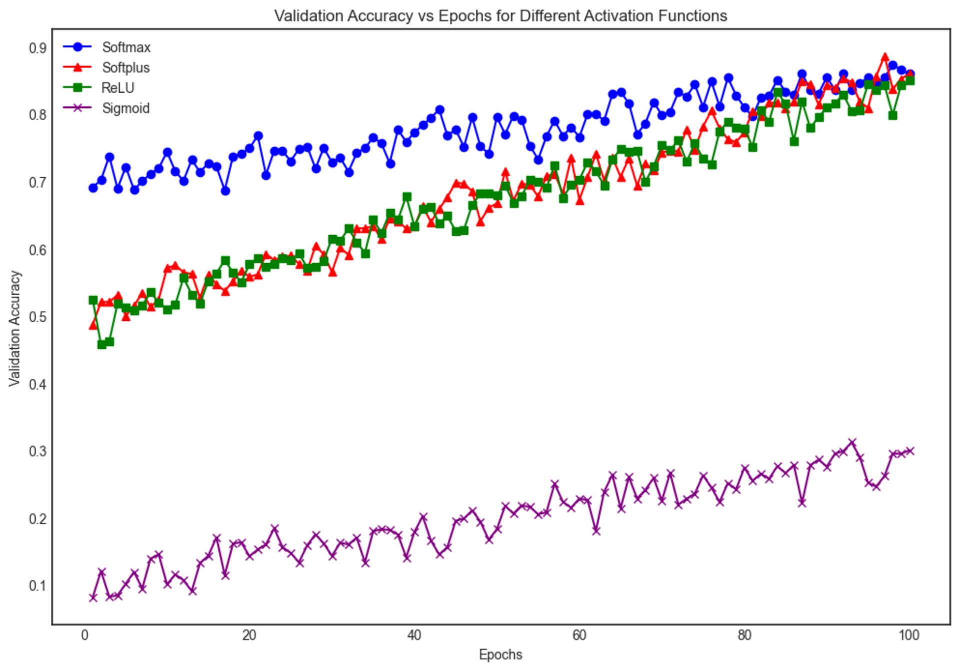

Figure 12. Activation Accuracy (update this figure 100 epochs)

Table 12: Result of Hyperparameters

Class	Hyper	Precision	Recall	F1-Score
	Adagrad	19%	28%	20%
	Adam	86%	79%	79%
Optimizer	Adamax	88%	80%	80%
	Nadam	43%	49%	42%
	Softmax	86%	85%	85%
	Softplus	86%	83%	83%
Activation	Sigmoid	1%	10%	2%
	ReLu	85%	83%	83%
	0.1	88%	87%	87%
	0.2	86%	86%	86%
Dropout Rate	0.3	86%	84%	84%
	0.4	82%	79%	79%

As presented in Table 12, each trained model underwent individual training with distinct optimizers, learning rates, epochs, dropout rates, and batch sizes. These variations were meticulously analyzed to determine their impact on model performance. The confusion matrix generated during the analysis facilitated the calculation of key classification metrics, including True Positive (TP), False Positive (FP), True Negative (TN), and False Negative (FN). These metrics are instrumental in evaluating the model's classification accuracy and error. To provide clarity, these classifications can be understood in the context of multi-class image classification as follows: True Positive (TP): Refers to images correctly categorized within their respective classes. False Positive (FP): Denotes images from relevant categories that were inaccurately classified as belonging to unrelated categories. True Negative (TN): Signifies correctly categorized images, excluding those within the relevant categories. False Negative (FN): Represents images from unrelated categories that were erroneously classified as belonging to relevant categories. These classifications were employed to compute performance metrics using the equations presented. The experimentation conducted in this study leveraged the Jupyter Notebook and focused on CNN models. The proposed research study centered on the potato, apple, and corn dataset, consisting of 9175 images divided into train, test, and validation sets. The potato dataset encompassed three distinct classes: healthy, early blight disease, and late blight disease, apple and corn also have divided with their respective diseases. In this experiment, various CNN models underwent training using different optimization approaches (specifically, Adam, Adamax, Adagrad, and Nadam) with fixed learning rates, epochs, and batch sizes. In table 12 the classification results for different hyperparameter settings are presented. Notably, the highest accuracy observed is 88%, achieved with the "0.1" dropout rate using the Adagrad optimizer. Conversely, the lowest accuracy, 1%, was obtained with the Sigmoid activation function. These results showcase the significant impact of hyperparameter selection on the performance of the plant disease detection model. Further analysis and discussion of these outcomes are provided in the following sections.

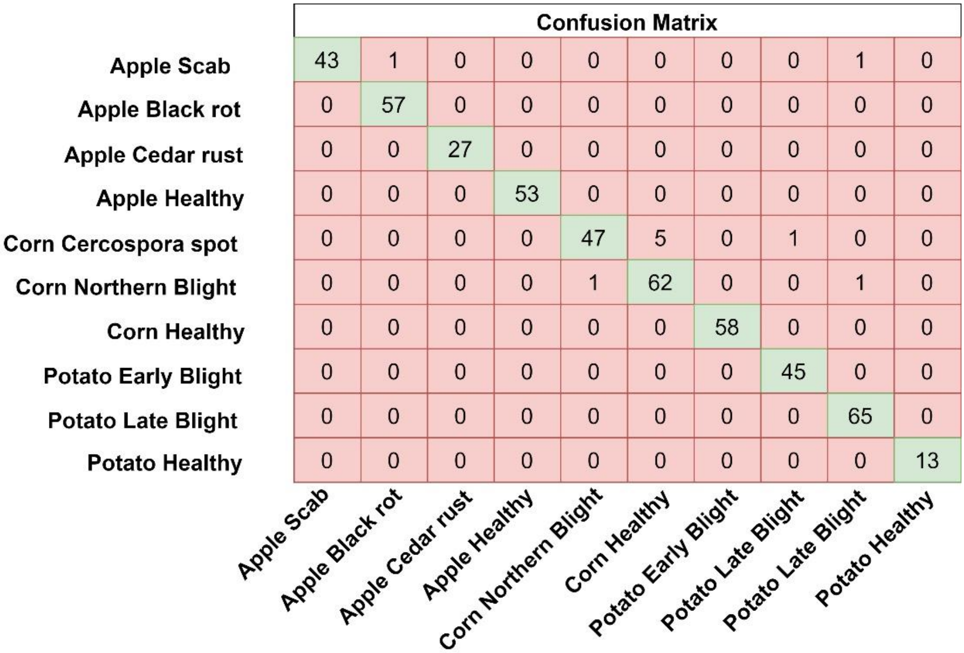

Figure 13. Proposed Confusion Matrix

The model's overall reliability was assessed through accuracy, and equations (1) to (4) were employed to calculate various performance measures using the confusion matrix. The results indicated that as the number of iterations increased, accuracy improved while model loss decreased, as illustrated in the accompanying graphs. To prevent overfitting, both training and validation accuracy were evaluated, with the model being trained over 100 epochs. In summary: In the 10th epoch, the CNN model trained on the mentioned dataset achieved an accuracy of 75%. By the 100th epoch, the CNN model's validation accuracy for the potato dataset reached an impressive 98.17%.

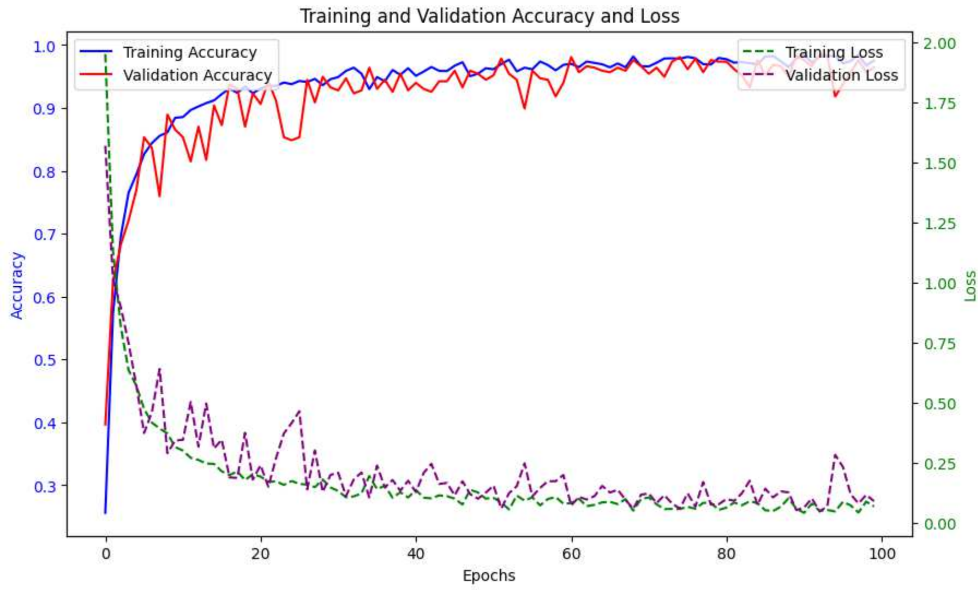

Figure 14: Loss and Accuracy of Proposed Enhanced CNN Model

This research also included a comprehensive Classification Report that considered different optimizers, activation functions, and dropout rates. Table 12 provides numerical representations and insights into various aspects of the experimentation, including the validation accuracy, the impact of different activation functions, and the effect of varying dropout rates. The results indicate promising outcomes, with Softmax activation and a dropout rate of 0.1 emerging as particularly effective configurations for the model. Further experimentation and exploration of these findings could lead to improved plant disease detection models with real-world applications. In Figure 10 (e) and (f), our pre-trained VGG-16 model demonstrated an impressive 91% accuracy while utilizing 23 layers with a total of 138 parameters. The associated image dataset occupied 528 MB. It is evident that our VGG-16 model outperformed the study in terms of accuracy. Moving on to Figure 10(c) and (d), our fine-tuned model based on the pre-trained AlexNet [44], [45] achieved an accuracy of 87.56% with just 8 layers and a minimal parameter count of 60. The image dataset used for this model was 240 MB in size. Thus, our AlexNet model showcased superior accuracy compared to the referenced study. Now, in Figure 10 (a) and (b), our pre-trained ResNet50 model exhibited remarkable accuracy, reaching an impressive 98%, while boasting a substantial 50-layer depth with a parameter count of only 23. The image dataset occupied 98 MB. Clearly, our ResNet50 model excelled in accuracy. In Figure 9 (a) and (b), our pre-trained DenseNet 121 model achieved an accuracy of 92.20% with a substantial 121 layers. The specific parameter count is not mentioned. The image dataset size was 93 MB. It's worth noting that despite its increased accuracy, the parameter count has been reduced from 4.24 million to a mere 3.47 million. Lastly, in Figure 9 (c) and (d), our pre-trained MobileNet model attained an accuracy of 84.02% using 88 layers and a parameter count of 3.37. The image dataset was relatively compact at 14 MB in size.

As shown in Table 13, we present a comparison of pretrained models for plant disease detection using various metrics such as the number of layers, parameter count, dataset size, accuracy, and the types of diseases considered. Table 13, the accuracy levels achieved in different studies related to crop disease detection are provided. The highest accuracy recorded among these studies is 98.17%, which is achieved by the proposed system for Potato disease detection using a CNN approach on the PlantVillage dataset. On the other hand, the lowest accuracy in the table is 73.50%, achieved by Bi et al. in the case of Apple disease detection using MobileNet with 300+ captured images. This comparison serves as a reference point to evaluate the performance of the proposed system and its significance in the field of crop disease detection. Further discussions and implications of these results are elaborated upon in the subsequent sections.

Table 13: Baseline and target performance levels

Reference	Crop	Approach	Accuracy	Dataset
Padol et al. (2016) [28]	Grape	SVM	88.89%	137 captured images
Fuentes et al (2017) [46]	Tomato	Faster R-CNN	83% (Testing acc)	Own
Hlaing et al. (2017) [47]	Cotton	SVM	84.7%	3474 images from PlantVillage
Hlaing et al. (2018) [48]	Tomato	SVM	85.1 %	3535 images from PlantVillage
Atole et al (2018) [49]	Rice	AlexNet	91.23%	850+ images
Ramcharan et al (2019) [50]	Cassava	MobileNet	80.6%	2000+ images
Ahmad et al. (2020) [22]	Tomato	ResNet	85%	300+ images
Bi et al. (2020) [51]	Apple	MobileNet	73.50%	300+ captured images
Proposed E-CNN Model	Multiple Plants	CNN	**98.17%**	PlantVillage

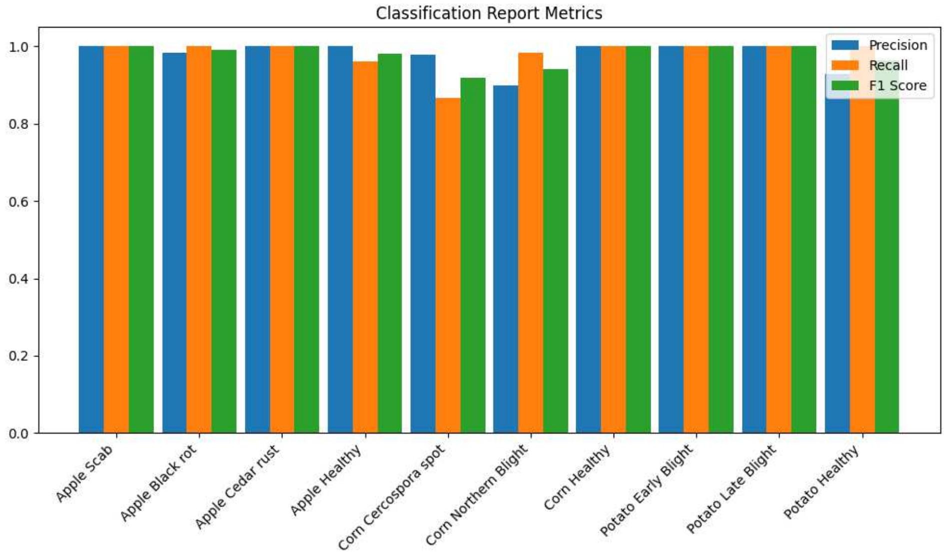

Figure 15: Classification report matrix of our proposed improved E-CNN model which has Epoch 100, Optimizer 'Adamax', Activation Function 'Softmax', and Dropout '0.1'.

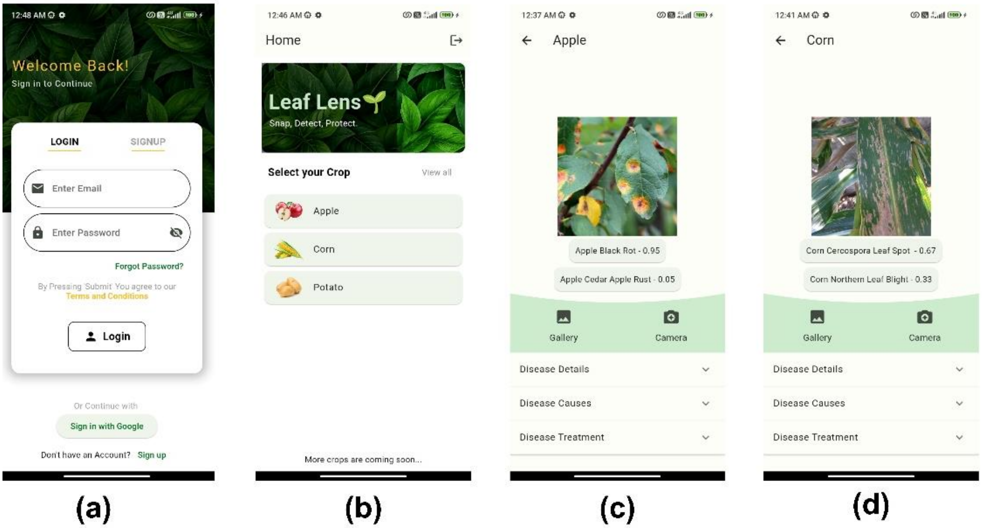

Figure 16: Integration of TF model in flutter mobile application

5 Conclusion

This paper introduces an advanced model for plant disease classification, leveraging an enhanced Convolutional Neural Network (CNN) methodology. The core CNN structure underwent modifications, including an increase in the number of network layers and the incorporation of a Global Average Pooling (GAP) layer and a Batch Normalization (BN) layer. In comparison to contemporary image classification techniques, our proposed model E-CNN demonstrates superior classification accuracy, and the structural adjustments contribute to a reduction in the number of parameters. The experimental results affirm the efficacy of the proposed method, particularly in the domain of plant disease classification. Notably, the suggested E-CNN model found successful application in identifying citrus illnesses. However, it is imperative to acknowledge certain limitations in the present study, such as the utilization of a relatively dataset comprising 9175 photos. Models trained on smaller datasets are susceptible to overfitting, leading to inaccurate assessments. To address this, data augmentation was employed to artificially enhance the photos. Nonetheless, the introduction of augmentation poses a challenge: if the source data exhibits biases, the augmented images may inherit the same issues. Thus, the selection of an appropriate augmentation technique becomes a complex task. In future research endeavors, our focus will extend to training the proposed network to handle larger and more varied dataset in plant disease recognition and classification. This forward-looking approach aims to overcome current limitations and contribute to a more robust and versatile solution for plant disease analysis.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

Ethical Approval This paper does not contain any studies with human participants or animals performed by any of the authors.

Human and Animal Rights This paper does not contain any studies with human participants or animals performed by any of the authors.

Informed Consent Informed consent was obtained from all individual participants included in the study.

Acknowledgement: This work is funded by the QNL, Qatar.

References

[1] Bangari, S., Rachana, P., Gupta, N., Sudi, P. S., & Baniya, K. K. (2022, February). A survey on disease detection of a potato leaf using cnn. In 2022 Second International Conference on Artificial Intelligence and Smart Energy (ICAIS) (pp. 144-149). IEEE.

[2] Singh, V., Chug, A., & Singh, A. P. (2023). Classification of Beans Leaf Diseases using Fine Tuned CNN Model. Procedia Computer Science, 218, 348-356.

[3] Asif, M. K. R., Rahman, M. A., & Hena, M. H. (2020, December). CNN based disease detection approach on potato leaves. In 2020 3rd International Conference on Intelligent Sustainable Systems (ICISS) (pp. 428-432). IEEE.

[4] Camargo, A., & Smith, J. S. (2009). Image pattern classification for the identification of disease causing agents in plants. Computers and electronics in agriculture, 66(2), 121-125.

[5] Zhang, Y., Wa, S., Liu, Y., Zhou, X., Sun, P., & Ma, Q. (2021). High-accuracy detection of maize leaf diseases CNN based on multi-pathway activation function module. Remote Sensing, 13(21), 4218.

[6] Sathya, K., & Rajalakshmi, M. (2022). RDA-CNN: Enhanced Super Resolution Method for Rice Plant Disease Classification. Computer Systems Science & Engineering, 42(1).

[7] Islam, M. T. (2020). Plant disease detection using CNN model and image processing. Int. J. Eng. Res. Technol, 9(10), 291-297.

[8] Verma, S., Chug, A., & Singh, A. P. (2020). Impact of hyperparameter tuning on deep learning based estimation of disease severity in grape plant. In Recent Advances on Soft Computing and Data Mining: Proceedings of the Fourth International Conference on Soft Computing and Data Mining (SCDM 2020), Melaka, Malaysia, January 22– 23, 2020 (pp. 161-171). Springer International Publishing.

[9] Sankaran, S., Mishra, A., Ehsani, R., & Davis, C. (2010). A review of advanced techniques for

detecting plant diseases. Computers and electronics in agriculture, 72(1), 1-13.
[10] Barman, U., Sahu, D., Barman, G. G., & Das, J. (2020, July). Comparative assessment of deep learning to detect the leaf diseases of potato based on data augmentation. In 2020 International Conference on Computational Performance Evaluation (ComPE) (pp. 682-687). IEEE.
[11] Poojary, R., & Pai, A. (2019, November). Comparative study of model optimization techniques in fine-tuned CNN models. In 2019 International Conference on Electrical and Computing Technologies and Applications (ICECTA) (pp. 1-4). IEEE.
[12] Butale, N. M., & Kodavade, D. V. (2019). Detection of plant leaf diseases using image processing and soft-computing techniques. International Research Journal of Engineering and Technology, 6, 3288-3291.
[13] Saleem, M. H., Potgieter, J., & Arif, K. M. (2020). Plant disease classification: A comparative evaluation of convolutional neural networks and deep learning optimizers. Plants, 9(10), 1319.
[14] Yi, N., Li, C., Feng, X., & Shi, M. (2018, June). Research and improvement of convolutional neural network. In 2018 IEEE/ACIS 17th International Conference on Computer and Information Science (ICIS) (pp. 637-640). IEEE.
[15] B. Liu, Y. Zhang, D. He, and Y. Li, "Identification of Apple Leaf Diseases Based on Deep Convolutional Neural Networks," Symmetry, vol. 10, no. 1, Art. no. 1, Jan. 2018, doi: 10.3390/sym10010011.
[16] Subramanian, M., Lv, N. P., & VE, S. (2022). Hyperparameter optimization for transfer learning of VGG16 for disease identification in corn leaves using Bayesian optimization. Big Data, 10(3), 215-229.
[17] Pujari, D., Yakkundimath, R., & Byadgi, A. S. (2016). SVM and ANN based classification of plant diseases using feature reduction technique. IJIMAI, 3(7), 6-14.
[18] Sharma, P., Berwal, Y. P. S., & Ghai, W. Performance analysis of deep learning CNN models for disease detection in plants using image segmentation. Inf. Process. Agric.(2019).
[19] Durmuş, H., Güneş, E. O., & Kırcı, M. (2017, August). Disease detection on the leaves of the tomato plants by using deep learning. In 2017 6th International conference on agro-geoinformatics (pp. 1-5). IEEE.
[20] DeChant, C., Wiesner-Hanks, T., Chen, S., Stewart, E. L., Yosinski, J., Gore, M. A., ... & Lipson, H. (2017). Automated identification of northern leaf blight-infected maize plants from field imagery using deep learning. Phytopathology, 107(11), 1426-1432.
[21] "PlantVillage Dataset." Accessed: Dec. 05, 2023. [Online]. Available: https://www.kaggle.com/datasets/abdallahalidev/plantvillage-dataset
[22] Ahmad, I., Hamid, M., Yousaf, S., Shah, S. T., & Ahmad, M. O. (2020). Optimizing pretrained convolutional neural networks for tomato leaf disease detection. Complexity, 2020, 1-6.
[23] Nachtigall, L. G. (2016). Classificação de distúrbios em folhas de macieiras utilizando redes neurais convolucionais (Master's thesis, Universidade Federal de Pelotas).
[24] Daphal, S. D., & Koli, S. M. (2023). Enhancing sugarcane disease classification with ensemble deep learning: A comparative study with transfer learning techniques. Heliyon, 9(8).
[25] Barbedo, J. G. A., Koenigkan, L. V., & Santos, T. T. (2016). Identifying multiple plant diseases using digital image processing. Biosystems engineering, 147, 104-116.
[26] Kaur, I., Aggarwal, G., & Verma, A. (2016). Detection and classification of disease affected region of plant leaves using image processing. Indian J. Sci. Technol, 9, 48.
[27] Mohanty, S. P., Hughes, D. P., & Salathé, M. (2016). Using deep learning for image-based plant disease detection. Frontiers in plant science, 7, 1419.
[28] Padol, P. B., & Yadav, A. A. (2016, June). SVM classifier based grape leaf disease detection. In 2016 Conference on advances in signal processing (CASP) (pp. 175-179). IEEE.
[29] Lu, Y., Yi, S., Zeng, N., Liu, Y., & Zhang, Y. (2017). Identification of rice diseases using deep convolutional neural networks. Neurocomputing, 267, 378-384.
[30] Verma, S., Chug, A., & Singh, A. P. (2018, September). Prediction models for identification and diagnosis of tomato plant diseases. In 2018 International Conference on advances in computing, communications and informatics (ICACCI) (pp. 1557-1563). IEEE.
[31] Adedoja, A., Owolawi, P. A., & Mapayi, T. (2019, August). Deep learning based on nasnet for plant disease recognition using leave images. In 2019 international conference on advances in big data, computing and data communication systems (icABCD) (pp. 1-5). IEEE.
[32] Ozguven, M. M., & Adem, K. (2019). Automatic detection and classification of leaf spot disease in sugar beet using deep learning algorithms. Physica A: statistical mechanics and its applications, 535, 122537.
[33] Hu, G., Yang, X., Zhang, Y., & Wan, M. (2019). Identification of tea leaf diseases by using an improved deep convolutional neural network. Sustainable Computing: Informatics and Systems, 24, 100353.
[34] Li, Y., Nie, J., & Chao, X. (2020). Do we really need deep CNN for plant diseases identification?. Computers and Electronics in Agriculture, 178, 105803.

[35] Chen, J., Chen, J., Zhang, D., Sun, Y., & Nanehkaran, Y. A. (2020). Using deep transfer learning for image-based plant disease identification. Computers and Electronics in Agriculture, 173, 105393.
[36] Atila, Ü., Uçar, M., Akyol, K., & Uçar, E. (2021). Plant leaf disease classification using EfficientNet deep learning model. Ecological Informatics, 61, 101182.
[37] Tuncer, A. (2021). Cost-optimized hybrid convolutional neural networks for detection of plant leaf diseases. Journal of Ambient Intelligence and Humanized Computing, 12(8), 8625-8636.
[38] Devi, R. S., Kumar, V. R., & Sivakumar, P. (2023). EfficientNetV2 Model for Plant Disease Classification and Pest Recognition. Computer Systems Science & Engineering, 45(2).
[39] Shrestha, G., Das, M., & Dey, N. (2020, October). Plant disease detection using CNN. In 2020 IEEE applied signal processing conference (ASPCON) (pp. 109-113). IEEE.
[40] Chauhan, R., Ghanshala, K. K., & Joshi, R. C. (2018, December). Convolutional neural network (CNN) for image detection and recognition. In 2018 first international conference on secure cyber computing and communication (ICSCCC) (pp. 278-282). IEEE.
[41] Enkvetchakul, P., & Surinta, O. (2022). Effective data augmentation and training techniques for improving deep learning in plant leaf disease recognition. Applied Science and Engineering Progress, 15(3), 3810-3810.
[42] Al-Shemarry, M. S., Li, Y., & Abdulla, S. (2019). An efficient texture descriptor for the detection of license plates from vehicle images in difficult conditions. IEEE transactions on intelligent transportation systems, 21(2), 553-564.
[43] "How to Build Mobile Apps with Flutter." Accessed: Dec. 05, 2023. [Online]. Available: https://www.freecodecamp.org/news/how-to-build-mobile-apps-with-flutter/
[44] Soujanya, K., & Jabez, J. (2021, October). Recognition of Plant Diseases by Leaf Image Classification Based on Improved AlexNet. In 2021 2nd International Conference on Smart Electronics and Communication (ICOSEC) (pp. 1306-1313). IEEE.
[45] Arya, S., & Singh, R. (2019, September). A Comparative Study of CNN and AlexNet for Detection of Disease in Potato and Mango leaf. In 2019 International conference on issues and challenges in intelligent computing techniques (ICICT) (Vol. 1, pp. 1-6). IEEE.
[46] Fuentes, A., Yoon, S., Kim, S. C., & Park, D. S. (2017). A robust deep-learning-based detector for real-time tomato plant diseases and pests recognition. Sensors, 17(9), 2022.
[47] Hlaing, C. S., & Zaw, S. M. M. (2017, December). Model-based statistical features for mobile phone image of tomato plant disease classification. In 2017 18th international conference on parallel and distributed computing, applications and technologies (PDCAT) (pp. 223-229). IEEE.
[48] Hlaing, C. S., & Zaw, S. M. M. (2018, June). Tomato plant diseases classification using statistical texture feature and color feature. In 2018 IEEE/ACIS 17th International Conference on Computer and Information Science (ICIS) (pp. 439-444). IEEE.
[49] Atole, R. R., & Park, D. (2018). A multiclass deep convolutional neural network classifier for detection of common rice plant anomalies. International Journal of Advanced Computer Science and Applications, 9(1).
[50] Ramcharan, A., McCloskey, P., Baranowski, K., Mbilinyi, N., Mrisho, L., Ndalahwa, M., ... & Hughes, D. P. (2019). A mobile-based deep learning model for cassava disease diagnosis. Frontiers in plant science, 10, 425916.
[51] Bi, C., Wang, J., Duan, Y., Fu, B., Kang, J. R., & Shi, Y. (2022). MobileNet based apple leaf diseases identification. Mobile Networks and Applications, 1-9.