Deep learning-based disease detection in peanut cultivars utilizing transfer learning

Kashif Mahmood
University of Sargodha
Qaisar Abbas
Islamic University of Madinah
Ume Habiba
University of Sargodha
Mahwish Ilyas
University of Rasul
Turki Alghamdi
Islamic University of Madinah
Hani Almoamari
Islamic University of Madinah
Muhammad Ramzan
mramzansf@gmail.com
University of Sargodha

Article

Keywords:

Posted Date: June 1st, 2026

DOI: https://doi.org/10.21203/rs.3.rs-9294795/v1

Additional Declarations: No competing interests reported.

Deep learning-based disease detection in peanut cultivars utilizing transfer learning

Kashif Mahmood [1], Qaisar Abbas [2], Ume Habiba [1], Mahwish Ilyas [3], Turki Alghamdi [2], Hani Almoamari [2], Muhammad Ramzan [1]

[1] Department of Software Engineering, University of Sargodha, Sargodha, 40100, Pakistan
[2] Faculty of Computer and Information Systems, Islamic University of Madinah, Madinah 42351, Saudi Arabia
[3] Department of Computer Science, University of Rasul, Mandi Bahauddin, 50370, Pakistan

Corresponding author: Muhammad Ramzan (email: mramzansf@gmail.com)

Abstract

Agriculture is a highly dynamic area that sustains global food security, and crop health plays a critical role in obtaining agricultural output. Peanuts, commonly known as groundnuts, hold a significant value due to their nutritional importance and economic benefits in many regions. However, it faces numerous challenges due to its vulnerability to a range of diseases that have a severe impact on both quality and yield. Disease detection methods based on traditional techniques were time-consuming, vulnerable to human mistakes and Labor-intensive. To handle these issues, we suggest a cutting-edge technique for the detection of peanut diseases using deep learning models. In this research work, we initially proposed some pre-trained deep learning models such as EfficientNet-B0, EfficientNet-B4, and ConvNeXt-Base, but none of them produced state-of-the-art results. To address this challenge, we leveraged the ResNet-50 Architecture using transfer learning enriched with the combination of advanced methodologies such as Data augmentation, OneCycleLR Learning rate scheduler, and weighted loss. This approach dramatically boosted the performance across various metrics, demonstrating the strength of transfer learning to handle imbalanced data and refine generalization. Deep learning models were trained with 1720 publicly available images of the dataset. The dataset includes both healthy and diseased images of groundnut leaves, including Alternaria leaf spot, Rosette, Rust and Leaf spot (early and late). The dataset was partitioned into 80% training images, 10% test images, and 10% value accuracy images. Our proposed methodology outperformed on the same dataset and gave better results than the older ones on the same dataset. The earlier same dataset had an accuracy rate of 96.51%. Our experiments showed that the suggested technique achieved a 97.28% accuracy rate and outperformed the existing state-of-the-art models. Results from these experiments demonstrate the merit of the proposed model for application in real-world agricultural problems, establishing a new baseline for detecting groundnut leaf diseases and establishing the feasibility of AI-based solutions for enhancing transferable sustainable agricultural practices.

Introduction

The word agriculture encloses a broad range of practices that domestic animals and crop plants use to support the global needs of the human population's sustenance and other essential products. Agriculture's critical demanding function has been widely acknowledged for a long time in development. The speed at which technological advancements are happening in the agricultural field has resulted in improved output and efficiency, ultimately enhancing economic expansion.

AI-powered solutions, such as using technologies like machine learning, deep learning, IoT-based systems, etc., will enable the farmers to generate more output with less input[1].

Arachis Hypogaea, commonly known as the peanut, is a highly acknowledged oilseed-based plant of the nuts family. In context of enriching soil, livestock food, and human food consumption with nitrogen, peanut serves as an oil plant [2]. Its kernels are rich in essential minerals and vitamins. Despite all these benefits of groundnut, farmers are facing problems in its cultivation due to several prevalent diseases. According to the estimation of the United Nation's Food and Agriculture Organization, annually between 20 and 40% of the world's crop production is destroyed due to infestations and diseases, even though roughly two million tons of pesticides are being used [3].

Peanut growth is significantly disrupted by the leaf diseases which can affect quality and yield in a negative manner. That's why peanut leaf diseases identification is of the utmost interest. The detection of groundnut diseases in time and accurately is of much importance.

Traditionally, techniques that were being used to detect peanut leaf diseases were manual. They required a lot of human effort and time. Due to the complex environmental conditions and poor distinction between peanut disease symptoms, it becomes hard to detect groundnut diseases by using conventional methods and reduces the quality and efficacy of crop yield. Evaluating the early-stage condition of leaf diseases can be challenging primarily because of an inadequate number of skilled personnel and research tools [4].

In recent times, there are several new techniques that are being used to address all these issues and identify the peanut diseases problems. Over the past few years, deep learning computational models have proven its increasingly crucial role in the diagnosis of crop leaf diseases and showed better performance than many other traditional methods [5].

In our research, we used several deep learning techniques to detect peanut diseases. These techniques have garnered a lot of attention and consideration as a result of their promising and massive outputs in various applications, such as the classification and identification of groundnut leaf diseases. Several models, such as EfficientNet-B0, EfficientNet-B4, ConvNeXt-Base, and a technique of transfer learning using ResNet-50 architecture are used to effectively evaluate the groundnut leaf diseases.

We used a dataset of total 1720 images including 5 classes as Alternaria Leaf spot, Healthy, Rust, Leaf spot (early and late) and Rossette.

In our paper, our main work is to identify and detect groundnut diseases using some deep learning algorithms. Following are the few contributions of this study's proposed work:

- We assessed various pre-trained deep learning models such as EfficientNet-B0, EfficientNet-B4 and ConvNeXt-Base as baseline models for disease classification, demonstrating the shortcomings of pre-trained models in handling the imbalance dataset.
- To enhance the performance, we utilized the technique of transfer learning via ResNet-50 architecture. The fine-tuned model performed significantly and surpassed all baseline approaches, demonstrating its capability to extract discriminative features for disease detection.
- We have overcome the significant issue of class unbalancing by dynamically computing the class weights using inverse frequency method and incorporating weights

in the CrossEntropyLoss function. This ensures that the model focus appropriately on overlooked classes.

- To culminate the model's generalization and resilience, we implemented an array of augmentation techniques including Vertical and Horizontal Flipping, RandomAffine, ColorJitter, and RandomErasing.
- We employed the OneCycleLR scheduler which adjusts the learning rate dynamically during training which leads to rapid convergence and enhanced model generalization over conventional decay methods.
- ResNet-50 architecture utilizing transfer learning gave the best outcomes in our experimentation and showed better accuracy than the previous experimentation on the same publicly available dataset.

The subsequent sections of this study will be structured as follows: In section 2, we will be reviewing the relevant literature. Section 3 will provide detailed explanation of the models being utilized in the research work and their detailed methodology. Detailed description of the dataset being used in the paper will be given. Last section will be concluding the paper by providing all the results of the models being used.

Literature review of existing papers on peanut diseases

Peanut diseases are no different than other plant diseases that are being notably affected, and they represent a great challenge for sustainable agricultural production. Numerous studies have been done in the domain of groundnut disease detection. In this section, various studies conducted on the detection of peanut disease are presented. As a result, the main existing papers in the most related work in this domain are mentioned below under the sections of deep learning and machine learning papers.

Existing work using deep learning models

Usually, deep learning is employed to use neural networks with more than one hidden layer to classify and find regions of interest in an image, for example, plant disease symptoms. It builds an end-to-end system that processes leaf images and predicts the likelihood of a leaf being diseased with specific diseases.

Sasmal et al [6] designed a dataset containing a total of 1720 original groundnut images. The dataset included the peanut images as Healthy, Alternaria leaf spot, Rust, Leaf spot (early and late), and Rosette, which were carefully obtained from West Bengal, India. They utilized several models as InceptionV3, DenseNet201, AlexNet, and ResNet50 and gained the highest accuracy of 96.51%. Urbano et al [7] used deep learning techniques such as VGG16, VGG19, MobileNet, DenseNet, Xception, InceptionResNetV2, InceptionV3, and ResNet50 to identify peanut leaf spot diseases. Researchers used the dataset, including 1000 images, to identify leaves having leaf spot diseases. They achieved the highest accuracy of 98%. Anbumozhi et al [8] primarily focused on the identification of self-collected dataset under several environmental conditions. They proposed the Progressive Groundnut Convolutional Neural Network (PGCNN) model for the disclosure of leaf diseases at the early stages. They gained an overall accuracy of 97.58%. Paramanandham et al [9] suggested the LeafNet model to identify and detect the peanut leaf diseases. Their dataset included the total 10,361 images, including the 6 classes of diseased as well as healthy images. The testing accuracy obtained by them was 97.225%. Aishwarya et al [10] performed their work on a self-created dataset. They used the Tri-CNN ensemble architecture models, such as DenseNet169,

Inception, and Xception models. Their proposed model performed well on the real-world peanut leaf dataset and achieved the accuracy rates of 98.46%. Rakholia et al [11] proposed a deep learning model in which progressive resizing was effectively performed to identify peanut leaf disease and for the classification work. They involved the five categories of the peanut leaf diseases dataset. In their paper, accuracy obtained was 96.12%. Chen et al [12] suggested two methods of imaging such as push-boom FX10 and Snapshot. Peanut issues were detected by using the deep learning models. Their dataset included total 1560 images and attained the overall maximum accuracy of 98.5%. Wang et al [13] used the faster RCNN method. In their work, dataset including 1300 sets was collected to perform the experimentations. Lv et al [14) collected the dataset consisting of 700 images of peanut rust diseases, scorch spot, and leaf spot for the identification of peanut leaf diseases. They employed YOLOX-Tiny network for the experimentation purposes of groundnut leaf diseases and achieved the enhanced accuracy. Sivaganesan et al [15] proposed various CNN models as VGG-16, EfficientNet, and MobileNet for the identification of the leaf diseases. Rajmohan et al [16] used the peanut images dataset to detect the peanut leaf diseases at the early stages. They employed the convolutional neural network and k-means clustering methodology to correctly detect the infected images. They gained the accuracy of 93%.

"Table 1" is used to represent the literature review of existing papers of Groundnut diseases based on deep learning models.

Table 1: Existing papers of peanut diseases based on deep learning techniques

Reference	Year	Dataset	Models
[6)	2024	1720 images	Deep learning models
[7]	2022	1000 images	VGG16, Xception, VGG19, MobileNet, InceptionV3, DenseNet, ResNet50, and InceptionResNetV2
[8]	2023	Self-collected	Progressive Groundnut Convolutional Neural Network (PGCNN), VGG Models, AlexNet, CNN
[9]	2024	10,361 images	LeafNet
[10]	2023	Self-collected	tri-CNN ensemble model as DenseNet169, Inception, and Xception models
[11]	2022	Manually created	Deep learning model
[12]	2023	1560 peanuts images	Deep learning
[13]	2023	1300 sets	Faster RCNN
[14]	2025	700 images	YOLOX-Tiny network
[15]	2023	Groundnut images	different CNN models, such as MobileNet, EfficientNet, and VGG-16
[16]	2022	Peanut leaf images	CNN model and k-means method

Existing work using machine learning models

Gowrishankar et al [17] proposed the machine learning models support vector machine (SVM), and such as artificial neural network (ANN) for the examination of peanut leaf diseases. They used

the groundnut images and integrated several image processing methods for detection purposes. In their experiments, they found out that ANN performed better than SVM. Zou et al [18) utilized a dataset of total 600 images including the five kinds of groundnut for the identification of peanuts mildew. They employed numerous machine learning models as GBDT (Gradient Boosting Decision Trees), XGBoost (Extreme Gradient Boosting), LightGBM (Light Gradient Boosting Machine), and CatBoost in their work. At the end, LightGBM (Light Gradient Boosting Machine), proved to be the best among all showing the identification rate of 99.10%. Existing papers on Peanut diseases based on machine learning models are outlined in "Table 2".

Table 2: Existing papers of peanut diseases based on machine learning techniques

Reference	Year	Dataset	Models
[17]	2020	Groundnut Images	Machine learning→ ANN, SVM
[18]	2022	600	Several Machine learning models as XGBoost, GBDT, LightGBM CatBoost

Proposed research methodology:

The knowledge of the system under study is essential for the formulation of the methodology for the research, and this is why the proposed research methodology section is a vital tool in sketching the systematic approach that will be adopted in order to attain the study's objectives. It provides a detailed blueprint for the procedures, tools and techniques that will be used to gather and analyze data to entice with research that has been conducted in a rigorous and transparent manner. The complete description of data, suggested models and their detailed overview, and performance measures are included in this section in order to secure the credibility and validity of the research. "Fig 1" is representing the overall methodology of our proposed models used in our research work.

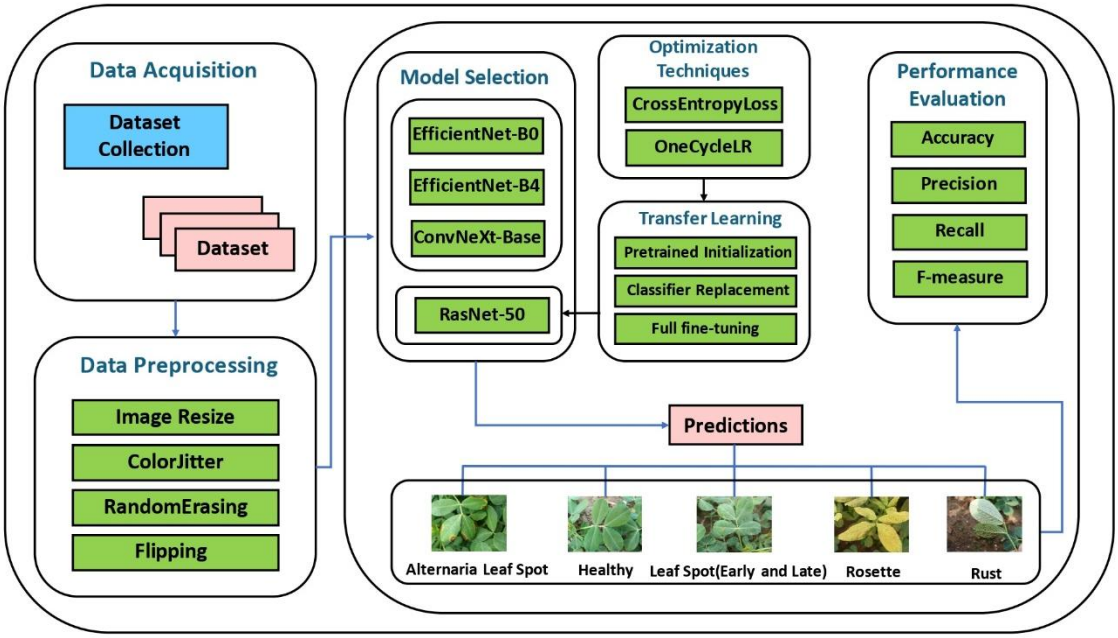


Fig 1. Proposed architecture workflow of peanut diseases detection

Data acquisition for our work

A publicly available dataset [6] was used for research purposes. In the original dataset, groundnut leaves that were diseased with Leaf spot (early and late), Rust, Alternaria leaf spot, Rosette, and Healthy had been collected from agricultural land and brought for analysis. The dataset holds 1,720 JPG files, all in the pixel format, together with the associated image number and names. The healthy data, Leaf spot (early and late) images, Alternaria leaf spot images, Rust and Rosette diseases images data was respectively uploaded to 5 different folders; healthy data, Leaf spot (early and late), Rossette, Alternaria leaf spot, and Rust. Each folder contained further 3 folders indicating as test, train, and value accuracy images for each disease. Images were divided as 80% training, 10% testing, and 10% value accuracy images. "Fig 2". Include the five classes of peanut images present in our dataset.

Fig 2. Showing the images included in our dataset

Additionally, the name of each folder was the same as the corresponding image class. The Healthy class directory contains a collection of groundnut leaf images in their healthy state. The Leaf Spot (early and late) folder contains the photos of groundnut leaves exhibiting the symptoms of leaf spot (early and late). The folder named Alternaria Leaf Spot includes the images of groundnut leaves that are infected by Alternaria leaf spot. Images of groundnut leaves affected by Rust disease are contained in the Rust named folder. The Rosette folder holds the visuals of peanut foliage affected by Rosette disease. DISTINCT directories were set up for the experimental procedures on image data for loading the data. The most commonly occurring diseases are considered for the procedure of leaf disease identification. Complete details of the dataset used in our research work is demonstrated in "Table 3".

Table 3: Description of used dataset in detail

Class name	Number of images	Images in train folder	Images in test folder	Images in value accuracy folder
Healthy	600	480	60	60
Leaf spot (Early and Late)	450	360	45	45
Alternaria Leaf Spot	450	360	45	45
Rust	120	96	12	12
Rossette	100	80	10	10
Total	1720	1376	172	172

In all these classes different number of images are present. We carefully curated all these images dataset in the separate folders to avoid any further issues. A visual overview of the dataset in shown below in the “Fig 3”.

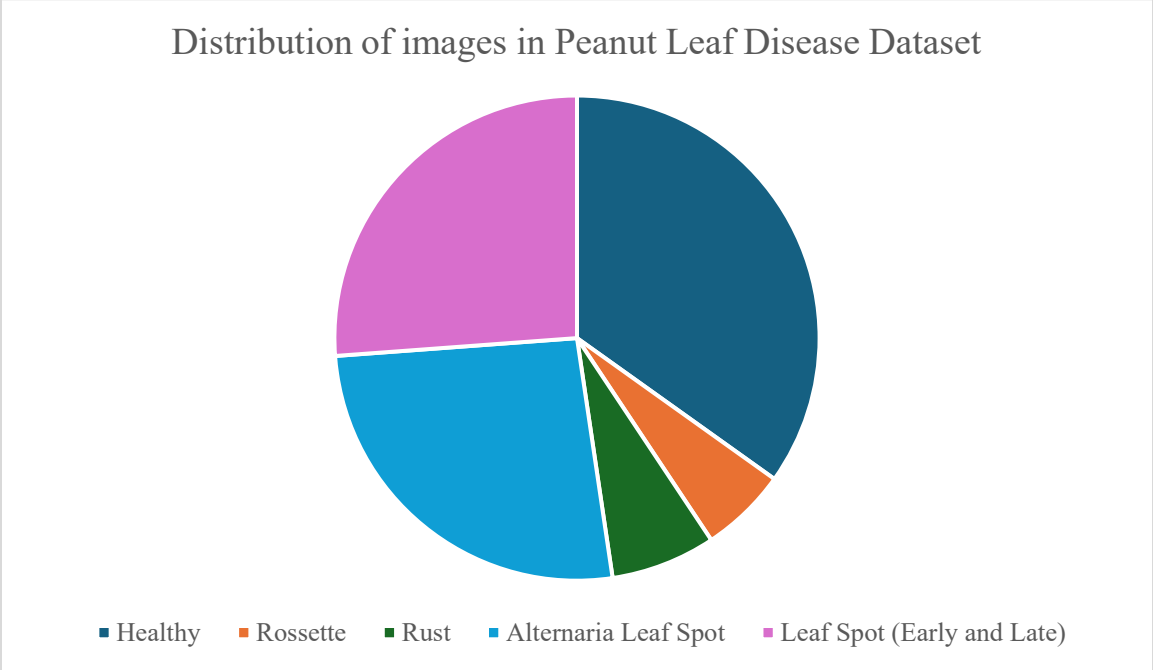


Fig 3. Visual representation of dataset

Data pre-processing

It is very important to mention that image preprocessing in DL is very important, as it increases model efficiency and accuracy of the model. The dataset taken for research was adopted from a publicly available source that had already done the necessary preprocessing steps on it. The dimensions of the images were also made uniform by first processing the images to give a uniform dimension. The non-square images were cropped into square dimensions, including the full leaf area. To better analyze further, all images were uniformly scaled to 224 × 224 pixels. Besides, augmentation techniques on images, including scaling, cropping, rotation, flipping, and adding noise, were considered to enhance the diversity and quantity of the training dataset; however, in this study, these techniques were applied because the dataset for evaluating the performance of a deep learning model would suffice. Instead, with this approach we were able to begin training and evaluation of models without any additional preprocessing. Images were given to the pre-trained deep learning model after the application of preprocessing techniques.

Data augmentation and regularization

To combat overfitting and boost generalization, we employed an array of data augmentation Techniques such as Vertical Flipping, ColorJitter, RandomAffine and RandomErasing. These transformations introduced diversity in scale, color and orientation, enabling the model to perform better in real-world scenarios

Augmentation techniques

The applied augmentation techniques are as follows:

- Random erasing involves eliminating the rectangular regions from the image, simulating occlusions and strengthening the model's spatial robustness
- ColorJitter Stochastically alters the image properties, including saturation, color, hue and brightness, enabling the model to perform under diverse illumination conditions in real-world scenarios.
- RandomAffine incorporates affine transformations like rotation, scaling and translation to enhance models' capability to generalize across various topological arrangements.
- Vertical flipping is applied to input data with a specific probability, enhancing the model's capability to become robust regardless of directional and orientation-related anomalies in real-world scenarios.

Leveraging deep learning models for semantic analysis

To extract semantic features from input images, initially, we utilized an array of pre-trained convolutional neural network (CNN) architectures, but as these models did not meet expectations, we employed transfer learning using the ResNet-101 architecture to improve both feature extraction and classification accuracy. The models employed in this study included:

- EfficientNet-B0
- EfficientNet-B4
- ConvNeXt-Base
- ResNet-101 (with transfer learning)

Transfer learning via ResNet-50

Among all the evaluated architectures, ResNet-50 with transfer learning performed state-of-the-art. The following transfer learning methodology is implemented as:

- The model parameters were initialized with ImageNet weights, enabling the network to benefit from semantically rich and transformable features.
- The original fully connected layer was substituted with a new custom output layer, developed particularly for the target disease classes.
- Fine-tuning was applied across the whole network to optimize the high-level features for the specific task of classifying the disease classes.

Training strategy

To overcome the challenges caused by class imbalance and sensitivity of the Learning rate, we leverage the following strategies:

Weighted loss function

We implemented CrossEntropyLoss with class weights computed dynamically through inverse class frequencies by compute_class_weight function of scikit-learn [22], ensuring the greater contribution of under-represented classes during training, which mitigates the effect of class imbalance.

$$w\_c = N / (K * n\_c) \quad [1]$$

Where N denotes the total sample count, K represents the number of classes, and n_c indicates the number of samples belonging to class c.

The final loss function is:

$$L = -\sum w\_\{y\_i\} * log(\hat{y}\_\{i, y\_i\}) \quad [2]$$

OneCycleLR scheduler

We employed the OneCycleLR Scheduler [23] to speed up the convergence and eliminate the need for manual learning rate tuning. It dynamically adjusts the learning rate during training by initially boosting it to a peak value and then reducing it gradually. The schedule is typically defined as:

$$\eta(t) = \eta\_min + (\eta\_max - \eta\_min) * (t / T\_up), \text{ if } 0 \leq t < T\_up \quad [3]$$

$$\eta\_max - (\eta\_max - \eta\_final) * ((t - T\_up) / T\_down), \text{ if } T\_up \leq t \leq T$$

Where η_max and η_min represent the maximum and minimum learning rates, respectively. η(t) denotes the learning rate at time step t, while η_final is the final learning rate at the training. T represents the total number of training steps, and t is the current training step. T_up and T_down correspond to the warm-up and cool-down durations, respectively.

Feature extraction and classification

In our research work, we proposed and applied five deep learning pre-trained models as EfficientNet-B0, EfficientNet-B4, ConvNeXt-Base, and ResNet-50 with transfer learning:

EfficientNet-B0 model structure

Mingxing Tan and Quoc V. Le, authors of the paper "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks," proposed EfficientNet-B0 [19]

In that respect, EfficientNet is a completely different way of exploring the limits of convolutional neural networks (CNNs): models only scaled in either depth, resolution or width, none combined at once. However, there was no systematic methodology in this approach. In order to solve this problem, Google Brain researchers implemented the EfficientNet model, which used a composite scaling method. It enhances all three dimensions in proportion to obtain the optimal results by using a set of scaling coefficients that are defined. The EfficientNet-B0 is the lightest architecture of the family and has fewer parameters and lower computational cost, which enables it to perform competitively on resource-constrained devices with relatively high accuracy.

The EfficientNet-B0 consists of one Stem layer, multiple Inverted Residual Bottleneck (IRB) blocks, one Head layer and one fully connected layer. Specifically:

- In stem layer, the image has been convolved with a 3x3 convolution and a stride value of 2 and a padding value of 1 to output 32 channels. The initial feature extraction is taken care of by this layer.
- 8 Inverted Residual Bottleneck (IRB) blocks total are used, each having a depth-wise separable convolution, expansion layer, 3x3 deeper convolution and squeeze & excitation (SE) module. On the other hand, each block has a varying number of channels, expansion ratio, and stride. The first IRB block is an example of 16 channels, expansion ratio of 1, stride of 1 and containing an SE module. “Table 4” provides the detailed structure of the IRB blocks.

- After 1 × 1 convolution with 1280 channels and 7 × 7 adaptive average pooling layers to down sample the size of the feature map, the outcomes of the IRB blocks will pass through this stage called head layer
- The output from the Head layer is flattened and fed to a dense neural layer (fully connected layer) with 1000 neurons (that can be a number of classes for a particular task), and the outputs generate classification results as the final result.

Table 4: Structural overview of EfficientNet-B0 model

Stage	Operator	Resolution	#Channels	#Layers
1	Conv3x3, s=2	224 × 224 → 112 × 112	32	1
2	MBConv1, k3x3, s=1	112 × 112	16	1
3	MBConv6, k3x3, s=2	112 × 112 → 56 × 56	24	2
4	MBConv6, k5x5, s=2	56 × 56 → 28 × 28	40	2
5	MBConv6, k3x3, s=2	28 × 28 → 14 × 14	80	3
6	MBConv6, k5x5, s=1	14 × 14	112	3
7	MBConv6, k5x5, s=2	14 × 14 → 7 × 7	192	4
8	MBConv6, k3x3, s=1	7 × 7	320	1
9	Conv1x1 & Pooling & FC	7 × 7 → 1 × 1 → 1 × 1	1280 → 1000	1

The key to EfficientNet is its compound scaling method, which simultaneously scales the width, depth, and resolution of the network using fixed scaling coefficients. The formula is:

$$\phi = \alpha \cdot \beta^2 \cdot \gamma^2 \approx 2 \quad (4)$$

The scaling parameters for depth, width and resolution are designated here to be α, β and γ, respectively. To compute α = 1.2, β = 1.1, and γ = 1.15 using the standard network EfficientNet-B0 (Φ = 1), the authors performed a grid search. Thus, with these coefficients fixed, different Φ values are used to scale EfficientNet-B0 into a family of neural networks: EfficientNet-B1 to B7.

Mobile inverted bottleneck MBConv blocks with squeeze and excitation optimization are used in baseline network, mathematically expressed as:

$$MBConv(x) = SE(DepthwiseConv(ExpandConv(x))) + x \quad (5)$$

Where the expansion convolution increases channel dimensionality before depthwise spatial filtering, followed by channel-wise feature recalibration through squeeze-excitation (SE) layers. EfficientNet-B0's architectural efficiency stems from its optimal ϕ=1 configuration, achieving 77.1% ImageNet accuracy with only 5.3M parameters through this balanced scaling approach.

EfficientNet-B4 model structure

EfficientNet-B4 was first proposed by Mingxing Tan and Quoc V. Le in the paper "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks"[19)

Prior to EfficientNet, known scaling of CNNs mostly came via increasing depth, width, or resolution individually. However, this process had no systematic methodology. In order to address this issue, Google Brain researchers introduced EfficientNet model family based upon a compound scaling method. It scales in all three dimensions proportionally using defined scaling coefficients in order to get optimal performance without separately increasing depth, width or resolution. The EfficientNet family has medium sized model named as EfficientNet B4 which is an adjustment in between the accuracy and computation efficacy. It was trained using a multi-objective neural architecture search with the same aim as MnasNet, of optimizing accuracy as well as FLOPs.

Specifically, EfficientNet-B4 contains a Stem layer, multiple Inverted Residual Bottleneck (IRB) blocks, a Head layer, and a fully connected layer.

Specifically, the architecture is as follows:

- The first layer, known as stem layer, contains a 3×3 convolution with a stride value of 2, padding value of 1 on input image to get 48 channels, which is output from Stem Layer. It is this layer which is the first to take care of the initial feature extraction.
- The total number of IRB blocks in the IRB layer is 14, each containing a (depthwise separable) convolution layer, an expansion layer, a (depth-wise) 3 × 3 convolution layer, and a squeeze-and-excitation (SE) block. Across blocks, the number of channels, expansion ratio and stride are different. For instance, the first IRB block has 24 channels, an expansion ratio of 1, a stride of 1 and has an SE module.
- Output of these blocks is then fed to a 1×1 convolution of 1792 channels, often called head layer and 7×7 adaptive average layer of pooling to remove the spatial dimension of feature maps.
- This means that the output of the Head layer is flattened and passed to a dense neural layer with 1000 neurons for a final classification, since we will be using 1000 neurons as per the task.

Table 5: Layered structure of EfficientNet-B4 model

Stage	Operator	Resolution	#Channels	#Layers
1	Conv3x3, s=2	380 × 380 → 190 × 190	48	1
2	MBConv1, k3x3, s=1	190 × 190	24	1
3	MBConv6, k3x3, s=2	190 × 190 → 95 × 95	32	2
4	MBConv6, k5x5, s=2	95 × 95, 48 × 48	56	2
5	MBConv6, k3x3, s=2	48 × 48 → 24 × 24	112	3
6	MBConv6, k5x5, s=1	24 × 24	160	3
7	MBConv6, k5x5, s=2	24 × 24 → 12 × 12	272	4
8	MBConv6, k3x3, s=1	12 × 12	448	1

9	Conv1x1 & Pooling & FC	12 × 12 → 1 × 1 → 1 × 1	1792 → 1000	1

Its compound scaling method comprises scaling in network (width, depth, and resolution) simultaneously with defined coefficients of scaling being the core of EfficientNet.

The formula is:

$$\phi = \alpha \cdot \beta^2 \cdot \gamma^2 \approx 2 \quad (6)$$

Here, α, β, and γ indicates the scaling factors for depth, width, and resolution, respectively. Based on the baseline network EfficientNet-B0 (Φ=1), the authors determined α=1.2, β=1.1, and γ=1.15 through grid search. With these coefficients fixed, different Φ values are used to scale EfficientNet-B0 to derive a family of neural networks, EfficientNet-B1 to B7. EfficientNet-B4 corresponds to Φ=4.

The total FLOPS increase by a factor of (α·β2·γ2]ϕ when scaling the network.

EfficientNet-B0 model's architecture contain seven blocks and the most important building block of this model is MBConv, which is a mobile inverted bottleneck convolution [19). EfficientNet-B4 model scaled up from the EfficientNet-B0 model by using the method of compound of compound scaling. This model is having more depth and its blocks are very deep [19].

Below "Fig 4" is showing the basic layered framework of the EfficientNet model.

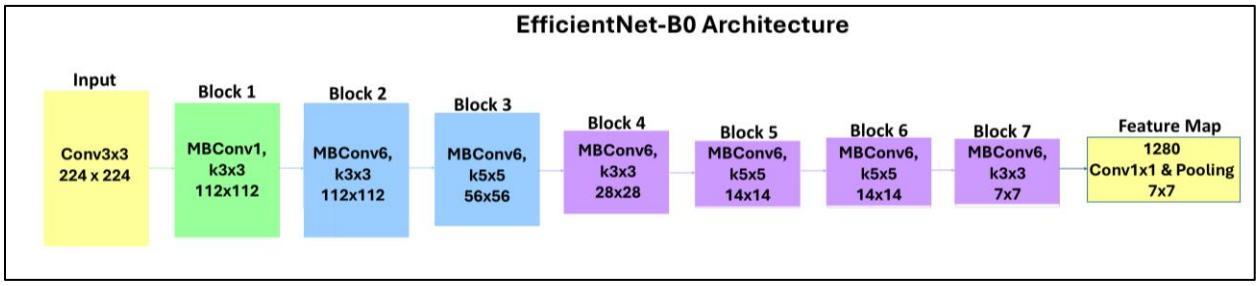


Fig 4. Layered architecture of EfficientNet-B0 model

ConvNeXt-Base model structure

The ConvNeXt model was first proposed by Liu et al. in the paper "A ConvNet for the 2020s"[20] published in 2022.

Convolutional neural networks (CNNs) have been profitably applied to the functions of classifying images over the recent years, due to the improvements and advancements of deep learning. But in the last recent years, ViTs have also shown good performance in many vision tasks. Based on ViTs, the authors of ConvNeXt reinvented the design space of CNNs and proposed the ConvNeXt model. Inherit core ideas of CNNs and some design ideas from ViTs, like hierarchical feature maps and self attention mechanisms, ConvNeXt is the model. This achieves competitive performance with the advantages of CNNs in terms of computational efficiency and interoperability. Four stages with each stage including multiple blocks are the components of ConvNeXt-Base.

The full architecture is as follows:

- The first stage has 4×4 convolution of the input image with a stride 4 thus resulting in 96 channels output. It is the first feature extraction stage.

- The stage 2 contains 3 blocks. The block is each 7×7 depthwise convolution followed with a LayerNorm module and 1×1 pointwise convolution and another 1×1 pointwise convolution with GELU. There are 192 channels in this stage.
- 9 blocks are present in this stage, and the structure is same as of the blocks in the Stage 2. This stage comprises 786 channels.
- The output of stage 4 is the full classification results.

The model employs a 4-stage hierarchical structure with progressive downsampling through strided convolutions:

$$Stage_i\,(x) = ConvNeXtBlock^{N_i}\,(Patchify(x)) \quad (7)$$

Where N1=3N1=3, N2=3N2=3, N3=27N3=27, N4=3N4=3 blocks per stage, and patchify layers use 4×4 kernels for spatial reduction.

Each ConvNeXt Block implements:

$$x_{out} = x + DW(Conv(GELU(LayerNorm(MLP(x))))) \quad (8)$$

Utilizing 7×7 depthwise convolutions for large receptive fields and inverted bottlenecks with 4× expansion ratios. The base configuration contains 88M parameters, achieving 84.1% ImageNet accuracy through its macro design that mimics Swin Transformers' stage compute distribution.

ResNet-50 model structure

The ResNet-50 model was proposed by Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun in their influential paper titled “Deep Residual Learning for Image Recognition” [24) published in 2015. ResNet introduced the concept of residual learning to combat the issue of performance degradation in neural networks, where extended depth tends to perform worse than shallower ones. The skip connection is the key innovation, facilitating gradients to flow smoothly during backpropagation and facilitating the training of extremely deep neural networks. ResNet-50 is one of the most popular variants of the ResNet family, comprising 50 layers built using residual blocks with bottleneck design to boost computational efficiency. The architecture of ResNet-50 can be split into different stages, each consisting of a specific number of Bottleneck Residual Blocks. Each block consists of 1×1, 3×3, and 1×1 convolutions and utilizing identity or projection shortcuts. Conv1 is the first convolution layer with a 7×7 kernel, followed by max pooling. Conv2_x to Conv5_x are residual block stages that gradually increase the number of filters while reducing spatial resolution. Average Pooling is global average pooling after the final block. The Fully Connected Layer is the dense layer with 1000 units (can be customized for a specific number of classes). These are depicted below in “Table 6”:

Table 6: Layered structure of ResNet-50 model

Stage	Block Type	Output Size	Residual Blocks	#Channels
Conv1	7x7, 64, stride 2	112x112	1	64
	3x3 max pool, stride 2	56x56	-	-
Conv2_x	[1x1, 64], [3x3,64], [1x1, 256] x 3	56x56	3	64 → 256
Conv3_x	[1x1, 128], [3x3,128], [1x1, 512] x 4	28x28	4	128 → 512

Conv4_x	[1x1, 256], [3x3,256], [1x1, 1024] x 6	14 x 14	6	256 → 1024
Conv5_x	[1x1, 512], [3x3,512], [1x1, 2048] x 3	7 x 7	3	512 → 2048
AvgPool	Global Average Pooling	1 x 1	-	-
FC	Fully Connected (Dense)	1 x 1	1	1000 (or custom classes)

The residual block allows the neural network to capture the residual mapping rather than the entire transformation. This process simplifies the optimization process. The structure of a residual block is represented as:

$$b = F(a,\{Wi\}) + b \quad (9)$$

Where a serves as the input to the block, F(x,{Wi}) is the residual function to be learned (typically two or three stacked convolutional layers), and the output of the block is represented by b.

If the input and output dimensions differ, a linear projection Ws is applied to match the dimensions:

$$B = F(a,\{Wi\}) + Wsa \quad (10)$$

This identity mapping enables smoother gradient propagation and makes training deeper networks more feasible.

Sections 3.3, 3.4 and 3.5 of the methodology outline the in-depth explanation of the methodologies used to enhance the architecture of ResNet for our disease classification task, including the transfer learning, OneCycleLR Scheduler, along with class rebalancing using weighted CrossEntropyLoss.

Performance measures

Conventional methods for estimating deep learning classifiers rely on confusion metrics that compare the rock-bottom ground truth in the dataset with the predictions of model, specifically using TP, TN, FP, and FN to represent true positives, true negatives, false positives, and false negatives, respectively.

Accuracy

Accuracy of a model is how accurate it is at spotting the outcome. The total correct outcomes can be divided by the total predicted to get accuracy.

The fraction proportion of the outcomes that are successfully categorised as follows expressed using the equation:

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \quad [11]$$

Precision

Precision is measured in a way that true positive is the number of instances, for which predicted value by the model is positive, and it is correct; and total positive instances, which includes wrong and correct predicted positive values. This is represented as:

$$Precision = \frac{TP}{TP + FP} \quad [12]$$

F-measure / F-score

F-measure (also known as F1 score) balances Precision and Recall by taking their harmonic mean. It provides a unified metric to evaluate the model when Precision and Recall both are equally important and the class distribution is uneven.

$$F = \frac{2X\ precision\ \times recall}{precision + recall} \quad [13]$$

Recall

Recall or sensitivity (also known as sensitivity or actual positive rate) is used to measure the accuracy of a model in identifying actual positive instances.

$$R = \frac{TP}{TP + FN} \quad (14)$$

Results and discussion

In the section below, a comprehensive analysis of the results is provided which is obtained by using these four models: EfficientNet-B0, EfficientNet-B4, ConvNeXt-Base, and ResNet-50 (with transfer learning). Each model is being evaluated across various parameters as accuracy, precision, recall, and F-measure.

Results obtained by using EfficientNet-B0 model

With an accuracy of 87.50%, EfficientNet-B0 is azure by an overall precision of 92.13%, but lacks recall at 78.88% and F1-score of 81.21%.

Classification report metrics of EfficientNet-B0 model

The performance across different classes is quite variable in a classification report of EfficientNet-B0, shown in “Fig 5”. The model has a very high precision (92.13%) however its recall (78.88%) is extremely low, which is 81.21% according to F1 score. More specifically, the 'Healthy' class has a remarkable precision (95.6%) as well as recall (98.3%) of healthy samples, which shows strong performance on healthy samples. Despite such problems, the recall of the ‘Leaf Spot (Early and Late)’ class [17.5%) is poor and indicates that early-stage leaf spot conditions cannot be easily identified. Recall is 37.5% and precision is 30.0%, due at least in part to the visual similarity to other stress induced patterns of imagery that belong to the 'Rosette' class.

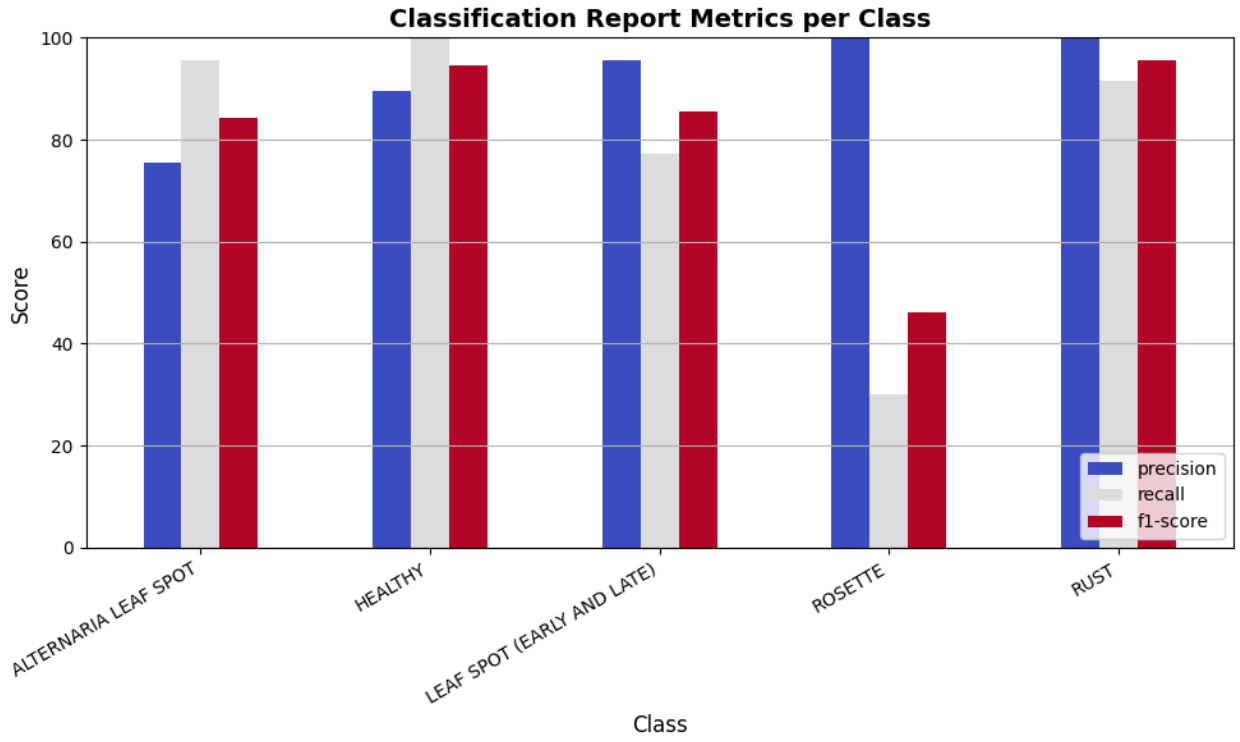


Fig 5. Classification report using EfficientNet-B0 model

Confusion matrix using EfficientNet-B0 model

"Fig 6". Further shows these challenges in the confusion matrix. Since the model classifies correctly for most samples, and the diagonal dominance shows it, the model mistakenly classifies "Rosette" with "Alternaria Leaf Spot" (6 misclassifications), and "Healthy" with "Leaf Spot (Early and Late)" (also 6 misclassifications). The lower recall in problematic classes is impact by these errors.

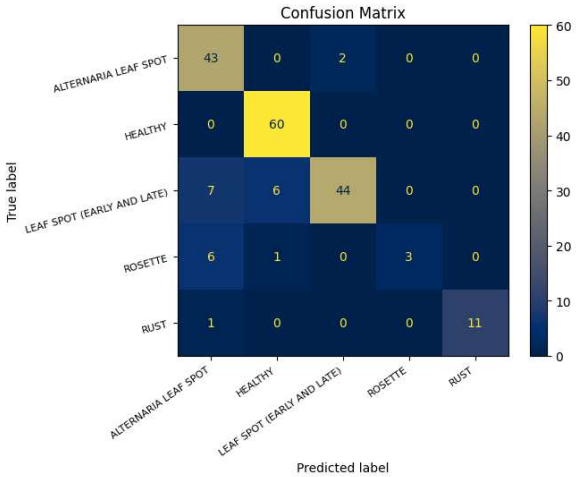


Fig 6. Confusion matrix using efficientnet-b0 model

Loss dynamics related to EfficientNet-B0 model

As shown in “Fig 7” (a), there is a steady decline of training loss up to the 20th epoch, and it converges at around 0.85. The fact it is effective learning during training indicates it is effective learning. But, although the validation loss (not shown) is slightly up after Epoch 15, it indicates a small amount of overfitting.

Training vs. validation accuracy using EfficientNet-B0 model

The accuracy curves in “Fig 7” (b) shows 92.13% training accuracy and 87.50% validation accuracy. This will confirm that overfitting, since the model memorizes training patterns and generalizes badly on validation data.

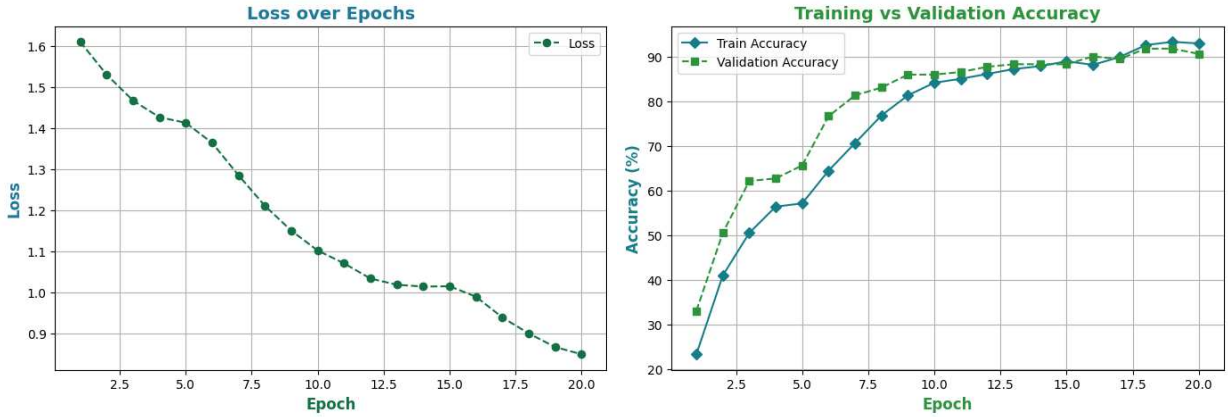


Fig 7. Loss dynamics using EfficientNet-B0 model (a) showing the loss dynamics (b) showing the training vs. validation accuracy

Results obtained by using EfficientNet-B4 model

EfficientNet-B4 model represented the 91.85% accuracy, precision of 74.61%, recall as 78.06%, and F1-score of 76.27%.

Classification report metrics of EfficientNet-B4 model

The performance of different classes is shown in the “Fig 8”. It can be observed in the figure that rust class showed the exceptional performance on the metrics of precision, recall, as well as f1-score.

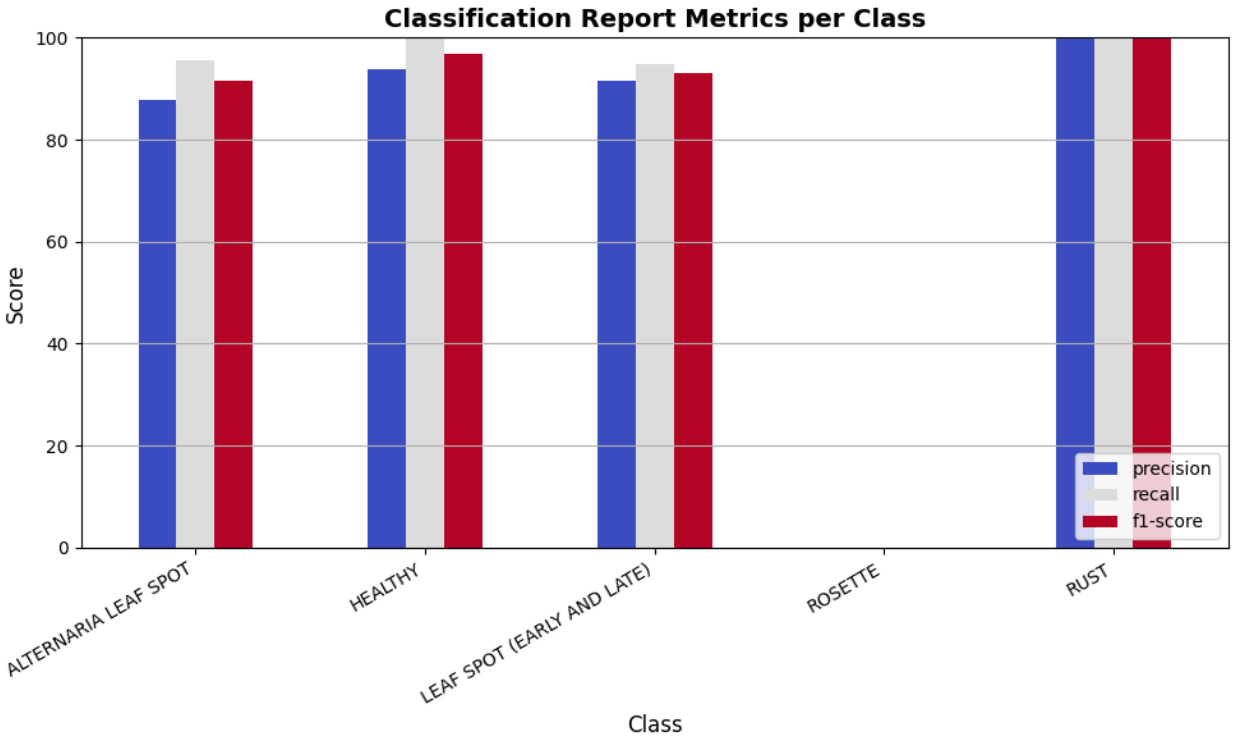


Fig 8. Classification report using EfficientNet-B4 model

Confusion matrix using EfficientNet-B4 model

"Fig 9" Illustrate the confusion matrix of all the classes using the EfficientNet-B4 model. This confusion matrix will provide a deeper analysis of the proposed model's performance on the provided dataset.

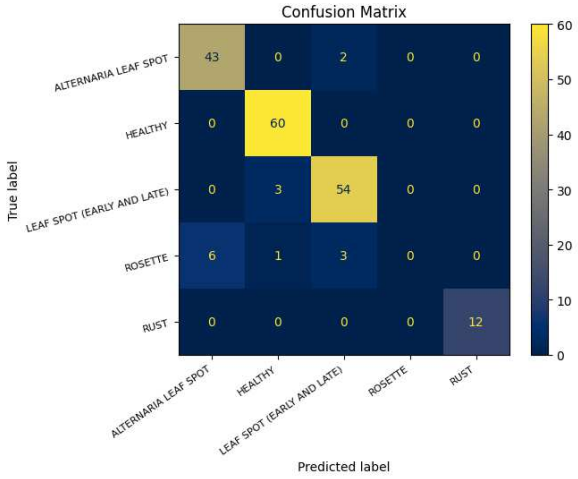


Fig 9. Confusion matrix using EfficientNet-B4 model

Loss dynamics related to EfficientNet-B4 model

In "Fig 10" (a), it can be seen that there is a stabilization over the 0.95. These loss metrics help to understand more about the used model in the research paper.

Training vs. validation accuracy using EfficientNet-B4 model

The training accuracy of 94.7% versus the 91.85% validation accuracy is shown in the “Fig 10” (b). There is an overfitting confirmation as the gap is about 2.85% and that can be due to the deeper structure of model’s architecture.

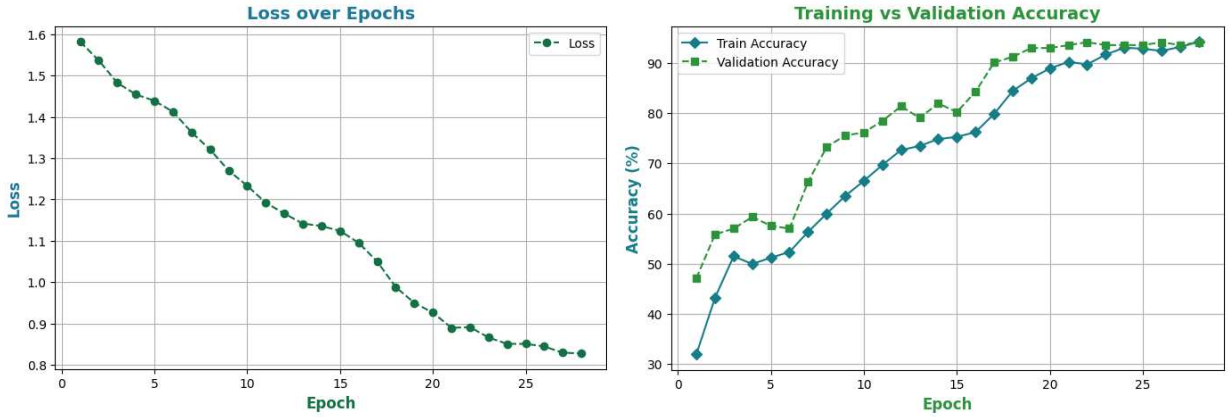


Fig 10. Loss dynamics using EfficientNet-B4 model (a) showing the loss dynamics (b) showing the training vs. validation accuracy

Results obtained by using ConvNeXt-Base model

Accuracy shown by ConvNeXt-Base model is 96.20%. The score of other matrices, such as precision, recall, and F1-score are 97.61%, 87.65%, and 90.02% respectively.

Classification report metrics of ConvNeXt-Base model

The classification report clearly shows that various classes performed well using the ConvNeXt-Base model. “Fig 11”. Demonstrates the results of all the classes as Healthy, Alternaria leaf spot, Leaf spot (early and late), rust, and rosette using the ConvNeXt-Base.

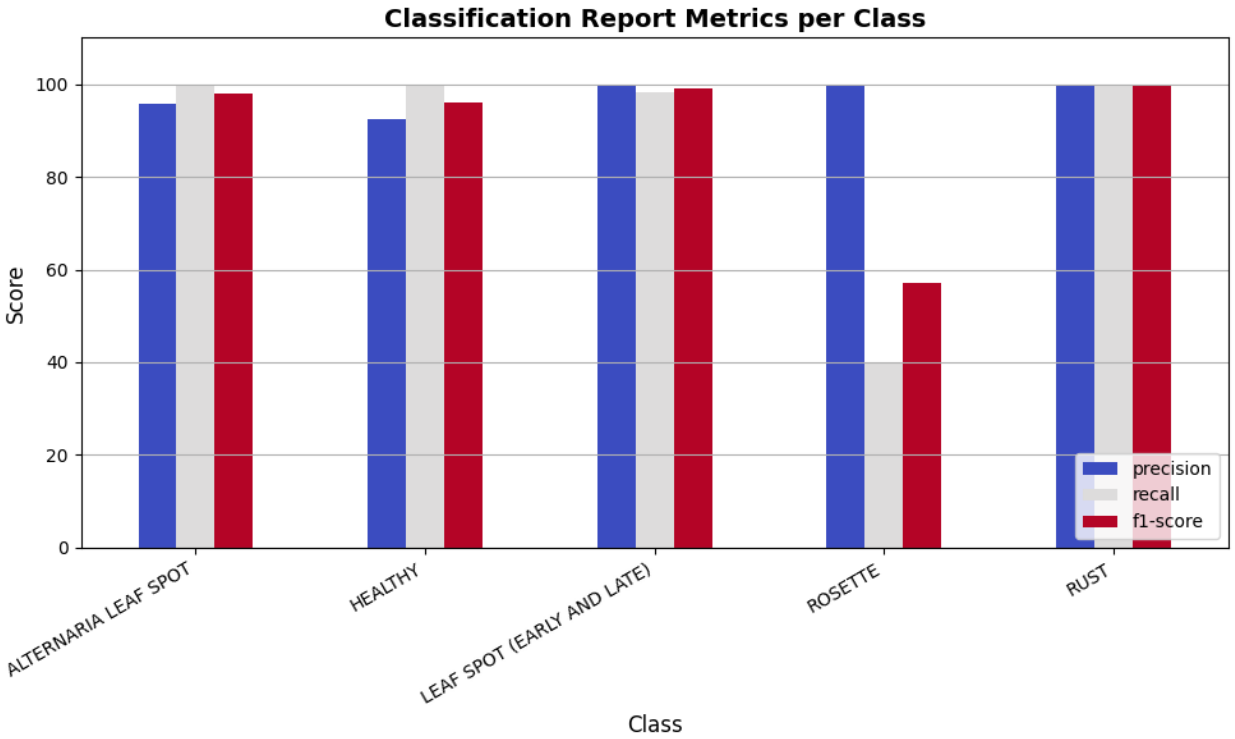


Fig 11. Classification report using ConvNeXt-Base model

Confusion matrix using ConvNeXt-Base model

Visual representation of all the classes present in our dataset is shown in the "Fig 12" which is representing the confusion matrix of all the peanut image classes using the ConvNeXt-Base model.

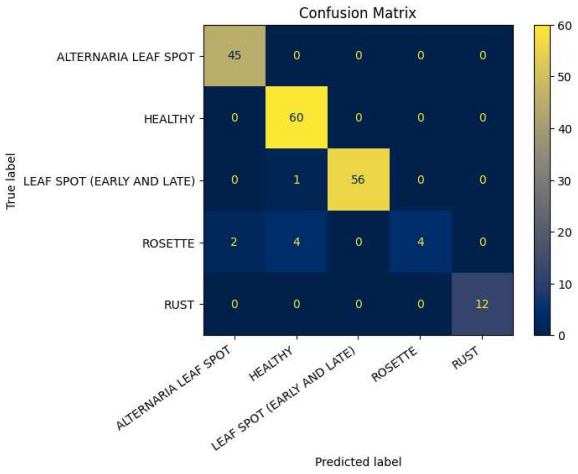


Fig 12. Confusion matrix using ConvNeXt-Base model

Loss dynamics related to ConvNeXt-Base model

Loss dynamics are shown in the "Fig 13" (a) which clearly demonstrates the value loss by using the ConvNeXt-Base model in our work.

Training vs. validation accuracy using ConvNeXt-Base model

By using the ConvNeXt-Base model in our work, the validation accuracy of 96.20% was obtained. “Fig 13” (b) Represents a visual view of training versus validation accuracy.

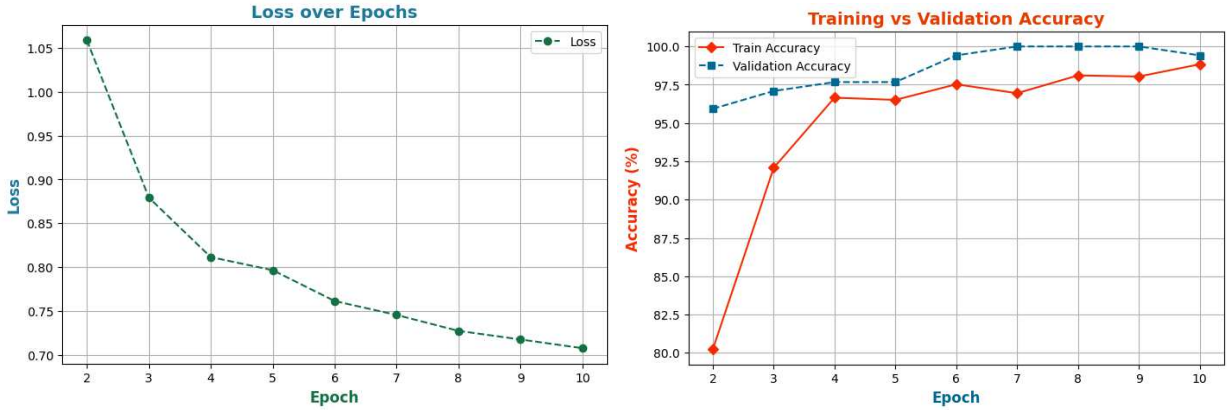


Fig 13. Using ConvNeXt-Base model (a) showing the loss dynamics (b) showing the training vs. validation accuracy

Results obtained by ResNet-50 model with transfer learning

ResNet-50 Architecture with transfer learning showed the exceptional results in our research work and proved to be the best among other modernized models applied on the same publicly available dataset.

The highest accuracy shown in our work is 98.37% by using transfer learning via ResNet-50. All other performance measure parameters showed the extraordinary results. Our experimental work got the Precision of 98.82%, a Recall value of 97.30%, and an F1-score of 97.99%.

Classification report metrics of ResNet-50 architecture

It is very evident from the “Fig 14”. Showing the classification report metrics of all the image classes of our dataset using the ResNet-50 with transfer learning and advance methodologies that our model performed best on all the classes.

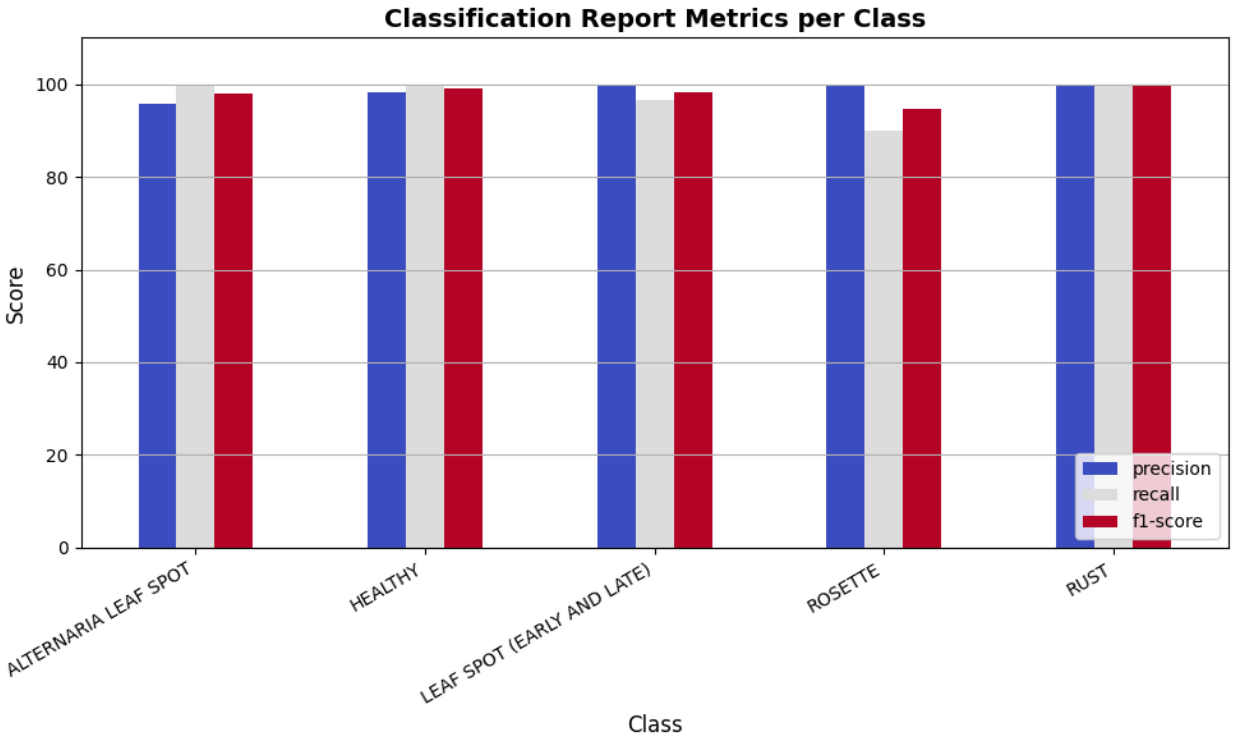


Fig 14. Classification report using ResNet-50 architecture

Confusion matrix using ResNet-50 architecture

The confusion matrix made by using the ResNet-50 Architecture with transfer learning on the peanut images of our dataset is given in the "Fig 15".

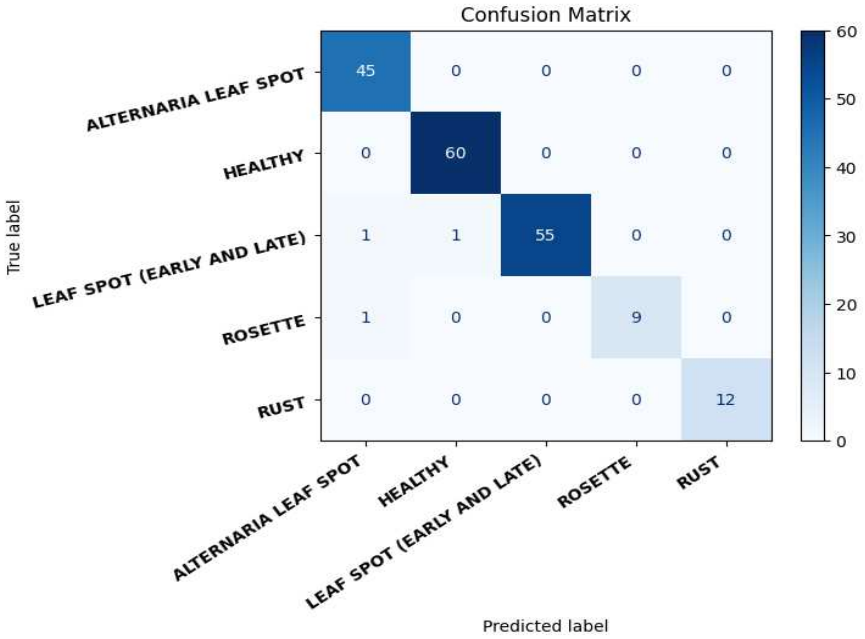


Fig 15. Confusion matrix using ResNet-50 architecture

Loss dynamics related to ResNet-50 architecture

“Fig 16” (a) is showing the loss dynamics attained by using the ResNet-50 architecture in our experimental work.

Training vs. validation accuracy using ResNet-50 architecture

The best validation accuracy of 98.91% was obtained by using the ResNet-50 Architecture with transfer learning in our experiment on the publicly available dataset. In this way it is evident in the “Fig 16” (b) that our model outperformed very well.

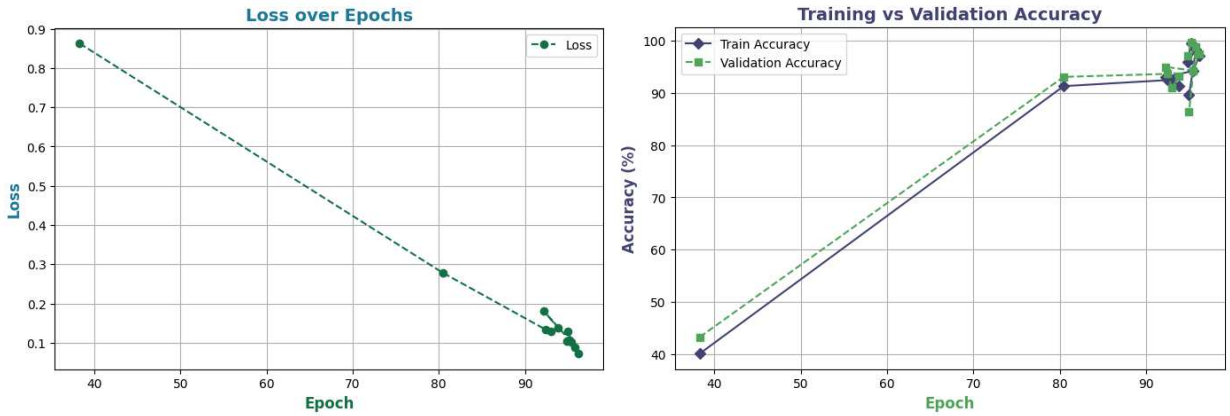


Fig 16. Using ResNet-50 architecture (a) showing the loss dynamics (b) showing the training vs. validation accuracy

Comparative view of all the performance measures using different models

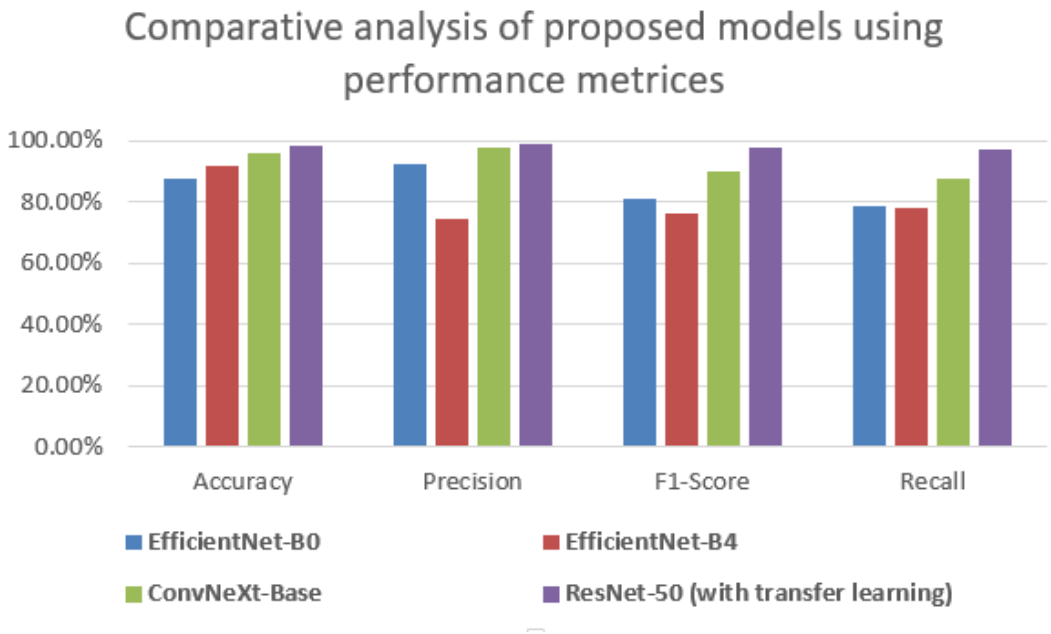
In the “Table 7”, we’ll be showing the results obtained related to different performance measures such as Accuracy, Precision, Recall, and F1-score using different models as EfficientNet-B0, EfficientNet-B4, ConvNeXt-Base, and ResNet-50 (with transfer learning) in our work.

Table 7: Showing the performance measures obtained using all the models of our proposed work

Model Name	Accuracy	Precision	F1-Score	Recall
EfficientNet-B0	87.50%	92.13%	81.21%	78.88%
EfficientNet-B4	91.85%	74.61%	76.27%	78.06%
ConvNeXt-Base	96.20%	97.61%	90.02%	87.65%
ResNet-50 (with transfer learning)	98.37%	98.82%	97.99%	97.30%

Moreover, “Fig 17” is a visual representation of all the results we have achieved using various proposed models.

Fig 17. Comparison of all performance metrics of proposed models



Conclusion and future work

For the fundamental need in sustainable agriculture, it has been considered for application of deep learning models for peanut disease detection. Application of ResNet-50 model with transfer learning and advance methodologies were demonstrated to be effective in the identification of all five peanut disease images. The models used in our experimental work showed the better results. This result demonstrates the potential of deep learning to offer farmers effective means of low cost, time appropriate disease detection tools to minimize crop loss and enhance food security. All of this however, leaves little capacity to scale the range of models globally for agricultural applications that see the disease in a variable fashion in a regional manner throughout the environment. The expansion of artificial intelligence driven technology’s abilities for environmentally friendly agricultural methods and food safety is what this study provides.

Future research may focus on the application to the presented investigation with advanced deep learning algorithms to further improve the model. Additional information from a wider range of locations globally may increase the model’s capability for generalizing well. Deployment of models to edge devices would make the detection of peanut diseases in real time more feasible in the harsh environmental field conditions. Therefore, the inclusion of multimodal data such as spectral or temporal information could help decode in a deeper way, disease progression and improve the diagnostic accuracy.

Data Availability Statement: This work uses a published dataset.

Acknowledgements:

Author Contribution: Conceptualization was done by K.M. and M.R; U.H.; methodology, M.I.; software, M.I.;

validation, Q.A. and T.A.; formal analysis, Q.A and H.A. ; investigation, M.R.; data curation, K.M.; writing 1st draft preparation, A.G.; writing—review and editing, R.A and A.S.A.; visualization, M.R and Q.A.; supervision. All authors have read and agreed to the published version of the manuscript.

Funding Statement: No funding available in this research.

Conflicts of Interest: The authors declare no conflicts of interest.

References:

1. Jha K, Doshi A, Patel P, Shah M. A comprehensive review on automation in agriculture using artificial intelligence. Artificial Intelligence in Agriculture. 2019 Jun 1;2:1–12.
2. Rui M, Ma C, Tang X, Yang J, Jiang F, Pan Y, et al. Phytotoxicity of Silver Nanoparticles to Peanut (Arachis hypogaea L.): Physiological Responses and Food Safety. ACS Sustain Chem Eng [Internet]. 2017 Aug 7 [cited 2025 Apr 23];5(8):6557–67. Available from: https://pubs.acs.org/doi/abs/10.1021/acssuschemeng.7b00736
3. King A. Technology: The Future of Agriculture. Nature 2017 544:7651 [Internet]. 2017 Apr 27 [cited 2025 Apr 23];544(7651):S21–3. Available from: https://www.nature.com/articles/544S21a
4. Anbumozhi A. Leaf Diseases Identification and Classification of Self-Collected Dataset on Groundnut Crop using Progressive Convolutional Neural Network (PGCNN). IJACSA) International Journal of Advanced Computer Science and Applications [Internet]. 2023 [cited 2025 Apr 23];14[2]. Available from: www.ijacsa.thesai.org
5. Efficient Deployment of Peanut Leaf Disease Detection Models on Edge AI Devices - ProQuest [Internet]. [cited 2025 Apr 23]. Available from: https://www.proquest.com/openview/ac4dd7c3017394d3929ca276f20b6d35/1?cbl=2032441&pq-origsite=gscholar
6. Sasmal B, Das A, Dhal KG, Saheb SkB, Khurma RA, Castillo PA. A novel groundnut leaf dataset for detection and classification of groundnut leaf diseases. Data Brief. 2024 Aug;55:110763.
7. Patayon UB, Crisostomo R V. Peanut leaf spot disease identification using pre-trained deep convolutional neural network. International Journal of Electrical and Computer Engineering (IJECE) [Internet]. 2022 Jun 1 [cited 2025 Apr 23];12[3]:3005–12. Available from: https://ijece.iaescore.com/index.php/IJECE/article/view/24919
8. Anbumozhi A, Shanthini A. Leaf Diseases Identification and Classification of Self-Collected Dataset on Groundnut Crop using Progressive Convolutional Neural Network (PGCNN). International Journal of Advanced Computer Science and Applications [Internet]. 2023 Dec 28 [cited 2025 Apr 23];14[2]:364–73. Available from: www.ijacsa.thesai.org
9. Paramanandham N, Sundhar S, Priya P. Enhancing Disease Detection With Weight Initialization and Residual Connections Using LeafNet for Groundnut Leaf Diseases. IEEE Access. 2024;12:91511–26.
10. M. P. A, Reddy P. Ensemble of CNN models for classification of groundnut plant leaf disease detection. Smart Agricultural Technology. 2023 Dec 1;6:100362.
11. Rakholia RM, Tailor JH, Saini JR, Kaur J, Pahuja H. Groundnuts Leaf Disease Recognition using Neural Network with Progressive Resizing. International Journal of Advanced Computer Science and Applications [Internet]. 2022 Autumn 30 [cited 2025 Apr 24];13(6):83–8. Available from: www.ijacsa.thesai.org
12. Chen SY, Kuo YM, Zhang RH, Cheng TY, Chu PY, Kang LW, et al. The Development of Peanut Defect Detection Technique Using Hyperspectral Images. 2023 Jun 2 [cited 2025 Apr 23]; Available from: https://scispace.com/papers/the-development-of-peanut-defect-detection-technique-using-36bm27t0
13. Wang Y, Ding Z, Song J, Ge Z, Deng Z, Liu Z, et al. Peanut Defect Identification Based on Multispectral Image and Deep Learning. Agronomy 2023, Vol 13, Page 1158 [Internet]. 2023 Apr 19 [cited 2025 Apr 23];13(4):1158. Available from: https://www.mdpi.com/2073-4395/13/4/1158/htm
14. Lv Z, Yang S, Ma S, Wang Q, Sun J, Du L, et al. Efficient Deployment of Peanut Leaf Disease Detection Models on Edge AI Devices. Agriculture 2025, Vol 15, Page 332 [Internet]. 2025 Feb 2 [cited 2025 Apr 23];15[3]:332. Available from: https://www.mdpi.com/2077-0472/15/3/332/htm
15. Sivaganesan D. Deep Learning Approaches for Disease Detection in Groundnut Crops using CNN Models. Journal of Soft Computing Paradigm. 2023 Dec;5(4):404–16.

16. Rajmohan M, Reddy DSS, Krishna CM, Krishna NMS. Groundnut leaf disease identification using image processing. AIP Conf Proc [Internet]. 2022 Oct 3 [cited 2025 Apr 23];2519[1). Available from: /aip/acp/article/2519/1/030110/2828013/Groundnut-leaf-disease-identification-using-image
17. Gowrishankar K, Prabha L. An Integrated Image Processing Approach for Diagnosis of Groundnut Plant Leaf Disease using ANN and GLCM. J Sci Ind Res (India). 2020;79:372–6.
18. ZOU Z, CHEN J, WANG L, WU W, YU T, WANG Y, et al. Nondestructive detection of peanuts mildew based on hyperspectral image technology and machine learning algorithm. Food Science and Technology. 2022;42.
19. Tan M, Le Q V. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. 36th International Conference on Machine Learning, ICML 2019 [Internet]. 2019 May 28 [cited 2025 Apr 25];2019-June:10691–700. Available from: https://arxiv.org/pdf/1905.11946
20. Liu Z, Mao H, Wu CY, Feichtenhofer C, Darrell T, Xie S. A ConvNet for the 2020s. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition [Internet]. 2022 Jan 10 [cited 2025 Apr 24];2022-June:11966–76. Available from: https://arxiv.org/pdf/2201.03545
21. ConvNeXt — Next Generation of Convolutional Networks | by Atakan Erdoğan | Medium [Internet]. [cited 2025 Apr 25]. Available from: https://medium.com/@atakanerdogan305/convnext-next-generation-of-convolutional-networks-325607a08c4
22. Scikit-learn Developers. `compute_class_weight` documentation, https://scikit-learn.org/stable/modules/generated/sklearn.utils.class_weight.compute_class_weight.html
23. L. N. Smith and N. Topin, "Super-convergence: Very fast training of neural networks using large learning rates," in Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications, vol. 11006, 2019. https://doi.org/10.1117/12.2520589
24. K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), 2016, pp. 770–778. https://doi.org/10.1109/CVPR.2016.90
25. P. Thakare and R. S. V., "Advanced pest detection strategy using hybrid optimization tuned deep convolutional neural network," Journal of Engineering, Design and Technology, vol. 22, no. 3, pp. 645–678, Apr. 2024, doi: 10.1108/JEDT-09-2021-0488.
26. S. Dev, "Application of CNN Architecture for Insect Identification and Labeling in Peanut Crops," in 2023 International Conference on Data Science and Network Security (ICDSNS), IEEE, Jul. 2023, pp. 1–8. doi: 10.1109/ICDSNS58469.2023.10244877