RhodaNet: A novel deep learning architecture for Rose disease classification

Pudumalar S [a,*], Muthuramalingam S [b]

[a,b] Department of Information Technology, Thiagarajar College of Engineering, Madurai, Tamilnadu, India,625020.

*spmit@tce.edu, *+91 9940529761, smrit@tce.edu

Abstract— **Technology adoption in agriculture has creatively solved and eased many farming problems. Home gardeners grow various plants throughout the year. Many of them show interest in growing roses at their houses. In a survey conducted with home gardeners 70% of them have roses in their garden and 80% of them added that their plants get infected often. Early and accurate diagnosis of the diseases may reduce the likelihood that the plant will suffer further harm and spread. The substantial advancements in deep learning have created the opportunity to improve the coordination and accuracy of the system for identifying plant diseases. An improvised ResNet architecture-RhodaNet is proposed to identify the rose disease at an early stage. RhodaNet architecture uses concatenation in stacked layers preserving both spatial and channel information even where input and output have different channel sizes or feature representations. RoseNet dataset from Mendeley data with 2993 images was considered for the study with 5 diseases Black spot, Downy Mildew Powdery Mildew. Mosaic, Botrytis Blight. Our proposed model gives an accuracy of 96% whereas ResNet and DenseNet gives 91.56% and 93.50% respectively. RhodaAPP predicts the type of disease and its remedy serves as an appropriate solution for home gardeners.**

Keywords—Rose diseases, ResNet, DenseNet, RhodaNet, RhodaApp

1.INTRODUCTION

Humans' innate urge to be with plants is at the heart of the history of garden. Ancient Egypt and China had some of the earliest known gardens. These societies gave rise to the two opposing traditions of formalism and naturalism in gardens. Formalism: Plants were seen as merely a building material and were designed and pruned to fit the design. The idea of "naturalism" in the garden might be understood as an effort to coexist with nature rather than try to control it. In India, Babar, one of the Mughal emperors, was a garden enthusiast with a keen eye for beauty. He created gardens in Panipat and Agra. The Archeological Department in Agra continues to provide excellent care for Aram Bagh. Formal gardening is always associated with Mughal gardens. The special characteristics of Mughal gardens are the square or rectangular flowerbeds. Exotic plants like cypress, rose, carnation, narcissus, daffodils, lilies, tulips, etc. were introduced significantly. Notable gardens are Budha Jayanti Park and the Rose Garden. For rural households in Southeast Asia, the home garden has historically been a crucial plot of land. A farming system that integrates several physical, social, and economic activities on the property surrounding the home might be referred to as a "home garden." The average backyard contains social spaces for gatherings, play areas for kids, and display gardens, as well as practical spaces for storing things, living, washing, and disposing of waste, and producing food, trees, medicinal plants, and fish. Although it is a place where people can live, it also generates a wide range of meals and other items for use in homes. In recent years, there has been a significant surge in the popularity of home gardening in India. This can be attributed to the growing awareness among individuals regarding the numerous benefits associated with cultivating their own flowers, fruits, vegetables, and herbs.

The rose, being one of the oldest flowers in cultivation, is considered to be the most adorable creation of nature. Rosaceae, the rose family of flowering plants, consists of 2500 species in over 90 genera. The cultivation of roses in gardens began some 5000 years ago in China. During the 16th century, when the Mughal Emperors assumed control over India, they transported significant quantities of roses via camel caravans. Since then, roses have been considered a great business plan in India. Rose cultivation on a commercial basis has provided greater profit among those cultivators because of its increasing demand in the global market. Polyphenols are present in rose petals and serve as antioxidants, which safeguard the body against cellular harm. Research studies have demonstrated that the presence of polyphenols in rose tea can effectively mitigate the likelihood of developing heart disease, diabetes, obesity, and cognitive diseases. Rose water and oil are used in making perfumes and cosmetics. The major challenge that every home gardener faces is the identification of plant diseases. Out of the survey conducted among the native home gardeners, rose, guava, aleo vera, and curry leaves are widely grown. With a high response rate of 80% among gardeners, this paper focuses on the detection of diseases in rose plants using deep learning models. Most common rose diseases include downy mildew, powdery mildew, black spot, mosaic, botrytis blight, and so on. People who grow roses at home will not be able to immediately detect these diseases and provide suitable remedies. To reduce this hassle among home gardeners, several machine learning algorithms like SVM and RF were employed to detect those diseases in an accurate manner. Diagnosing and classifying these diseases can be challenging due to their complex and diverse nature as well as the wide array of symptoms they present. The inherent complexity involved in developing machine learning models poses challenges for accurately predicting diseases. The accuracy of machine learning algorithms is heavily dependent on the quality of the input data. In order to address these limitations, transfer learning-based models were utilized for the purpose of plant disease detection.

Deep learning (DL), one of the primary subfields of machine learning, uses neurons to carry out learning tasks that resemble mathematical processes. They are frequently used to diagnose illnesses, and artificial neural networks (ANNs), like neural networks, have gained increased recognition in this area. Neural networks (NN) are frequently employed in decision-making processes. Artificial Neural Networks (ANN) are a deep learning technique that surpasses traditional methods in terms of performance by utilizing images with different levels of complexity. Advancements in deep learning have facilitated the resolution of challenging problems and the attainment of notable outcomes in various machine learning and computer vision applications. These applications encompass object detection, picture classification, speech, and language recognition. ResNet (residual network) and DenseNet (densely connected convolutional network), being variants of CNN, are examples of transfer learning-based models. In a ResNet architecture, the input is passed through a sequence of residual blocks after a series of convolutional layers. The network may learn residual functions because each residual block has a shortcut connection that skips one or more convolutional layers. The block's input and output are then combined, and a nonlinear activation function like ReLU is used to activate the block. Each dense block in a DenseNet design is made up of a number of layers, each of which is connected to all the layers that came before it, generating a dense connectivity pattern. This makes it possible for the network to access and reuse features from all earlier levels, improving feature propagation and training effectiveness. The batch normalization, a ReLU activation function, and a convolutional layer are typically the order of the layers in a dense block. Large datasets like ImageNet were used to pre-train these deep learning architectures.

Numerous scholars have extensively examined the occurrence of plant leaf diseases. However, it is unequivocally challenging to precisely diagnose diseases affecting leaves. The symptoms exhibited by infected plants can vary both between different plant species and within the same plant. As a result, several researchers conducted investigations on the impact of these diseases on a limited number of plants. The primary objective of this paper is to develop a system for detecting diseases in roses. Furthermore, our proposed procedure is based on an enhanced architecture of the current deep learning model, specifically RhodaNet. RhodaNet is constructed by merging the advantageous features of the ResNet and DenseNet models. ResNet utilizes residual connections to enhance the learning efficiency of the network. On the other hand, DenseNet establishes dense connections between all layers to maximize the reuse of features and minimize the parameter count. The concatenation feature utilized in DenseNet has been adopted and implemented within the ResNet skip connections. The utilization of concatenation offers several benefits, including enhanced feature reuse and the ability to capture a broader spectrum of features, resulting in improved accuracy for the network.

2.LITERATURE REVIEW

Agriculture and botanical research have seen an increase in the use of ML and DL-based methodologies in recent years. These methods have demonstrated significant promise in enhancing crop productivity, locating plant lesions, and enhancing plant growth. ML and DL-based methods provide a number of advantages over conventional methods, and they have the potential to change the fields of agricultural and botanical research. Traditionally, manual inspection and professional knowledge have been the mainstays of agricultural and botanical studies. These procedures are frequently laborious, physically taxing, and prone to human error. Contrarily, ML and DL-based methods can automate these operations, lowering the requirement for human intervention and increasing the process' accuracy and efficiency. Many academic researchers have improved and introduced a variety of deep CNN models that detect plant diseases without the intervention of experts guidance. Table 1. Shows the literature review of existing solutions that tried to solve plant disease classification problems.

Kaiming He et.al [1] introduced the concept of Residual Networks (ResNets), which are a type of deep neural network architecture. The key innovation of ResNets is the use of residual blocks that allow for the training of much deeper networks by addressing the vanishing gradient problem.As neural networks become deeper, their performance often saturates and then degrades rapidly, a phenomenon known as the degradation problem.The author used ResNets to solve this by introducing a residual learning framework. Instead of layers learning direct mappings, they learn residual functions with reference to the layer inputs. The model consists of 34 convolutional layers, with the number of filters progressively increasing from 64 to 128 and then to 256 as it traverses through the 34 layers. Additionally, it presents the notion of skip connections, also known as residual connections. Skip connections enable the direct transfer of information from one layer to another that is not sequentially adjacent in a neural network. The ResNet architecture incorporates a process where the input of a specific layer is combined with the output of a subsequent layer. The model is capable of learning residuals, which refer to the discrepancy between the input and the desired output. These residuals can subsequently be combined with the output of a subsequent layer to generate the final output. The ResNet architecture enables the network to efficiently access and reuse features from preceding layers, resulting in improved feature propagation and enhanced training efficiency. Hence, this study utilizes the advantage of the residual network.

Gao Huang et.al [2] the authors believed that convolutional networks can achieve greater depth, accuracy, and training efficiency when they incorporate shorter connections between layers close to the input and those close to the output. Building upon this insight, the authors introduce the Dense Convolutional Network (DenseNet). Unlike traditional convolutional networks, where each layer is connected only to its subsequent layer, DenseNet connects each layer to every other layer in a feed-forward manner. DenseNet, which utilizes the concept of skip connections in ResNet. In the DenseNet architecture, there is a direct connection between every layer and all other layers in a feedforward manner. This leads to the establishment of dense connections between the layers. The output feature maps of each layer are concatenated with the input feature maps of all subsequent layers in a specific manner. The process generates a "dense block" comprising multiple layers. Each layer in the block receives input that incorporates the output from all preceding layers within the block. DenseNet achieves a consistent level of feature reuse and propagation throughout the network by establishing connections between each layer and all subsequent layers. This design choice enhances the efficiency and ease of training compared to conventional deep neural networks. To address classification tasks, researchers frequently use DenseNet and ResNet.

Kamilaris et al. [3]analyzed the Plant Village dataset with various modern CNN architectures and showed that VGG outperformed other CNN architectures. These applications assume one-to-one mapping as 1 leaf: 1 disease and perform plant disease diagnosis. Tuning hypermeters gives better performance to improve the identification accuracy in VGG and Alex Net structures

Barbedo et al. [4] explore how the lack of an image database can be addressed by increasing the variability of data by considering all individual lesions and the spot's characteristics. Though data augmentation overcomes the problem of insufficient image databases, it fails to cover practical diversity. Without additional images, the author tries to use individual lesions and spots for identification rather than the entire leaf, allowing the identification of multiple diseases affecting the same leaf. The augmented images outperformed the original images by 12%. About ten diseases were considered, and even then, no crop had accuracies below 75%, proving deep learning techniques are effective for plant disease detection and recognition with large data sets. The original dataset contains 1567 images; after augmenting, the number of images increased to 46,409 images with 76 class labels for 14 plant species, the increase in dataset size resulted in 82% accuracy for Google Net.

Shahin et al. [5] developed a model to classify olive leaf disease detection using a dataset consisting of 3400 olive leaf samples of Aculusolearius and peacock spot disease. They compared their proposed model using both augmented and non-augmented data sets with the VGG16 and VGG19 architectures. The augmentation dataset reached the highest success rate of 95% with different optimization algorithms like Adam, Adagrad, SGD, and RMSProp.

Lamba et.al [6] proposed a model that combines Auto-Color Correlogram as an image filter with Deep Learning (DL) classifiers using different activation functions. Applying this model to four distinct datasets, addressing both binary and multiclass subcategories of plant diseases. The results for binary classification, the proposed model achieves 99.4% accuracy and 99.9% sensitivity. For multiclass classification, the accuracy reaches 99.2%. Comparing their approach to other methods such as LibSVM and SMO, the proposed model outperforms them in terms of F-measure, recall, Matthews correlation coefficient (MCC), specificity, and sensitivity1. This research highlights the potential of deep learning in enhancing disease detection accuracy and contributes to better understanding and managing plant health.

Wei SJ et.al [7] utilized the Plant Village dataset and employed image transformation techniques and down sampling to mitigate the problem of unbalanced class distribution. The study findings significantly indicated that DenseNet121 outperformed the other models with an accuracy of 96.4%. Furthermore, the study evaluated the model's performance on various runtime devices such as CPU, GPU, and VPU to test its consistency across different hardware and software configurations. The results showed that when the model was put on the VPU device, it kept its high recall (95.6%), precision (96.3%), and F1 (96%) scores. This suggests that it could be used as an embedded application to find plant diseases. Overall, the successful implementation of the model on the VPU device suggests its feasibility as an edge solution for detecting plant diseases. This research provides valuable insights into the use of CNN models for edge computing applications in the field of agriculture, particularly for plant disease detection.

Rasheed et.al [8] examines the performance of a CNN-based classification algorithm over other traditional classification algorithms like SVM. The study found that optimizing the parameters of the CNN algorithm is important for achieving optimal performance and avoiding overfitting. The researchers also noted that the CNN-based model outperformed the SVM and Softmax classifiers in terms of accuracy but had a slower execution time. Overall, the study highlights the benefits of using CNNs for classification tasks and the importance of selecting the appropriate algorithm based on the specific needs and requirements of the task at hand. The CNN-based model demonstrates a superior accuracy performance of 95.94% and an execution time of 58.8 ms. In comparison, the linear classifiers, specifically the SVM, achieve an accuracy of 75.62% with an execution time of 2.03 ms, while the Softmax achieves a testing accuracy of 62.9% with an execution time of 2.04 ms. Convolutional Neural Network Models: DenseNet121, VGG16, ResNet50, and AlexNet for Detecting Plant Diseases as an edge Solution

Chen et al. [9] advanced tomato disease detection by tailoring the AlexNet CNN architecture. They trained their model on a dataset with an 80/20 split between training and testing images, all in RGB and labeled by disease type. Utilizing the Adam optimizer and a learning rate of 0.0005, the model underwent 75 epochs of training with batches of 128. The cross-entropy loss function was meticulously optimized, leading to a model that boasts a 98% accuracy, 0.98 precision, 0.99 recall, and an F1-score of 0.98. The loss was recorded at 0.1331. These metrics underscore the model’s precision and reliability. The study reflects the evolution of deep learning since AlexNet, highlighting the superior performance of newer architectures like GoogleNet and ResNet, especially in mobile-optimized applications. This opens avenues for enhanced accuracy and speed in mobile-based image classification, promising for agricultural technology advancements. By exploring these architectures and their modifications, researchers can potentially achieve even better accuracy and faster classification speeds for mobile-based image classification applications

Borhani et.al [10] explored a deep learning-based approach for automated plant disease classification using a Vision Transformer (ViT). They proposed a lightweight model that combines ViT with classical convolutional neural network (CNN) methods. The combined approach aims to provide real-time, accurate plant disease classification while addressing the trade-off between accuracy and prediction speed. The models were trained and evaluated on multiple datasets, and the results indicated that while attention blocks in ViT increase accuracy, they also slow down prediction. However, integrating these with CNN blocks can help maintain speed, making the approach feasible for real-time applications.

Algani et.al [11] explored an innovative approach for detecting and classifying plant leaf diseases. The study leveraged a deep learning technique called Ant Colony Optimization with Convolution Neural Network (ACO-CNN) to improve the accuracy of disease diagnosis in plant leaves.The proposed ACO-CNN approach outperformed existing techniques in terms of accuracy.

Zhang et al.[12 &13]presents a method to extend the lifetime of DC–DC converters using a Levenberg–Marquardt backpropagation neural network (LM-BPNN) and a power routing strategy. The LM-BPNN is trained with voltage, current, and temperature data to estimate the health of each power cell in the system. By routing power differently based on the estimated damage of the cells, the system can delay the failure of more damaged cells. This approach has been validated through numerical simulations and experimental setups, demonstrating its potential for predictive maintenance in industrial systems.

Zhang et.al [14] focus on the accurate prediction of the remaining useful life (RUL) of lithium-ion batteries, which is critical for energy supply systems. The PF-BiGRU-TSAM (Particle Filter - Temporal Attention Mechanism - Bidirectional Gated Recurrent Unit) approach is a novel method proposed for predicting the remaining useful life (RUL) of lithium-ion batteries. The approach is data-model interactive which uses both historical data and a prediction model. The BiGRU-TSAM is trained offline through historical data. This model assigns corresponding significance to battery capacities at different time points. The Particle Filter is likely used for state estimation in the presence of noise, a common technique in robotics and control systems. The Temporal Attention Mechanism allows the model to pay different amounts of attention to different time steps in the input data. This approach is designed to enhance the precision and stability of battery RUL prediction.

Topaloglu [15] study introduces a novel image classification method for eye disease identification, specifically focusing on Diabetic Retinopathy. The research employs a unique “care model” that diverges from traditional CNN structures. This model enhances accuracy by rescaling data based on pixel count before pooling. The study utilizes the VGG19 model and a developed mathematical model, achieving 87% training accuracy, 88% testing accuracy, 93% precision, and 83% recall. These results suggest the model’s potential for accurate eye disease diagnosis using deep learning.

Khasawneh et al. [16] on the automatic identification of tomato leaf diseases using three compact CNN models, namely ResNet-18, ShuffleNet, and MobileNet. The study utilized transfer learning to extract deep features from the final fully connected layer of the CNNs for high-level representation. These features were then merged from the three CNNs to benefit from every CNN structure. The study used six classifiers to identify tomato leaf diseases, and the findings show that, with only 22 and 24 features, the K-nearest neighbor and support vector machine classifiers achieved the best accuracy, 99.92% and 99.90%, respectively. In contrast, Mukherjee et al. [17] found that Google Net with transfer learning performed best with an accuracy of 99.3% by conducting 60 different experiments and configurations in the Plant Village dataset.This study demonstrates how computer vision technology can be used to enhance crop quality and agricultural productivity in the context of smart farming. Uğuz et al., Athanikar et al., Brahimi et al., Batchuluun et al., Mohanty et al., Sladojevic et al., Kumar et al. Ashqar et al. Geetharamani et al., Militante et al., Ozguven et al., [18-28] outlines the state of the art in CNN applications for plant disease recognition, emphasizing their role in agricultural innovation .

3. HIGHLIGHTS

This study makes several contributions that address gaps in the current body of research.

1. To propose a novel architecture RhodaNet -an improvised ResNet capable of classifying rose leaf diseases into six distinct classes, namely black spot, powdery mildew, downy mildew, botrytis blight mosaic, and fresh leaf. Based on our current understanding, our proposed model represents a pioneering implementation for the identification and classification of five specific ailments affecting rose leaves. It is worth noting that the majority of existing techniques solely focus on classifying two diseases, namely black spot and downy mildew.

2. The proposed RhodaNet architecture enhances feature reuse and the ability to capture a broader spectrum of features, ultimately resulting in improved accuracy for the network than the existing deep learning model. Unlike ResNet, which uses identity mapping (addition) in its stacked layers, RhodaNet uses concatenation. In this proposed model, input given to the first layer is concatenated with the output of the second layer, and the concatenated result is given as input to the third layer. Input and output are concatenated along the channel dimension, and a convolutional layer is applied to reduce the number of channels. Concatenation preserves both spatial and channel information for input and output. This is beneficial in cases where input and output have different channel sizes or feature representations.

3.The proposed model demonstrates a higher accuracy percentage when compared to previous studies in the field, as reported in academic journals. RhodaAPP is developed to support the home gardeners, by just uploading the images, they get notified with the type of disease and remedy.

4. MATERIALS AND METHODS

This section presents the datasets used in the experiments, describes various disease recognition methodologies reported in the literature and introduces the proposed RhodaNet architecture.

4.1 Image Datasets

Rose diseases are a common problem in gardens and landscapes. There are different types of diseases that can affect roses, including fungal, bacterial, and viral diseases. These diseases can cause a range of symptoms, including discoloration of leaves, spots or patches on leaves and stems, wilting, stunted growth, and even the death of the plant. Some of the most common rose diseases include black spot, powdery mildew, rust, and crown gall. Black spot is a fungal disease that causes black spots to appear on leaves, eventually leading to defoliation. Powdery mildew is another fungal disease that appears as a white or grayish powder on leaves, stems, and buds. Rust is a fungal disease that causes orange or red pustules to appear on leaves, stems, and flowers. Crown gall is a bacterial disease that causes abnormal growths on the roots and stems of the plant, which can eventually lead to death.In the initial stage, a Rosenet dataset containing black spot, downy mildew, and fresh leaf classes has been collected from an openly accessible dataset, namely the Mendeley dataset https://data.mendeley.com/datasets/7z67nyc57w/2. Other than the existing dataset, mosaic, botrytis blight, and powdery mildew rose disease images have been captured using a camera device and were collected from various fields. The number of images used in training and testing for each class is shown in Table 2. The various field images of black spot, rust, downy and powdery mildew, Botrytis blight, and mosaic are shown in Figure 1.

Data Pre-processing and Data Augmentation: Following the completion of data collection, the subsequent step involves preprocessing, which entails carrying out geometric adjustments such as color orientation and resizing. The process of preprocessing is utilized to adequately prepare the samples for both training and classification purposes. In order to mitigate the issue of overfitting, the process of data augmentation is implemented using the Image Data Generator class in the Keras framework. Using this class, rotation range, zoom range, skew, shift in the width and height of the image, and its flip in the horizontal and vertical directions are defined. After defining these variables, fitting these measurements to the training data is done. Balancing the dataset is an important step in training the RhodaNet model. This ensures that the model is not biased towards one class and can accurately predict all classes. A data augmentation process is carried out to balance the dataset. Data augmentation techniques involve rotation, flipping, and cropping shown in figure 2. By applying these techniques to the input image dataset, new images will be created, which will increase the size of the dataset. This will be helpful in balancing a class in accordance with the other classes in the dataset. The augmented data sample shown in Figure 3.

Input Image Representation:

Color images being the input image, their representation will be in the 3D matrix format, with its 3 dimensions representing the height, width, and channels. Each channel corresponds to a red, blue, or green color channel. Before being fed to the RhodaNet model, the input images are preprocessed by resizing them to a fixed size (224x224x3 pixels). Resizing is carried out by subtracting the mean pixel across all training images and dividing by the standard deviation of pixel values across all training images. The performance of the RhodaNet model is improved by this normalization step by making the input more consistent across different images. Figure 4 represents the conversion of 3D to 1D, representing a single channel.

4.2 ResNet architecture:

The residual network is a deep neural network architecture that was first introduced by Kaiming He et al. in 2015. The purpose of its design is to tackle the issue of vanishing gradients in deep neural networks, which can result in subpar performance or failure to converge during the training process. The introduction of residual connections is a crucial element in this model. These connections facilitate the smoother flow of information through the network, effectively preventing the issue of vanishing gradients. Residual connections function by incorporating the input of a layer into its output, thereby establishing a shortcut that circumvents the layer. The inclusion of a shortcut connection in the network facilitates the acquisition of residual mappings, which represent the discrepancy between the input and output of a layer. This approach is advantageous as it simplifies the optimization process compared to direct mappings. The ResNet model is a refined convolutional neural network (CNN) architecture primarily employed for image classification tasks. Its structure comprises a series of convolutional layers, which are subsequently followed by global average pooling and a fully connected layer. Convolutional layers are structured in a hierarchical manner, forming blocks. Each block comprises several layers that possess identical output shapes. The initial block reduces the number of layers, resulting in a down-sampled input image. In contrast, the following blocks have a greater number of layers, allowing them to maintain the spatial resolution of the input. The building blocks of ResNet have two stacked layers, defined as follows:

$$H(x) = F\ (x,\ \{Wi\}) + x \qquad (1)$$

where x and H(x) are the input and output vectors of the building blocks. F(x,{Wi}) -residual mapping to be learned.
The formula F(x)+x can be realized by feedforward neural networks with identity shortcut connections. The identity shortcut connections perform identity mapping and the outputs are then added to the outputs of the stacked layers as shown in figure 5

4.3 Proposed RhodaNet model:

RhodaNet is an improvised ResNet architecture. Unlike ResNet, which uses identity mapping (addition) in its stacked layers, RhodaNet uses concatenation. In this proposed model, input given to the first layer is concatenated with the output of the second layer, and the concatenated result is given as input to the third layer. Input and output are concatenated along the channel dimension, and a convolutional layer is applied to reduce the number of channels. Concatenation preserves both spatial and channel information for input and output. This is beneficial in cases where input and output have different channel sizes or feature representations.

Rhoda Block:

Rhoda blocks are the building blocks of the Rhoda Net architecture. Each block consists of two convolution layers followed by batch normalization and activation functions (RELU). Along with this, there is also a shortcut connection that skips one layer. This shortcut connection helps avoid the vanishing gradient problem during the process of training and allows the model to learn in a better way. Figure 6 represents a single Rhoda block.

Convolutional Layer:

Within each Rhoda block, a layer is employed to execute a series of convolutions on the input data in order to extract the pertinent features. The layers in question utilize a series of filters that can be adjusted through learning processes. These filters are applied to the input image, resulting in the creation of a collection of output feature maps. In the convolutional layer, every filter traverses the input image and conducts a dot product operation between its weight and the corresponding pixel values in the input image. The output feature map serves the purpose of emphasizing specific patterns or features that are present in the input image. RhodaNet uses a 3x3 kernel size for the convolutional layer, which is followed by batch normalization and a ReLU activation function. This combination is utilized to introduce non-linearity into the network. The number of filters in each layer is incrementally increased as the network depth increases. This progressive increase enables the model to acquire a greater understanding of intricate features.

Batch Normalization:

The batch normalization process helps in normalizing the input to each activation function, which can accelerate the training process by reducing the internal covariate shift (change in distribution of input to a layer caused by the change in the previous layer's parameters during training). This process will be performed after convolution and before the ReLU activation function. The mean and variance of the input batch are calculated, and then the inputs are normalized based on these values. Normalized values are then scaled and shifted by learned parameters called Gamma and Beta. These learned parameters allow the model to learn the optimal scale and shift for the normalized inputs.

Activation Function:

The activation function is used after each batch normalization layer and convolution layer in a Rhoda block. It applies a non-linear transformation to the output of the batch normalization and thus introduces non-linearity into the model. ReLU sets all negative values to zero and keeps positive values as they are. ReLU is popularly used in many deep learning networks due to their simplicity and effectiveness. ReLU is defined as

$$f(x) = max\ (0, x) \qquad (2)$$

where x denotes the input to the function and f(x) is the output. By introducing non linearity into the network, ReLU allows the RhodaNet Model to learn more complex representations of input data.

Skip Connection:

This is the key component of the RhodaNet architecture. Skip connections are added to bypass one or more convolutional layers and directly connect the input to the output of the ResBlock, thereby addressing the degradation problem of deeper neural networks. The gradients flow directly from the output layer to the input layer, bypassing intermediate layers; this preserves the gradients and ensures that the model is able to learn complex representations effectively. In RhodaNet, skip connections are implemented using the concatenation operation. Here, the output of a layer is concatenated with the input of the subsequent layer. The Rhodanet model uses the concat function as the core operation for skip connections.

$$output = [x, f(x)] \qquad (3)$$

The above equation represents the concatenation of input x and output f(x) of the subsequent convolutional layer. The proposed RhodaNet layer structure in figure 7 and 8

Proposed RhodaNet Model:

Input: Rose leaves images (Healthy and Diseased)

Output: Classified Images as Ci

1. Preprocess images to improve visibility

3. Resize and transform images to 224 x 224 square and save as RGB channels

4. Start Training of model for all images I:

5. Extraction of raw features from the pre-processed images.

6. Start convolution and feature maps generation.

7.RhodaNet_layer function;

For Stack :0

Input Parameters: Filters:16; kernel size:3; strides:1; Activation function:'ReLu'; Normalisation: Batch; Conv_first=True

On calling ***RhodaNet_layer function***, a stack of RhodaNet block is created.

I. ***Perform Convolution*** *on the input matrix using the function* ***Conv2D*** *(Input: number; filters, kernel size, strides, padding, kernel initializer, kernel_regularizer)*

then

```
II. Perform Batch Normalization using the function Batch Normalization () (x), here x represents the input matrix the output obtained from convolution. (Input: Batch size: 32, epochs :20 and total number of classes is 6 (5 disease+1 healthy leaf).
 Step:1   Set Depth of RhodaNet = N * 6 + 2 where N = 3.
Step :2   Normalise input by dividing its size by 225, input
          size 224x224 pixels.
Step:3 Set number of filters:16 (within the function    rhodanet_v1)
III. Instantiate the stack layers. Apply ReLU activation function and obtain feature map.
Output: X
To instantiate
Inner Loop: Iterate Stack++
Outer Loop: Iterate Block++
RhodaNet layer: Input :X.
Set res_block=0, num_filters =16 and strides =1.
Output: Y
Next
RhodaNet Layer: Input: Y, Iterate (I) and (II) with no activation.
Output: Y
IV) Concatenate (X and Y)—> (Skip Connection)
    Until stack=0; (outer loop)
For Stack =1,2 and 3: {all instances stack>0}
    Perform down sampling,
    Call RhodaNet Layer function repeatedly.
                        For each increment in the outer loop, the number of filters gets incremented by multiples of two. (i..e..16, 32, 64…)
      Introduce linear projection to the shortcut connections to match the dimensions, set kernel size :1 and strides :2
      8.Max pooling with pool size:8.
      9.Flatten
      10.Softplus
      11.Classification as Ci belonging to i classes of
         disease.
      END
```

5. RESULTS AND DISCUSSION

In order to assess the effectiveness of the proposed method, the customized Mendeley data was partitioned into training and testing sets using an 80:20 ratio. The experiment was conducted on a Windows system equipped with an Intel Core i7-4790 v3 processor running at 4 GHz x 4, 16 GB of RAM, and a stable internet connection. The experiments were conducted using the Python framework and the Keras v0.1.1 library. The performance of the proposed model was analysed using an image size of 256 x 256. The NumPy package for matrix manipulation and the Pandas package for data processing are used. Keras is a high-level neural network API capable of running on top of several lower-level deep learning frameworks, including TensorFlow, Microsoft Cognitive Toolkit, and Thean. Though there are many plants grown by home gardeners, a survey was conducted to find out the needs of the stakeholders. The results of the survey narrowed down our research study to rose plants. The other top choices from the stakeholders were hibiscus, guava, and mango. As the quality of the data improved, a few geometric adjustments, such as color orientation and resizing, were carried out. This reduces the pixel size so that all the images have the same size and resolution before training them. To prevent overfitting, data augmentation is carried out with the Keras ImageDataGenerator class.

The pre-processed data were explored and visualized using Matplotlib and Seaborn libraries. 80:20 The rule was used for splitting the training and testing dataset into 1840 and 460 images, respectively. RhodaNet_layer Function is created by the function stack of Rhodanet blocks. The input parameters for Rhodanet_layer are a 224 x 224 image and a number of filters set to 16. Before instantiating the stack of layers, The input is passed to a rhodanet_layer function in which the input is passed to a convolutional layer with a number of filters: kernel size is 3, strides are 1. Batch size is set and tested for 32, 64, 128; the number of epoches is 25, 50, 100; and the total number of classes is 6 (5 disease and 1 healthy leaf). Initially, the depth is calculated as (depth= n * 6 + 2), where n = 3. The image size of 224 x 224 is normalized by dividing its size by 225. The input matrix performs convolution, and a feature map is obtained as a result. It is then passed to batch normalization, and the normalized output undergoes activation using the ReLU nonlinear activation function as shown in table. The output from the ReLU activation function is stored in the variable x. The total number of Rhodablocks is calculated using the formula int((depth-2)/6). To instantiate a stack of layers, two loops are introduced. with their range equal to the total number of rhoda blocks. Stack is the iterating variable of the first loop, and rhoda_block is the iterating variable of the next loop. For stack=0:

res_block = , strides = 1, given input = x to the rhodanet_layer function with num_filters =16 and strides =1. The output obtained from this function is stored in y.

This y is again given as input to the rhodanet_layer function with the activation specified as none. So this input goes inside the rhodanet_layer function and undergoes convolution and batch normalization without undergoing any activation.Again, the output is stored in the y variable. Using the keras.layers.concatenate function, x (which was supplied as input initially) and y (the final output obtained) are concatenated. Thus, the skip connection between the layers is carried out in this way. After concatenation, output is stored in x, and this output is again given to the ReLU activation function to convert negative values to 0 or positives. This process is carried out until stack (the outer loop iterable) equals 0.For all instances where stack >0 and rhodablock =0 , strides = 2 for the process of downsampling, and as done in previous iterations, the input is given to rhodanet_layer (conv+batch normalization+ReLU), and again, this output is given as input to the rhodanet_layer function. For each increment in the outer loop, the number of filters gets incremented by multiples of two. (i.e., 16, 32, 64...) For all instances where stack > 0 and rhodablock =0 (first layer in every stack other than the first stack), a linear projection is introduced to the shortcut connections to match the dimensions that were changed during the concatenation operation. In this linear projection process, x is given as input to the rhodanet_layer with kernel size = 1 and strides =2, activation is specified as None, and batch_normalization is specified as False. Thus, the input x will be given to rhodanet_layer, which will perform a convolution process, and the output obtained will again be stored in x. Once this stack of Rhodablocks is created, the MaxPooling process is carried out with a pool size of 8.The output is then flattened. The model's performance on the validation data will be evaluated at the end of each epoch. Once the model gets trained and all the experiments run successfully, it is then evaluated. The list of metrics values for the given test dataset It is typically used to report the final performance of a model after training and to compare the performance of different models or hyperparameters. Once the model is evaluated, test loss and test accuracy are obtained as a result. It depicts how accurate our model is when used to obtain predictions from a trained deep learning model on a test dataset. The predicted outputs can be used to evaluate the performance of the model or to make predictions on new, unseen data.

5.1 Training and Validation Accuracy:

The training and validation accuracy graph in the RhodaNet model illustrates the model's performance throughout the training and validation phases. The term "training accuracy" pertains to the accuracy of a model on the training set, which is the specific dataset employed for training the model. The accuracy is determined by comparing the predicted outputs of the model to the actual outputs in the training set. The term "validation accuracy" pertains to the accuracy of a model when applied to a validation set. This validation set is distinct from the training set and serves the purpose of assessing the model's ability to generalize. The accuracy is calculated using the validation set instead of the training set, following the same computation method as the training accuracy. The training and validation accuracy graph illustrates the progression of the model's accuracy throughout the training process, providing insights into its performance over time. The training accuracy indicates the level of fit between the model and the training data, while the validation accuracy measures the model's ability to generalize to new and unseen data. In an ideal scenario, it is expected that both training and validation accuracy will exhibit a consistent increase over time, eventually reaching a plateau. This observation suggests that the model has successfully acquired accurate knowledge or skills.

Figure 9 illustrates the accuracy rates of the training and validation data for the RhodaNet model.

5.2 Training and Validation Loss:

The loss metric quantifies the accuracy of the model's predictions in relation to the actual output. During the training process, the model iteratively adjusts its weights and biases in order to minimize the loss function. This loss function is commonly used to quantify the disparity between the predicted output and the actual output. The training loss refers to the loss function's value calculated on the training set, whereas the validation loss corresponds to the loss function's value calculated on a distinct validation set. The graph depicting the training and validation losses illustrates the progression of the training and validation loss values throughout the training process. The x-axis denotes the epochs, which correspond to the number of iterations the model has undergone, encompassing the complete training dataset. The vertical axis, commonly referred to as the y-axis, is used to represent the numerical value of the loss function. In an ideal scenario, the training loss is anticipated to decrease progressively as the model becomes more adept at fitting the training data. Conversely, the validation loss may initially decrease, but it will eventually begin to rise if the model overfits the data. Figure 10 illustrates the loss of both. The RhodaNet model utilizes training and validation data throughout its development process. It has been observed that when the training accuracy consistently improves over time while the validation accuracy remains stable or improves, it indicates that the model is effectively learning to generalize to new data and is not excessively fitting to the training data. The observed scenario aligns with the desired outcome, indicating satisfactory performance by the model. The accuracy graph of DenseNet is depicted in Figures 11 and 12. It is evident from the graph that the training accuracy is considerably high, whereas the validation accuracy is comparatively low. This observation indicates that the model is exhibiting signs of overfitting, as it is excessively tailored to the training data and lacks the ability to generalize effectively to new, unseen data. This issue may arise when the model is excessively intricate or when the available training data is insufficient to adequately capture the range of variations present in real-world data. When the training loss is significantly lower than the validation loss and the validation loss is observed to increase while the training loss decreases, it indicates that the model is exhibiting overfitting behavior towards the training data. This indicates that the model is effectively capturing the random variations present in the training data instead of the fundamental patterns, resulting in poor performance when applied to unseen data. In order to mitigate the issue of overfitting, several techniques can be employed, including early stopping, regularization, and dropout.

5.3 Confusion Matrix:

The performance of a classification model can be assessed by comparing the predicted class labels with the actual class labels.True positives (TP) refer to instances in which the model accurately predicts the positive class and the actual class is also positive. This observation suggests that the model is accurately detecting the positive cases.

False positives (FP) occur when the model makes a prediction that the class is positive, but in reality, the class is negative. The observed outcome suggests that the model is erroneously classifying certain negative instances as positive.

True negatives (TN) refer to instances where the model accurately predicts the negative class and the actual class is also negative. This observation suggests that the model is accurately detecting negative cases.

False negatives (FN) occur when the model incorrectly predicts the negative class despite the actual class being positive. This observation suggests that the model is exhibiting misclassification behavior where it incorrectly labels certain positive cases as negative. Figures 13 and 14 depict the confusion matrix for the RhodaNet model and DenseNet, where the correctly predicted samples in each class will be shown in the diagonal. Non-diagonal values represent the incorrect predictions for each class. For a model to be accurate or ideal, the values in the diagonal must be larger than the non-diagonal values.

5.4 Sensitivity Analysis-ROC Curve and AUC Score:

The ROC curve and AUC score for multi-class classification models are similar to those for binary classification models. A higher AUC score indicates a better performance of the model in distinguishing between the classes. From figure 15, it is observed that a curve closer to the upper left corner indicates better performance, while a curve that is closer to the diagonal line indicates poorer performance. The mean AUC score for the RhodaNet model and AUC scores for each class, including black spot, downy mildew, fresh leaf, powdery mildew, mosaic, and Botrytis blight, are shown in the diagram below. Classes that have higher true positive rates and lesser false positive rates are considered to be the most accurately predicted classes by the RhodaNet model.

5.5 Precision Recall Curve

The curve represents the trade-off between precision and recall for the different classes in the classification task. In a precision-recall curve, each point represents a particular threshold value for the classification decision boundary, where a sample is predicted as positive if its score is above the threshold. From figure 16, the precision-recall curve that is closer to the upper-right corner of the graph has high precision and high recall. A class with low precision and low recall will have a curve closer to the lower-left corner. By varying the epochs and batch sizes, the proposed model is analysed. Figure 14 shows the recall for ResNet, DenseNet, and RhodaNet.

Hyperparameters Tuning

Learning rate schedulers is the hyperparameter tuning technique used to find the ideal set of hyperparameters to prevent overfitting to the test set. Learning Rate Range :A test technique is used to evaluate the batch size, ranging from 32, 64, 128; epochs range from 25, 50, and 100. The AUC score, ROC curve, and map score are analyzed for optimal results.The hyperparameters tuned for the experimental setup are listed below in Table 3. The performance comparison of the ResNet, DenseNet, and RhodaNet models for Epochs 50 and Batch Size 32 is shown in Figure 17 and Table 4. The proposed model shows higher accuracy, lower error rates, and high performance in terms of speed and efficiency than existing models like ResNet and DenseNet for the Rose disease dataset.

RhodaApp-Mobile Application

The RhodaNet model saved with the tflite extension is used as a middleware for Android applications. The Android application written in Java makes use of the TFlite model to load the model inside the application. This application holds two buttons: capture and gallery. Capture Buttons uses a camera with appropriate user permission to capture the images. The captured image is directly loaded into the model after appropriate pre-processing like resizing and reshaping. The model, after processing the pre-processed image, will throw output in integers from 0 to 5, which indicates appropriate disease, for example, 0 for fresh leaf and 5 for botrytis blight. The corresponding disease is displayed with a remedy for the end-user. Similarly, the Gallery button allows us to input the image from the gallery (i.e already captured images) for processing. The model does the same thing and displays output with its remedy.Figure 18 shows the screenshots of RhodaApp

Conclusion:

In this study, a proposed automated method for disease classification and detection in roses. Black Spot, Downy Mildew, Powdery Mildew, Mosaic, and Botrytis Blight are just a few of the five disease classes that the suggested model is trained for. RhodaNet, the proposed model, is working well with a 96% accuracy rate. The model was fine-tuned using a number of use cases. To fine-tune the model, all potential values for hyperparameters including epoch, batch size, learning rate, and activation functions were evaluated. It has been noted that the performance of the model was significantly reduced when batch sizes increased by orders of 2 (16, 32, 64, etc.). Thus, a batch size of 32 is set as a constant because of its higher accuracy compared to that of 16, 64, and 128. Instead of feeding at a steady rate, a learning schedule is used. For model optimization, hybrid optimizers were also tested. Several optimizers, including Adadelta, Adam, Adagrad, and RMSprop, were merged with all probability to make a pair, however, they were unable to fine-tune the model. ReLU was found to be superior to Leaky ReLU, ELU, SoftMax, and Tanh, which were all different activation functions. As a result, this updated model, which has a 96% accuracy rate, serves as a middleware for both Web and Android applications. For those who take care of their indoor gardens and enjoy plants, the mobile application that was created works well. It is challenging to scan every plant for illness in large-scale gardens with an excessive number of rose plants. The future scope is to analyse a video recording from the large-scale farm every grid and feed that photo into a model for disease recognition and spread if any. Improvements to the current UI and UX of the app, as well as testing the model using various fruits and vegetable-based plants, must be made.

REFERENCES

1. K. He, X. Zhang, S. Ren and J. Sun, "Deep residual learning for image recognition," in 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, pp. 770-778. (2016). doi: 10.1109/CVPR.2016.90
2. G. Huang, Z. Liu, L. Van Der Maaten and K. Q. Weinberger, "Densely connected convolutional networks," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI, USA, pp. 2261-2269, (2017). doi: 10.1109/CVPR.2017.243.
3. Kamilaris, A., & Prenafeta-Boldú, F. X. "Deep learning in agriculture: A survey". *Computers and Electronics in Agriculture*, *147*, 70–90. (2018b). https://doi.org/10.1016/j.compag.2018.02.016
4. Barbedo, J. G. A. "Plant disease identification from individual lesions and spots using deep learning*". Biosystems Engineering*, *180*, 96–107. (2019). https://doi.org/10.1016/j.biosystemseng.2019.02.002
5. Alshammari, H. H., Taloba, A. I., & Shahin, O. R. "Identification of olive leaf disease through optimized deep learning approach". Alexandria Engineering Journal, 72, 213–224. (2023). https://doi.org/10.1016/j.aej.2023.03.081
6. Lamba, M., Gigras, Y., & Dhull, A. "Classification of plant diseases using machine and deep learning". Open Computer Science, 11(1), 491–508. (2021). https://doi.org/10.1515/comp-2020-0122

7. Wei, S. J., Riza, D. F. A., & Nugroho, H. "Comparative study on the performance of deep learning implementation in the edge computing: Case study on the plant leaf disease identification". Journal of Agriculture and Food Research, 10, 100389. (2022) https://doi.org/10.1016/j.jafr.2022.100389
8. Rasheed, A., Ali, N., Zafar, B., Shabbir, A., et al. "Handwritten urdu characters and digits recognition using transfer learning and augmentation with alexnet". *IEEE Access*, *10*, 102629–102645. (2022). https://doi.org/10.1109/access.2022.3208959
9. Chen H-C, Widodo AM, Wisnujati A, et al. "Alexnet convolutional neural network for disease detection and classification of tomato leaf". *Electronics*.; 11(6):951.(2022) https://doi.org/10.3390/electronics11060951
10. Borhani, Y., Khoramdel, J. & Najafi, E. "A deep learning-based approach for automated plant disease classification using vision transformer". *Sci Rep* **12**, 11554 (2022). https://doi.org/10.1038/s41598-022-15163-0
11. Abd Algani, Y. M., Marquez Caro, O. J., Robladillo Bravo, et al., "Leaf disease identification and classification using optimized deep learning", Measurement: Sensors, vol. 25, Art. no. 100643, (2023). doi:10.1016/j.measen.2022.100643

12. X. Zhang, X. Zhou, M. Lin and J. Sun, "Shufflenet: an extremely efficient convolutional neural network for mobile devices" *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Salt Lake City, UT, USA, pp. 6848-6856, (2018).doi: 10.1109/CVPR.2018.00716.
13. Zhang, J., Tian, J., Alcaide, A. M., et al., "Lifetime extension approach based on the levenberg–marquardt neural network and power routing of dc–dc converters". *IEEE Transactions on Power Electronics*, *38*(8), 10280-10291. (2023). doi: 10.1109/TPEL.2023.3275791.
14. Zhang, J., Huang, C., Chow, M. et al., "A data-model interactive remaining useful life prediction approach of lithium-ion batteries based on PF-BiGRU-TSAM". *IEEE Transactions on Industrial Informatics*, *20*(2), 1144-1154. (2023). doi: 10.1109/TII.2023.3266403.
15. Topaloglu, I. "Deep learning based convolutional neural network structured new image classification approach for eye disease identification". *Scientia Iranica*, *30*(5), 1731-1742. (2023). 10.24200/sci.2022.58049.5537
16. Khasawneh, N., Faouri, E., & Fraiwan, M. "Automatic detection of tomato diseases using deep transfer learning". *Applied Sciences*, *12*(17), 8467. (2022). https://doi.org/10.3390/app12178467
17. Mukherjee, P., Zhou, M., Lee, E., et al. "A shallow convolutional neural network predicts prognosis of lung cancer patients in multi-institutional computed tomography image datasets". *Nature machine intelligence*, *2*(5), 274-282. (2020). https://doi.org/10.1038/s42256-020-0173-6
18. Uğuz, S., Uysal, N. "Classification of olive leaf diseases using deep convolutional neural networks". *Neural Comput & Applic* **33**, 4133–4149 (2021). https://doi.org/10.1007/s00521-020-05235-5
19. Bharali, P., Bhuyan, C., Boruah, A. "Plant disease detection by leaf image classification using convolutional neural network". In: Gani, A., Das, P., Kharb, L., Chahal, D. (eds) Information, Communication and Computing Technology. ICICCT 2019. Communications in Computer and Information Science, vol 1025. Springer, Singapore. (2019). https://doi.org/10.1007/978-981-15-1384-8_16
20. Brahimi, M., Boukhalfa, K., & Moussaoui, A." Deep learning for tomato diseases: classification and symptoms visualization. Applied Artificial Intelligence", *31*(4), 299-315. (2017).
21. Batchuluun, G., Nam, S. H., & Park, K. R. "Deep learning-based plant classification and crop disease classification by thermal camera". *Journal of King Saud University-Computer and Information Sciences*, *34*(10), 10474-10486. (2022). https://doi.org/10.1016/j.jksuci.2022.11.003
22. Mohanty, S. P., Hughes, D. P., & Salathé, M. "Using deep learning for image-based plant disease detection". *Frontiers in plant science*, *7*, 1419. (2016). https://doi.org/10.3389/fpls.2016.01419
23. Sladojevic, S., Arsenovic, M., Anderla, A., et al., "Deep neural networks-based recognition of plant diseases by leaf image classification". *Computational intelligence and neuroscience*, *2016*(1), 3289801. (2016). https://doi.org/10.1155/2016/3289801
24. Kumar, M., Gupta, P., & Madhav, P. "Disease detection in coffee plants using convolutional neural network". In *2020 5th International Conference on Communication and Electronics Systems (ICCES)* (pp. 755-760). IEEE. (2020, June). 10.1109/ICCES48766.2020.9138000
25. Ashqar B, Abu-Naser S "Image-based tomato leaves diseases detection using deep learning". Int J Acad Eng Res (IJAER) 2(12):10–16. (2018)
26. Geetharamani, G., & Pandian, A, "Identification of plant leaf diseases using a nine-layer deep convolutional neural network". *Computers & Electrical Engineering*, *76*, 323-338. (2019).https://doi.org/10.1016/j.compeleceng.2019.04.011
27. Militante, S. V., Gerardo, B. D., & Dionisio, N. V. "Plant leaf detection and disease recognition using deep learning". In *2019 IEEE Eurasia conference on IOT, communication and engineering (ECICE)* (pp. 579-582). IEEE. (2019, October). 10.1109/ECICE47484.2019.8942686
28. Ozguven, M. M., & Adem, K. "Automatic detection and classification of leaf spot disease in sugar beet using deep learning algorithms". *Physica A: statistical mechanics and its applications*, *535*, 122537. (2019). https://doi.org/10.1016/j.physa.2019.122537

List of Figure Captions

List of Table Captions

Appendices

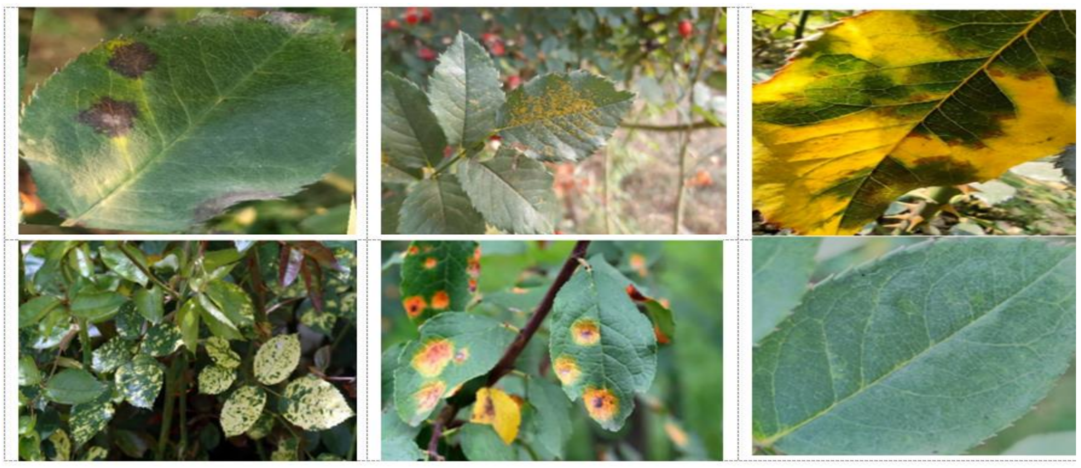


Figure 1. Rose Field’s Images captured while collecting customised data

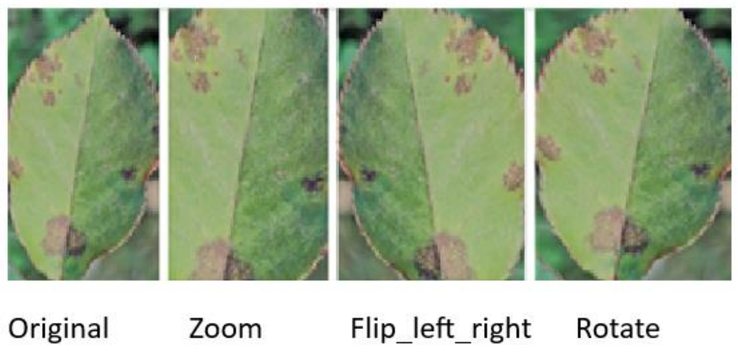


Figure 2: Data Augmentation techniques

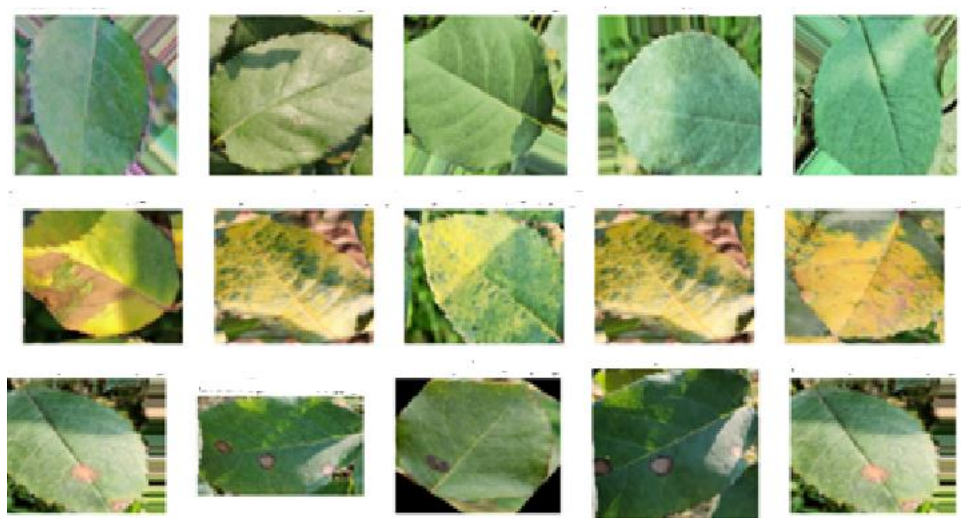
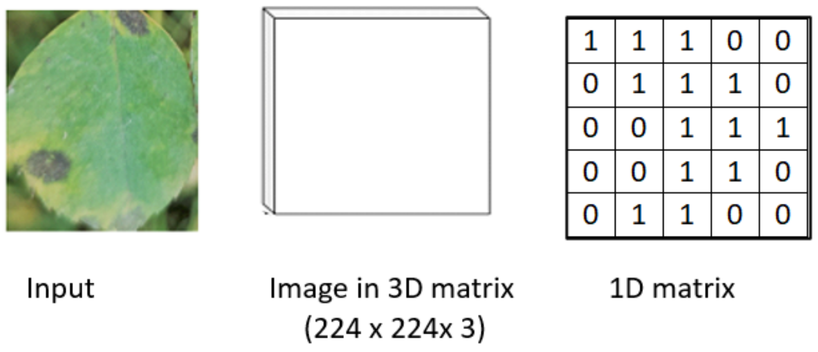


Figure 3 Data Augmented samples of Rose leaves -Fresh leaf, Downy Mildew, Black Spot



1	1	1	0	0
0	1	1	1	0
0	0	1	1	1
0	0	1	1	0
0	1	1	0	0

Figure 4 Input image to RhodaNet.

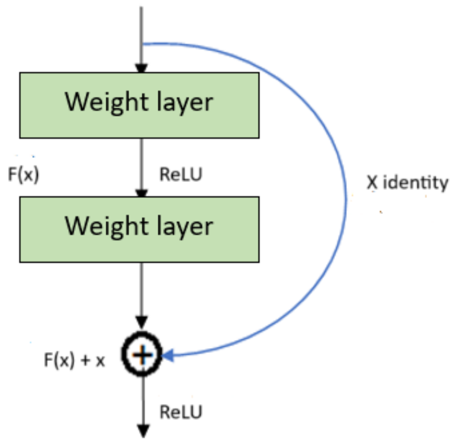


Figure 5 A single residual block

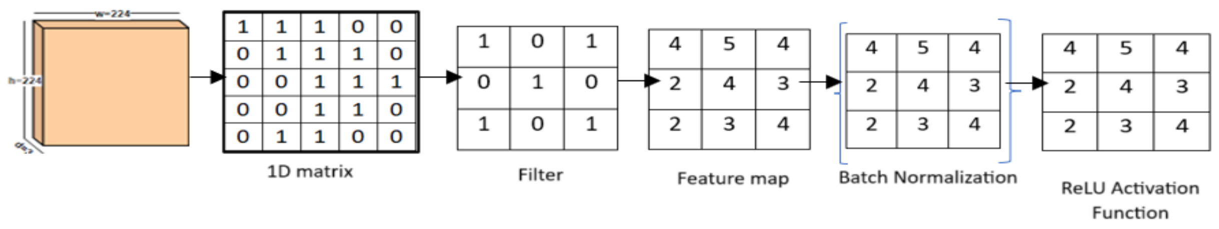


Figure 6: Single Rhoda Block

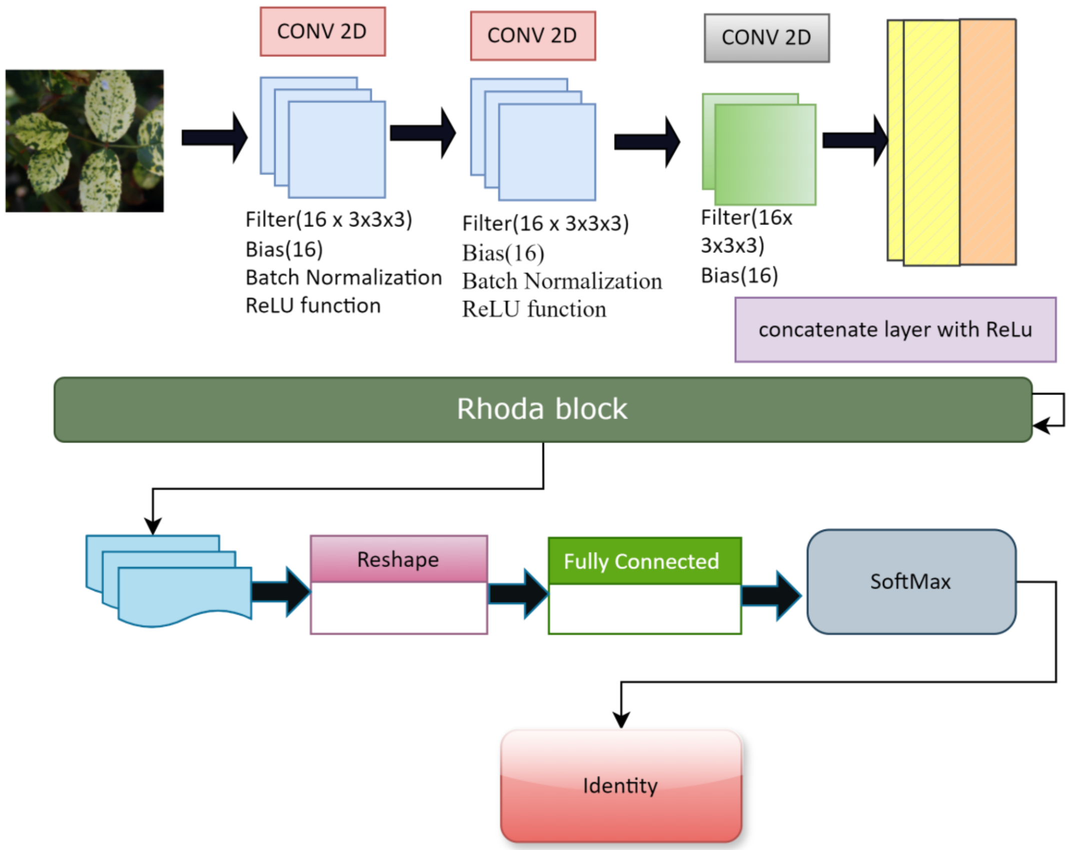


Figure 7: RhodaNet Architecture

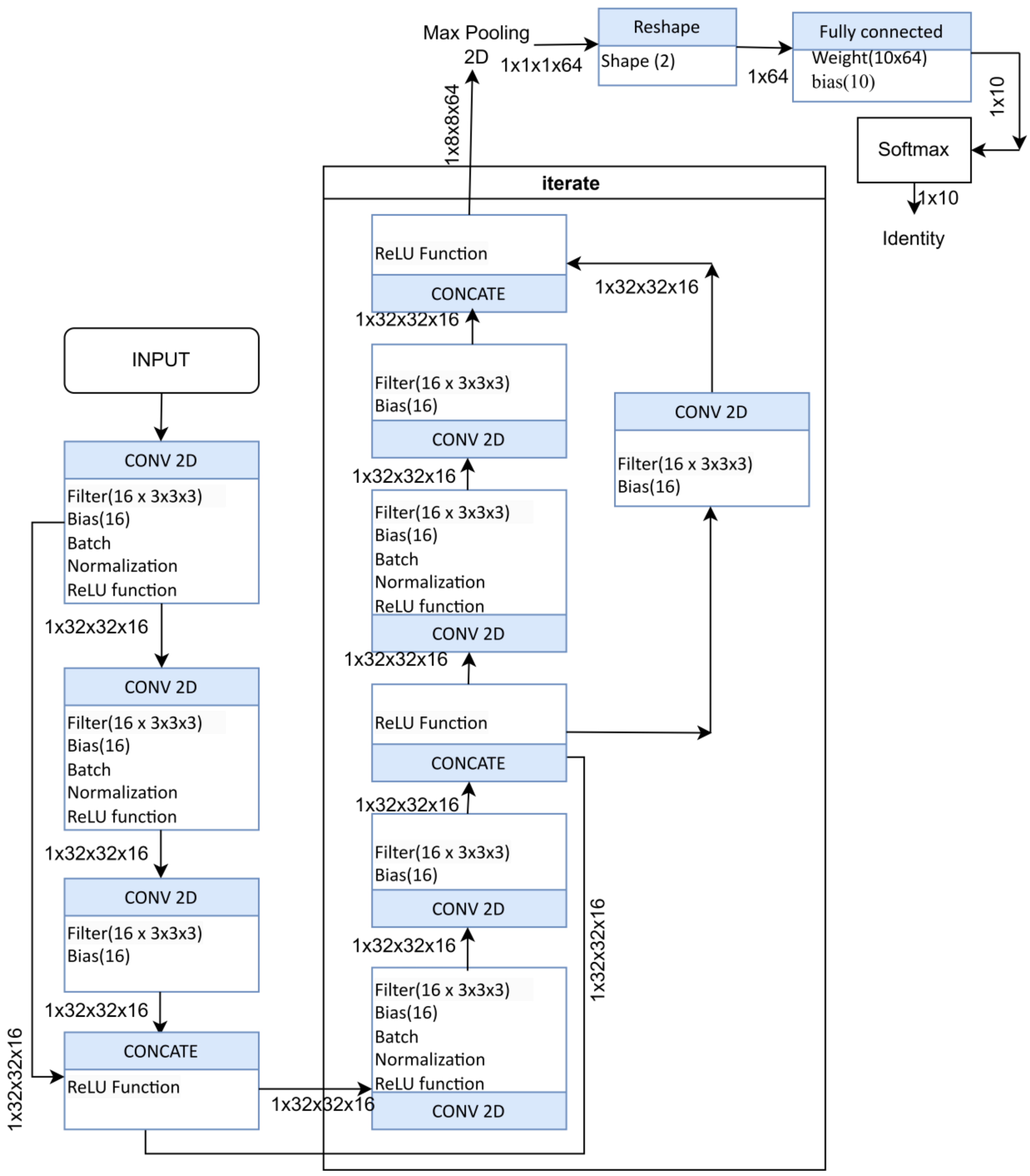


Figure 8: Proposed Model’s Layer Structure

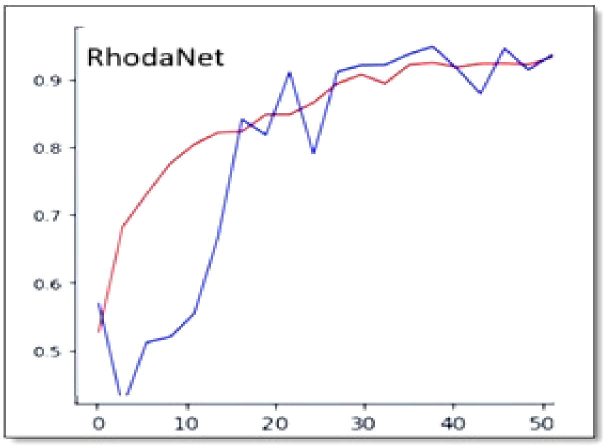


Figure 9 — Training and — Validation accuracy

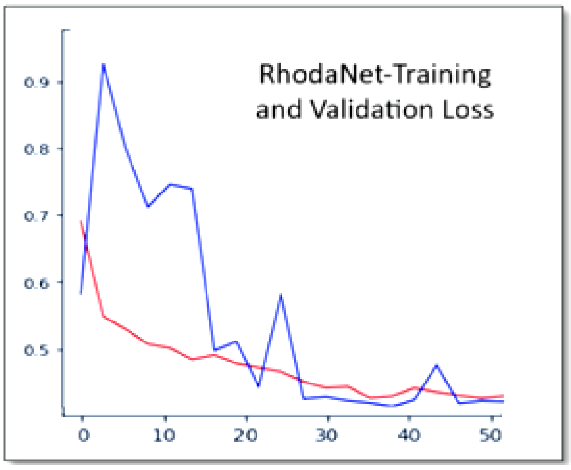


Figure 10 — Training and — Validation Loss

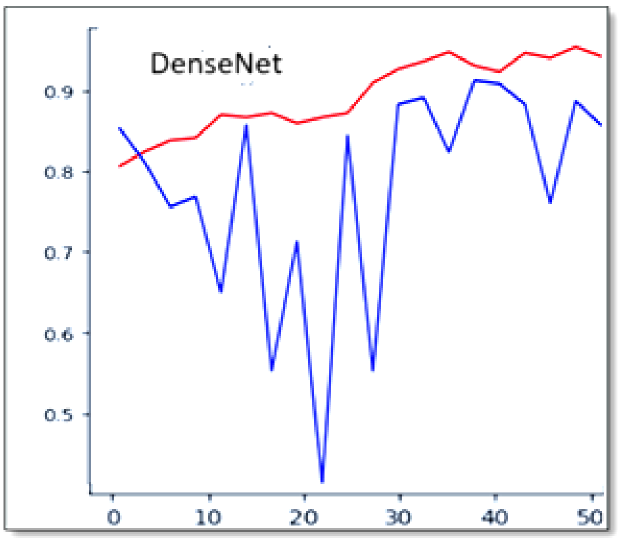


Figure 11 — Training and — Validation accuracy

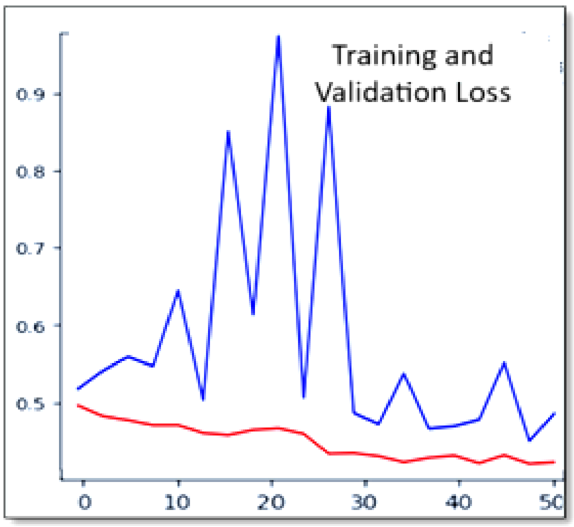


Figure 12 — Training and — Validation Loss

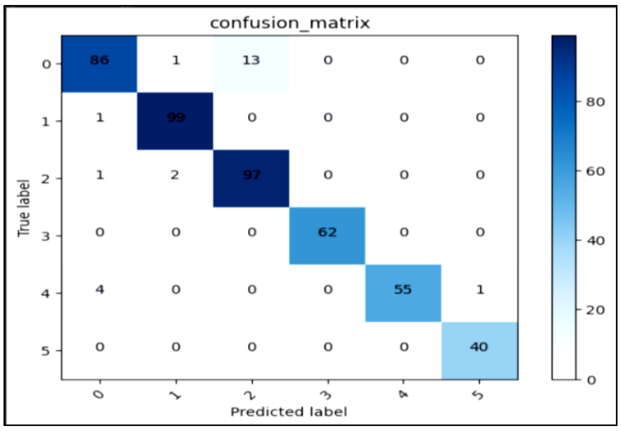


Figure 13. Confusion matrix_RhodaNet

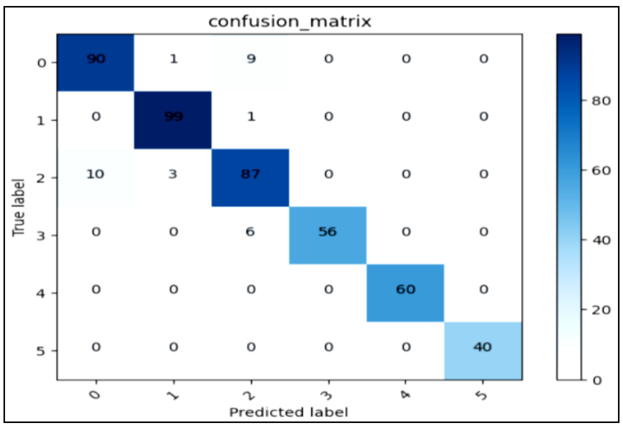
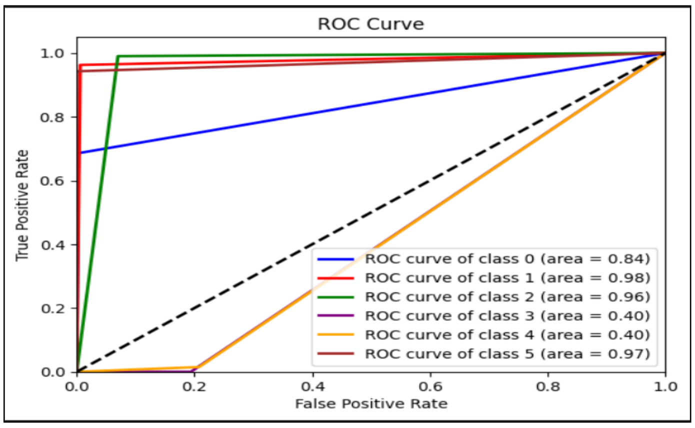


Figure 14 Confusion matrix_DenseNet



Average AUC score: 0.7595853680599435	
AUC score for class 0	0.8413927680798005
AUC score for class 1	0.9782514992503747
AUC score for class 2	0.9600174912543729
AUC score for class 3	0.40304847576211894
AUC score for class 4	0.4035519740129935
AUC score for class 5	0.97125

Figure 15: ROC Curve

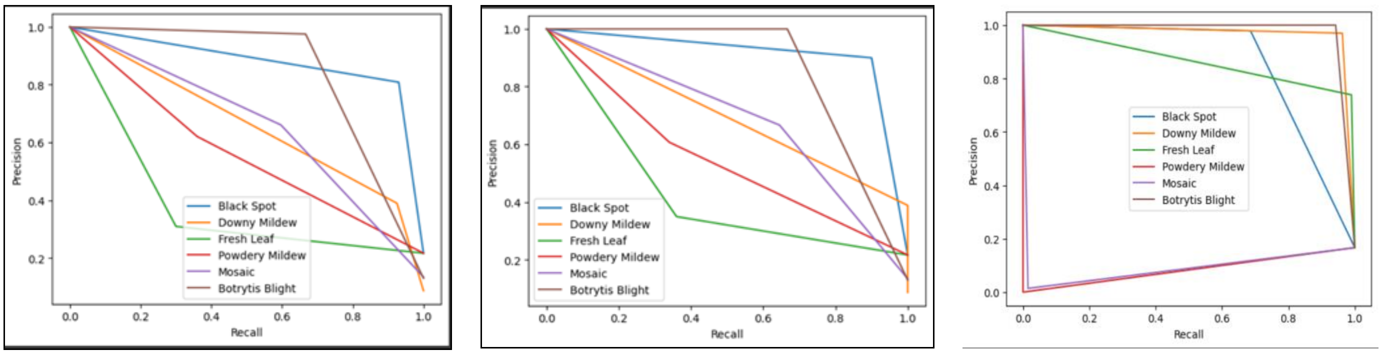


Figure 16: Recall_ ResNet, DenseNet and RhodaNet

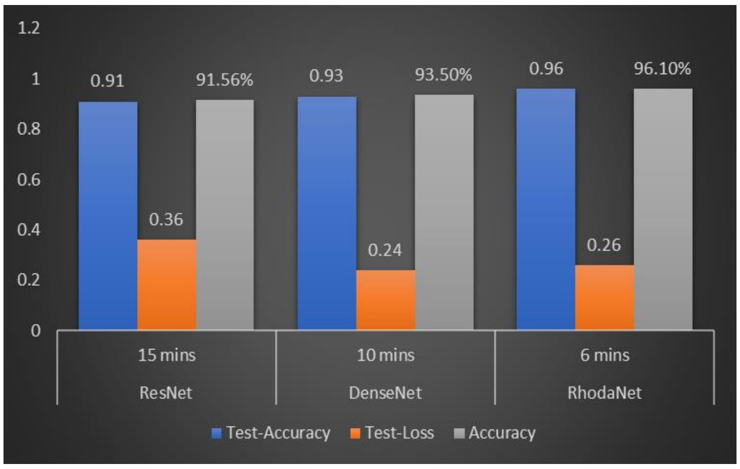
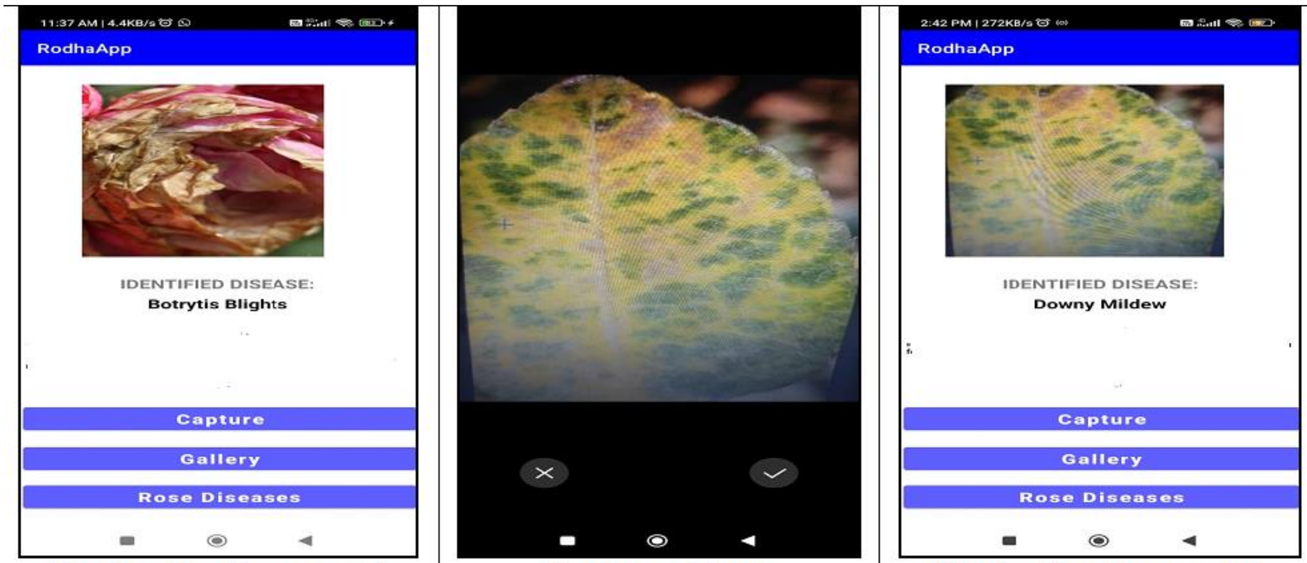


Figure 17 Performance Comparison graph



IDENTIFIED DISEASE:
Botrytis Blights

IDENTIFIED DISEASE:
Downy Mildew

Display the disease and Remedy	Capture picture	Display the disease and Remedy

Figure 18 Screenshots of RhodaApp

Table 1. Literature review of existing solutions

Author	Year	Objective	Image count	Algorithm	Accuracy
(Mohanty et al. 2016)	2016	Plant Disease Detection	54306	AlexNet	93.88
(Sladojevic et al. 2016)	2016	Plant Disease Detection	2589	CNN	96.3
(Kumar M et.al)	2020	Coffee Leaves Disease Detection	3700	CNN	92.88
(Ashqar and Abu-Naser 2018)	2018	Tomato Leaf Disease Detection	9000	CNN	95.54
(Geetharamani andPandian 2019)	2019	Plant Leaves Disease Identification	54305	CNN	96.46
(Militante et al. 2019)	2019	Tea Leaves Disease Identification	4980	VGG16	90
(Ozguven and Adem 2019)	2019	Sugar Beet Leaves Disease Detection	155	Faster RCNN	92.89

Table 2: Data Distribution

Rose Disease Name	Training images	Testing images	Total images
Black spot	400	100	500
Downy Mildew	400	100	500
PowderyMildew	400	100	500
Mosaic	400	100	500
Botrytis Blight	400	100	500
Fresh Leaf	400	100	500

Table 3 Hyperparameters

Parameters	***Values***
Optimizer	RMSprop
Batch size	32
Epochs	50
Activation function	Softmax/Soft plus
Normalization	Batch normalization
Learning rate	0.0001

Table 4 Performance Comparison

Epochs:50 Batch Size:32	**ResNet**	**DenseNet**	**RhodaNet**
Time Taken	15 mins	10 mins	6 mins
Test-Accuracy	0.91	0.93	0.96
Test-Loss	0.36	0.24	0.26
Accuracy	91.56%	93.50%	96.10%

Ms.S.Pudumalar is working as an Assistant Professor in the Department of Information Technology at Thiagarajar College of Engineering, Madurai. She received her UG and PG degrees with the specialization of Computer Science and Engineering. She has a passion for teaching as well as Research. She is an active member of pedagogical activities including Curriculum Design and Teaching Learning Process. Her main areas of research interest cover Machine learning algorithms, Deep learning architectures, and Applications of Deep learning for agriculture.

Muthuramalingam Sankayya received the bachelor degree from the CSE,Bharathidasan University in 1997, the Master of engineering from Kamaraj Ubiversity in 2002 and the Ph.D degree in Information and communication engineering from Anna University,Chennai in 2012.He is currently working as a Professor with the department of Information Technology,Thiagarajar college of Engineering,Madurai,India.He has published various quality articles in the reputed Journals.His main thrust areas are Computer Networks, Mobile Communication and IoT.