Journal of Agricultural Sciences
(Tarim Bilimleri Dergisi)

J Agr Sci-Tarim Bili
e-ISSN: 2148-9297
jas.ankara.edu.tr

JAS

Mobile Device-Based Detection System of Diseases and Pests in Rose Plants Using Deep Convolutional Neural Networks and Quantization

Burhan Duman[a]

[a]*Department of Computer Engineering, Isparta University of Applied Sciences, Isparta, TURKEY*

ARTICLE INFO
Research Article
Corresponding Author: Burhan Duman, E-mail: burhanduman@isparta.edu.tr
Received: 12 July 2024 / Revised: 13 October 2024 / Accepted: 27 October 2024 / Online: 25 March 2025

Cite this article
Duman B (2025). Mobile Device-Based Detection System of Diseases and Pests in Rose Plants Using Deep Convolutional Neural Networks and Quantization. *Journal of Agricultural Sciences (Tarim Bilimleri Dergisi),* 31(2):302-318. DOI: 10.15832/ankutbd.1514972

ABSTRACT
In agriculture, the rapid and accurate identification of plant diseases and pests is crucial for maintaining the quality and yield of agricultural products. This study focuses on detecting diseases and pests affecting *Rosa damascena* Mill. plants through an ensemble learning approach and deploying the model in an Android mobile application-a rarity in similar research. A new dataset was created using images from the natural habitat and season of *Rosa damascena* Mill., covering seven different diseases and pests.

For this approach, pre-training was performed with mixed-Convolutional Neural Network (CNN) models DenseNet169, ResNet152, MobileNetV2, VGG19, and NasNet. DenseNet169 and MobileNetV2, which are the models with the highest classification success obtained from mixed-CNN models, were combined in the new model by fine-tuning with the ensemble learning method. In the performance tests of the model, an accuracy of 95.17% was obtained.

In addition, this study introduces an Android mobile application integrating these models, a distinctive feature compared to other similar studies. The best performances of these models, DenseNet169 and MobileNetV2 in both flat buffered and quantized forms, were performed separately on a computer, a physical mobile device, and an Android emulator. MobileNetV2 outperformed DenseNet169 (2271 ms) by having the lowest average inference time (301 ms) on mobile devices. These results demonstrate the effectiveness of using a mobile device to detect rose plant diseases and pests efficiently in natural environments.

Keywords: Mobile application, Image classification, Deep łearning, Plant disease and pests, *Rosa damascena* Mill

1. Introduction

Agriculture is one of the most important and significant resources for the country's economy. However, agricultural products are severely affected by plant diseases. These challenges can be addressed by using modern agricultural technologies (Sharma & Singh 2015). These technologies, which are considered costly compared to traditional methods, can be made cost-effective by applying machine learning and deep learning approaches in image processing applications. Additionally, learning about leaf diseases can yield better results than traditional methods (Khitthuk et al. 2018).

Foliar diseases in plants depend on factors such as survival and climate, but most arise from attacks by viruses, bacteria, and fungi. Pathological diseases cause 85% of foliar diseases and most farmers in developing countries rely on traditional methods such as visual observation, physiological observation, manual inspection, laboratory tests and tissue analysis, which are more labor and time consuming. Typically, manual detection of foliar diseases is considered to be specialized and unsatisfactory due to the human factor. Therefore, it is becoming increasingly common to introduce a modern, deep learning-based approach to disease detection (Mutka & Bart 2015; Mahlein 2016; Weerakoon et al. 2017; Ferentinos 2018).

Rosa damascena Mill., which is called the “Isparta Rose” in Turkey, is a valuable industrial plant used in the food, medicine, cosmetics, and various other sectors. Introduced to Isparta from the Kazanlık region of Bulgaria in the late 19th century, *Rosa damascena* Mill. is now cultivated in the provinces of Isparta, Burdur, Afyon, and Denizli. This plant is generally harvested in May and June, with regional and seasonal variations, and should be harvested early in the morning to optimize oil production (Ersan & Başayiğit 2022). *Rosa damascena* Mill. thrives in temperate climates and is a shrub-type industrial plant known for its pink color, intense fragrance, and flowers that measure an average diameter of 7 cm and weigh 2-3 grams (see Figure 1).

Figure 1- *Rosa damascena* Mill

Products such as oil obtained from the Isparta oil rose have applications in various fields, particularly in the cosmetics and pharmaceutical sectors. Turkey generates an income of approximately 10 million dollars from rose products annually (Bıtrak & Hatırlı 2022). As with every plant species, the rose plant can also be affected by diseases and pests. These infections can lead to a reduction in flower yield and quality (Karanfil 2021). Managing various diseases and pests on *Rosa damascena* Mill. is essential. Of the numerous diseases, the main ones are rose rust (*Phragmidium mucronatum*) and rose powdery mildew (*Sphaerotheca rosae*). The most significant pests include cochlea (*Rhodococcus perornatus*), proboscis (*Mecorhis umgaricus*), sprout wasp (*Syrista parreyssi*), sprout aphid (*Ardis brunniventris*), scissors beetle (*Perotis chlorana*), rose moth (*Cnaemidophorus rhododactyla*), comma aphid (*Leulmidosa*), rose aphid (*Macrosiphum rosae*), spider mite (*Tetranychus urticae Koch*), and leafhopper (Thrips meridionalis Priesner) (Baydar 2016).

Poor timing of control against diseases and pests in *Rosa damascena* Mill. results in biomass losses and a decrease in the oil content, hence poor quality products with lower yields (Yılmaz 2015). Therefore, early detection of diseases and pests in RDM and prompt application of spraying will help prevent adverse outcomes as much as possible. Failure to do so may result in the spread of these diseases and pests to other rose plants, causing significant damage (Fazili et al. 2024).

Farmers typically rely on traditional methods, primarily visual observation, to identify diseases in *Rosa damascena* Mill. Visual classification with the naked eye requires prior knowledge and experience, especially in diagnosing leaf-based diseases. This manual process can be time-consuming, and diseases found in plants may be misidentified. Inaccurate disease detection can lead to incorrect treatment.

In the literature, academic studies on rose plants can be classified into two categories: cut rose and oil rose. Cut roses are often grown for landscaping and commercial sales. The main characteristic that distinguishes oil rose from cut rose is its ability to be industrially processed and converted into various commercial products. Deep learning methods play a significant role in academic studies on cut roses. For instance, Swetharani & Prasad (2021) used Convolutional Neural Networks to classify leaf diseases with an accuracy of 97.30% using images from cut rose leaves. Rajbongshi et al. (2020) utilized the MobileNet architecture to classify four rose disease types with 95.63% accuracy, using 400 images. Ma et al. (2020) applied CNN-based Alexnet, VGG16, and neural discriminative dimensionality reduction (NDDR) methods to detect Black Spot in roses in China. They found that the NDDR-CNN model yielded the best results. Khaleel et al. (2022) employed a Deep Learning-based CNN model to detect four diseases in rose plant leaves, achieving an accuracy rate of 64.35%. Yin et al. (2021) distinguished between the early, middle, and advanced stages of Black Spot disease in the leaves of Chinese roses and healthy leaves, achieving a success rate of 84.16% with the Faster R-CNN technique.

In recent years, many computer vision and deep learning methods have been developed to address the challenges arising from manual techniques (Chen et al. 2020). These methods focus on creating an image database, performing feature extraction, and conducting classification analysis. Fuentes et al. (2018) using a database of 5,000 images and deep convolutional neural networks, achieved a 96% success rate in tomato disease and pest identification. Zhong & Zhao (2020) achieved a 93.30% F1-Score classification success with the DenseNet-121 deep convolution network using 2,462 apple leaf images across six different diseases. Sethy et al. (2020) used a database of 5,932 rice leaf images covering four diseases in their study. The study, which employed thirteen different CNN models, reached an F1-Score of 97.96% with MobileNetV2 feature extraction and SVM classification.

In this study, a system is presented to address the above-mentioned disadvantages and enable quicker, more accurate classification performance. The system, which includes computer vision and a mobile application, offers an accessible solution that aims to detect diseases faster and more accurately without the need for prior knowledge and experience. As a result, more precise measures can be taken in a short period. Timely spraying will reduce pesticide residues that can negatively impact the export of other products, especially rose oil, and prevent economic losses due to decreased harvests.

The main contributions of this study to the literature are:

- The creation of a new dataset on plant diseases and pests affecting the oil rose, which is one of the most important economic products in Turkey and the world.
- Transfer learning and fine-tuning methods were applied to the CNN models to achieve optimal results in classification-based CNN models
- An ensemble method using mixed-CNN models was used to improve the classification performance of the detecting system.
- A mobile application was developed that integrates flatbuffered and quantized versions of the models to utilize the performance of the best models in real-time and without network access in the field.

The second section of the paper describes the characteristics of the dataset created, the deep learning algorithms used in the study, ensemble learning, and the performance evaluation criteria. The third section examines the results of the algorithms used in the experimental study, while the fourth section discusses the conclusions of the study and future research directions.

2. Development of the Proposed System

The architecture for the proposed disease and plant pest classification approach, including the model training, refinement, and testing processes, is given in Figure 2. For the training and testing of the models, a study-specific original dataset from the rose field was created and labeled. Classification performances of the prepared dataset were observed in a mixed manner with CNN-based approaches. After the obtained classification predictions, the mixed CNN-based model was selected by applying the ensemble learning method on CNN-based algorithms to ensure maximum accuracy performance. During this process, the weights of each model were saved separately in a .h5 file format. The weight file of the models was converted to *.tflite* format and transferred to the developed mobile application.

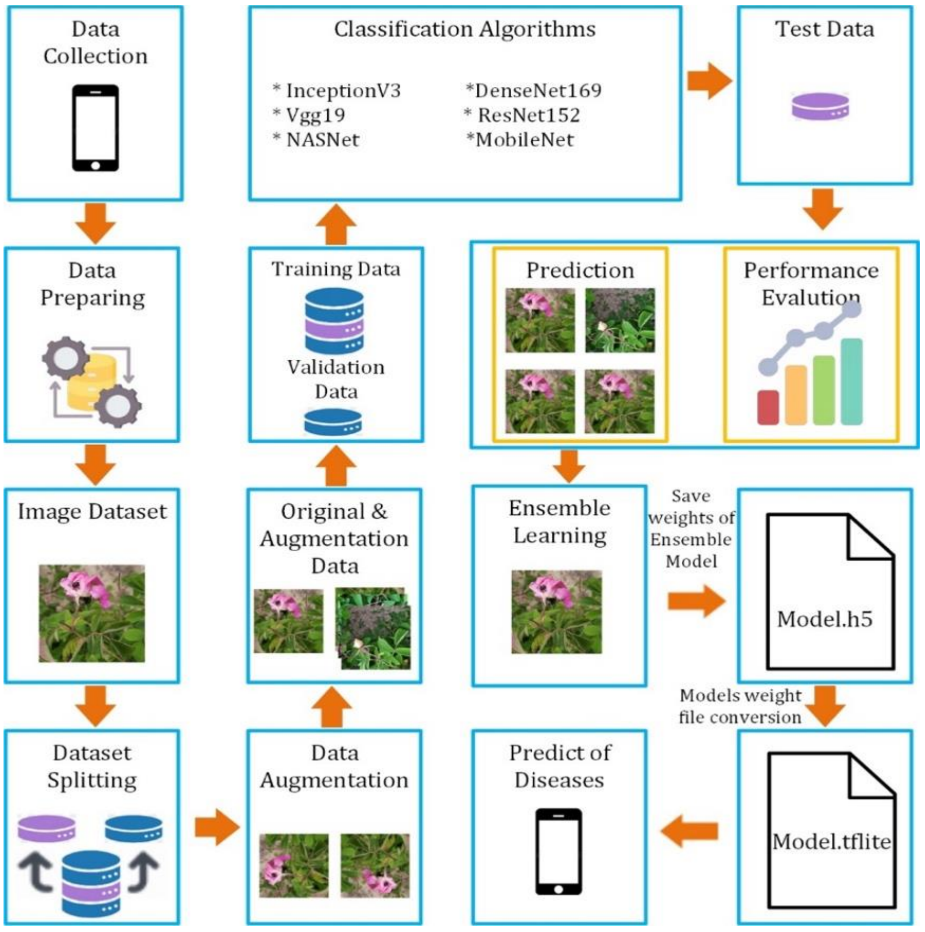


Figure 2- Schematic illustration of proposed system architecture

2.1 Image datasets

The dataset for this study consists of images collected from various rose gardens located in different districts and villages in the Isparta province between May & July 2022. In total, 567 images were captured under natural conditions using a Samsung SM-M317F mobile phone. The dataset includes images of rose plant diseases and pests, such as *Botrytis spp.*, *Ardis brunniventris*, *Mecorhis umgaricus, Rhodococcus perornatus, Sphaerotheca rosae, Phragmidium mucronatum*, and *Macrosiphum rosae*, as well as healthy rose plants. The images were labeled with the assistance of an expert agricultural engineer and by referencing relevant literature. A selection of these images is shown in Figure 3.

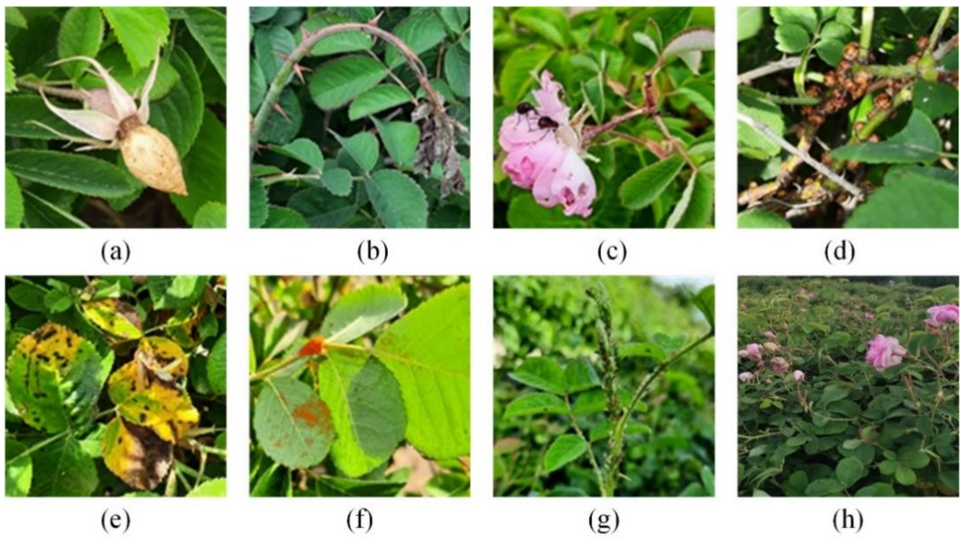


Figure 3- Healthy and diseased variants in the dataset (a) *Botrytis spp.*, (b) *Ardis brunniventris*, (c) *Mecorhis ungarica*, (d) *Rhodococcus perornatus*, (e) *Sphaerotheca rosae*, (f) *Phragmidium mucronatum*, (g) *Macrosiphum rosae*, (h) Healthy

2.2. Data preparing

For the experimental study on the deep learning-based classification of Isparta oil rose diseases, a dataset was created using leaf images from seven diseased plants and one healthy plant. Images collected from the natural environment and labeled according to disease types were carefully reviewed to exclude unsuitable images; these images were subsequently excluded from the dataset. To optimize processing speed, the high file size images in the dataset, originally sized at 3 456 × 3 456 pixels, were resized to 600 × 600 pixels. These images were not cropped. The 567 images in the dataset were randomly selected and divided into training (65%), validation (15%), and test (20%) datasets. In the training and testing processes of mixed CNN models, each image was extracted using Equations 1 - 5 to create ideal images and visual diversity.

$$area = X_{height}\ x\ Y_{width} \tag{1}$$

$$X_{height} = picture.boundingRect(first\_pic) \tag{2}$$

$$Y_{width} = picture.boundingRect(first\_pic) \tag{3}$$

$$Aspect_{ratio} = \frac{X_{height}}{Y_{width}} \tag{4}$$

$$extent = \frac{object\_area}{bounding\_rectange\_area} \tag{5}$$

These formulas calculate ratios between an object and its surrounding rectangle in dataset images, hence enabling the detection of the size. This is especially useful in resizing images that don't fit the 600×600 target size. The advantage of this is that the method will preserve the visual integrity of the objects without degrading image resolution. Stretching images while preserving aspect ratios will keep pictures from deforming through the processing. These operations increase the consistency of data and results for more reliable analysis. This ensures that all the images in the dataset are homogeneous before feeding into any CNN model.

Data augmentation techniques were applied to the images within the dataset to increase the variety. In total, 2 633 images were generated for training and testing, utilizing seven different augmentation techniques: shear_range, fill_mode, horizontal_flip, height_shift_range, width_shift_range, rotation_range, and zoom_range. Table 1 illustrates the disease categories used in the study, their labels, and the number of training, validation, and test images in the original and augmented data.

Table 1- Labels, disease categories and the count of images in these categories

		Image counts (without data augmentation)			*Image counts (with data augmentation)*		
Label	*Category*	***Train***	***Validation***	***Test***	***Train***	***Validation***	***Test***
0	Healthy	25	6	6	115	29	28
1	Ardis brunniventris	38	9	9	176	42	42
2	Botrytis spp.	44	11	11	204	51	51
3	Macrosiphum rosae	41	10	10	187	48	48
4	Phragmidium mucronatum	51	13	12	236	59	58
5	Rhodococcus perornatus	61	16	15	282	73	72
6	Mecorhis ungarica	70	18	17	325	82	81
7	Sphaerotheca rosae	50	12	12	234	55	55
	Total	380	95	92	1759	439	435
			567			**2633**	

The dataset categories show an unbalanced distribution of images due to the uneven prevalence of diseases and pests during the time period when the dataset images were collected from the natural environment.

2.3. Integrating Mixed-CNN models to system

The disease detection model utilizes transfer learning. Knowledge gained from CNN algorithms, previously trained with large datasets and proven effective, was fine-tuned for new datasets and different classes by adjusting network parameters.

In the algorithm of the system, the dataset and hyperparameters serve as input variables for each designed CNN model. The base CNN model, with initial parameters (weights, input shape, pooling, etc.), is loaded from the Keras applications library. Since the prior learning dataset differs from the dataset used in the target task, the first layers are not trained, allowing prior knowledge to be preserved.

Subsequent layers in the base model and newly added layers (pooling, dropout, dense) are trained with the dataset and hyperparameters specific to the target task. As a result, the model parameters are updated and learning occurs based on the new dataset. The weights providing the best performance during training are saved. Performance metrics are derived from evaluating the trained model with test data it has not seen before. Performance can be gauged by assessing the metrics obtained for each model.

At this point, ensemble learning was applied using basic deep learning models. Through the majority voting method, $C(n,r) = n!/((r!\,(n-r)!))\ p$ predictions from n deep learning models with different combinations of r elements ($1 < r \leq n$) were combined and the accuracy of the ensemble model was measured. This process led to the ensemble model achieving the highest classification performance.

2.3.1. Deep convolutional neural networks

Deep learning models are widely applied across various research domains, including agriculture, health, and space studies, for tasks such as object detection and classification. In agriculture, they are used for predicting crop yield, while in health, they aid in the early detection of diseases. Moreover, deep learning models contribute to space studies through the classification of constellations (Jung et al. 2021; Oikonomidis et al. 2023; Yu et al. 2023).

These algorithms, which contain multiple hidden layers, excel at tasks such as prediction, classification, speech recognition, visual object recognition, object detection, and drug discovery. Notable deep learning algorithms include Convolutional Neural Networks (CNNs), Generative Adversarial Networks (GANs), Long-Short Term Memory (LSTM) networks, Multilayer Perceptrons (MLPs), and Self-Organizing Maps (SOMs). Among these, pre-trained CNN algorithms, in particular, have demonstrated substantial success in classification tasks (Abulwafa 2022; Dewangan 2023).

The use of pre-trained CNN algorithms for classification is often referred to as transfer learning. This approach facilitates the rapid achievement of successful results with relatively small datasets. By utilizing knowledge gained from models trained on different datasets, transfer learning enables prediction, feature extraction, and fine-tuning in new applications (Iman et al. 2023; Vrbančič & Podgorelec 2020).

Researchers use different pre-trained CNN architectures to train models with their own data in classification applications. In this experimental study, InceptionV3, DenseNet169, ResNet152, VGG19, MobileNet, and NASNet, which are popular pre-trained CNN models in this field, were used to classify the disease and health status of oil rose in Isparta province.

DenseNet

DenseNet (Dense Convolutional Network) has been brought to the literature by Cornwell University, Tsinghua University and Facebook AI Research (FAIR). It received the best article award at the 2017 CVPR with more than 2000 citations. It is a network architecture where each layer is connected to every other layer immediately (Huang et al. 2017). The feature maps of all preceding layers are handled as separate inputs for each layer, while the feature maps of each subsequent layer are passed as input to each layer's own feature maps. This connection model provides the most advanced accuracies. DenseNet-121, DenseNet-169, DenseNet-201, and DenseNet-264 models are used where DenseNet's different Dense Blocks are used (Pan et al. 2019).

InceptionV3

Szegedy et al. (2015) confirmed that InceptionV3 is the Winner of the ILSVRC 2014 image classification competition. In the model of this network, there are symmetrical and asymmetrical blocks. Each block has different convolution, mean or maximum pooling, and concats layers. On the last side of the network, there is the Softmax layer for convolution, average pooling, forgetting, full coupling, and classification, respectively.

MobileNet

TensorFlow's first mobile computer vision model is the MobileNet design, which was put forth in 2017. Deeply separable convolutions are used by MobileNet. As a result, compared to a network with regular convolutions of the same depth, the number of parameters is considerably reduced. A neural network called MobileNet is made up of highly separable convolution from existing convolution. A concept called "deeply separable convolution" is used to factor an existing convolution and entails two steps; point convolution conducts the sum processing, while deep convolution filters the convolution. Deep convolution applies a single filter to every channel of incoming data and then outputs the final product (Lee 2020).

NASNet

Neural Architecture Search (NAS) was developed by Google Brain and, in 2017, achieved a classification accuracy on ImageNet that was 1.2% higher than that of previously published studies (Zoph & Le 2016). NAS algorithm, which consists of three components: Search Space, Search Strategy, Performance Estimation Strategy, are scalable CNN architectures and consists of reinforcing learning method and separable convolution and reduction blocks. In addition, NAS algorithms can be classified into three different categories: Evolutionary Computation (EC), gradient, and Reinforcement Learning (RL) (Liu et al. 2021).

ResNet

ResNet was introduced to the literature in 2015 by Kaiming He, Xiangyu Zhang, Shaoqing Ren and Jian Sun. More layers have been added over the years for deep learning architectures to solve more complex tasks. Adding more layers causes accuracy saturation, rapid drops, and increase in the training error. The reason for this is the degradation/optimization problem encountered in the optimization of deep models. To solve this problem, ResNet architecture, which uses a 34-layer flat network architecture with less filters and lower complexity than VGG networks based on VGG-19 architecture, has been developed. The architecture was then converted into a residual mesh by adding jump links or residual blocks to this flat mesh (Reddy et al. 2021).

VGG19

The VGG19 architecture is a 19-layer version of the VGG (Visual Geometry Group) architecture, consisting of 16 convolutional layers and three fully connected layers. The input image is a fixed-size 224×224 RGB image. The VGG19 architecture includes similar layers which have weight sharing. This architecture uses 3×3 convolutional filters and usually contains 5 convolutional layers with filter numbers 64, 128, 256, 512 and 512 respectively. In addition, the VGG19 architecture includes 3 fully connected layers and uses the SoftMax function for multiclassification (Simonyan & Zisserman 2014).

2.3.2. Ensemble learning

Ensemble learning is a machine learning technique used across various artificial intelligence applications to improve generalization by combining multiple base learners. This method enhances average model predictions by incorporating diverse training data and models, with the final prediction based on majority voting among the classifiers (Yang et al. 2023). Ensemble learning seeks to surpass traditional single learning methods, benefiting from factors such as mitigating overfitting, computational advantages, and enhanced representativeness. Additionally, ensemble methods address challenges in machine learning, such as class imbalance, concept drift, and high-dimensional data. Ensemble classifiers combine the outputs of multiple base models using distribution aggregation methods, such as summing conditional probability vectors, to arrive at the final class prediction. Mathematical formulations for ensemble classifiers are described, with Equation 6 defining the final class prediction and Equation 7 detailing the aggregation process (Sagi & Rokach 2018).

$$\hat{y} = argmax(\varphi(x_i)) \quad (6)$$

$$\varphi(x_i) = \sum_{k=1}^{K} f_k(x_i) \quad (7)$$

Where; x_i represents the input features of a single example, $\hat{y} \in R$ for regression problems and $\hat{y} \in Z$ for classification problems.

In Equation 6, the model assigns a class label corresponding to the maximum score $\varphi(x_i)$returned by the function $\varphi(x_i)$.The function $\varphi(x_i)$ is further elaborated in Equation 7 as the sum of individual contributions from different classifiers or feature functions. This allows us to view the final prediction as a sum of many feature-specific evaluations; hence, more robust decision-making is achieved.

2.3.3. Adding transfer learning to fine-tunning

Fine-tuning is the process of retraining a pre-trained model for a specific task, typically trained on a larger dataset. The model is adjusted to perform better on a different task by using the general features it has learned. The convolutional layers of the model are frozen, allowing training to be conducted solely on the newly added dense layers.

Due to the small size of the dataset and the necessity of achieving high performance with faster training, state-of-the-art deep learning models were implemented using the transfer learning method. As part of the fine-tuning process, the base convolutional layers of models pre-trained on large-scale image datasets were frozen. These layers have learned to extract general features, such as edges and textures, from input images, which can be utilized for various classification tasks. The weights of these frozen layers remain unchanged during the new training process, thereby reducing the computational cost of the training.

Since the labels from the original classification task differ from the class labels used for disease and pest detection, new dense layers that are specific to the dataset have been added in place of the original classification layers. As shown in Table 2, this new model architecture has been redesigned to include *global average pooling*, *dropout*, *dense* and *batch normalization* layers sequentially. The ReLU activation function was employed in the intermediate layers, while *softmax* was used in the output layer. The number of classes in the output layer was adjusted according to the number of diseases and pests (Figure 4).

Table 2- New layers and parameters added for fine-tuning

Modified Model Architecture	
Basemodel (trainable = False)	*DenseNet169, MobileNetV2, VGG19, NASNet, InceptionV3, ResNet152*
New added layers and their parameters	keras.layers.*GlobalAveragePooling2D(),* keras.layers.*Dropout(0.4),* keras.layers.*Dense(256, activation='relu'),* keras.layers.*BatchNormalization(),* keras.layers.*Dropout(0.3),* keras.layers.*Dense(128, activation='relu'),* keras.layers.*BatchNormalization(),* keras.layers.*Dropout(0.2),* keras.layers.*Dense(8, activation='softmax')*

Fine-tuned models were tested using the Adam and SGD optimization algorithms. The categorical cross-entropy loss function was employed, and accuracy was selected as the metric to evaluate the model's performance. Various epoch values were tested to determine the most suitable model. Controls such as *reduce_lr_on_plateau*, *early_stopping*, and *model_checkpoint* were incorporated into the model to update the learning rate and prevent overfitting. While the new layers were updated during the training process, the frozen convolutional layers continued to extract general features from the input images. After assessing the training and validation accuracy, the classification performance was measured using test data. A confusion matrix was generated, and metrics such as precision, recall, and F1-score were calculated.

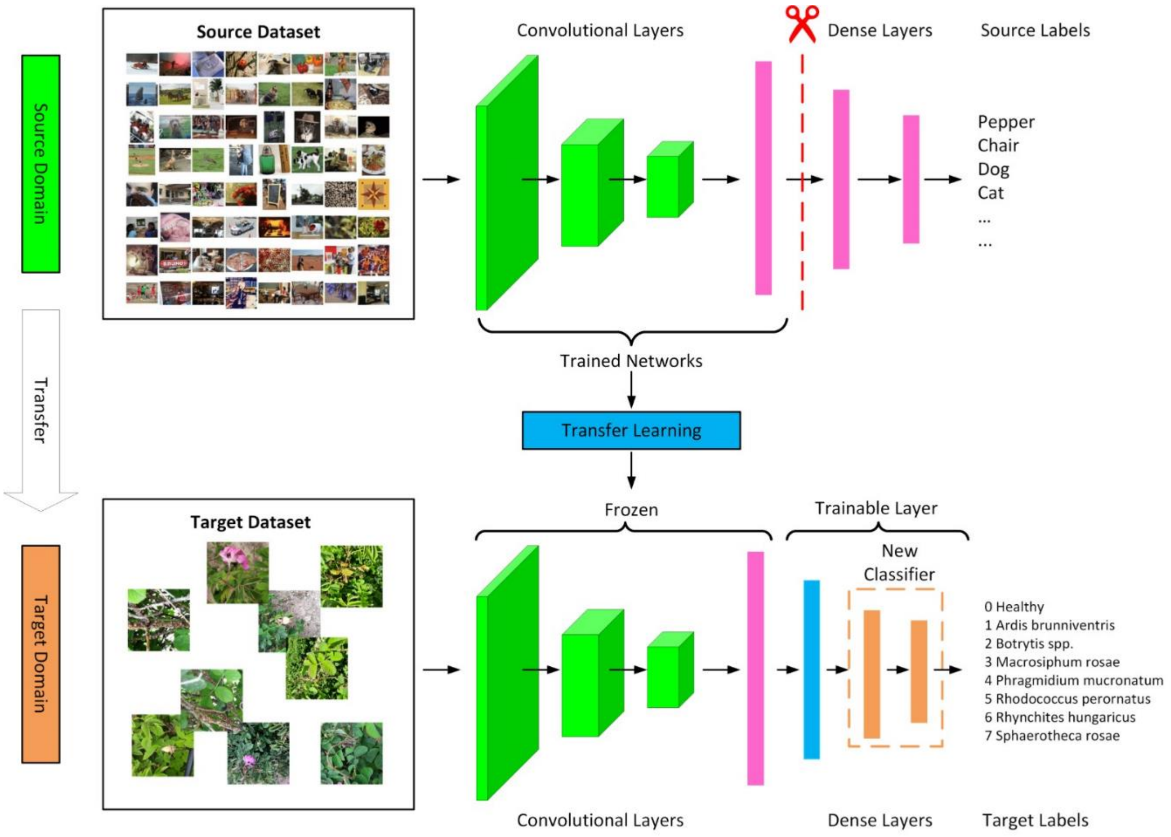


Figure 4- Demonstration of the application of the transfer learning (adopted from Tao et al. 2020)

2.3.4. Quantization

Quantization is an essential technique used in Deep Neural Network (DNN) models to reduce computational and storage requirements, making them suitable for resource-constrained mobile and embedded devices. By reducing the precision of numerical data, weights, deviations, and activations, quantization allows for the use of lower precision representations, such as 8-bit integers instead of 32-bit floating point values (Chen et al. 2020). This optimization technique is crucial for adapting deep neural networks for deployment in resource-constrained environments, but it involves trade-offs in model accuracy (Ahn et al. 2023). Quantization can be achieved through Quantization-Aware Training (QAT) or Post-Training Quantization (PTQ) methods. QAT, which involves retraining the model, preserves accuracy but is computationally expensive. In contrast, PTQ, applied after training, is faster but often less accurate (Gholami et al. 2022). Various options for PTQ include dynamic range, integer, and float16 quantization. This study focuses on dynamic range post-training quantization.

2.4. Performance evaluation

The metrics outlined in Equations 8-11 were used to calculate the performance values of the eight deep learning models employed in the study. Accuracy measures the proportion of correctly classified samples within the classified sample data. Precision quantifies how many of the sample data classified as positive are actually positive. Recall indicates how much of the sample data that should be predicted as positive was actually classified as positive. The F1-Score represents the harmonic mean of the precision and recall metrics.

$$\gamma_{accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{8}$$

$$\gamma_{precision} = \frac{TP}{TP + FP} \tag{9}$$

$$\gamma_{recall} = \frac{TP}{TP + FN} \tag{10}$$

$$\gamma_{f1score} = 2 \times \left(\frac{\gamma_{precision} \times \gamma_{recall}}{\gamma_{precision} + \gamma_{recall}} \right) \tag{11}$$

- **True Positive (TP):** A correct prediction that a plant is diseased or not diseased.

- **True Negative (TN)**: A prediction that a healthy plant is not diseased.

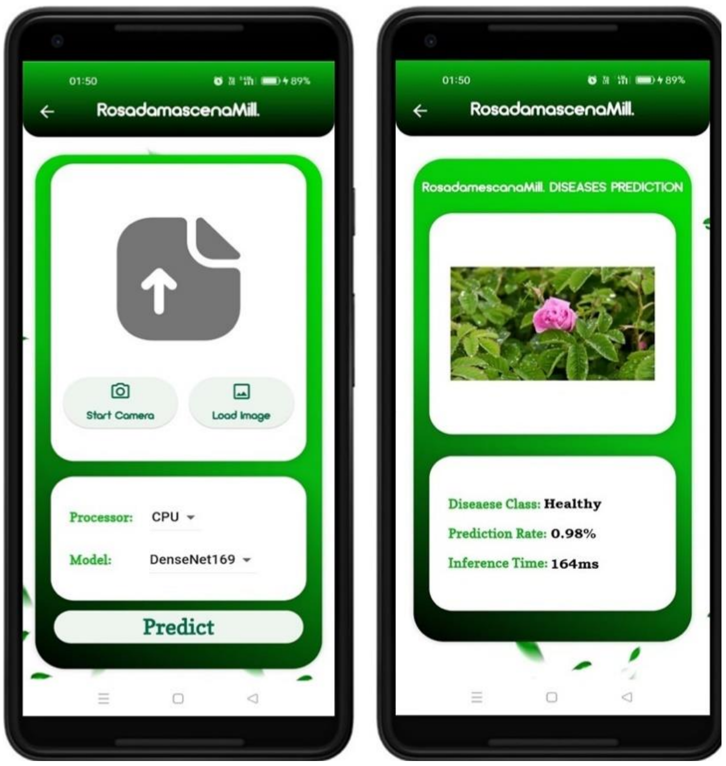
- **False Positive (FP):** A prediction of a healthy plant as diseased.

- **False Negative (FN)**: A prediction state indicating that a diseased plant is healthy.

2.5. Mobile application

A mobile application has been developed, which performs classification for the detection of diseases and plant pests, for real-time use. In this way, it will be possible to classify plants taken with a mobile phone or tablet instantly. In the Flutter-based application, TensorFlow Lite library was preferred for efficient use of deep learning tools. Here TensorFlow Lite is used to convert the models into FlatBuffer files (*.tflite*). These converted models were then integrated into the application's folder using TensorFlow version 2.4.1.

For mobile application development, the Android Studio Graffe and Visual Studio Code environments were used, alongside Flutter 2.3.0 -a popular cross platform framework- supported in these environments. Dart, an object-oriented programming language, was employed for coding in the Visual Studio Code environment. The code underwent thorough testing for performance and functionality on the Pixel 2 Android x86 emulator, after which an APK was generated and installed on a smartphone. Figure 5 shows the developed mobile application and the implementation of the models.



Processor: CPU
Model: DenseNet169

Diseaese Class: Healthy
Prediction Rate: 0.98%
Inference Time: 164ms

Figure 5- Representative screenshot of the developed mobile application

3. Results and Discussion

3.1 Experimental procedure

Experimental studies were conducted on a virtual machine in a cloud environment. A Tesla T4 GPU with CUDA 11.4 and TensorFlow 2.4.1 deep learning library were used alongside the Python programming language.

The hyperparameters used for model training in the experimental study, which employed six deep learning models to classify oil rose diseases, are provided in Table 3.

Table 3- Train hyperparameters

Parameter	*Setting*
Image size	224×224×3
Batch size	40
Optimizer	Adam
Loss function	Categorical Cross Entropy
Epochs	10, 20, 30, ….,150
Output Activation	Softmax
Learning rate	0.00001 - 0.01

3.2. Comparison performance of CNN-based models

With the dataset created for the study, the InceptionV3, DenseNet169, ResNet152, VGG19, MobileNet, and NASNet deep learning models were trained both with and without data augmentation. The training results, including loss and accuracy graphs based on incremental data are presented in Figure 6.

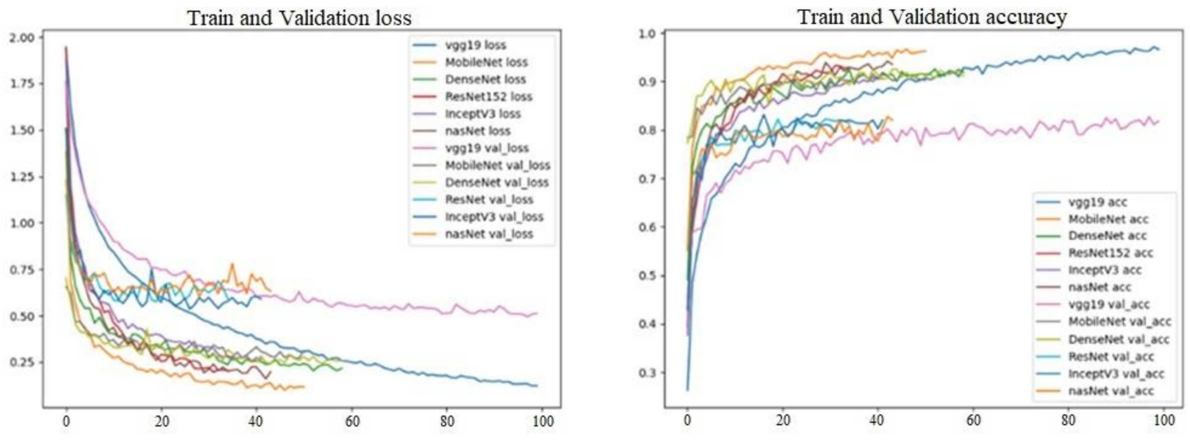


Figure 6- Training and validation loss-accuracy graphs of deep learning models after data augmentation

During model training, the training process is automatically terminated when the validation loss rate fails to improve due to adjustments in the *learning rate (lr)* applied to the model with the *reduce_lr_loss* and *early_stop* parameters. As a result, the maximum number of epochs specified as a model parameter may not be reached, causing the number of epochs in the graphs to vary.

The minimum validation loss during model training with non-augmented data was obtained as follows: 0.74 for the VGG19 model, 0.67 for the DenseNet169 model, 0.81 for the NASNet model, 0.83 for the InceptionV3 model, 0.86 for the ResNet152 model, and 0.70 for the MobileNet model. The minimum validation loss results from training with augmented data are as follows: 0.51 for the VGG19 model, 0.25 for the DenseNet169 model, 0.63 for the NASNet model, 0.58 for the InceptionV3 model, 0.59 for the ResNet152 model, and 0.31 for the MobileNet model.

When training the VGG19, DenseNet169, NASNet, InceptionV3, ResNet152, and MobileNet models with non-augmented data, the highest validation accuracies achieved were 0.81, 0.86, 0.81, 0.80, 0.75, and 0.83 respectively. In contrast, when training with augmented data, the validation accuracies were as follows: 0.82 for VGG19, 0.93 for DenseNet169, 0.82 for NASNet, 0.83 for InceptionV3, 0.91 for ResNet152, and 0.83 for MobileNet.

The precision, recall, and F1-score metrics from the test results are presented in Table 4 to identify the model with the best performance. The highest score from the test results was achieved with the DenseNet169 model. The lowest performance was observed with the InceptionV3 model. All six models achieved the highest accuracy in predicting the healthy class with label 0.

Table 5 presents the data for six different deep learning models designed for the study, including both original and augmented data, as well as the corresponding accuracy rates achieved during the training and testing phases. Among the pre-trained models trained with the original data, the InceptionV3 model attained a training accuracy of 98%, while the DenseNet169 model achieved validation and test accuracies of 86% and 80%, respectively.

When trained with augmented data, the InceptionV3 model yielded the lowest training accuracy, while the VGG19 model achieved the highest. An average validation accuracy of 85% was observed, with the DenseNet169 model achieving the highest validation accuracy of 93%, while other models ranged between 82% and 93%. During evaluation with the test data, the fine-tuned DenseNet169 model exhibited the highest performance at 93%, followed by the fine-tuned MobileNetV2 model with a performance of 90% (see Figure 7).

Table 4- Precision, recall and f1-score values of test results

Models and metrics		***Disease Label***							
		0	*1*	*2*	*3*	*4*	*5*	*6*	*7*
InceptionV3	precision	0.93	0.70	0.73	0.81	0.81	**0.98**	0.70	0.82
	recall	0.93	0.84	0.80	0.73	0.60	0.88	0.78	0.85
	f1-score	0.93	0.76	0.77	0.77	0.69	0.92	0.74	0.84
DenseNet169	precision	**1.00**	**0.93**	0.90	**0.90**	**0.98**	**0.98**	**0.90**	**0.96**
	recall	**1.00**	0.90	**0.98**	**0.97**	**0.90**	**0.94**	**0.82**	**1.00**
	f1-score	**1.00**	**0.92**	**0.94**	**0.93**	**0.94**	**0.96**	**0.86**	**0.98**
ResNet152	precision	0.98	0.82	0.67	0.89	0.70	0.89	0.69	0.88
	recall	0.98	0.87	0.85	0.74	0.64	0.86	0.64	0.90
	f1-score	0.98	0.85	0.75	0.81	0.67	0.88	0.67	0.89
VGG19	precision	0.92	0.81	0.80	0.89	0.71	0.90	0.76	0.92
	recall	0.98	0.87	0.85	0.83	0.68	0.88	0.64	0.94
	f1-score	0.95	0.84	0.82	0.86	0.69	0.89	0.70	0.93
MobileNet	precision	0.97	0.84	**0.92**	0.88	0.93	**0.98**	0.80	**0.96**
	recall	**1.00**	**0.97**	0.87	0.88	0.82	0.92	0.80	0.96
	f1-score	0.98	0.90	0.90	0.88	0.87	0.95	0.80	0.96
NasNet	precision	0.88	0.74	0.83	0.81	0.73	0.95	0.64	0.93
	recall	**1.00**	0.86	0.82	0.76	0.66	0.78	0.76	0.79
	f1-score	0.94	0.79	0.83	0.78	0.69	0.85	0.69	0.85

Table 5- Accuracy values of models for Training, Validation, and Testing

Dataset	***Accuracy***	***InceptionV3***	***DenseNet169***	***ResNet152***	***VGG19***	***MobileNet***	***NasNet***	***Average***
Original	Train	**0.98**	0.96	0.95	0.96	0.97	0.95	0.96
	Validation	0.80	**0.86**	0.75	0.81	0.83	0.81	0.82
	Test	0.70	**0.80**	0.73	0.75	0.78	0.67	0.75
Augmentation	Train	0.91	0.93	0.93	**0.97**	0.96	0.94	0.94
	Validation	0.83	**0.93**	0.82	0.82	0.91	0.82	0.85
	Test	0.80	**0.93**	0.81	0.84	0.90	0.80	0.85

In the confusion matrix, the main diagonal shows correctly predicted images, while the off-diagonal entries represent incorrectly predicted images. Looking at the normalized confusion matrix in Figure 8, it is evident that classification is successful for nearly all classes. However, the images labeled as class 6 show a relatively lower classification performance compared to those labeled with other categories, and there are examples of misclassification. In the row where the true classification label is 6, it can be observed that misclassification occurs more frequently for label 2 (normalized value 0.13).

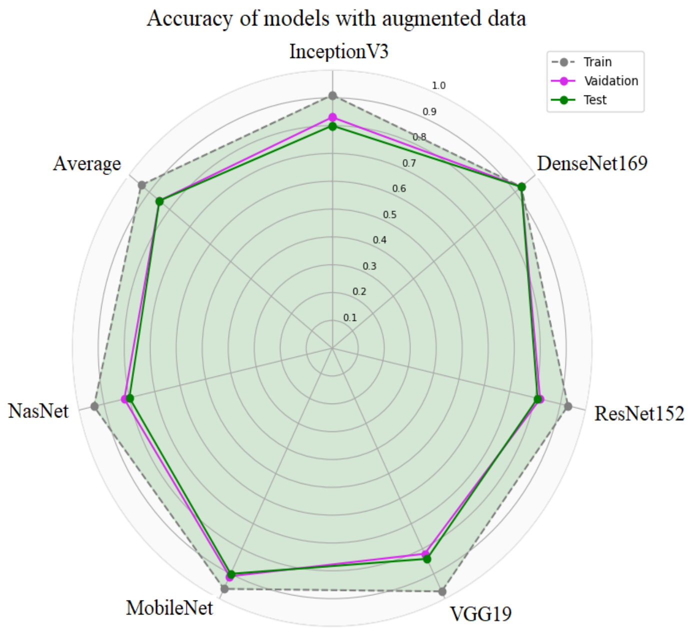
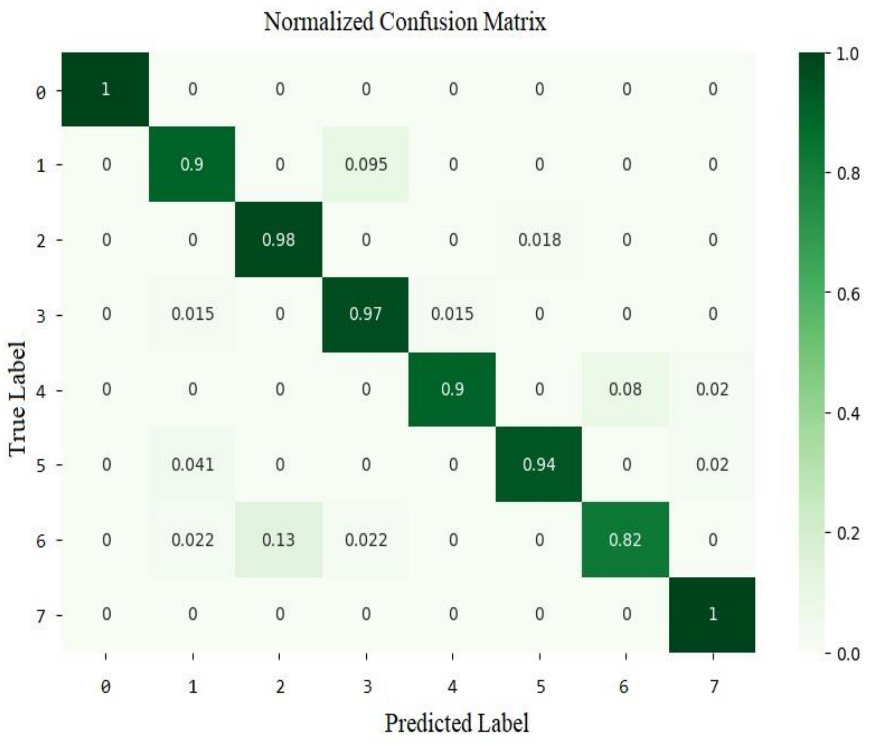


Figure 7- Training, validation and testing performances of models with augmented data



True \ Predicted	0	1	2	3	4	5	6	7
0	1	0	0	0	0	0	0	0
1	0	0.9	0	0.095	0	0	0	0
2	0	0	0.98	0	0	0.018	0	0
3	0	0.015	0	0.97	0.015	0	0	0
4	0	0	0	0	0.9	0	0.08	0.02
5	0	0.041	0	0	0	0.94	0	0.02
6	0	0.022	0.13	0.022	0	0	0.82	0
7	0	0	0	0	0	0	0	1

Figure 8- Confusion matrix of the DenseNet169 model

This may be attributed to the presence of both *Botrytis_spp* and *Mecorhis ungarica* diseases in the images of the dataset. In Figure 9, the images corresponding to both of these diseases are presented together.

The ensemble test accuracy was determined by combining the predictions of deep learning models using various combinations in the ensemble learning approach employed in the study to improve classification performance. Among these combinations, the combination of DenseNet169, ResNet152, MobileNet, VGG19, and NASNet models produced the best performance using the maximum voting method. The classification success rate of the deep learning model, which was 93%, increased to 95.17% with ensemble learning.

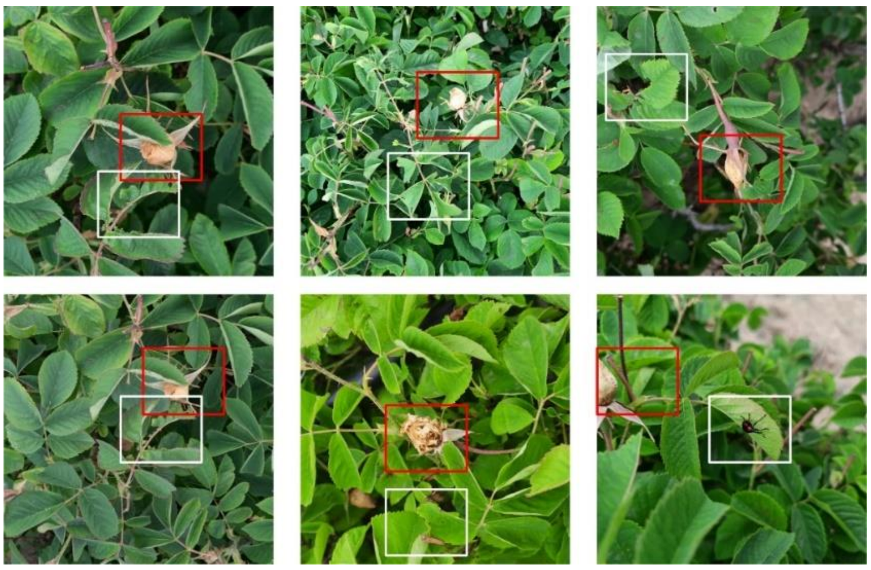


Figure 9- Images of *Botrytis_spp* (red rectangle) and *Mecorhis ungarica* (white rectangle) diseases together

3.3. Performance of mobile application

DenseNet169 and MobileNetV2 emerged as the top-performing models among the six models that were trained and tested on a desktop computer. The performances of these models were evaluated in the mobile application developed for disease and pest detection in the natural environment of the oil rose.

Given the limited hardware resources available on mobile devices, it is essential for inference models to be resource-efficient. Furthermore, these models are expected to deliver satisfactory processing times during inference.

If one examines the file sizes of the two models both before and after conversion to flatbuffer files, it becomes apparent that TensorFlow Lite inherently reduces the file size. The DenseNet169 model had a file size of 55.4MB in .h5 format, which was reduced to 50MB after conversion to *.tflite* format, and further reduced to 13.3MB after post-quantization. The MobileNetV2 model, originally 13MB, was reduced to 9.9MB after conversion to *.tflite* format, and further down to 2.77MB after post-quantization.

To evaluate the inference time and confidence in the developed mobile application, a test was conducted using 10 images from each class. The values were obtained by running the mobile application in the Visual Studio Code environment using an Android emulator and on a physical device.

When using the DenseNet169 model for predictions on the emulator, the inference time ranged from 1 266 ms to 1 577 ms, yielding confidence values between 50.1% and 98.4% for complete accuracy. When evaluated with the quantized DenseNet169 model, the inference time ranged from 21875 ms to 24087 ms, resulting in confidence values between 56.5% and 99.7% for full accuracy. Predictions made with the MobileNetV2 model showed inference times ranging from 137 ms to 225 ms, with confidence values between 88.3% and 99.3% for complete accuracy. For the tests conducted with the quantized MobileNetV2 model, the inference time varied between 2 842 ms and 3 591 ms, yielding confidence values between 62.3% and 99.9% for full accuracy.

The same tests were carried out on the physical mobile device. With the DenseNet169 model, the inference time ranged from 2 113 ms to 2 238 ms, and the confidence value was between 88.1% and 99.9%. For tests with the quantized DenseNet169 model, the inference time ranged from 2 485 ms to 2 932 ms, with the confidence value ranging from 80.8% to 99.9% for full accuracy. Predictions made using the MobileNetV2 model showed inference times ranging from 293 ms to 303 ms, yielding confidence values between 99.7% and 99.9% for full accuracy. In tests conducted with the quantized MobileNetV2 model, the inference time varied between 404 ms and 530 ms, generating confidence values ranging from 89.7% to 99.9% for full accuracy.

3.4. Overall evaluation

Table 6 illustrates the utilization of various algorithms for classifying diseases in different plant species. Deep learning algorithms and ensemble learning notably demonstrate substantial success in this area. The performance of these algorithms seems to be influenced by factors such as the number of classes in the dataset and the volume of images available. In many studies, established datasets such as the PlantVillage dataset were used, while others curated their datasets by collecting images from the internet. A significant portion of dataset images was obtained under controlled environmental conditions. Additionally, some studies included an image enhancement process to refine the dataset further. The highest performance was achieved by the models developed by Vallabhajosyula et al. (2022) and Ma et al. (2020), which used datasets with two classes and a relatively high number of images. Ma et al. (2022) concentrated solely on black spot disease.

Unlike the proposals by Harbola et al. (2024) and Prathiksha et al. (2024), this paper adopts an ensemble learning approach, where the accuracy rates are high but fails to put into consideration the realization on a mobile application, and the images are mostly taken in a controlled environment.

When comparing the performance of the study with other studies on rose plants, particularly those involving similar species and a comparable number of classes, it becomes apparent that the study's performance is relatively higher. Despite using data augmentation, the small size of the original dataset and the fact that the images were obtained in an uncontrolled environment limit the model performance from reaching nearly 100%. While other studies focus on detection from single-leaf images taken in controlled environments, this study conducts detection from plant images in their natural environment.

In addition, there have been no deep learning-based studies focused on disease detection in *Rosa damascena* Mill. plants. Moreover, this study deployed the deep learning model with the highest classification performance in a mobile application, a rare occurrence in similar studies. In their study on a mobile application, Hemalatha et al. (2022) implemented only a single model within the mobile application and did not perform a comparative analysis of inference times across different platforms or with various model configurations.

Table 6- Comparison of similar studies in the literature

Ref.	*Types of Plant*	*Num. of class*	*Base algorithm used for classification*	*Image count*	*Accuracy (%)*	*Deployment to any environment*
Vallabhajosyula et al. (2022)	14 crops	38	Ensemble (Deep learn.)	54305 (plant village)	100	-
Fuentes et al. (2018)	Tomato	10	CNN	5000	96	-
Astani et al. (2022)	Tomato	13	Ensemble (Machine learn.)	23811	95.98	-
Zhong & Zhao (2020)	Apple	6	DenseNet-121	2462	93.30	-
Sethy et al. (2020)	Rice	4	MobileNet+SVM	5932	97.96	-
Hemalatha et al. (2022)	Sugarcane	6	LeNet-5	2940	96	Android application
Ahila et al. (2019)	Maize	4	LeNet	4365	97.89	-
Wang et al. (2021)	Cucumber	4	DUNet(DeepLabV3 + and U-Net)	1000	92.85	-
Uğuz & Uysal (2021)	Olive leaf	3	Custom CNN	3400	95	Web app.
Swetharani & Prasad (2021)	Rose	-	CNN	-	97.3	-
Nuanmeesri (2021)	Rose	15	Vgg16+SVM	4032	88.33	-
Rajbongshi et al. (2020)	Rose	4	MobileNet	400	95.63	-
Ma et al. (2020)	Rose	2	AlexNet-VGG16-NDDR	900	99.10	-
Khaleel et al. (2022)	Rose	4	CNN	(not known)	64.35	-
Yin et al. (2021)	Rose	2	FasterR-CNN	650	84.16	-
This Study	*Rosa damascena* Mill.	8	DenseNet169 Ensemble model	567	93 95.1	Mobile application (with and without quantization)

In this study, the performance of the models was evaluated based on inference time across different environments, such as a personal computer, a mobile phone emulator, and a mobile device. In the computer environment, the designed DenseNet169 and MobileNetV2 models were tested, while in the mobile environment, the post-quantized models were also tested. The obtained results are shown in Table 7 as average values.

Table 7- Inference time of deep learning models in different environments

Model	***Average Inference Time (ms)***		
	Computer*	***Emulator*****	***Physical Mobile Device******
MobileNetV2	71	182	301
DenseNet169	119	1394	2271
Quantized MobileNetV2	NA	3303	434
Quantized DenseNet169	NA	22491	2624

*** Intel$^{(R)}$ Core$^{(TM)}$ i9-10920X CPU 3.50GHz 32GB, Win 10 Pro**
**** Pixel 2 API**
***** Samsung A6+: Cortex A53 CPU 1.8GHz 4GB, Android 10**

An examination of the inference times shows that the designed MobileNetV2 model has the lowest inference times in both virtual and physical environments. The fastest inference time is obtained on a desktop computer, followed by the emulator and then the physical mobile device. The post-quantized MobileNetV2 model shows slightly slower inference times in mobile environments compared to the MobileNetV2 base model but is significantly faster than the DenseNet169 models. Both the post-quantized MobileNetV2 model and the DenseNet169 model show slower inference times compared to their base models.

Rakib et al. (2024) proposed a model that is even independent of an internet connection, but it works with a very small dataset and a trivial application process. Adebisi et al. (2024) studied on the efficiency of multiple deep learning models on FPGA hardware, but failed to delve into finding an actual solution for mobile application or real-time usage. While the work of Padeiro et al. (2024) developed certain techniques of improvement for quantized models, their work did not include the integration of mobile applications. Actually, Katumba et al. (2024) studied on the quantization of models such as EfficientNetv2B0 and BeanWatchNet, integrated them in a mobile application, but they did not use real natural environment data, with accuracy rates not high but only up to 90-91%.

4. Conclusions and Future Work

Detecting plant diseases is challenging due to the large variety of diseases and the similarities between some of them. A mixed-CNN approach for disease and pest detection is presented using the new dataset created in the study. Six different newly designed deep learning models were trained with the new dataset. All models achieved training accuracies over 90%, with the VGG19 model performing best at 97%. Validation accuracies ranged from 82% to 93%, with DenseNet169 emerging as the best-performing model. Testing accuracies ranged from 80% to 93%, with the DenseNet169 model achieving a classification accuracy of 93%. Ensemble deep learning further improved test accuracy to 95.17%. Nevertheless, the small size and uncontrolled nature of the dataset may have limited test accuracy.

A comparative study was conducted on different platforms with the DenseNet169 and MobileNetV2 models using tflite and quantized variants. The results show that the MobileNetV2 model achieved the lowest inference times: 71 ms on a desktop computer, 182 ms on an emulator, and 301 ms on a physical mobile device. In contrast, inference times increased in mobile environments with the post-quantization of the models. However, the size of the post-quantized MobileNetV2 model decreased from 9.9MB to 2.77MB, and the size of the DenseNet169 model from 50MB to 13.3MB. In scenarios where inference time is critical, the tflite-converted MobileNetV2 model or the quantized MobileNetV2 model are better options for optimizing memory usage and power consumption.

The evidence supports the claim that the developed detection system can effectively diagnose rose diseases and pests with high accuracy. The system's real-time disease classification on smartphones suggests that the resulting mobile application could be a valuable and cost-effective tool for growers.

In future studies, the dataset will be expanded by collecting more images through the mobile application developed and a higher accuracy will be achieved. The second state will explore detecting multiple diseases in a single image and recommending appropriate treatments. Additionally, plans include transferring inference models embedded in the mobile application to cloud environments for broader accessibility via API.

Acknowledgements

I would like to thank Assist. Prof. Sinan Demir, member of the Faculty of Agriculture, Isparta University of Applied Sciences for his support in the systematic classification of diseases and pests. I am also thankful to Kaggle and Google for supplying free GPU computation platform.

Funding

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

Declaration of competing interest

The authors report no conflict of interest.

Data availability

The dataset is publicly available on the Mendeley data platform and can be referenced as "Duman, Burhan (2023), 'ISPGULDataset8C,' Mendeley Data, V1, doi: 10.17632/8ptzbtryhj.1."

Code availability

To access the source code of the mobile application, please visit the following link:
https://github.com/burhanduman/RosaDamascenaMillDiseasesV1.git

References

Abulwafa A E (2022). A Survey of Deep Learning Algorithms and its Applications. *Nile Journal of Communication and Computer Science* 3(1): 28-49

Adebisi J, Srinu S & Mitonga V (2024). Deep Learning Algorithm Analysis of Potato Disease Classification for System on Chip Implementation. *Journal of Digital Food, Energy & Water Systems* 5(1)

Ahila P, Arivazhagan R S, Arun M & Mirnalini A (2019). Maize leaf disease classification using deep convolutional neural networks. *Neural Comput. Appl.* 31: 8887-8895

Ahn H, Chen T, Alnaasan N, Shafi A, Abduljabbar M & Subramoni H (2023). Performance Characterization of using Quantization for DNN Inference on Edge Devices: Extended Version arXiv preprint arXiv:2303.05016

Astani M, Hasheminejad M & Vaghefi M (2022). A diverse ensemble classifier for tomato disease recognition. *Computers and Electronics in Agriculture* 198: 107054

Baydar H (2016). Oil Rose Cultivation and Industry. Science and Technology of Medicinal and Aromatic Plants (5th Expanded Edition). Süleyman Demirel University Press, 51: 290-325 (In Turkish)

Bıtrak O O & Hatırlı S A (2022). Global Oil Rose Market and Turkey's Role. *Selçuk University Akşehir Vocational School Journal of Social Sciences* 13: 85-94 (In Turkish)

Chen J, Chen J, Zhang D, Sun Y & Nanehkaran Y A (2020). Using deep transfer learning for image-based plant disease identification. *Computers and Electronics in Agriculture* 173: 105393

Chen Y, Zheng B, Zhang Z, Wang Q, Shen C & Zhang Q (2020). Deep learning on mobile and embedded devices: State-of-the-art, challenges, and future directions. ACM Computing Surveys (CSUR) 53(4): 1-37

Dewangan O (2023). Study and Innovative Approach of Deep Learning Algorithms and Architecture. In Exploring Future Opportunities of Brain-Inspired Artificial Intelligence (pp. 28-45). IGI Global

Ersan R & Başayiğit L (2022). Ecological modelling of potential Isparta Rosa areas (*Rosa damascena* Mill.). *Industrial Crops and Products* 176: 114427

Fazili M A, Ganie I B & Hassan Q P (2024). Studies on pharmacological aspects, integrated pest management and economic importance of Rosa damascena L. *South African Journal of Botany* 174: 534-541

Ferentinos K P (2018). Deep learning models for plant disease detection and diagnosis. *Computers and Electronics in Agriculture* 145: 311-318

Fuentes A F, Yoon S, Lee J & Park D S (2018). High-performance deep neural network-based tomato plant diseases and pests diagnosis system with refinement filter bank. *Frontiers in Plant Science* 9: 1162

Gholami A, Kim S, Dong Z, Yao Z, Mahoney M W & Keutzer K (2022). A survey of quantization methods for efficient neural network inference. In Low-Power Computer Vision (pp. 291-326). Chapman and Hall/CRC

Harbola G, Rawat M S, Gupta A & Gupta R (2024, May). Intelligent Diagnosis of Potato Leaf Diseases using Deep Learning. In 2024 4th International Conference on Pervasive Computing and Social Networking (ICPCSN) 372-377 IEEE

Hemalatha N K, Brunda R N, Prakruthi G S, Prabhu B V B, Shukla A & Narasipura O S J (2022). Sugarcane leaf disease detection through deep learning. In Deep Learning for Sustainable Agriculture, 297-323 https://doi.org/10.1016/B978-0-323-85214-2.00003-3

Huang G, Liu Z, Van Der Maaten L & Weinberger K Q (2017). Densely connected convolutional networks. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 4700-4708

Iman M, Arabnia H R & Rasheed K (2023). A review of deep transfer learning and recent advancements. *Technologies* 11(2): 40

Jung K, Lee J I, Kim N, Oh S & Seo D W (2021). Classification of space objects by using deep learning with micro-Doppler signature images. *Sensors* 21(13): 4365

Karanfil A (2021). Prevalence and molecular characterization of Turkish isolates of the rose viruses. *Crop Protection* 143: 105565

Katumba A, Okello W S, Murindanyi S, Nakatumba-Nabende J, Bomera M, Mugalu B W & Acur A (2024). Leveraging edge computing and deep learning for the real-time identification of bean plant pathologies. *Smart Agricultural Technology* 9: 100627. https://doi.org/10.1016/j.atech.2024.100627

Khaleel M I, Sai P G, Kumar A U R, Raja P & Hoang V T (2022). Rose Plant Leaves: Disease Detection and Pesticide Management using CNN. 2022 6th International Conference on Trends in Electronics and Informatics (ICOEI), 1067-1072

Khitthuk C, Srikaew A, Attakitmongcol K & Kumsawat P (2018). Plant leaf disease diagnosis from color imagery using co-occurrence matrix and artificial intelligence system. 2018 International Electrical Engineering Congress (IEECON), 1-4

Lee S H (2020). Deep learning based face mask recognition for access control. *Journal of the Korea Academia-Industrial Cooperation Society* 21(8): 395-400

Liu Y, Sun Y, Xue B, Zhang M, Yen G G & Tan, K C (2021). A survey on evolutionary neural architecture search. IEEE Transactions on Neural Networks and Learning Systems.

Ma J, Pang L, Yan L & Xiao J (2020). Detection of black spot of rose based on hyperspectral imaging and convolutional neural network. *AgriEngineering* 2(4): 556-567

Mahlein AK (2016). Plant disease detection by imaging sensors–parallels and specific demands for precision agriculture and plant phenotyping. *Plant Disease* 100(2): 241-251

Mutka A M & Bart R S (2015). Image-based phenotyping of plant disease symptoms. *Frontiers in Plant Science* 5: 734

Nuanmeesri S (2021). A hybrid deep learning and optimized machine learning approach for rose leaf disease classification. *Eng. Technol. Appl. Sci. Res.* 11(5): 7678-7683

Oikonomidis A, Catal C & Kassahun A (2023). Deep learning for crop yield prediction: a systematic literature review. *New Zealand Journal of Crop and Horticultural Science* 51(1): 1-26

Pan W, Qin J, Xiang X, Wu Y, Tan Y & Xiang L (2019). A smart mobile diagnosis system for citrus diseases based on densely connected convolutional networks. *IEEE Access* 7: 87534-87542

Prathiksha B J, Kumar V, Krishnamoorthi M, Poovizhi P, Sowmiya D & Thrishaa, B. (2024, April). Early Accurate Identification of Grape leaf Disease Detection using CNN based VGG-19 model. In 2024 International Conference on Cognitive Robotics and Intelligent Systems (ICC-ROBINS) (pp. 263-269). IEEE

Rajbongshi A, Sarker T, Ahamad M M & Rahman M M (2020). Rose diseases recognition using MobileNet. 2020 4th International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT) 1-7

Rakib A F, Rahman R, Razi A A & Hasan A T (2024). A lightweight quantized CNN model for plant disease recognition. Arabian *Journal for Science and Engineering* 49(3): 4097-4108

Reddy D S, Rajalakshmi P & Mateen M A (2021). A deep learning based approach for classification of abdominal organs using ultrasound images. *Biocybernetics and Biomedical Engineering* 41(2): 779-791

Sagi O & Rokach L (2018). Ensemble learning: A survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 8(4): e1249

Sethy P K, Barpanda N K, Rath A K & Behera S K (2020). Deep feature based rice leaf disease identification using support vector machine. *Computers and Electronics in Agriculture* 175: 105527

Sharma R & Singh G (2015). Access to modern agricultural technologies and farmer household welfare: Evidence from India. *Millennial Asia* 6(1): 19-43

Simonyan K & Zisserman A (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint* arXiv:1409.1556

Swetharani K & Prasad G V (2021). Design and implementation of an efficient rose leaf disease detection and classification using convolutional neural network. *International Journal of Image Mining* 4(1): 98-113

Szegedy C, Liu W, Jia Y, Sermanet P, Reed S, Anguelov D, Erhan D, Vanhoucke V & Rabinovich A (2015). Going deeper with convolutions. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 1-9

Tao W, Al-Amin M, Chen H, Leu M C, Yin Z & Qin R (2020). Real-time assembly operation recognition with fog computing and transfer learning for human-centered intelligent manufacturing. *Procedia Manufacturing* 48: 926-931

Uğuz S & Uysal N (2021). Classification of olive leaf diseases using deep convolutional neural networks. *Neural Comput. & Applic.* 33: 4133–4149.https://doi.org/10.1007/s00521-020-05235-5

Vallabhajosyula S, Sistla V & Kolli V K K (2022). Transfer learning-based deep ensemble neural network for plant leaf disease detection. *J. Plant Dis. Prot.* 129: 545-558. https://doi.org/10.1007/s41348-021-00465-8

Vrbančič G & Podgorelec V (2020). Transfer learning with adaptive fine-tuning. *IEEE Access* 8: 196197-196211

Wang C, Du P, Wu H, Li J, Zhao C & Zhu H (2021). A cucumber leaf disease severity classification method based on the fusion of DeepLabV3+ and U-Net. *Computers and Electronics in Agriculture* 189: 106373

Weerakoon W M W et al. (2017). Effect of leaf color chart based nitrogen management on the performance of rice crop. *Field Crops Research* 151: 15-23

Yang Y, Lv H & Chen N (2023). A survey on ensemble learning under the era of deep learning. *Artificial Intelligence Review* 56(6): 5545-5589

Yilmaz H (2015). Estimating the economic costs and level of pesticide use in oil rose (Rosa damascena Mill.) orchards: evidence from a survey for the lakes region of Turkey. *Erwerbs-obstbau* 57(4): 195-202

Yin J, Yu D, Li Z, Guo W & Zhu H (2021). Chinese Rose flower disease recognition method based on deep learning. Frontier Computing: Proceedings of FC 2020, 1309-1319

Yu Z, Wang K, Wan Z, Xie S & Lv Z (2023). Popular deep learning algorithms for disease prediction: a review. *Cluster Computing* 26(2): 1231-1251

Zhong Y & Zhao M (2020). Research on deep learning in apple leaf disease recognition. *Computers and Electronics in Agriculture* 168: 105146

Zoph B & Le Q V (2016). Neural architecture search with reinforcement learning. ArXiv Preprint, ArXiv:1611.01578