ARQII Publication
http://arqiipubl.com/ams
APPLICATIONS OF
MODELLING AND SIMULATION
eISSN 2600-8084
VOL 7, 2023, 146-155

Android Based-App Papaya Leaf Disease Identification Using Convolution Neural Network

Bo Xuan Wong[1], Chang Choon Chew[1], Kim Gaik Tay[1*], Osamah Al-qershi[2], Audrey Huong[1] and Shehab Abdulhabib Alzaeemi[1]

[1]Faculty of Electrical and Electronics Engineering, Universiti Tun Hussein Onn Malaysia, Malaysia
[2]University of Melbourne, Melbourne, Victoria, Australia

[*]Corresponding author: tay@uthm.edu.my

Submitted 20 August 2023, Revised 05 October 2023, Accepted 20 October 2023, Available online 05 November 2023.

Abstract: Papaya plant disease can lead to substantial harvest and financial losses for crop owners, potentially affecting the overall income of Malaysia's agriculture sector. Integrating Artificial Intelligence into the agriculture sector can be a significant leap in attracting young agropreneurs and assisting both existing and young farmers in identifying papaya plant diseases. In line with these challenges, this project proposes papaya plant disease identification using a Convolutional Neural Network (CNN). A total of 225 images were collected from Google sources and real-life captured images, consisting of 3 different classes: Healthy, Ring Spot, and Black Spot. After the preprocessing and augmentation process, a total of 600 images were obtained. Out of 25 Keras API pre-trained CNN models, InceptionV3 was selected as the best base model, as it achieved the highest validation accuracy during a 10-epoch run through Google Colab. Hyperparameters were tuned to obtain the best results by inputting the training images from the top of the base model and extracting 2048 output features from the last layer of the Inception V3 model. The extracted features were saved in the form of NumPy arrays to be employed in the pipeline for hyperparameter tuning, thereby improving the tuning efficiency. The results show the tuned training data achieved a validation accuracy of 1.0 using a batch size of 4, a learning rate of 0.01, and 50 epochs with the SGD optimizer. With this set of hyperparameters, a full model training was conducted with training images as input, resulting in a training accuracy of 1.0 and validation accuracy of 0.96. The trained model was exported in the form of tflite file format and used in the app development through Android Studio. Testing the app's accuracy involved importing selected 15 images from each class into the designed app, resulting in a precision of 0.8889, recall of 0.8889, f1-Score of 0.8889, and accuracy of 0.8889. These results demonstrate that the accuracy of identifying papaya plant disease through papaya leaves using the designed Android-based app was relatively high.

Keywords: Agriculture; Agropreneur; Android Studio; Convolution Neural Network; Papaya.

1. INTRODUCTION

The agriculture sector has been one of the most important sources of income in Malaysia for many years. The most famous agricultures commodities that generate the highest profits for Malaysia are rubber and palm oil. In addition to rubber and palm oil, various other types of agriculture also contribute significantly to Malaysia's profitability, including cocoa, rice, tropical fruits, and wood [1]. Nowadays, the main employees who work in the agriculture industry in Malaysia are mostly elderly or foreign workers, highlighting the need for the participation of younger agropreneurs to sustain the economic status of the agriculture sector of Malaysia [2]. Since many tasks in the agriculture sector are traditionally performed manually, the integration of technology in this industry will be a significant leap in attracting young agropreneurs to participate [3].

Papaya, scientifically known as the Carica plant was initially planted in Malaysia in the year 1972 as a smallholder crop. After 15 years, it gained popularity and transitioned into a plantation crop. Today, papaya is one of the famous fruits in Malaysia. Regular pest and disease control measures need to be implemented in the plantations to prevent losses or damage. Common plant diseases can be easily identified by observing the color and condition of their leaves, but this requires the workers' experience to accurately identify the plant disease based on leaf conditions [4]. Plant disease on papaya plants can affect the overall harvest of the entire plantation and cause significant financial losses for the owner. A breakout of papaya dieback disease caused by bacteria occurred at Batu Pahat and Bidor, Perak, in the years 2003 and 2004, respectively. This disease spread to five other states on the west coast of Peninsular Malaysia by the end of the year 2006. This outbreak destroyed approximately 800 hectares of crops, resulting in the destruction of around 1 million papaya trees nationwide. This led to a

yield loss of 200,000 metric tons of papaya fruit, equivalent to 58 million US dollars. The production of papaya fruit in Malaysia decreased by about 40%, and the export value declined up to 70%. The papaya dieback disease is the primary reason behind these disappointing phenomena [5]. Papaya Ringspot Virus (PRSV) caused millions of Ringgits in losses to the farmers of papaya crops at Perak, Johor Bahru and Lanchang [6].

Nowadays, smartphones enable easy access to multiple applications developed by various organizations. An app equipped with artificial intelligence features can be introduced to assist young agropreneurs in easily identifying plant diseases through leaves. By using the camera of their smartphone, the app will classify the captured image if the plant has a disease or not. The Convolution Neural Network (CNN) encompasses various pre-trained models such as Inception, VGGNet, and Resnet. Utilizing transfer learning from a pre-trained model is recommended for classifying new applications as it can save the time and energy required for training a model from scratch, which is a time-consuming and resource-intensive process. However, the accuracy of pre-trained models may vary depending on the specific applications. Therefore, conducting a test using the same training and validation sets is essential to determine which model can achieve the highest accuracy for the project [7].

This project intends to develop an Android phone app using the best pre-trained CNN model and the best hyperparameters to identify whether the entire papaya plant has plant disease or is healthy. The app can identify 3 classes of outputs which are healthy, blackspot disease, and ringspot disease of papaya leaf. The selection of monitored diseases was based on those that caused significant financial losses for the farmers, produced the most substantial damage, and were the most prevalent in the observed crop [6]. These diseases include blackspot disease and ringspot disease of papaya leaves. This research does not focus on other diseases due to their lower occurrence rates compared to blackspot and ringspot diseases. Black spots and ring spots are two of the most common diseases affecting papaya leaves.

The contribution of this study involves:

1. Creation of a dataset consisting of three classes of papaya leaf disease.
2. Identification of the best pre-trained CNN model by implementing a script that examined 25 pre-trained models from Keras Api.
3. Optimization of hyperparameter for InceptionV3 using the GridSearch method.
4. Development of an Android app for the identification of three classes of papaya leaf diseases.

2. RELATED WORKS

Sanjay *et al.* [8] proposed an image-processing method based on Spatial Gray Level Dependence Matrices (SGDM) for detecting plant diseases caused by bacteria, fungi, and viruses. The method involves four main steps: Firstly, transformation of RGB to HSI: the original RGB (Red, Green, Blue) colour space is converted to HSI (hue, saturation, intensity) as RGB is primarily used for colour generation, while HSI serves as a more suitable colour descriptor. To accomplish this, they constructed a colour transformation structure. Next, they masked and removed the green pixels using a specific threshold value. Then the image segmentation was performed to extract meaningful segments from the leaf images. Finally, the statistics of the leaf texture were computed by using SGDM matrices and the classification task was done using the squared distance technique. They do believe that the proposed method would help to detect and classify plant leaf disease.

Yashodharan [9] prepared and enhanced four classes of papaya leaf images: black spot, ring spot, mealybug and healthy leaves. They extracted colour features from the images that helped in detecting the regions infected by the respective diseases. To analyze and pinpoint the infected regions on the papaya leaves, they employed the k-medoid clustering algorithm. The classification task was performed by using a Multilayer Perceptron (MLP). The results obtained from their study show that the accuracy of sample image disease recognition was 94.61%. This high accuracy indicates that the k-medoid clustering algorithm is an effective method in the context of this study. It demonstrates its usefulness in detecting and classifying different diseases on papaya leaves based on the extracted color features and clustering analysis.

Habib *et al*. [10] developed a papaya disease recognition system to assist distant farmers in identifying diseases affecting their papaya plants. To construct this system, they first collected images from the internet, which were then converted into fixed-size images through bicubic interpolation. The image quality was enhanced using the histogram equalization technique. These improved images were subsequently transformed from the original colour space to Lab (longitude, latitude, and altitude) colour space. To facilitate disease segmentation, they employed the k-means clustering technique, which separated the faulty parts of the image into two different feature vectors: co-occurrence and statistical features. These two different feature vectors were then used for constructing multiple Support Vector Machines (SVMs) to handle multiclass classification problems. The system allows distant farmers to capture plant images using their mobile phones or handheld devices. The images are sent to the expert system proposed by them. Upon receiving the image, it is forwarded to the backend server, where it is processed by the system. After processing, feedback about the disease is provided and sent back to the user's device in the form of an SMS. Testing the proposed method using image data samples, the system achieved an impressive accuracy of 90.15%. They have successfully demonstrated that their proposed system has enormous potential in assisting distant farmers with papaya identification.

Mohanty *et al*. [11] collected a total of 54306 greyscale images of 38 classes of plant leaves. The images were resized to a resolution of 256x256 pixels. An automated script was used to segment the images, which were then analyzed using two different colour spaces: HSB (Hue, Saturation, Brightness) and Lab (longitude, latitude, and altitude). The processed data were analyzed by using two famous pre-trained models: AlexNet and GoogLeNet, which utilised transfer learning. The models were tested with various training testing set distributions. The study concluded that the transfer learning method outperformed the train-from-scratch approach. Among the models, GoogLeNet achieved the best performance, reaching 99.34% when trained

with a distribution of 80% training data and 20% testing data using colour images. Even when using segmented data, GoogLeNet still achieved a high accuracy of 99.25%. This research demonstrates the potential of applying transfer learning to identify plant diseases based on leaf images. The high accuracy obtained by the models indicates that this approach holds significant promise in assisting farmers in identifying their plant disease through leaf images with their mobile phones.

Leong *et al*. [12] collected a dataset of tomato leaves with four different classes: healthy, bacteria disease, Septoria leaf spot, and Alternaria. The dataset was split into training and testing sets using an 8:2 ratio. To preprocess the images, they applied two approaches: resizing them to 224x224 and enhancing contrast. For the segmentation process, they employed two methods: K-means clustering and the color threshold method. In the colour thresholding method, the images were converted from RGB colour space to HSV colour space. Then, they limited H (hue) and S (saturation) values to a certain range to extract useful colour properties. In the feature-extracting stage, they applied two different methods: CNN and Grey-Level Co-Occurrence Matrix (GLCM). For the CNN method, they applied the pre-trained ResNet-50 model to extract features. On the other hand, for the GLCM, the images were converted to greyscale and eight features were extracted from each image, namely Variance, Mean, Contrast, Correlation, Dissimilarity, Homogeneity, Entropy and Angular Second Moment. The extracted features were then fed into the Support Vector Machine (SVM) for classification. Using these two different feature-extraction methods, they achieved an average accuracy of 82.63% for GLCM and 96.63% for the CNN method. Consequently, they concluded that the CNN method provides better accuracy compared to the GLCM method. As a future recommendation, they suggested improving the segmentation stage by applying deep CNN methods to effectively and accurately process images with complex backgrounds.

Shelar *et al*. [13] developed an Android smartphone plant disease detection APP using VGG-19 and TensorFlow Lite. They collected 38 classes of image data from the PlantVillage dataset. The image dataset underwent preprocessing and augmentation through the image-data generator of the Keras API. During training, the model achieved an impressive accuracy of 95.6% after 50 epochs. Subsequently, they conducted testing and found that the model can accurately identify plant diseases from 13 different species. These species include tomato, strawberry, soybean, raspberry, potato, corn, pepper bell, peach, orange, grape, cherry, blueberry, and apple. The study showcases the effectiveness of utilizing VGG-19 and TensorFlow Lite in creating a practical and accurate plant disease detection application for Android smartphones.

Ajra *et al*. [14] collected approximately 4000 leaf images affected by four different plant diseases: Early blight and late blight for potatoes and tomatoes. Additionally, they gathered 2000 images representing the healthy class. Consequently, the model was designed with five classes. The researchers conducted a four-stage image pre-processing procedure: cleaning, integration, reduction, and transformation. The primary aim of this pre-processing was to eliminate unwanted or inconsistent data, retaining only useful data and ensuring data linearity for easier processing. Following pre-processing, data augmentation techniques were applied, involving resizing, random flipping, rotating, and cropping of the images. Furthermore, the images were transformed into RGB color format. For classification, two different pre-trained models: AlexNet and ResNet-50 from Keras API were employed. Analyzing the results, ResNet-50 achieved an overall accuracy of 97% in classifying whether the leaves are healthy or unhealthy while AlexNet obtained an overall accuracy of 96.5%. Additionally, for leaf disease detection, ResNet-50 achieved 96.1% overall accuracy while AlexNet obtained an overall accuracy of 95.3%. The overall findings indicated that the ResNet-50 model outperformed AlexNet in leaf disease detection. The study demonstrates the effectiveness of using pre-trained models, particularly ResNet-50, in accurately identifying and classifying plant diseases from leaf images.

Kumar *et al.* [15] conducted a study on plant disease detection using CNN. They collected a dataset of 54303 images, encompassing 38 classes of leaf images categorized by species and disease for analysis. From this dataset, they selected 36148 samples, consisting of eight types of leaf categories for further analysis. The images underwent preprocessing, where they were resized to 256x256 pixels and augmented using rotation, shear, and brightness techniques. The dataset was then split into training and testing sets using a ratio of 7:3. For the model architecture, a last channel architecture was used, which consisted of Convolutional layers followed by ReLU activation and Pooling layers. Additionally, they designed two sets of (Convolutional => ReLU) layers followed by Pooling blocks and then integrated fully connected layers into the model. As an optimizer, they chose Adam's Hard Optimizer. They achieved an overall accuracy of 94.6% for this approach using this model.

Guan [16] prepared a dataset comprising 36258 leaf images, which included ten different plant species. Each plant species had one class representing healthy leaves and multiple classes for various plant diseases. The dataset consisted of a total of 61 classes. 31718 images were used for training while the remaining images were allocated for validation. The images underwent augmentation, involving random horizontal flip, vertical flip, random degree rotation, random brightness, and contrast tuning. Four different pre-trained models were employed for transfer learning: ResNet, Inception, Inception Combine ResNet, and DenseNet. Each model was trained for 20 epochs, with a batch size of 16, and a learning rate of 0.001 to 0.00001. Upon completing 20 epochs, the evaluation results indicated that the combination of Inception and ResNet achieved the highest accuracy of 84.07%, followed by DenseNet (83.44%), ResNet (82.78%), and Inception Net (82.22%). He suggested that future research could focus on reducing the amount of substantial computation power. Additionally, the number of species and diseases in the dataset could be expanded. The current research could only determine whether the plant was infected or not but it is unable to predict the probability of the plant getting infected so further research could be focused on this.

Hassan *et al*. [17] prepared 54305 images consisting of healthy and infected plant leaves obtained from the standard open-access PlantVillage dataset. The dataset contains 38 different classes, including 14 species of plants, each having healthy and infected plant images. The images were converted to 3 different formats: colored, segmented, and grayscale. The images were used to train four different pre-trained models: InceptionV3, InceptionResNetV3, MobileNetV2, and EfficientNetB0. The training process involved using 30, 45, and 50 epochs for different train-test ratios: 8:2, 7:3, and 6:4. For the ratio of 8:2, 50

epochs were used, followed by 45 epochs for 7:3 and 30 epochs for 6:4. Among all the approaches, the EfficientNetB0 model with the split ratio of 8:2 and colored images achieved the highest overall accuracy of 99.56%.

Sardogan *et al*. [18] proposed a tomato leaf disease detection and classification based on CNN with the learning Vector Quantization (LVQ) algorithm. The dataset that they used consisted of 500 images, categorized into 5 different classes: healthy, bacterial spot, late blight, Septoria leaf spot, and yellow leaf curl. A train-test split with a ratio of 8:2 was applied to the dataset during the training session. To prepare the images for their CNN model, they resized them to a size of 512x512 pixels. The proposed CNN architecture takes the RGB matrix of the images as input and utilised 4 Rectified Linear Unit (ReLu) activation functions within the convolution layer. For the classification layer, they employed the LVQ algorithm. The Kohonen layer was configured with 50 neurons, meaning 10 neurons for each class, and the output layer contained 5 neurons corresponding to the 5 classes. The training process was run with a maximum 300 epochs, utilising a learning rate of 0.1. With this approach, they achieved an average accuracy of 86%. In light of their findings, they suggested applying different filtering methods in the future to increase the accuracy.

Tejaswini *et al*. [19] conducted a study on rice leaf disease detection using CNN. The dataset they used comprised a total of 1600 images, containing four different classes with the following distribution: 500 for Hispa, 400 for Brownspot, 300 for Leafblast, and 400 for healthy leaves. All images had an original resolution of 1449x1449 with a white background. The dataset underwent preprocessing in three stages: data cleaning, transformation, and reduction. The main objective was to enhance data quality and reduce dimensions to facilitate the calculation process. In their investigation, Tejaswini *et al*. utilized five pre-trained models: VGG16, VGG19, Xception, ResNet-50, and 5-Layer Convolution. The training was conducted over 30 epochs. The results indicated that their custom 5-layer CNN architecture achieved the highest accuracy of 78.2% among all the models tested. They mentioned that by adjusting the learning rate, number of epochs, and optimizer, it is possible to construct a model with fewer layers while maintaining high accuracy compared to traditional models. For their future work, they planned to expand the number of diseases and explore different algorithms to further improve the efficiency of disease detection in rice leaves.

Srinidhi *et al*. [20] gathered 3600 real-life apple leaf images belonging to 4 different classes: healthy, scab, rust, and multiple diseases. The image dataset underwent preprocessing using image annotation and augmentation techniques including Canny edge detection, blurring, rotating, skewing, and flipping. The augmentation process utilized an Image data generator, which resulted in the generation of 10,000 augmented images. Subsequently, the augmented images were fed into two different pre-rained models: EfficientNet-B7 and DenseNet, and both models were trained using 40 epochs. The results obtained show that the EfficientNet-B7 achieved the highest accuracy of 99.81%, slightly surpassing DenseNet's accuracy of 99.75%. They suggested that exploring stacking, ensembling, and employing better validation techniques could lead to more accurate and better models.

Veeraballi [21] identified two classes of papaya leaf diseases: leaf curl and papaya Mosaic using ResNet 50. They used 9,470 training images and 200 testing images, achieving an average accuracy of 85.1%. Meanwhile, Bacus [22] developed a GUI in Raspberry Pi and Android to identify four types of papaya leaf diseases: Black spot, Brown Spot, Mealybug, Powdery Mildew, one healthy class and one unknown class. They used MobileNet for the classification of these 6 classes and achieved an accuracy of 91.67%.

In summary, the literature review conducted reveals that CNN outperformed other feature extraction-machine learning methods. Transfer learning from a pre-trained CNN model performed better than training a CNN from scratch. Out of the 15 reviewed related works, only four researchers [9-10] and [21-22] conducted studies on papaya leaf disease identification. Studies [9-10] employed ML techniques, but these techniques require the determination of significant features for classification. Meanwhile, study [21] employed ResNet50, while [22] used MobileNet. None of the studies on papaya leaf disease recognition investigated the best pre-trained CNN model to be employed in their applications. Additionally, none of them optimized the hyperparameters used in their CNN models. Furthermore, none of the researchers utilized InceptionV3 in identifying papaya leaf diseases. Therefore, this study aims to develop an Android phone app using the best pre-trained CNN model (InceptionV3) and the optimal hyperparameters to identify whether the entire papaya plant has a plant disease or is healthy. This app is easy for farmers to use.

3. MATERIALS AND METHODS

Figure 1 illustrates the flowchart of the entire project, starting from image acquisition, image preprocessing, and the selection of the best pre-trained model for papaya leaf disease detection, followed by hyperparameters tuning, CNN training session, app development phase, and app testing session.

3.1 Image Acquisition and Preprocessing

For the training, validation, and testing datasets, a total of 225 images were acquired, consisting of three different classes: Healthy, Ring Spot, and Black Spot. Their distribution can be seen in Table 1. The image dataset was collected from various sources, including the Google search engine and real-life captures. To capture the images, a smartphone camera was used in real-life settings, specifically in the residential area and Bistari Hostel in Parit Raja, Batu Pahat. The camera was positioned at a 90-degree angle and placed approximately 20 cm away from the papaya leaf attached to the plant. Figure 2 shows samples of healthy, black spot and ring spot of papaya leaves. After acquiring the image dataset, it underwent pre-processing, which included data cleaning to eliminate low-resolution images that did not contain useful information. Subsequently, the image quality was enhanced by using Irfan View. The enhanced images were then uploaded to Google Drive in preparation for

augmentation. After augmenting using the techniques given in Table 1 and saving the images, the dataset expanded to a total of 600 images. Among these, 170 images were allocated for training, while 15 images were used for both testing and validation for each class as seen in Table 1. Augmentation can increase the generalization of the training model. Therefore, this dataset is deemed sufficient for training an accurate model to identify black spot and ring spot diseases and make decisions. It is worth noting that other research studies also utilized their own and limited datasets [23, 24].

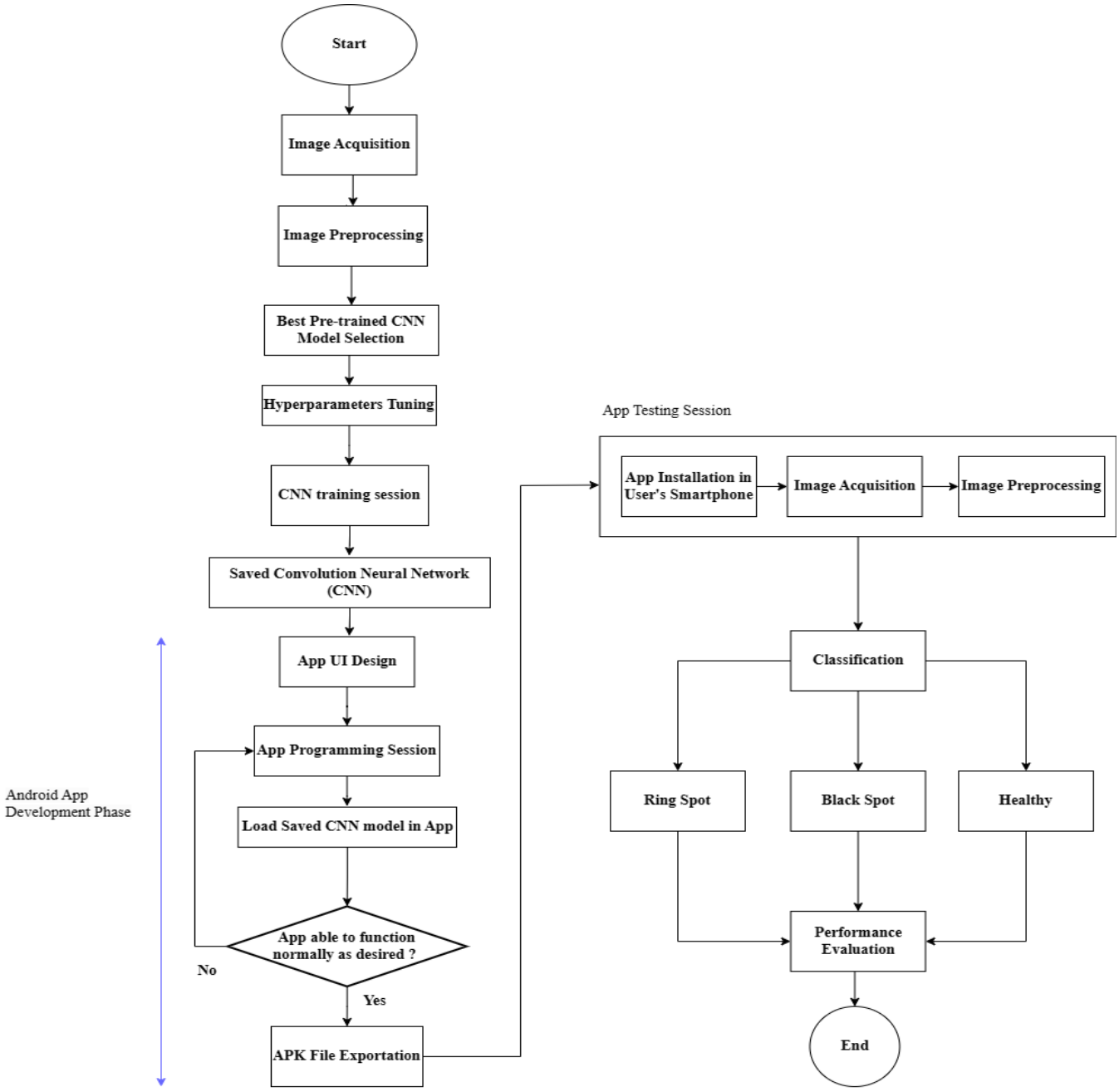
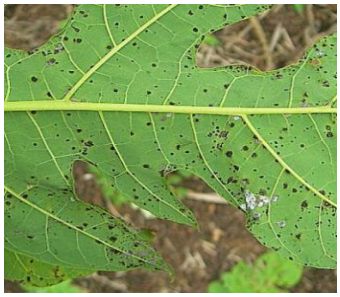


Figure 1. Project flowchart

Table 1. Image dataset distribution with classes

Augmentation Technique	rescale = 1/255, shear range = 0.2, zoom range = 0.2 to 1.2, horizontal flip, brightness range= 0.2 to 0.8					
	Raw Image Data			New Image Dataset		
	Healthy	Black Spot	Ring Spot	Healthy	Black Spot	Ring Spot
Train	45	45	45	170	170	170
Validate	15	15	15	15	15	15
Test	15	15	15	15	15	15


(a) Healthy Papaya Leaf


(b) Blackspot disease


(c) Ringspot disease

Figure 2. Example of image dataset

3.2 The Best Pre-trained CNN Model Selection

The image dataset was input to a script on Google Colaboratory. The script aimed to run training using 25 KERAS pre-trained CNN models to identify the pre-trained model with the highest validation accuracy. This selected model would then serve as the base model for the next stage.

3.3 Hyperparameter Tunning

After selecting the pre-trained model, the hyperparameters such as the number of epochs, optimizers, batch sizes, and learning rates were fine-tuned to obtain the best possible set of hyperparameters. To tune the hyperparameter, the second script was coded to extract the features from the pre-trained models using transfer learning. The third script takes the features as input for finding the best hyperparameter. These best hyperparameters were used in CNN model training to export the optimal model in the form of tflite file for the Android App development stage.

3.4 CNN Model Training

The tuned hyperparameters were used in this CNN model training session. The image dataset was used as input, starting from the top of the base model, and extending to the last dense layer, which consists of 3 output nodes for each class. This approach was chosen to avoid extensive tuning sessions with numerous hyperparameter combinations. At the end of the training, graphs illustrating the training and validation accuracies and losses were plotted. Subsequently, the model was extracted in tflite file format for app development.

3.5 App Development

Android app development commenced after saving the CNN model. Once the Android GUI development and app base programming were completed, the CNN model was integrated into the app. The app's functionality was then tested to ensure it functions as intended. If there are no issues with the functionality of the app when using the loaded model, the APK files were prepared for export to the user's device. The Android-based mobile app was developed using the Android Studio Electric Eel. The Android Studio could be programmed with either Java or Kotlin. Java was the main programming language during this app development.

3.6 App Testing

The app testing phase involves installing the app on a user's device and using it for the easy identification of papaya leaf diseases. Users can accomplish this by either inserting an image of a papaya leaf from their device's gallery or capturing real-time images of papaya leaves. The app will then classify the leaf as either healthy, affected by black spot, or afflicted with ring spot diseases.

3.7 Performance Evaluation

The performance of the developed app is evaluated using precision, recall, f1-score and accuracy as follows:

$$\text{Precision} = \frac{TP}{TP + FP} \tag{1}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{2}$$

$$\text{f1-score} = 2 * \frac{(\text{Recall} * \text{Precision})}{(\text{Recall} + \text{Precision})} \tag{3}$$

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FN + FP} \tag{4}$$

where TP: Positive predicted and it is true, FP: Positive predicted but it is false, TN: Negative predicted and it is true, and FN: Negative predicted but it is false.

The instances of TP, FP, TN, and FN can be referred to a binary classification of confusion matrix as shown in Table 2. The formula of Precision can be easily calculated from the ratio of correctly predicted positive class (TP) to the total sum of a total number of instances that the model predicted as positive (TP + FP). Meanwhile, recall also known as True Positive Rate (TPR) or Sensitivity indicates the ratio of correctly predicted positive class (TP) to the total number of actual positive classes (TP + FN). The f1-score is the harmonic mean of precision and recall. Accuracy measures the ratio of correctly predicted instances (TP and TN) to the total number of images.

Table 2. Confusion matrix

		Actual Class		
		P	N	
Predicted Class	P	TP	FP	$\text{Precision} = \frac{TP}{TP + FP}$
	N	FN	TN	
		$\text{Recall} = \frac{TP}{TP + FN}$		$\text{Accuracy} = \frac{TP + TN}{TP + TN + FN + FP}$

4. RESULTS AND DISCUSSION

This section discusses the results of the best pre-trained model, hyperparameter tuning, model training, and app testing.

4.1 The best pre-trained model

Table 3 demonstrates that InceptionV3 achieved the highest accuracy of 45.8% among 25 pre-trained models after running them for 10 epochs. Therefore, InceptionV3 was chosen as the base pre-trained model for this study.

4.2 Hyperparameters Tuning

The input images were passed through at the top of the InceptionV3 model and processed until the last layer, where the 2048 features were extracted from the final layer and lastly saved in NumPy arrays. These arrays were then loaded as input for the hyperparameters tuning pipeline of a Keras-based deep learning model using Grid Search with cross-validation as the hyperparameters range given in Table 4. Table 5 shows that the best hyperparameters were obtained as a batch size of 4, a learning rate of 0.01, and 50 epochs with the SGD optimizer, resulting in a validation accuracy of 1.0.

Table 3. Validation accuracy for all pre-trained CNN models from KERAS, Python

	Model name	Num model params	Validation accuracy		Model name	Num model params	Validation accuracy
23	MobileNetV3Small	939120	0.333333	1	DenseNet169	12642880	0.333333
21	MobileNetV2	2257984	0.416667	14	EfficientNetV2B3	12930622	0.333333
22	MobileNetV3Large	2996352	0.333333	7	EfficientNetB4	17673823	0.416667
20	MobileNet	3228864	0.250000	2	DenseNet201	18321984	0.291667
3	EfficientNetBO	4049571	0.333333	17	EfficientNetV2S	20331360	0.375000
11	EfficientNetV2B0	5919312	0.250000	19	Inception V3	21802784	0.458333
4	EfficientNetB1	6575239	0.250000	8	EfficientNetB5	28513527	0.416667
12	EfficientNetV2B1	6931124	0.333333	9	EfficientNetB6	40960143	0.333333
20	DenseNet121	7037504	0.375000	16	EfficientNetV2M	53150388	0.333333
5	EfficientNetB2	7768569	0.333333	18	InceptionResNetV2	54336736	0.375000
13	EfficientNetV2B2	8769374	0.416667	10	EfficientNetB7	64097687	0.250000
6	EfficientNetB3	10783535	0.250000	15	EfficientNetV2L	117746848	0.250000

Table 4. Hyperparameter range

Hyperparameters	Range
Batch size	[4, 8, 16],
Optimizer	['SGD', 'RMSprop', 'Adagrad', 'Adadelta', 'Adam', 'Adamax', 'Nadam'],
Learning rate	[0.01, 0.001, 0.0001],
Epochs	[50, 40, 30],

Table 5. Tuned hyperparameters

Hyperparameters	Best Value
Batch size	4
Optimizer	SGD
Learning rate	0.01
Epochs	50

Train Classification Report

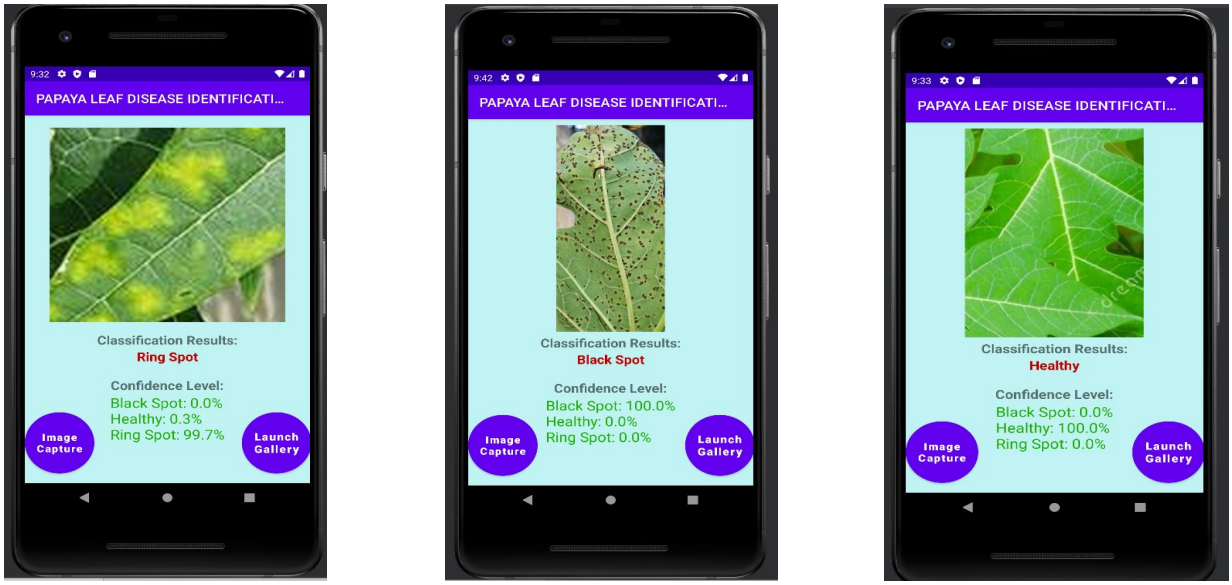
	precision	recall	f1-score	support
Black Spot	1.00	1.00	1.00	170
Healthy	1.00	1.00	1.00	170
Ring Spot	1.00	1.00	1.00	170
accuracy			1.00	510
macro avg	1.00	1.00	1.00	510
weighted avg	1.00	1.00	1.00	510

Figure 3. Training classification report of model training with the best hyperparameters

Validate Classification Report

	precision	recall	f1-score	support
Black Spot	1.00	1.00	1.00	15
Healthy	1.00	0.87	0.93	15
Ring Spot	0.88	1.00	0.94	15
accuracy			0.96	45
macro avg	0.96	0.96	0.96	45
weighted avg	0.96	0.96	0.96	45

Figure 4. Validation classification report of model training with the best hyperparameters



(a) Ring spot (b) Black spot (a) Healthy

Figure 5. Designed App operation for papaya leaf disease identification

4.3 Model Training

Figures 3 and 4 show the Model Training session obtained a training accuracy of 1.0 and a validation accuracy of 0.96 utilizing the above best hyperparameters. The difference of 0.4 in validation accuracy compared to the tuning session can be attributed to variances in the feature extraction method and the utilization of sequential layers between the two scripts. In the hyperparameters tuning session, the extracted features from the final layer of the InceptionV3 model were employed to enhance tuning efficiency. Conversely, in the model training session, the image dataset was directly applied and inputted from the top of the base model until the last dense layer which consists of 3 output nodes for each class. As the model training session did not involve extensive tuning with multiple combinations of hyperparameters, minor differences in validation accuracy can be expected.

4.4 App Testing

The trained model was exported in the tflite file format and was integrated into the app development using Android Studio. Figure 5 shows samples of the classification results for each class using the developed app. The app allows users to take real-life images using a camera from a handphone or images from a phone gallery. The image is shown on top, and its identification result is shown in the middle with the confidence level for each class shown at the bottom.

The duration for image classification takes only a second. 15 unseen images were selected from each class and imported into the designed app for classification. Table 6 shows the confusion matrix of this testing. On the other hand, Table 7 shows the average test precision of 0.8889, recall of 0.8889, f1-Score of 0.8889, and accuracy of 08889. These results indicate that the designed Android-based app performs with relatively high accuracy in identifying papaya plant disease through papaya leaves.

Table 6. Confusion matrix of designed android App test

		Predicted Class		
		Black Spot	Healthy	Ring Spot
Actual Class	Black Spot	14	0	1
	Healthy	1	13	1
	Ring Spot	0	2	13

Table 7. Classification performance evaluation of designed android App

Class	Precision	Recall	f1-score
Black Spot	0.9333	0.9333	0.9333
Healthy	0.8667	0.8667	0.8667
Ring Spot	0.8667	0.8667	0.8667
Average	0.8889	0.8889	0.8889
Accuracy	0.8889		

Since there is no readily available benchmark dataset for papaya leaf diseases, each study used self-collected images, resulting in a non-standardized basis for comparison. Nevertheless, this study attempted to compare our results with other studies that also classified papaya leaf diseases. Our study achieved an accuracy of 88.89%, surpassing the study in [21], which attained an accuracy of 85.51%. On top of that, variations in the dataset and the number of images used in each study exist. Therefore, the comparison of accuracy should be interpreted as a guideline rather than a direct comparison based on identical criteria. Utilizing InceptionV3 as the base model for feature extraction was a strategic choice in our study. InceptionV3 is widely recognized for its exceptional performance in image classification tasks. This architectural advantage allowed us to harness the power of a pre-trained deep neural network, making our feature extraction technique more efficient and effective than training a full deep neural network from scratch. This decision not only expedited the training process but also leveraged the wealth of knowledge encoded within InceptionV3's layers, enabling our model to capture intricate patterns and features crucial for accurate papaya leaf disease classification.

5. CONCLUSION

A total of 225 papaya leaf images, consisting of 3 classes were collected and augmented to create a dataset of 600 images for the overall model training. InceptionV3 was determined as the best pre-trained CNN model out of 25 Keras API CNN pre-trained models using 10 epochs. Using InceptionV3 as the base model, the training images were fed into the top of the base model, and the 2048 output features were extracted from the last layer of the InceptionV3. These output features were saved as NumPy arrays and subsequently loaded into the hyperparameter tuning pipeline to determine the best hyperparameters, including batch size, epochs, optimizer, and learning rate. This approach enhances the tuning efficiency.

The tuning results revealed that the best set of hyperparameters consisted of a batch size of 4, a learning rate of 0.01, and 50 epochs with the SGD optimizer, leading to a tuning validation of 1.0. Utilizing these hyperparameters, a full model training was conducted, resulting in a training accuracy of 1.0 and a validation accuracy of 0.96. A difference of 0.4 from the validation accuracy was obtained from the tuning session due to the difference in the feature extracting method and usage of sequential layers of both scripts. The trained model was exported in the form of tflite file format and was used in the app development by using Android Studio. When importing a selected set of 15 images from each class into the designed app, the test accuracy achieved an average precision of 0.8889, a recall of 0.8889, an f1-Score of 0.8889, and an accuracy of 0.8889. These results demonstrate that the Android-based app exhibits a relatively high accuracy in identifying plant disease of papaya through papaya leaves.

In the future, further testing can be conducted by capturing papaya leaf images in any papaya plantation crops to explore the accuracy and limitations of the app. Additionally, enhancements can be made to the app by acquiring more training images for the current class and adding additional classes to the model, enabling it to detect a broader range of papaya plant leaf diseases.

ACKNOWLEDGEMENT AND FUNDING

This research was supported by Universiti Tun Hussein Onn Malaysia (UTHM) through Tier 1 (Vote Q489).

DECLARATION OF CONFLICTING INTERESTS

The authors declare no potential conflicts of interest with respect to the research and publication of this article.

REFERENCES

[1] USDA Foreign Agricultural Service Malaysia, Agricultural sector. https://www.trade.gov/country-commercial-guides/malaysia-agricultural-sector (accessed 08.06.2023)

[2] The Sun Daily, More young agropreneurs needed as agriculture industry involves. https://www.thesundaily.my/local/more-young-agropreneurs-needed-as-agriculture-industry-evolves-DD8603544 (accessed 08.10.2023)

[3] R. A. Dardak, Overview of the agriculture sector during the 11th Malaysian Development Plan (2016-2020). https://ap.fftc.org.tw/article/3010 (accessed 08.06.2023)

[4] L. Kaur and S. G. Sharma, Identification of plant diseases and distinct approaches for their management, *Bulletin National Research Cen*tre, 45, 2021, 1-10.

[5] R. Sekeli, M. H. Hamid, R. A. Razak, C. Y. Wee and J. Ong-Abdullah, Malaysian Carica papaya l. var. eksotika: current research strategies fronting challenges, *Frontiers in Plant Science*, 9, 2018, 1-9.

[6] Consumer's Association of Penang, Papaya trees in Perak hit by virus attack. https://consumer.org.my/papaya-trees-in-perak-hit-by-virus-attack/ (accessed 08.06.2023)

[7] D. Gupta, Transfer Learning and the Art of Using Pre-trained Models in Deep Learning, *Analytics Vidhya*. https://www.analyticsvidhya.com/blog/2017/06/transfer-learning-the-art-of-fine-tuning-a-pre-trained-model/ (accessed 08.06.2023)

[8] P. Sanjay, B. Dhaygude, M. Nitin and P. Kumbhar, Agricultural plant leaf disease detection using image processing, *International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering*, 2, 2013, 599-602.

[9] A. V. S. Yashodharan, Neural network for papaya leaf disease detection, *Acta Graphica*, 30, 2019, 11-24.

[10] M. T. Habib, A. Majumder, A. Z. M Jakaria, M. Akter, M. S. Uddin and F. Ahmed, Machine vision based papaya disease recognition, *Journal of King Saud University - Computer and Information Sciences*, 32, 2020, 300-309.

[11] S. P. Mohanty, D. P. Hughes and M. Salathé, Using deep learning for image-based plant disease detection, *Frontiers in Plant Science*, 7, 2016, 1-10.

[12] K. K. Leong and L. L. Tze, Plant leaf diseases identification using convolutional neural network with treatment handling system, *Proceeding of 2020 IEEE International Conference on Automatic Control and Intelligent Systems,* Malaysia, 2020, 39-44.

[13] N. Shelar, S. Shinde, S. Sawant, S. Dhumal and K. Fakir, Plant disease detection using CNN, *Proceeding of International Conference on Automation, Computing and Communication 2022*, Mumbai, 2022, 1-6.

[14] H. Ajra, M. K. Nahar, L. Sarkar and M. S. Islam, Disease detection of plant leaf using image processing and CNN with Preventive Measures. *Proceeding of 2020 Emerging Technology in Computing, Communication and Electronics,* Dhaka, 2020, 1-6.

[15] S. Kumar, V. Chaudhary, M. Supriya and K. Chandra, Plant disease detection using CNN, *Turkish Journal of Computer and Mathematics Education*, 12, 2021, 2106-2112.

[16] X. Guan, A novel method of plant leaf disease detection based on deep learning and convolutional neural network, *Proceeding of 2021 IEEE 6th International Conference on Intelligent Computing and Signal Processing,* Xi'an, China, 2021, 816-819.

[17] S. M. Hassan, A. K. Maji, M. Jasiński, Z. Leonowicz and E. Jasińska, Identification of plant-leaf diseases using CNN and transfer-learning approach, *Electronics*, 10, 2021, 1-19.

[18] M. Sardogan, A. Tuncer and Y. Ozen, Plant leaf disease detection and classification based on CNN with LVQ algorithm, *Proceeding of 2018 3rd International Conference on Computer Science and Engineering,* USA, 2018, 382-385.

[19] P. Tejaswini, P. Singh, M. Ramchandani, Y. K. Rathore and R. R. Janghel, Rice leaf disease classification using CNN, *Proceeding of International Conference on Advances in Earth and Environmental Studies*, India, 2022, 1-14.

[20] V. V. Srinidhi, A. Sahay and K. Deeba, Plant pathology disease detection in apple leaves using deep convolutional neural networks: apple leaves disease detection using EfficientNet and DenseNet, *Proceedings of 5th International Conference on Computing Methodologies and Communication,* India*,* 2021, 1119-1127.

[21] R. K. Veeraballi, M. S. Nagugari, C. S. R. Annavarapu and E. V. Gownipuram, Deep learning based approach for classification and detection of papaya leaf diseases, *Intelligent Systems Design and Applications. Advances in Intelligent Systems and Computing*, 940, 2018, 291-302.

[22] J. A. Bacus and N. B. Linsangan, Detection and identification with analysis of Carica papaya leaf using Android, *Journal of Advances in Information Technology*, 13, 2022, 162-166.

[23] H. Ali, M. I. Lali, M. Z. Nawaz, M. Sharif and B. A. Saleem, Symptom based automated detection of citrus diseases using color histogram and textural descriptors, *Computers and Electronics in Agriculture*, 138, 2017, 92-104.

[24] M. Sharif, M. A. Khan, Z. Iqbal, M. F. Azam, M. I. U. Lali and M. Y. Javed, Detection and classification of citrus diseases in agriculture based on optimized weighted segmentation and feature selection, *Computers and Electronics in Agriculture*, 150, 2018, 220-234.