Computers and Electronics in Agriculture 182 (2021) 105986

Contents lists available at ScienceDirect

Computers and Electronics in Agriculture

journal homepage: www.elsevier.com/locate/compag

Original papers

Disease and pest infection detection in coconut tree through deep learning techniques

Piyush Singh [a], Abhishek Verma [a], John Sahaya Rani Alex [b,*]

[a] School of Electronics Engineering, Vellore Institute of Technology, Chennai 600127, India
[b] Division of Healthcare Advancement, Innovation, and Research, Vellore Institute of Technology, Chennai 600127, India

ARTICLE INFO

Keywords:
Coconut tree
Deep learning
InceptionResNetV2
MobileNet
Transfer learning

ABSTRACT

The coconut palm plantation industry relies heavily on expert advice to identify and treat infections. Computer vision in deep learning technology opened up an avenue in the agriculture domain to find a solution. This study focuses on the development of an end-to-end framework to detect stem bleeding disease, leaf blight disease, and pest infection by Red palm weevil in coconut trees by applying image processing and deep learning technology. A set of hand-collected images of healthy and unhealthy coconut tree images were segmented by employing popular segmentation algorithms to easily locate the abnormal boundaries. The custom-designed deep 2D-Convolutional Neural Network (CNN) is trained to predict diseases and pest infections. Also, the state of the art Keras pre-trained CNN models VGG16, VGG19, InceptionV3, DenseNet201, MobileNet, Xception, InceptionResNetV2, and NASNetMobile were fine-tuned to classify the images either as infected or as healthy through the inductive transfer learning method. The empirical study ascertains that k-means clustering segmentation was more effective than the Thresholding and Watershed segmentation methods. Furthermore, InceptionResNetV2 and MobileNet obtained a classification accuracy of 81.48% and 82.10%, respectively, and Cohen's Kappa values of 0.77 and 0.74, respectively. The hand-designed CNN model achieved 96.94% validation accuracy with a Kappa value of 0.91. The MobileNet model and customized 2D-CNN model were deployed in the web application through the micro-web framework Flask to automatically detect the coconut tree disease or pest infection.

1. Introduction

Coconut (*Cocos nucifera* L.) is a palm plantation vital for its various uses from its fruit to trunk. India is the third-largest producer of coconut (Sreejith et al., 2013) and its by-products in the world. The southern states of India contribute a majority of the production in the country (Snehalatharani et al., 2016). Any disease affecting the yield of the coconut plantation eventually affects the related industries and the livelihood of the families who depend on the coconut economy. The major diseases that affect coconut trees are Ganoderma – basal stem rot (BSR), Root (wilt) disease (RWD), bud rot, leaf blight, stem bleeding, and leaf rot (Michel et al., 2012), and the common pests are Rhinoceros beetle, Red palm weevil, Black-headed caterpillar, Coconut Eriophyid Mite and Termite (Arulandoo et al., 2016; Chandy, 2019; Nampoothiri et al., 2019). To detect the symptoms of the above-mentioned diseases and pest infections using technology, a large collection of a dataset is required. The focus of this work has been limited to detect only stem bleeding disease, leaf blight disease, and pest infection by Red palm weevil, as collecting data to include information from all the diseases and the pests that affect the coconut tree is an arduous and time-consuming task.

The symptoms of stem bleeding disease are fluid seeping down the trunk and reddish/dark–brown stains on the trunk that becomes darker and turn black over time (Warwick and Passos, 2009). This disease is caused by *Chalara paradoxa,* a soilborne plant-pathogenic fungus that thrives on plant debris in the soil. If untreated or undetected, it leads to irreversible loss causing the death of the tree and also affects the neighbouring plant (Michel et al., 2012). Regular inspection of the plant for the above symptoms is required. Although an expert system developed to manage coconut diseases based on knowledge and rule-based agents (Dath and Balakrishnan, 2016) is found in literature, it does not provide automatic classification of coconut diseases.

Leaf blight disease is an airborne disease prevalent in the summer season; it gets aggravated because of the drought conditions. It affects the yield of nut production. The early symptoms are yellowish-brown spots on the leaflet which turn grey over time. At an advanced stage,

* Corresponding author.
E-mail address: jsranialex@vit.ac.in (J.S.R. Alex).

https://doi.org/10.1016/j.compag.2021.105986
Received 28 July 2020; Received in revised form 26 December 2020; Accepted 4 January 2021
Available online 27 January 2021

the leaf appears to be burnt (Michel et al., 2012). Image processing along with a support vector machine algorithm is applied to detect diseases in coconut leaves. State-of-the-art technology such as the deep learning method is applied to other plant leaves such as tomato and maize to detect leaf blight disease (Sun et al., 2020; Zhang et al., 2020) but not implemented for coconut.

Red palm weevil (RPW) is one of the lethal pests of the coconut tree and it starts infecting the tree at the larval stage (Giblin-Davis et al., 2013). The symptoms of severe infestations are wilting, the formation of a hole in the stem, and the death of the leaves at the crown. Literature reports that trained sniff dogs and acoustic sensors are used to detect early infection of RPW at the larval stage (Harith-Fadzilah et al., 2020). However, they report poor accuracy and implementation issues on a large scale.

In the literature, advanced methods were applied to detect disease in oil-palm plantations. However, not much was done for coconut palm plantations even though both plants are from the same palm family. Deep learning algorithms that extract discriminative features from the images implicitly gained popularity in computer vision. This motivated us to use it for coconut disease and pest infection detection. In this paper, the intention is to use image processing algorithms along with deep learning models to automate the detection of the infected coconut tree. Hence, the objectives of this work are to explore efficient segmentation techniques to identify the infected region in the image, to develop an optimized deep learning model to identify the infection from the segmented images, to compare the efficacy of pre-trained deep learning models using inductive transfer learning technique for predicting the pest infection and disease and finally to interface the optimized deep learning model with a web application to instantly detect coconut tree disease and pest infection in real-time.

2. Materials and methods

2.1. Dataset construction

In this work, images of coconut tree diseases and pest infections were captured. The dataset is collected from coconut farms located around Chennai, Tamil Nadu, India. These farms belong to farmers in Kayar village (50 km south of Chennai) and Kanchipuram (45 km east of Chennai) as identified by a plant pathologist, ICAR-KVK, Kattupakkam, TANUVAS, Chennai. Images captured for the dataset include images of healthy plants, stem bleeding infected plants, leaf blight infected plants, and pest affected (RPW) plants. Data annotation was given by the farmers and then verified by the plant pathologist. The count for the number of images goes as Healthy (445 images), Stem Bleeding (151 images), pest infection by RPW (146 images), and Leaf Blight (822 images), constituting a total number of 1564 images. The dataset is split into an independent training set, validation set, and testing set in the ratio of 8:1:1. Each of these sets is further split into 4 sub-classes i.e., healthy and the three diseases. To alleviate the illumination and brightness issues across the two farms, image capturing was done at around 11:30 am in both places on a sunny day. These images were captured by a Digital-SLR Camera (CanonEOS 200D) with an image quality set as 3.8 MP with 2400*1600 resolution. The file size of images taken with these settings varies between 1.5 MB and 3.0 MB. Hence, the images in the dataset have been resized (64x64) for further processing.

The images present in the dataset were pre-processed using the ImageData-Generator class which generates batches of tensor images. The images were rescaled by a factor up to 1/255. The images were randomly flipped in the horizontal direction to generate randomness while training the model. Images were sheared in the counter-clockwise direction up to 0.2 degrees and the zoom range for the images was set to be about 0.2 to provide random zoom. Thus, for each image in the dataset, the ImageData-Generator class generated 255 images which increase our dataset to a collection of approximately 39,000 images.

2.2. Image segmentation

Image segmentation is the process of partitioning regions of homogeneous characteristics in a digital image. The characteristic may be texture, grey level, colour, and so on depending upon the application of interest. Image segmentation is the basis for digital image processing analysis and further helps in digital image classification or recognition. Several image segmentation techniques are discussed in the literature (Yuheng and Hao, 2017). The taxonomy of the segmentation splits it into region-based segmentation, clustering-based segmentation, and edge-based segmentation. There is no single specific segmentation method that is effective for all digital images. Thus, in this research, Thresholding segmentation (THS), Watershed segmentation (WSS), and k-means clustering (KMC) segmentation are employed to segment the images before going for the classification process.

2.2.1. Thresholding segmentation

The most simple method is Thresholding (TH) segmentation (Yuheng and Hao, 2017) which is a region-based segmentation. The basic principle of TH is to set a threshold (T) value and compare each pixel (x, y) value of an image with T and the pixels are labelled based on the comparison with the T value and represented by Eq. (1). The segmented image g (x, y) has two regions namely R1, R2.

$$g(x,y) = \begin{cases} R1 & if \quad f(x,y) > T \\ R2 & if \quad f(x,y) \le T \end{cases} \tag{1}$$

Though the basic TH equation is given by (1), modified TH methods are found in research. In global TH, the T value is unchanged across all the images. If TH changes for every image, then it is local TH. For this research, global TH based on Otsu's method (Sha et al., 2016) is employed. Otsu's TH uses the difference between the objects in the image and the background to minimize the overlap between the two classes. The grey-value histogram of the given image is used by the Otsu method. The algorithm's main aim is to select a T that maximizes the variance between the class (σ_b^2), which is given by (2).

$$\sigma_b^2 = P_1(\mu_1 - \mu)^2 + P_2(\mu_2 - \mu)^2 = P_1P_2(\mu_1 - \mu_2)^2 \tag{2}$$

Where P_1 and P_2 denote class probabilities and μ_i the means of the object and background classes.

2.2.2. Watershed segmentation

Watershed segmentation is based on the application of watershed transformation, which is a region-based segmentation, to an image. There are two models of watershed segmentation. In this research, the flooding model Boundary Likelihood Map (BLM) algorithm is used (Shafarenko et al., 1997). In WSS, to replicate the real object boundary in the segmented images, the Bayesian classification is combined with morphological WSS to get the most conspicuous boundary pixels near the object boundary. In the BLM algorithm, the pixels close to the boundary of the objects and with high colour gradients are collected with boundary probability assigned to each pixel. WSS is a global segmentation and the greyscale image is visualized in topographic representation. The literature claims that high accuracy is achieved in segmentation due to this property.

2.2.3. K-Means Clustering segmentation

K-Means clustering segmentation (Dhanachandra et al., 2015), derived from the clustering method used in pattern recognition problems, is done in feature space. The same algorithm is applied in the spatial domain for image segmentation. The KMC formula is given in (3)

$$J = \sum_{j=1}^{k} \sum_{i=1}^{n} \left| x_i^{(j)} - c_j \right|^2 \tag{3}$$

where J is the objective function, 'k' represents the number of clusters, 'n' is the number of cases and c_j represents the centroid of cluster 'j'.

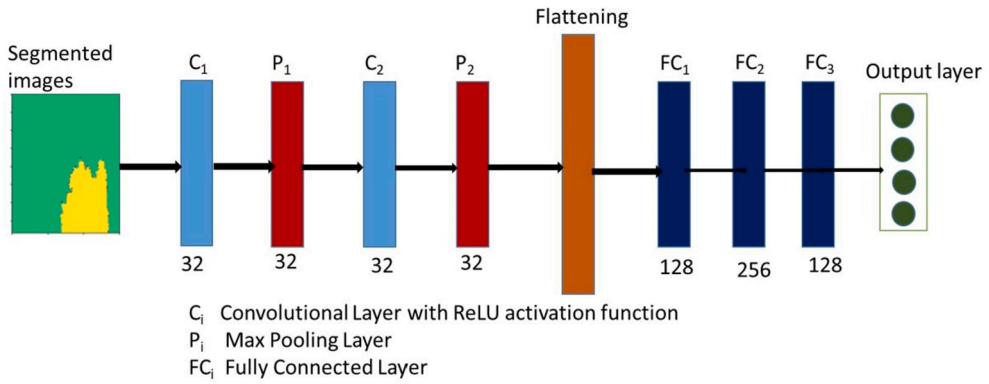


Fig. 1. 2D-CNN Architecture.

The distance measurement in the clustering equation is done with Euclidean distance measurement. The clustering algorithm is a type of unsupervised learning algorithm, which has been employed in this research to work as an image segmentation technique. To segment the image using the KMC technique, the different objects in the images are clustered or segmented into different categories or partitions depending upon the number of clusters pre-defined in the KMC algorithm. Images used for creating a dataset in this research paper had nearly 5 major components that include the coconut tree part, the sky, the background nearby bushes/other nearby trees, the soil/the land part, and the root part. These components usually added 5 different contrast and shade patterns to the image and thus to precisely segment these parts, 'k' has been assigned as 5. The KMC algorithm has only been used as an image segmentation technique in our research and not for classification purposes.

2.3. Convolutional neural network

Convolutional Neural Network (CNN) is a popular state of art neural network architecture efficacious in computer vision applications (Kieffer et al., 2018). Typically, CNN is hand-designed for optimum accuracy for a specific application. A successful CNN designed for one application may not work well for another computer vision application. In this research, CNN is hand-designed to classify images of a coconut tree to detect infections. CNN has a sequence of layers which include the input layer, the convolution layer, the pooling layer, the flatten layer, the fully connected layer, and the output layer.

The hand-designed CNN model is shown in Fig. 1. It consists of 2 convolution layers that are followed by a max-pooling layer. Also, the number of feature maps associated with the convolutional layer and the max-pooling is 32. The output of the second max-pooling layer is then flattened and given to three fully connected layers each with 128, 256, and 128 hidden neurons, respectively. The output layer has 4 nodes representing the output classes such as Healthy, Stem Bleeding, Pest infection, and Leaf Blight. This CNN architecture is trained, validated, and tested with segmented images from Section 2.2. It is also trained and tested with normal images without a segmentation algorithm.

2.4. Transfer learning

Deep learning models are efficacious in image classification, which depends on the number of layers and the amount of data with which the models are trained with. The hand-collected dataset is limited to about 1564 images; after data augmentation with the image data generator class, the number of images increased to 39,000 images. However, a large scale of the annotated dataset is required for deep learning technology. To achieve a good classification accuracy with a limited amount of data, research suggests the usage of pre-trained deep learning (DL) models via transfer learning (Pan and Yang, 2010). Keras pre-trained models (Chollet, 2015) are deep learning models that are made available alongside pre-trained weights. These models can be used for prediction, feature extraction, and fine-tuning (Kieffer et al., 2018). In this research, eight DL models namely VGG16 (Simonyan and Zisserman, 2015), VGG19, Xception (Chollet, 2017), InceptionV3 (Szegedy et al., 2015), InceptionResNetV2 (Szegedy et al., 2017) MobileNet (Sandler et al., 2018), DenseNet201 (Huang et al., 2017) and NASNetMobile (Zoph et al., 2018) are employed with transfer learning technique for coconut tree disease classification using data augmented dataset. All these models' weights are optimized for ImageNet (Russakovsky et al., 2015) data classification. ImageNet is a dataset; a collection of 15 million high-resolution images belonging to 22,000 categories in which 4,400 classes belong to plant, flora, and plant life. The pre-trained models' top layers are not included as they are responsible for the classification into 22,000 classes. The rest of the architecture is kept intact. Also, the Global Average Pooling 2D layer, as proposed by Lin et al. (2014), along with the 4 output nodes using sequential class is added as top layers. SoftMax activation function (Duan et al., 2003) is used in the output layer to get the probability distributions. The Deep Learning (DL) architecture details of the selected Keras pre-trained models are listed in Table 1.

Table 1
Hyperparameters of the selected deep learning models.

Network	Depth	Size	Parameters (Millions)
VGG16	16	528 MB	138.3
VGG19	19	549 MB	143.6
InceptionV3	48	92 MB	23.8
DenseNet201	201	80 MB	20.2
MobileNet	53	16 MB	4.2
Xception	71	85 MB	22.9
InceptionResNetV2	164	215 MB	55.8
NASNetMobile	*	23 MB	5.3

2.4.1. VGGNet

VGG16 is a CNN model (Simonyan and Zisserman, 2015) consisting of 13 convolution layers and 3 fully connected layers, a total of 16 wt layers. The model parameters are over 533 MB. Though the network is deep, and the computing complexity is huge, the performance is well over AlexNet, GoogLeNet and it is also easy to implement. This model achieved 92.70% accuracy in the ILSVRC-2014 competition on ImageNet for 1000 classes. VGG19 is a modified version of VGG16 with an additional 3 convolutional layers.

2.4.2. InceptionV3

InceptionV3 model (Szegedy et al., 2015) was proposed with the idea of factoring bigger convolution filters into multiple smaller convolutions, thus reducing the parameters without degrading the performance of the network. Other changes introduced were auxiliary classifier on

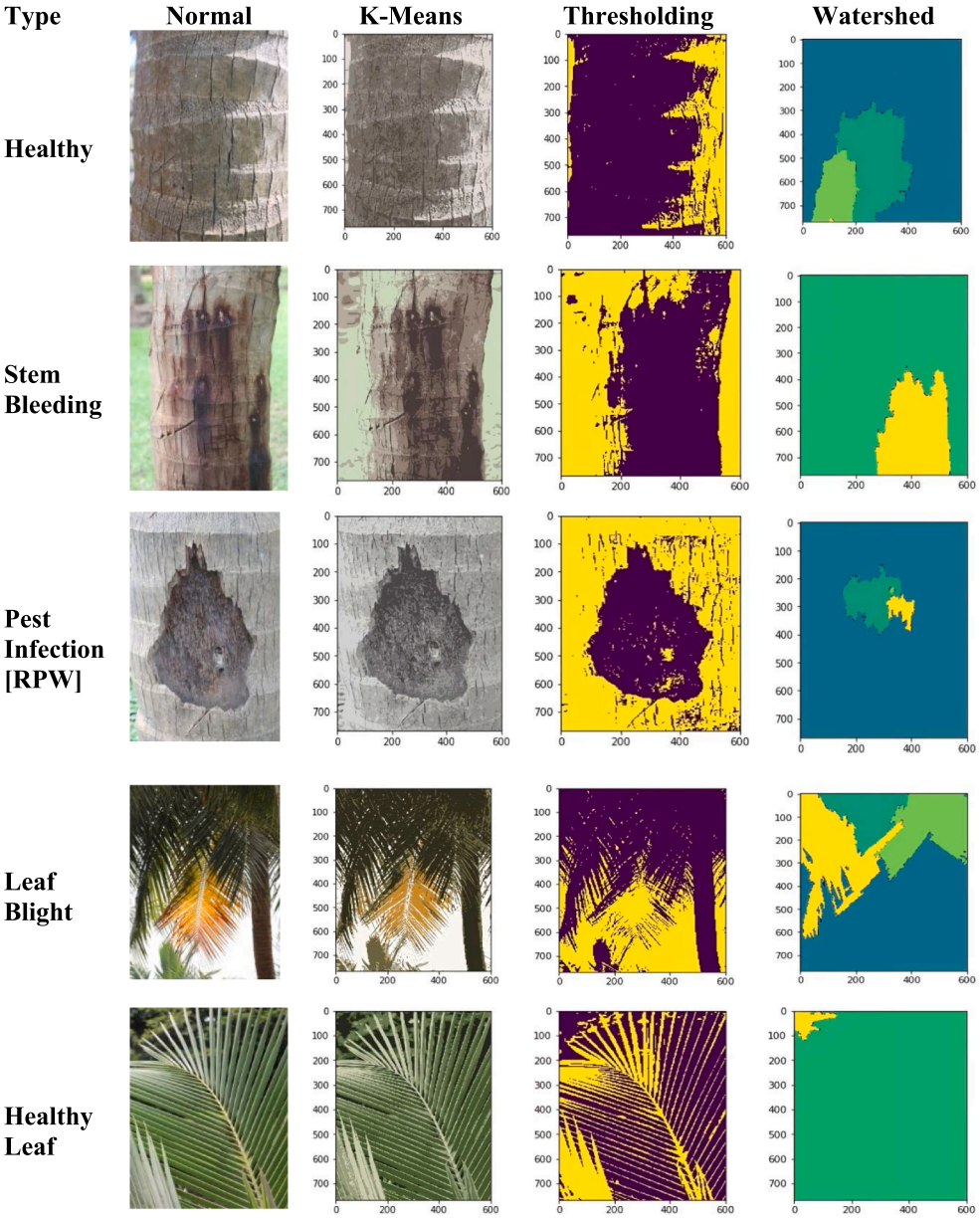


Fig 2. Comparison of segmentation methods for different categories.

Table 2
Validation accuracies of hand-designed 2D-CNN model.

Segmentation Method	Validation Accuracy in %
Threshold segmentation	78.09
Watershed segmentation	68.31
K-means clustering segmentation	85.23
No segmentation	80.63

top of the last layer and feature map downsizing achieved with efficient grid size reduction. Due to these features, the performance was better than VGGNet in ILSVRC 2015 despite the model having 42 layers.

2.4.3. InceptionResNetV2

It is a hybrid architecture of Inception and ResNet, thus having a significantly improved performance over the earlier architectures.

2.4.4. Xception

Xception architecture is called extreme Inception, a linear stack of depth-wise residual models. This has the same number of parameters as InceptionV3 but it is easier to implement. The performance is better than the earlier architectures VGGNet, InceptionV3, and ResNet. The residual models in the architecture induce fast convergence.

2.4.5. MobileNet

An efficient CNN model for mobile and embedded platforms, MobileNet is a depth-wise separable convolution model with a focus to reduce latency and is a small network in size (Howard et al., 2017). There is a trade-off between the accuracy and the choice of multiplier width and thus the size of the network. MobileNet has been chosen for transfer learning to implement our model in the mobile platform for the public.

2.4.6. DenseNet201

As the name indicates, it has dense connections in such a way that the i^{th} layer contains i connections instead of one connection from the previous layer as in the case of the traditional feed-forward network. This removes the vanishing-gradient problem, reinforces feature propagation, improves feature reuse, and significantly reduces the number of parameters.

2.4.7. NASNetMobile

It is a state-of-the-art architecture, proposed by Google Brain, which exhibits smaller model sizes with reduced floating-point operations and outperforms previous models. Neural architecture search (NAS) net searches for cells on a smaller dataset based on reinforcement learning search.

2.5. Web application

A web application was developed to deploy the trained neural network model for disease detection in coconut trees. The web application was developed to enable the instant use of the neural network model for real-world applications. Flask, a micro web framework written in Python, was used to develop the web application. The micro-framework is named so because it does not require any particular tools or libraries to run on different machines. However, Flask supports extensions that allow additional application features to the web application. The web application allows the user to either capture a photo using the mobile phone's camera or uses the image stored in the device. The web application's upload button then uploads the image to the neural network model's H5 file which then classifies the image using the pre-stored weights of the model. Accordingly, the resulting template is rendered, and the user gets the classified disease name and the instant remedy on their screen.

Table 3
Performance comparison of hand-designed CNN model with various numbers of convolution layers.

No of Convolution Layers	Validation Accuracy in %	Cohen's Kappa coefficient
2	85.63	0.80
3	89.07	0.86
4	96.94	0.91
5	90.21	0.88

3. Experimental setup

The self-collected images were pre-processed and applied with various segmentation algorithms discussed in Section 2. The segmented images are evaluated with the 2D-CNN model that has been designed.

The models discussed in Section 2 were compiled with loss function as categorical cross-entropy because of the 4 classes to classify. The RMSprop optimizer (Dauphin et al., 2015) with the learning rate set to 0.0001 is applied in this work. The Keras pretrained DL models used in transfer learning tend to overfit as they memorize the training data very quickly. They are complex and have a high number of parameters. To counter this tendency, L2 regularization (sum of the squared weights) (Cortes et al., 2004) was set to 0.75, with early stopping. For early stopping, validation loss with patience as 5 epochs was monitored. All the selected DL models were initially set to train up to 150 epochs. However, only VGG16 and VGG19 were trained till 150 epochs whereas the rest of the DL models were early stopped. After this initial training, the models are fine-tuned with early stopping for 50 more epochs to capture the application-specific fine features. The experiments were performed with 2.20 GHz Intel(R) Core (TM) i7-8750H CPU, 16 GB DDR4 RAM, NVIDIA GeForce 1050 Ti 4 GB DDR5 GPU. All the selected DL models were trained and fine-tuned through Keras library. Finally, the validated optimum deep learning network is deployed using Flask in the web portal.

4. Results and discussions

The segmentation results of the image processing algorithm are shown in Fig. 2. The results indicate that the KMC segmentation algorithm performed better than WSS and THS methods for segmenting the regions. Also, it shows that WSS, when applied to the image, fails in segmenting the relevant features in all four categories. THS does a good job in segmenting the portion of the trunk with the hole; however, in other categories, it did not perform very well. For example, a healthy trunk, identifies the shadow part of the trunk to be a trunk. It also does not segment stem bled portions of the trunk properly. Furthermore, THS fails to segment the portion of the leaf with leaf blight. On the other hand, KMC segmentation carries out its function to segment relevant regions in every category without problems. For example, it does not misidentify shadow in the trunk be a trunk. Similarly, stem bled regions and sections with holes are properly segmented. Unlike THS, a portion of a leaf with blight is segmented appropriately.

The segmented images are then applied as input to the hand-designed CNN model. The accuracy of the CNN model for various segmented images is shown in Table 2. The perceptive measurement, which was shown in Fig. 2, matched with the objective measurement of the CNN model listed in Table 2.

The KMC segmented images delivered a higher classification accuracy than the other segmentation methods

The hand-designed CNN model is experimented with 2, 3, 4, and 5 convolution layers. An input image of size 64 × 64 is fed into a convolution layer of 32 feature maps with a filter kernel size of 3 × 3. The convolutional layer's activation function- Rectified Linear Unit (ReLU) output is given to the max-pooling layer with a size of 2 × 2 for downsampling. Convolution layers followed by the max-pooling layer extract the high-level features from the segmented images in the spatial

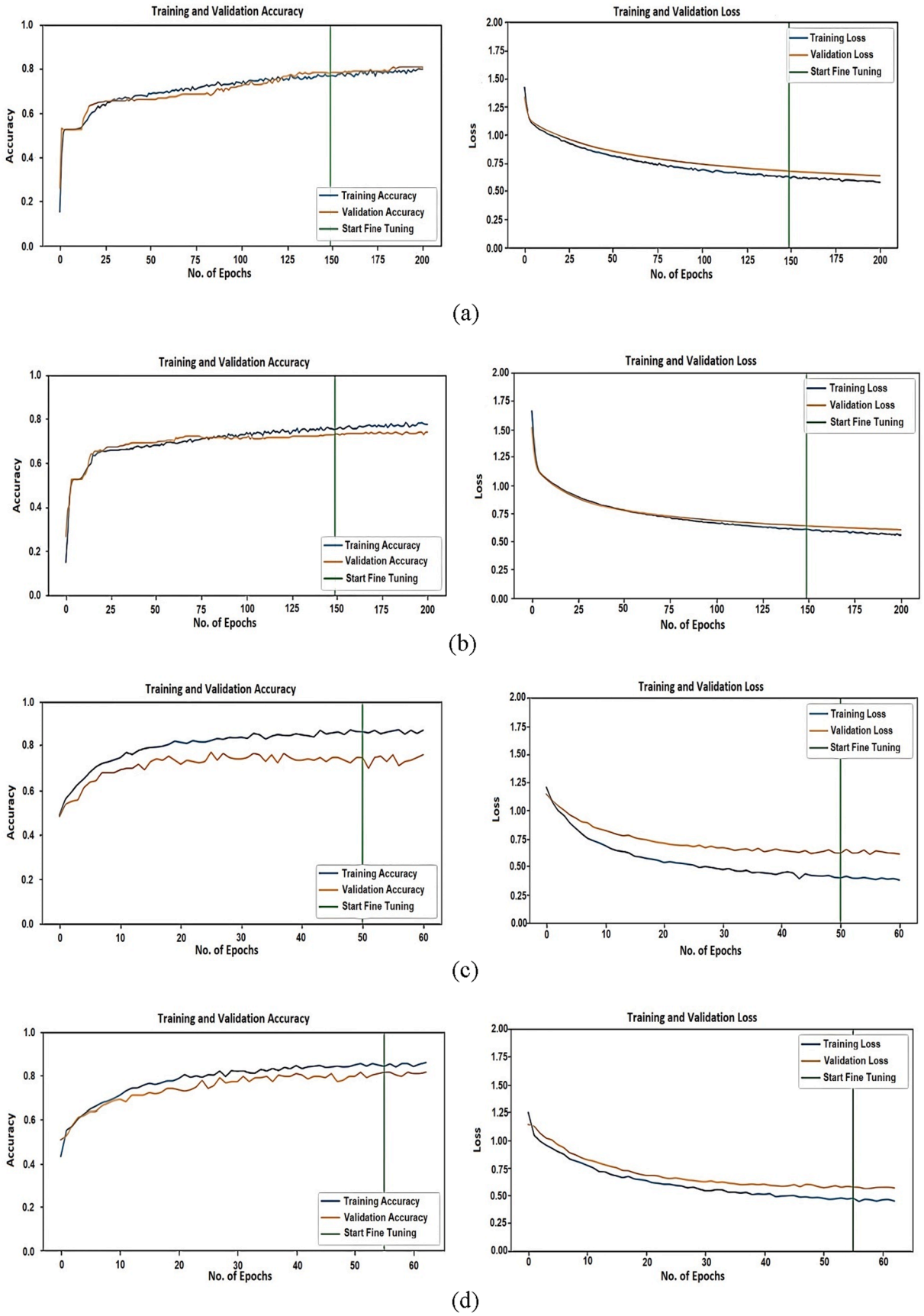


Fig. 3. Performance of eight deep learning models for coconut disease detection through inductive transfer learning. (a) VGG16 (b) VGG19 (c) InceptionV3 (d) InceptionResNetV2 (e) Xception (f) MobileNet (g) DenseNet201 (h) NASNetMobile.

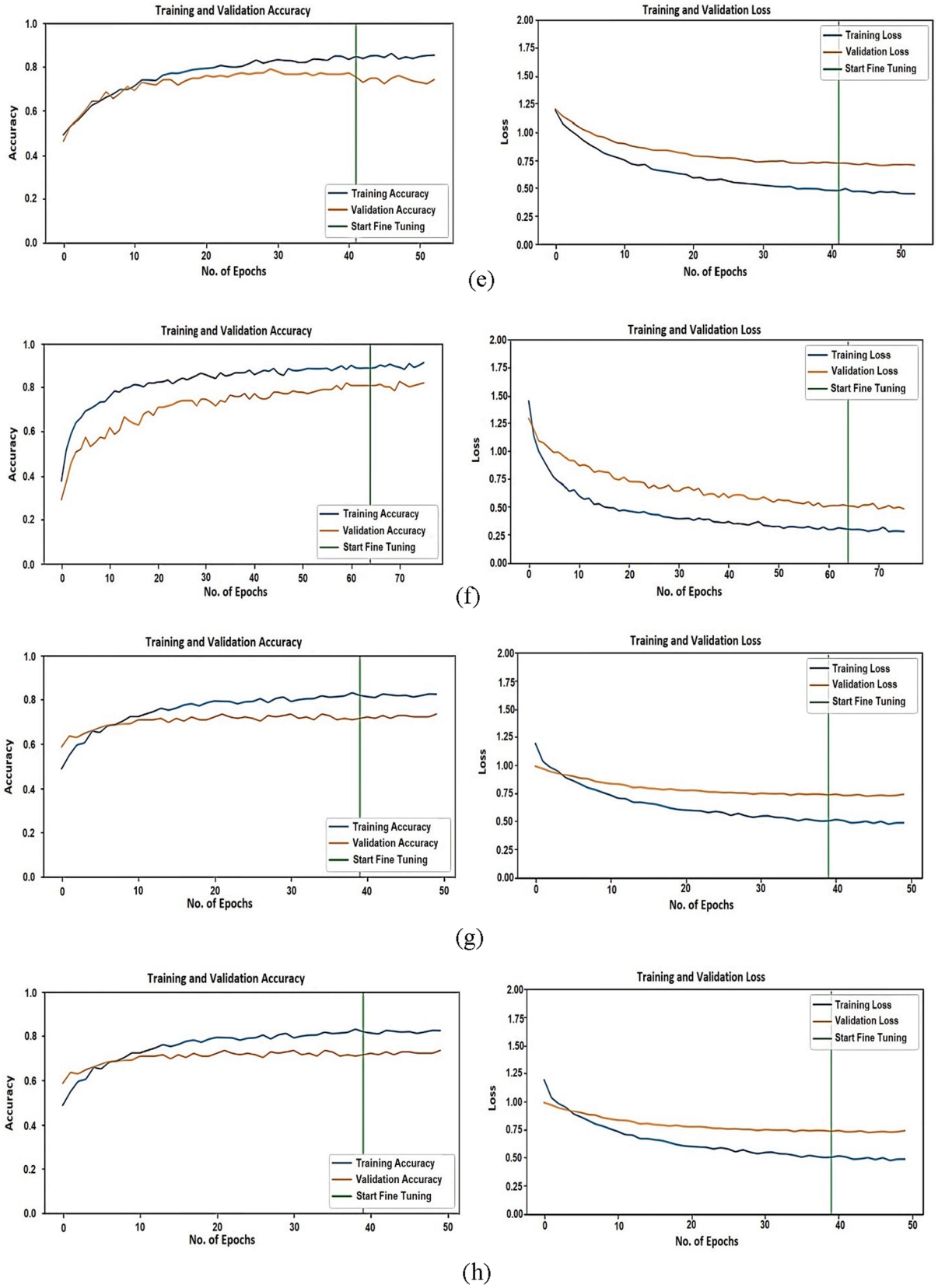


Fig. 3. (*continued*).

Table 4
Performance metrics of selected deep learning models.

Deep Learning Model	Accuracy (before fine-tuning)	Accuracy (after fine Tuning)	Cohen's Kappa coefficient
VGG16	78.40	80.86	0.73
VGG19	72.84	74.07	0.69
InceptionV3	74.69	75.93	0.70
DenseNet201	78.40	79.01	0.62
MobileNet	80.86	82.10	0.74
Xception	75.31	74.07	0.71
InceptionResNetV2	81.48	81.48	0.77
NASNetMobile	71.60	73.46	0.60

domain. The number of convolution layers in between the input layer and the flattened layer is experimented with for high classification accuracy. Next, the two-dimensional output from the last max-pooling layer is converted to a single dimension. It is then given as input to a traditional three-layer full connected neural network for classification of the high-level features obtained from CNN layers. The traditional hidden layers are implemented with the ReLU activation function. The final output layer has a sigmoid function for categorical classification. Based on empirical study, high accuracy is obtained for four convolution layers. Upon increasing the number of layers to more than four, the accuracy tends to decrease. This is because it tends to generalize on the training data. The parameters that are experimented with to get more accuracy are image augmentation, learning rate, choice of the optimizer, number of convolution layers, kernel size of the filter, and batch normalization. On increasing the number of fully connected layers, the CNN model started to over-fit, and to eliminate this, the fully connected layers were limited to 3 to get the best generalization results. Table 3 gives the validation accuracy, Cohen's Kappa coefficient for a varied number of convolution layers, which reveals that a hand-designed CNN model with 4 Convolutional layers obtained good validation accuracy and kappa score and the number of convolution layers is inversely

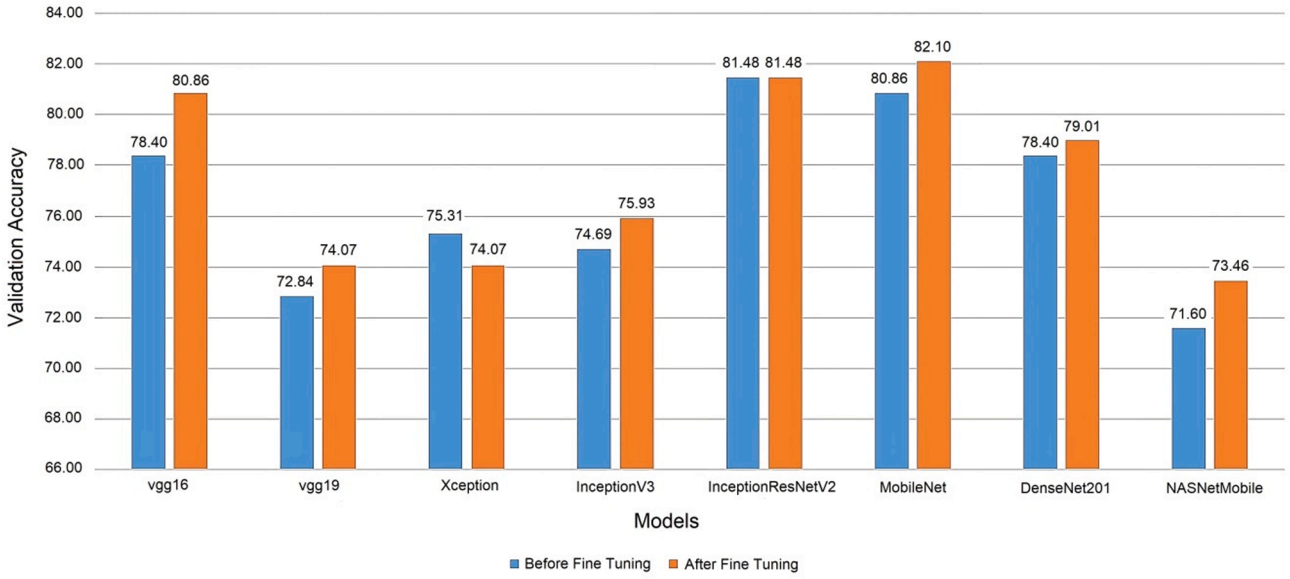


Fig. 4. Comparison of eight deep learning models validation accuracy before and after fine-tuning.

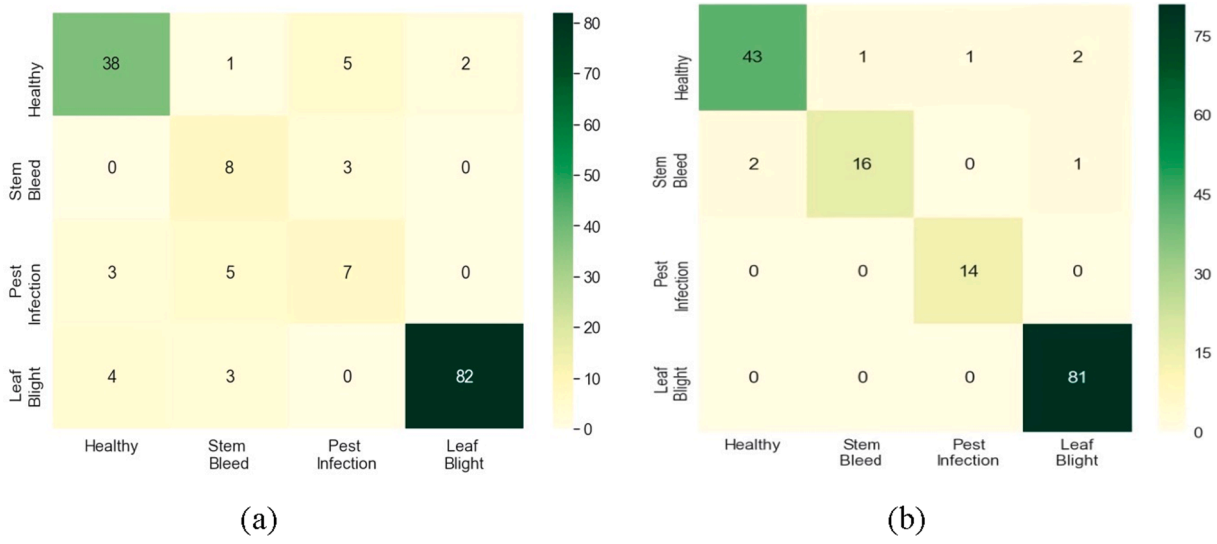
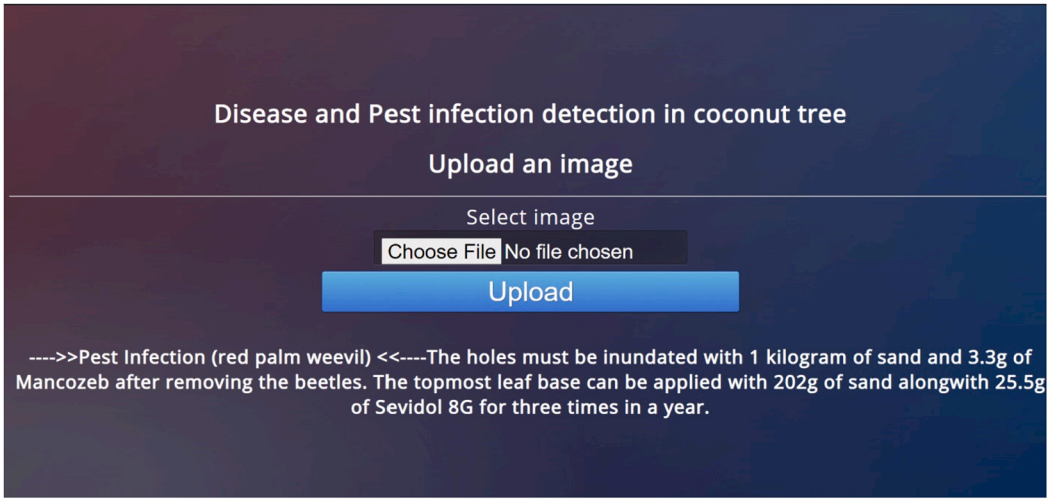


Fig. 5. Confusion matrix. (a) MobileNet (b) 2D-CNN with 4 layers.

(a)



(b)

(c)

Fig. 6. (a) Web application screenshot of uploading image (b) Web application screenshot of a coconut tree infected with pest infection. (c) Web application screenshot of a healthy coconut tree.

proportional to accuracy.

The discussed state-of-the-art Keras deep learning models in section 2 were pre-trained with ImageNet corpus for 1000 class classification. Therefore, all the models' top layers which are the fully connected layer, SoftMax layer, and output layer of 100 classes were replaced with a global average layer followed by a sequential layer with SoftMax function and an output layer of four classes as required for this application. The weights of all the layers are frozen and the top layers are trained with target data of k-means segmented coconut images. The training was set to 150 epochs. The results in Fig. 3 specify that VGGNet is better trained than other models in terms of overfitting. VGG16 and VGG19 are the models with the highest number of parameters as listed in Table 1, but with top layers removed, their number of parameters for training is one of the lowest among the models used, i.e., 14 M & 20 M, respectively. Except for VGGNet, all other models were early stopped at around 50 epochs as shown in Fig. 3a and b. They were stopped early because the validation accuracy was not as high as the training accuracy and also the validation loss was not as small as the level of training loss. The early stopped models were more generalized on the source data of ImageNet. Therefore, the deep learning models are then fine-tuned for another 50 epochs. The experiment was carried out for 1/2th, 3/4th, and 9/10th of the total number of convolutional layers. The highest accuracy is achieved when fine-tuning was done from 9/10th of the total number of convolutional layers. Fig. 3 graphs also show the epoch number of fine-tuning for each model. It also shows validation accuracy before and after fine-tuning.

Table 4 gives the performance metrics of the eight Keras pretrained model. From Table 4, the results represent that though the MobileNet accuracy improved by fine-tuning to 82.10%, Cohen's kappa coefficient is less than InceptionResNetV2.

The validation accuracy of the eight deep learning models before and after fine-tuning are displayed in Fig. 4. All the models have decreased validation loss after fine-tuning. Validation accuracy can also be seen increasing, except for the Xception model. Fig. 4 shows MobileNet obtained the highest classification accuracy of 82.10% after fine-tuning. InceptionResNetV2 did not show any improvement with fine-tuning, the classification accuracy stayed at 81.48%. The fine-tuning of the other models provided an improved classification accuracy of 1% to 2%.

The confusion matrix of the best model MobileNet and hand-designed CNN model with 4 layers are shown in Fig. 5a, b.

The trend to get high accuracy in the deep learning architecture is to increase in number of layers. However, this increases the complexity of the network, the number of hyperparameters, and the network latency. To counter this, the MobileNet is designed with depth-wise separable convolution layers which use factorization to reduce the number of parameters drastically by $[1/N + 1/D_k^2]$ where N is the number of layers and D_k is the kernel size. MobileNet employs a 3x3 kernel size, $D_k = 3$ and it has 28 layers ($N = 28$). This produces a small model with low latency (Howard et al., 2017) which makes the model perfect for embedded mobile vision application that satisfies the objective of this work.

The fine-tuned MobileNet model was deployed using a micro-web framework namely Flask. The screenshot of the web application interface is shown in Fig. 6. The image of the coconut tree captured in real-time through mobile phone is uploaded on the web in Fig. 6a and the classification of the upoaded image is displayed on the web page as a result. The sample webpage of disease deteced and a healthy plant are shown in Fig. 6b and c respectively. For the images detected with infections, the remedies are also displayed on the same web page. The best Keras pretrained model and hand-designed CNN model are deployed and tested.

5. Conclusions

In this paper, an end-to-end framework for the detection of disease and pest infection in coconut tree using the state-of-the-art deep learning technology was proposed, developed, trained, tested, and implemented. For this objective, a dataset of 1564 images was collected. This dataset consists of images of healthy trees, trees infected with Stem Bleeding disease or Leaf Blight disease, and pest infection by RPW. Popular image segmentation algorithms are applied to the dataset to increase the classification accuracy of the hand-designed CNN model. Further to improve the accuracy with our limited dataset, inductive transfer learning on the pre-trained Keras models [VGGNet, InceptionV3, DenseNet, InceptionResNetV2, MobileNet, and NASNetMobile] were applied. Among these, MobileNet has the highest validation accuracy of 82.10%. The CNN model, on the other hand, achieves validation accuracy of 85.63% with 2 convolution layers and 96.94% with 4 convolution layers, respectively. Therefore, the optimized CNN model and Keras pretrained best model MobileNet were deployed for disease and pest infection classification of the coconut tree. In the future, segmentation using deep learning technology will be explored; a large dataset for a variety of diseases/pest infections in the coconut tree will be collected. Furthermore, the severity level of the infections will also be included.

CRediT authorship contribution statement

Piyush Singh: Data curation, Methodology, Software, Validation, Visualization, Writing - original draft. **Abhishek Verma:** Data curation, Methodology, Software, Validation, Visualization, Writing - original draft. **John Sahaya Rani Alex:** Conceptualization, Methodology, Supervision, Writing - original draft, Writing - review & editing.

Declaration of Competing Interest

None.

Acknowledgments

The authors would like to thank Dr. Gayathri Subbiah, Plant Pathologist, ICAR-KVK, Kattupakkam, TANUVAS, Chennai, for the assistance provided during field data collection.

Appendix A. Supplementary material

Supplementary data to this article can be found online at https://doi.org/10.1016/j.compag.2021.105986.

References:

Arulandoo, X., Sritharan, K., Subramaniam, M., 2016. The Coconut Palm, Second Edi. ed, Encyclopedia of Applied Plant Sciences. Elsevier. https://doi.org/10.1016/B978-0-12-394807-6.00237-9.

Chandy, A., 2019. Pest Infestation Identification in Coconut Trees Using Deep Learning. J. Artif. Intell. Capsul. Networks 01, 10–18. https://doi.org/10.36548/jaicn.2019.1.002.

Chollet, F., 2017. Xception: Deep learning with depthwise separable convolutions. In: Proc. - 30th IEEE Conf. Comput. Vis. Pattern Recognition, CVPR 2017 2017-Janua, 1800–1807. https://doi.org/10.1109/CVPR.2017.195.

Chollet, F., n.d. Keras [WWW Document]. URL https://keras.io.

Cortes, C., Research, G., York, N., 2004. L 2 Regularization for Learning Kernels. In: Proc. Twenty-Fifth Conf. Uncertain. Artif. Intell.

Dath, A., Balakrishnan, M., 2016. Expert System on Coconut Disease Management and Variety Selection 5, 242–246. https://doi.org/10.17148/IJARCCE.2016.5462.

Dauphin, Y.N., De Vries, H., Bengio, Y., 2015. Equilibrated adaptive learning rates for non-convex optimization. Adv. Neural Inf. Process. Syst. 2015-Janua, 1504–1512.

Dhanachandra, N., Manglem, K., Chanu, Y.J., 2015. Image Segmentation Using K-means Clustering Algorithm and Subtractive Clustering Algorithm. Proc. Comput. Sci. 54, 764–771. https://doi.org/10.1016/j.procs.2015.06.090.

Duan, K., Keerthi, S.S., Chu, W., Shevade, S.K., Poo, A.N., 2003. Multi-category classification by soft-max combination of binary classifiers. Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics) 2709, 125–134. https://doi.org/10.1007/3-540-44938-8_13.

Giblin-Davis, R.M., Faleiro, J.R., Jacas, J.A., Peña, J.E., Vidyasagar, P.S.P.V., 2013. Biology and management of the red palm weevil, rhynchophorus ferrugineus. Potential Invasive Pests Agric. Crop. 1–34 https://doi.org/10.1079/9781845938291.0001.

Harith-Fadzilah, N., Haris-Hussain, M., Ghani, I.A., Zakaria, A., Amit, S., Zainal, Z., Azmi, W.A., Jalinas, J., Hassan, M., 2020. Physical and physiological monitoring on red palm weevil-infested oil palms. Insects 11, 1–12. https://doi.org/10.3390/insects11070407.

Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., Adam, H., 2017. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications.

Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q., 2017. Densely connected convolutional networks. In: Proc. - 30th IEEE Conf. Comput. Vis. Pattern Recognition, CVPR 2017 2017-Janua, 2261–2269. https://doi.org/10.1109/CVPR.2017.243.

Kieffer, B., Babaie, M., Kalra, S., Tizhoosh, H.R., 2018. Convolutional neural networks for histopathology image classification: Training vs. Using pre-trained networks. In: Proc. 7th Int. Conf. Image Process. Theory, Tools Appl. IPTA 2017 2018-Janua, 1–6. https://doi.org/10.1109/IPTA.2017.8310149.

Lin, M., Chen, Q., Yan, S., 2014. Network in network. In: 2nd Int. Conf. Learn. Represent. ICLR 2014 - Conf. Track Proc. 1–10.

Michel, D., Franqueville, H.D., Ducamp, M., 2012. Bud rot and other major diseases of coconut, a potential threat to oil palm. Exist. Emerg. Pests Dis. Oil Palm - Adv. Res. Manag. 13–14.

Nampoothiri, K.U.K., Krishnakumar, V., Thampan, P.K., Achuthan Nair, M., 2019. The coconut palm (Cocos nucifera L.) - Research and development perspectives, The Coconut Palm (Cocos nucifera L.) - Research and Development Perspectives. https://doi.org/10.1007/978-981-13-2754-4.

Pan, S.J., Yang, Q., 2010. A survey on transfer learning. IEEE Trans. Knowl. Data Eng. 22, 1345–1359. https://doi.org/10.1109/TKDE.2009.191.

Olga Russakovsky*, Jia Deng*, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg and Li Fei-Fei. (* = equal contribution) ImageNet Large Scale Visual Recognition Challenge. IJCV, 2015.

Warwick, R.N.D., Passos, E.M.E., 2009. Outbreak of stem bleeding in coconuts caused by Thielaviopsis paradoxa in Sergipe. Brazil. Trop. Plant Pathol. 34, 175–177. https://doi.org/10.1590/s1982-56762009000300007.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A.C., Fei-Fei, L., 2015. ImageNet Large Scale Visual Recognition Challenge. Int. J. Comput. Vis. 115, 211–252. https://doi.org/10.1007/s11263-015-0816-y.

Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L.C., 2018. MobileNetV2: Inverted Residuals and Linear Bottlenecks. Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. 4510–4520 https://doi.org/10.1109/CVPR.2018.00474.

Sha, C., Hou, J., Cui, H., 2016. A robust 2D Otsu's thresholding method in image segmentation. J. Vis. Commun. Image Represent. 41, 339–351. https://doi.org/10.1016/j.jvcir.2016.10.013.

Shafarenko, L., Petrou, M., Kittler, J., 1997. Automatic watershed segmentation of randomly textured color images. IEEE Trans. Image Process. 6, 1530–1544. https://doi.org/10.1109/83.641413.

Simonyan, K., Zisserman, A., 2015. Very deep convolutional networks for large-scale image recognition. In: 3rd Int. Conf. Learn. Represent. ICLR 2015 - Conf. Track Proc. 1–14.

Snehalatharani, A., Maheswarappa, H.P., Devappa, V., Malhotra, S.K., 2016. Status of coconut basal stem rot disease in India - A review. Indian J. Agric. Sci. 86, 1519–1529.

Sreejith, C.C., Muraleedharan, C., Arun, P., 2013. Life cycle assessment of producer gas derived from coconut shell and its comparison with coal gas: An Indian perspective. Int. J. Energy Environ. Eng. 4, 1–22. https://doi.org/10.1186/2251-6832-4-8.

Sun, J., Yang, Y., He, X., Wu, X., 2020. Northern Maize Leaf Blight Detection under Complex Field Environment Based on Deep Learning. IEEE Access 8, 33679–33688. https://doi.org/10.1109/ACCESS.2020.2973658.

Szegedy, C., Ioffe, S., Vanhoucke, V., Alemi, A.A., 2017. Inception-v4, inception-ResNet and the impact of residual connections on learning. 31st AAAI Conf. Artif. Intell. AAAI 2017, 4278–4284.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A., 2015. Going deeper with convolutions. Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. 1–9. https://doi.org/10.1109/CVPR.2015.7298594.

Yuheng, S., Hao, Y., 2017. Image Segmentation Algorithms Overview 1.

Zhang, Y., Song, C., Zhang, D., 2020. Deep Learning-Based Object Detection Improvement for Tomato Disease. IEEE Access 8, 56607–56614. https://doi.org/10.1109/ACCESS.2020.2982456.

Zoph, B., Vasudevan, V., Shlens, J., Le, Q.V., 2018. Learning Transferable Architectures for Scalable Image Recognition. Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. 8697–8710 https://doi.org/10.1109/CVPR.2018.00907.