Identification and categorization of diseases in arecanut: a machine learning approach

Ajit Hegde, Vijaya Shetty Sadanand, Chinmay Ganapati Hegde, Krishnamurthy Manjunath Naik, Kanaad Deepak Shastri
Department of Computer Science and Engineering, Nitte Meenakshi Institute of Technology, Bengaluru, India

ABSTRACT
Areca nut is one of the prominent commercial crops that are grown worldwide for traditional medicines, furniture, cosmetics, food, veterinary preparations, and textile industries. It experiences a variety of diseases during its existence, from the bottom to the tip. The conventional method for detection of diseases is through visual inspection and it is also necessary to have properly designed laboratories to check these harvests. It is a time consuming and tedious task to inspect these crops across wide acres of plantations. The proposed system has been developed that uses convolutional neural network (CNN) to identify and categorize diseases in arecanuts, trunks and leaves also suggesting effective preventative measures. Proprietary dataset consists of 1,100 photos of healthy and diseased arecas. The ratio between the train and test data is 80:20. Binary cross entropy is employed as the loss function for model construction, with accuracy serving as the metrics and Adam serving as the optimizing function. In identification and categorization of arecanut diseases, the suggested approach was shown to be efficient with 93.05% accuracy.

Keywords: Adam optimizer, Areca nut disease, Binary cross entropy, Convolutional neural network, Machine learning

This is an open access article under the CC BY-SA license.

Corresponding Author:
Ajit Hegde
Department of Computer Science and Engineering, Nitte Meenakshi Institute of Technology, Bengaluru, India
Email: ajithegde3520@gmail.com

1. INTRODUCTION
A widely known commercial palm arecanut grows in large portions of the Tropical Pacific, Asia and East Africa. Large, evergreen leaves on plants are spirally arranged at the apex of the stem and are either palmately or pinnately compound. The various factors including climatic conditions, soil conditions, diseases and more, have an impact on growth of the crops. Fruit rot, stem bleeding, yellow leaf spot and nut split disease are the common diseases that affect arecanut trees. The identification and categorization of diseases in arecanut plants is a challenging task as the symptoms can vary greatly depending on the type of disease and the stage of infection. Moreover, the lack of expert knowledge in this area makes it difficult for farmers to accurately identify and treat diseases. Timely and accurate diagnosis is crucial for effective disease management, but this is often hindered by the lack of appropriate technology and resources.

Puneeth and Nethravathi [1] focused on diseases in arecanut through the classification of healthy and unhealthy arecanuts using image processing techniques. Dhanuja and Kumar [2] proposed on building a fully automated image classification system for the disease detection on multiple arecanut by using algorithms at all detection stages. Siddesha and Niranjan [3] presented detection of affected region from an infected arecanut image using K-means clustering and Otsu method. Anilkumar et al. [4] defined on detection of arecanut diseases such as fruit rot, yellow leaf spot and stem bleeding. Akshay and Hegde [5] divided the dataset into healthy and unhealthy classes. The decision tree method has been used to train the model. Guo et al. [6] took
all of inputs and process factors that is vegetable index, VI value, p-Value and mainly uses satellite images for recognition. A segmentation technique utilising the hue saturation value (HSV) colour model was presented by Dhanesha and Naika [7]. Irawan et al. [8] focused on expert system architecture and chaining method for detection which involves user interface (UI) features.

Kanan and Kumar [9] proposed the method where comparison between the data values are done by using a regression algorithm to find similarities in the pattern of conditions using IoT sensors. Lei et al. [10] focused on using many machine learning algorithms like back propagation in neural network (BPNN), decision tree (DT), naive Bayes and k-nearest neighbors algorithm (KNN). A disease detection and classification system based on Android was proposed by Tlhobogang and Wannous [11]. The arecanut diseases can be prevented by using machine learning models, various techniques have been employed such as deep learning [12], convolutional neural network (CNN) [13], [14], image processing [15], [16], K-means [17], [18], support vector machine (SVM) [19], learning and machine perception (LAMP) [20] and real time identification of diseases [21]. Also, a simple practical architecture with three stages illumination normalization, feces detection and trait identification for CNN classification is proposed [22]. Automatic segmentation and classification of the region of interest are performed using the StoolNet [23] shallow convolutional neural network can be the feature.

Identification and categorization refer to different but related stages of the detection process. Identification entails determining the precise class label of the arecanut plant based on the likelihood score that the machine learning (ML) model gives to each class label. Whereas, categorization entails classifying the many arecanut plant class labels according to the traits they have in common. The categorization stage is classifying the class labels according to the type of plant disease or condition. The objectives of the proposed system include:
- Create the dataset that contains healthy and unhealthy images of arecanut, trunk and leaves.
- Implementing an algorithm for detection and classification of diseases in arecanut.
- Implementing an algorithm that would suggest remedies for the detected diseases.

2. METHOD

The methodology as shown in the Figure 1 for the proposed model is:
- Dataset creation and pre-processing.
- Training a neural network for the identification of arecanut diseases.

![Figure 1. Working design model](image)

2.1. Dataset acquisition

The proprietary dataset consists the collection of healthy and unhealthy images of arecanut, trunk and leaves. Some of the images from the dataset are shown in Figure 2, where Figure 2(a) is yellow leaf, Figure 2(b) is healthy leaf, Figure 2(c) is nut split, Figure 2(d) is stem bleeding, Figure 2(e) is fruit rot and
Figure 2(f) is healthy arecanut. Digital camera images were captured at half a meter from the source. The images of healthy and diseased arecanuts are gathered from the coastal and malnad regions of Karnataka. These images are shot with assistance of knowledgeable arecanut growers as well as researchers. The dataset collection has more than 1,100 photos in total which are stored in a defined hierarchy for further training.

Figure 2. Sample dataset (a) yellow leaf, (b) healthy leaf, (c) nut split, (d) stem bleeding, (e) fruit rot, and (f) healthy arecanut

2.2. Pre-processing

Before training CNN model, images are reduced to 256*256 resolution by performing resizing and reshaping of the images. Resizing involves scaling the images to a standardized size, while reshaping involves converting the input images into a standardized shape. Each pixel’s RGB value, which ranges from 0 to 255, is contained in an array. In this study, the pixel values of the arecanut leaf images were normalized to a range of [0, 1] by using the factor of 255. Data augmentation techniques were applied to the training dataset to increase its size and improve the performance. The data is randomly divided into two sets using the train-test split technique. According to the 80/20 split ratio, 80% of the data is utilized for training and 20% is used for testing.

2.3. Training model using CNN

A CNN is a type of deep neural network that is designed to process and analyze the data that has a grid-like topology, such as an image [24], [25]. As shown in the Figure 3, the network consists of a series of convolutional layers that apply filters to the input data followed by pooling layers that down sample the output which completes feature extraction. The final layers of the network typically include one or more fully connected layers that perform classification.

Feature extraction consists of, Input layer: This layer takes in the raw image data as input and prepares it for processing by the rest of the network, that is the input layer would take in an image of arecanut. The input image can be represented as a 3D tensor of shape (height, width, channels). Convolutional 2D layer: This layer
applies a set of 32/64/128 filters of size 3*3 to the dataset, which helps to extract features from the image. The resulting feature maps highlight important patterns and edges in the image that can be used for classification. The output of a convolutional layer can be computed using (1).

\[ h[i, j, k] = f \left( \sum \sum \text{kernel}[l, m, n] \ast x[i+l, j+m, n] + \text{bias}[k] \right) \]  

Where, \( h[i,j,k] \) is the output activation at position \((i,j)\) for the \(k\)-th filter, \( f \) is a non-linear activation function, \( \text{kernel}[l, m, n] \) is the weight for the filter at position \((l,m)\) and channel \( n \), \( x[i+l, j+m, n] \) is the pixel value at position \((i+l,j+m)\) and channel \( n \) in the input image, \( \text{bias}[k] \) bias term. Activation function rectified linear unit (ReLU): The output of the convolutional layer is subjected to an activation function known as the ReLU layer. Batch Normalization layer: It is a technique used to normalize the input to each layer of the network. Each mini batch of data is individually processed by calculating the mean and dividing by the input's standard deviation. Pooling 2D layer (Max Pooling): This layer reduces the dimensionality of the feature maps with 2*2 or 3*3 by down sampling using maximum. The output of a max-pooling layer can be computed using (2).

\[ h[i, j, k] = \max(x[i+\text{pool}_\text{size}, j+\text{pool}_\text{size}, k]) \]  

Where, \( h[i,j,k] \) is output activation at position \((i,j)\), \( x[i+\text{pool}_\text{size}, j+\text{pool}_\text{size}, k] \) is the receptive field of the pooling operation with the position \((i,j)\), \( \text{pool}_\text{size} \) size of pooling window. Dropout layer: This layer randomly drops out a certain percentage of the neurons in the network during training which helps to prevent overfitting and improve the network’s generalization ability. Classification consists of, fully connected layer: This layer takes the flattened feature maps from the previous layers and applies a set of weights and biases to produce an output which has been passed through one or more fully connected layers that is hidden layers such as dense layer for classification. Output of fully connected layer is obtained using (3).

\[ y[j] = f \left( \sum w[i, j] \ast x[i] + b[j] \right) \]  

Where, \( y[j] \) is the output activation for jth neuron in the fully connected layer, \( f \) is a non-linear activation function, \( w[i,j] \) is the weight for the connection between the \(i\)th input neuron and the \(j\)th output neuron, \( x[i] \) is the input activation for the \(i\)th input neuron, \( b[j] \) is the bias term for the \(j\)th output neuron. Output layer: This layer produces the final output of the network, which is a probability distribution over the different disease classes. Table 1 shows that the proposed CNN model consists of several layers beginning with a 2D convolutional layer with 32 filters of size 3*3, stride 1 with ‘same’ padding working on preprocessed image. Then comes a max pooling layer with a pool size of 3*3 and stride of 1 which is followed by an activation layer that employs ReLU, a batch normalization layer and a layer of max pooling. Having a dropout rate of 0.25 the following layer is a dropout layer.

<table>
<thead>
<tr>
<th>Layers</th>
<th>Output shape</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conv2d</td>
<td>(256, 256, 32)</td>
<td>896</td>
</tr>
<tr>
<td>Activation function (ReLU)</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Batch normalization</td>
<td>(256, 256, 32)</td>
<td>128</td>
</tr>
<tr>
<td>Pooling 2D (MaxPooling)</td>
<td>(85, 85, 32)</td>
<td>-</td>
</tr>
<tr>
<td>Dropout</td>
<td>(85, 85, 32)</td>
<td>-</td>
</tr>
<tr>
<td>Flatten</td>
<td>(56448)</td>
<td>-</td>
</tr>
<tr>
<td>Dense</td>
<td>(1024)</td>
<td>57803776</td>
</tr>
</tbody>
</table>

Alternating 2D convolutional layers, activation layers, batch normalization layers, max pooling layers and dropout layers are used in the succeeding layers. The convolutional layers have more filters than before 32, 64, and 128. The stride, padding and kernel sizes are kept 1 and ‘same,’ respectively. The last layers consist of a flatten layer to reshape the output from the preceding layer, a fully linked dense layer with 1,024 units, another activation layer, a batch normalization layer and a dropout layer. Seven units with a softmax activation function make up the last dense layer. The model comprises 58,094,471 parameters of which 58,091,591 are trainable. In a multiclass CNN before compiling the model, the binary cross entropy loss function is extended. To minimize the loss Adam has been used for optimizing bias and weights of the network during the training process. The dataset with a specified batch size is used to train the model for 25 epochs of 28 steps per epoch.
2.4. CNN based classification

The pre-processed image is passed through the trained model and the output is the probability distribution over the 7 classes. The SoftMax function is applied to the output to convert the probabilities into a probability distribution that sums to one. The predicted class is the class with the highest probability in the output of the model. This is the class that the model predicts the input image that it belongs to. The prediction and remedies are displayed along with the input image that can be used to make decisions.

3. RESULTS AND DISCUSSION

The proposed method was evaluated on the proprietary dataset of arecanut with 7 classes namely healthy leaves, healthy nuts, healthy trunks, fruit rot, stem bleeding, yellow leaf spot and also nut split disease [26], [27]. The CNN model was trained by the training set and the accuracy of the network was evaluated on the test set. As indicated in Figure 4 the graph plots the accuracy of the model on the y-axis and the number of epochs on the x-axis. The training accuracy vs epoch graph shows the model’s performance on the training dataset over time. As the number of epochs increases, accuracy of the training should improve because the network is learning more about the dataset and becoming better at classifying it.

On the other hand, the validation accuracy vs epoch graph shows the performance of the network on a validation data over time, this aids in avoiding overfitting. Ideally, both the training accuracy and validation accuracy should increase over the time and plateau as the model converges to an optimal solution as in the graph. If the training accuracy continues to increase while the validation accuracy plateaus or even decreases, it may indicate overfitting. In contrast, if both the training and validation accuracies are low and do not improve over time, it may indicate underfitting, which occurs when the model is too simple to capture the complexity.

As indicated in Figure 5 the training loss vs epoch graph shows the loss function’s value during each epoch of training on the training data. A decrease in the loss function over time indicates that the model is learning and improving. The validation loss vs epoch graph shows the loss function’s value during each epoch of training on the validation data. A decrease in the validation loss function over time indicates that the model is not overfitting. Ideally, there will be a decreasing trend in both training and validation loss over the epochs. However, if the training loss continues to decrease but the validation loss starts increasing, it indicates overfitting. On the other hand, if both the training and validation loss functions are high, it indicates underfitting. Results show that the model converges over the epochs, the training and validation loss functions decrease steadily over time indicating that the model is learning and generalizing well to new data. As indicated in Figure 4, the test accuracy after model training was 93%. The trained model recognizes the condition of arecanut based on probability score of the classes and remedies for the predicted diseases are displayed for the end users.

<table>
<thead>
<tr>
<th>Figure 4. Accuracy v/s epoch</th>
<th>Figure 5. Loss v/s epoch</th>
</tr>
</thead>
</table>

3.1. Comparative analysis

Table 2 exhibits the performance evaluation of different methods used for classification of arecanut diseases. For comparison we have chosen classification algorithms like nearest neighbors, convolutional neural network, decision tree, support vector machine. The Table 2 makes it abundantly evident that the proposed CNN model has produced a better result and performance. This precision can be linked to CNN’s capacity to extract intricate features from images which makes it more effective at detecting diseases than other models that depend on less complicated feature extraction techniques. Convolutional layers of CNN also allow for the retention of spatial information improving the ability to recognize patterns and structures in the images.
4. CONCLUSION

The novel work demonstrated the potential of using CNN for the identification and categorization of diseases in arecanut. Through the development and testing of the model it was possible to achieve a high accuracy rate of 93.05% in classifying the different diseases affecting arecanut which includes fruit rot, stem bleeding, yellow leaf spot and newly spreading nut split disease. The results of the study can have significant implications for farmers and agricultural researchers who are looking for more efficient and accurate ways to diagnose and manage diseases in arecanut. However, future studies could benefit from the use of larger distributed datasets to increase the reliability of the model. Additionally, there may be other environmental and climatic factors that could affect the accuracy of the model in real-world. With further research and development, this technology could become a valuable tool for the management of crop diseases. Consequently, this approach helps in encouraging farmers to engage in intelligent farming and giving them the tools they need to make better decisions regarding yield.

ACKNOWLEDGEMENTS

The authors would like to thank Karnataka State Council of Science and Technology (KSCST) for funding this research.

REFERENCES


**BIOGRAPHIES OF AUTHORS**

**Ajit Hegde** is pursuing Bachelor of Engineering in Computer Science at Nitte Meenakshi Institute of Technology, Bengaluru. He has completed the projects in Cloud Computing and ML. His research interests include machine learning, cloud computing, and big data. Currently he is working as an associate consultant intern at Oracle. He can be contacted at email: ajithegde3520@gmail.com.

**Dr. Vijaya Shetty Sadanand** is currently working as Professor and Head in the Department of Computer Science and Engineering at Nitte Meenakshi Institute of Technology, Bengaluru. She is currently executing a project in the domain of Deep Learning funded by the Vision Group on Science and Technology (VGST). Her research interests include data mining, machine learning, deep learning and distributed computing. She is a Life member of Indian Society for Technical Education (ISTE), member of IEEE and Computer Society of India (CSI). She can be contacted at email: vijayashetty.s@nmit.ac.in.
Chinmay Ganapati Hegde is pursuing Bachelor of Engineering in Computer Science at Nitte Meenakshi Institute of Technology, Bengaluru. He has completed the projects in Web Development, UI and ML. His research interests include AI, machine learning and data mining. Currently he is working as a quality analyst intern at Quinbay Technologies. He can be contacted at email: chinmayhegdehere@gmail.com.

Krishnamurthy Manjunath Naik is pursuing Bachelor of Engineering in Computer Science at Nitte Meenakshi Institute of Technology, Bengaluru. He has completed the projects in Android Development, AI, and ML. His research interests include AI, machine learning and big data. Currently he is working as a graduate trainee engineer intern at Siemens. He can be contacted at email: krishnamurthynaik876@gmail.com.

Kanaad Deepak Shastri is pursuing Bachelor of Engineering in Computer Science at Nitte Meenakshi Institute of Technology, Bengaluru. He has completed the projects in web development and data science. His research interests include machine learning, IoT, and 5G. Currently he is working as a quality analyst intern at Zscaler. He can be contacted at email: kanaadds@gmail.com.