ISSN: 2096-3246
Volume 58, Issue 02, 2026

AUXILIARY GENERATIVE ADVERSARIAL NETWORK (AGAN) BASED IMAGE GENERATION AND MULTI-SCALE ENCODER DECODER SELF-ATTENTION NETWORK (MSEDSAN) FOR DISEASE CLASSIFICATION IN COCONUT TREE

C. Sivaraj[1] and Dr. A. Selvakumar[2]
[1]Research Scholar & Assistant Professor, Department of Computer Science,
Sree Saraswathi Thyagaraja College, Pollachi-642107.
Email: sivazck@gmail.com
[2] Associate Professor, Department of Computer Science,
Sree Saraswathi Thyagaraja College, Pollachi-642107.
Email: arshariniselva@gmail.com

ABSTRACT:

Coconut trees are vital fasten crops that are widely grown in coastal regions, contributing significantly to the national economy. However, diverse types of diseases affect coconut tree health, including leaf spot diseases, which affect crop health and productivity. Deep learning algorithms that extract discriminative features from the images implicitly gained popularity in computer vision. In this paper, Auxiliary Generative Adversarial Network (AGAN) based image augmentation is employed to generate synthetic images, improving dataset diversity and reducing class imbalance. The augmented images are then processed using Transfer Learning-based Enhanced Visual Geometry Group 16 (EVGG16) to extract local features and a Swin Transformer to capture global contextual information. The extracted features are subsequently processed using the proposed Multi-Scale Encoder Decoder Self-Attention Network (MSEDSAN) to learn comprehensive and discriminative disease representations at multiple scales. MSEDSAN, Cross-Scale Attention (CSA) establishes interactions among shallow and deep feature representations to effectively integrate fine local disease features with high-level semantic information. Furthermore, a Spatial-Channel Attention Fusion (SCAF) adaptively emphasizes disease-relevant regions and informative feature channels while suppressing complex backgrounds and redundant information. The resulting features are further processed through a Multi-Scale Feature Enhancement (MSFE) to enhance disease-specific representations across multiple scales. Finally, Global Average Pooling (GAP) transforms the refined feature maps into a compact feature vector, which is passed through a fully connected layer with Softmax activation to classify the coconut leaf images into healthy and different disease categories. Proposed approach achieves improved disease classification performance and provides more reliable and explainable predictions for practical agricultural applications.

Keywords: Deep learning, Coconut trees, Auxiliary Generative Adversarial Network (AGAN), Image generation, Swin Transformer, Multi-Scale Encoder Decoder Self-Attention Network (MSEDSAN), and classification.

1. INTRODUCTION

Over the last several years, the substantial drop in agricultural output has threatened international food supplies. The coconut tree is an important plantation because of its many useful products [1-2]. In many regions globally, the excellent nutritional value and diverse array of coconut products have made them important to daily life and several enterprises [2]. Despite rising consumer

demand, many coconut farmers struggle to make ends meet due to the high labour cost and the time-consuming process of making coconut goods [3]. Coconuts' fibre, vitamins, and minerals make them a healthy snack. The coconut is the most valuable source of nutrition since it can be used to make food, fuel, and even medication [4]. The coconut fruit grows on the coconut palm and has been used for centuries. Humans have relied on it for thousands of years to meet their basic needs [5]. In modern times, people of many different faiths and cultures have found coconut a useful food and medicinal resource. Some diseases that attack coconut leaves are very harmful, weakening the leaves over time and resulting in significant crop losses [6].

Analyzing healthy and unhealthy leaves is an important process for successful agriculture development. Identification of the infected plants is an important process to protect the uninfected leaves from infected leaves. The leaves of plants are the main source for detecting leaf infection because most of the signs of the diseases may visible on the leaves. Leaf disease detection is the most recommended process to detect plant infection by recognizing different symptoms of different infections. Conventional plant disease diagnostic techniques originate from standard laboratory technology and hand examination. Conventional methods may encounter problems in extracting meaningful features or accurate segmentation in the presence of complex backgrounds, lighting conditions, or shadows in the images. Primarily in the domains of artificial intelligence (AI) and machine learning (ML), have the potential to fundamentally alter the detection of plant diseases. Traditional ML techniques, such as feature extraction and classification, have been widely used in the field of plant disease detection. These methods extract features from images, such as color, texture, and shape, to train a classifier that can differentiate between healthy and diseased plants.

These methods have been widely used for the detection of diseases and nutrient deficiency but have limitations in accurately identifying subtle symptoms of diseases and early-stage disease detection [7-8]. Recently, DL techniques such as convolutional neural network (CNN) and deep belief network (DBN) have been proposed for plant disease detection. These methods involve training a network to learn the underlying features of the images, enabling the identification of subtle symptoms of diseases that traditional image processing methods may be able to detect. However, these DL methods require a large amount of labeled training data and may not be suitable for unseen diseases. Furthermore, DL models are computationally expensive, which may be a limitation for some applications [9].

Despite their superior performance, DL models have several limitations [9-10]. It require large-scale annotated datasets for effective training, are computationally expensive, demand high memory and processing resources, and often exhibit reduced performance when applied to unseen diseases or datasets with different image characteristics. Furthermore, training deep learning models from scratch is time-consuming and may lead to overfitting when only limited labeled data are available. Transfer learning has emerged as an effective solution to address these challenges. Instead of training a deep neural network from scratch, transfer learning utilizes pre-trained models, such as Visual Geometry Group (VGG), ResNet, DenseNet, EfficientNet, and Vision Transformers have been already learned rich and generalized feature representations from large-scale image datasets [11-12]. By fine-tuning these models on plant leaf images, transfer learning significantly reduces the requirement for large labeled datasets, shortens training time, lowers computational cost, and

improves convergence. Moreover, transfer learning enhances feature extraction capability, increases classification accuracy, improves robustness and generalization to unseen data, and reduces the risk of overfitting [13]. These advantages make transfer learning particularly suitable for real-world agricultural applications where labeled disease images are limited and rapid, accurate disease diagnosis is essential [14].

The existing methods perform well for coconut disease classification; however, it still depends on supervised learning and limited labeled RGB image data. Conventional augmentation may not sufficiently represent disease variability, and CNN-based models may focus mainly on local texture patterns while missing global disease relationships. In addition, the base model provides limited interpretability regarding which infected regions influence the final prediction. Therefore, a more robust, explainable, and feature-rich classification framework is required for RGB coconut leaf disease images.

The proposed framework begins with coconut leaf images, which are preprocessed through resizing, normalization, denoising, and contrast enhancement. Auxiliary Generative Adversarial Network (AGAN) based data augmentation is employed to generate synthetic images, improving dataset diversity and reducing class imbalance. The augmented images are then processed using Transfer Learning-based Enhanced Visual Geometry Group 16 (EVGG16) to extract local features and a Swin Transformer to capture global contextual information. Swin Transformer is a vision transformer architecture that computes self-attention within local image windows and uses a shifted window mechanism to enable interaction between neighboring regions. An attention-based feature fusion module integrates these complementary features into a unified representation. Multi-Scale Encoder Decoder Self-Attention Network (MSEDSAN) based classification model, where the encoder learns compact and discriminative feature representations and the attached classification layers classify the coconut leaf images into healthy or disease categories using decoder, producing the final disease prediction. Proposed approach achieves improved disease classification performance and provides more reliable and explainable predictions for practical agricultural applications.

2. LITERATURE REVIEW

Parvathi and Selvi [15] proposed an improved faster region-based convolutional neural network (Faster R–CNN) model for the detection of two important maturity stages for coconuts in complex backgrounds. The detection of the maturation stages of coconuts for harvesting without human intervention involves challenges because of the complexity of the environment and the similarity between fruits and their backgrounds. Images of coconut and mature coconut bunches were collected from coconut farms. These images were augmented using rotation and colour transformation techniques. These augmented images were used along with original images during model training. The Faster R–CNN algorithm with the ResNet-50 network was used to enhance the detection score of nuts with two major maturity stages. Following training, the detection performance was tested with a dataset that included real-time images as well as Google images. The test results showed that the detection performance achieved using the improved Faster R–CNN model was greater than that for other object detectors such as the single shot detector (SSD) you only look once (YOLO-V3) and Region-based Fully Convolutional Networks (R–FCN). The promising results

obtained from this study provided the motivation to develop an application tool for detecting coconut maturity from real-time images on farms.

Vigneshwaran and Tamburi [16] focused on the automatic identification and mapping of individual coconut palm trees from very-high-resolution satellite imagery using deep learning. The study uses WorldView-2 satellite data and develops an object-detection model in ArcGIS pro based on the Single Shot Detector (SSD) algorithm. SSD enables individual coconut palms to be detected directly from satellite images, reducing the need for labor-intensive manual surveys. The resulting detections can be converted into spatial maps and used to estimate the number and distribution of coconut trees within plantation areas. The authors highlight the usefulness of combining remote sensing, Global Information System (GIS), and deep learning for large-scale coconut plantation inventory, monitoring, and management. The proposed approach also provides a foundation for future applications involving individual-tree health monitoring and plantation assessment.

Ramesh et al., [17] explored a Transformer Convolutional Neural Network (TCNN) approach to automatically identify nutrient deficiencies in coconut trees. The model achieved impressive accuracy in classifying various deficiencies, including nitrogen, iron, potassium, and calcium. By analyzing leaf images, the system effectively detects these issues, demonstrating its ability to function well under different environmental conditions. A dataset of real-time images showcasing the four main deficiency categories was created for training and validating the model. Dataset consists of 8990 nutrition deficiency images namely calcium, iron, nitrogen and potassium. There are 2390 images under calcium, 2200 images under iron 2200 images under nitrogen and 2200 potassium, the effectiveness of the model using a variety of metrics, including accuracy, F1 score and the confusion matrix.

de Luna et al., [18] employed CNN particularly transfer learning models for categorizing coconuts into three groups: mature, pre-mature, and young. A collection of 3,000 coconut images was collected and augmented, with an equal number at each of the three stages of ripeness and then enhanced to increase variation. Before training, the images were preprocessed by resizing and normalizing before applying three pre-trained models (EfficientNet, MobileNetV2, and Visual Geometry Group 16 (VGG16)). The device was made by incorporating the high-performing VGG16 model into the Jetson Nano microprocessor. This offers a feasible and scalable answer for enhancing farming practices and production methods within the coconut industry. The assessment involved measurements such as accuracy, rate of errors, sensitivity, specificity, false positive rate, precision, and F1-score. This development could greatly enhance efficiency and standards in coconut farming, which would help farmers and the overall agricultural industry, ultimately promoting economic growth and sustainability.

Usman et al., [19] proposed a **VGG16 based approach for automatically classifying local coconut fruit varieties** from images. The primary objective was to improve classification accuracy while supporting the preservation of coconut quality and genetic diversity in Indonesia. The study applied **image augmentation**, particularly **color jitter**, to increase image diversity and improve the model's ability to generalize. Exploratory Data Analysis (EDA) was also performed to better understand the dataset. For classification, **VGG16 architecture** is used to extract discriminative visual features from coconut fruit images and distinguish among local coconut varieties. Model

performance was evaluated using a confusion matrix and metrics including accuracy, precision, recall, and F1-score.

Huang et al., [20] proposed an Enhanced Visual Geometry Group (EVGG16) model by transfer learning for detecting disease infections in coconut trees. The EVGG16 model achieves effective training with a limited quantity of data, utilizing the weight parameters of the convolution layer and pooling layer from the pre-training model to perform transfer VGG16 network model. VGG16 model is enhanced by adding Batch Normalization (BN) and Global Average Pooling (GAP) layers. In addition, the proposed enhanced VGG16 model utilizes transfer learning, adjusting hyperparameters and training batches to achieve more accurate recognition and obtain a more stable and accurate prediction model. For hyperparameters optimization, the Stochastic Gradient Descent (SGD) algorithm is used to process a training sample in each iteration and then update the parameters, so that the training speed is fast. Through hyperparameter tuning and optimized training batch configurations, achieved enhanced recognition accuracy, facilitating the development of more robust and stable predictive models.

Santhiya et al., [21] the leaflet diseases in coconut plants are those that impact individual leaflets, or leaf segments, on the fronds of coconut palms. Drying leaflets in coconut tree leaves can result from a variety of factors, including biotic (living organisms) and abiotic (environmental or non-living) impacts. In coconut trees, chlorosis, or the yellowing of the leaves, is a prevalent issue that can be caused by some factors. The CNN model is trained on a sizable set of high-resolution images of both healthy and diseased coconut leaves. Each disease's unique characteristics and patterns are automatically acquired with the aid of the deep learning architecture, enabling accurate classification. Using an additional set of annotated images of coconut tree leaves, the CNN model is assessed with remarkable accuracy in identifying illnesses. Comparing the system to traditional manual diagnosis procedures demonstrates the technology's efficiency and potential for early disease identification.

Shanmugam et al., [22] presented a novel framework for early detection of coconut leaf diseases by utilizing deep learning-based approach that incorporates image processing and classification techniques to develop an automated system for precision agriculture. The method starts by converting the Red Green Blue (RGB) images into Hue Saturation and Value (HSV) format so that the color difference improves and the complex backgrounds can be eliminated from the input image. Binary images are then created based on hue and saturation values, since it contains useful information to separate healthy from diseased areas. Capsule Networks are utilized for feature extraction since they can maintain spatial hierarchies and capture intricate patterns in image data. The features that are extracted later are classified with the Inception V3 model which is known to be quite a powerful model for image recognition due to its efficient architecture on Mendeley Data coconut leaf disease dataset. HSV-based preprocessing, then using CapsNet as a feature extractor, then adding Inception V3 as a classifier significantly improves detection. The study offers a fast and accurate method for early detection of diseases in coconut cultivation, aiding in early intervention and greater crop management in precision agriculture.

Cherian [23] presented a comprehensive approach to the classification of coconut leaf images through a robust image processing integrated with transformer technique. The coconut diseases and

pest infestations dataset is applied pre processing stage using a Guided Image Filter (GIF), which enhances image quality by preserving edges while suppressing noise. Following this, Active Contour Method (ACM) is employed for precise segmentation, enabling accurate delineation of leaf boundaries. To extract meaningful texture features, the Local Ternary Pattern (LTP) technique is applied, offering improved noise resistance and discriminative pattern compared to traditional texture descriptors. Finally, a lightweight yet powerful transformer based deep learning (DL) model, the Mobile Vision Transformer (MobileViT), is employed to classify healthy and infected coconut leaves. Using Python software, the proposed framework achieved higher accuracy, F1 score, precision and recall.

Vidhanaarachchi et al., [24] presented a study conducted in Sri Lanka, demonstrating the effectiveness of employing transfer learning-based CNN and Mask Region-based-CNN (Mask R-CNN) to identify Weligama Coconut Leaf Wilt Disease (WCWLD) and Coconut Caterpillar Infestation (CCI) damage at their early stages and to assess disease progression. The proposed approach employed transfer learning-based CNN for disease recognition and Mask R-CNN for identifying and segmenting affected regions associated with WCWLD and CCI. In addition, the study utilized You Only Look Once (YOLO) object-detection architectures, including YOLOv5, YOLOv8, and YOLO11, to automatically detect and count caterpillars distributed across coconut leaflets. The models were trained and validated using real-field image datasets collected from coconut-growing regions in Matara, Puttalam, and Makandura, Sri Lanka, thereby allowing the framework to be evaluated under realistic agricultural conditions.

Jaidka et al. [25] proposed a deep learning-based approach for the automatic detection and classification of coconut leaf diseases to support early disease diagnosis and precision agriculture. The authors developed a hybrid Neural Network Support Vector Coco Leaf Disease (NNSVCLD) model that integrates Convolutional Neural Networks (CNNs) for extracting discriminative features from coconut leaf images with a Support Vector Machine (SVM) for disease classification. The proposed framework was evaluated on coconut disease images covering five disease categories using performance measures such as accuracy, precision, recall, specificity, and F1-score. In the experimentation, 3-fold, 5-fold, and 10-fold cross-validation strategies were employed to assess the reliability of model. NNSVCLD model has demonstrating its effectiveness in distinguishing different coconut leaf diseases. The results indicated that combining CNN-based deep feature extraction and SVM classification provides a robust approach for automated coconut disease identification and facilitates timely disease management in coconut plantations.

3. PROPOSED METHODOLOGY

Auxiliary Generative Adversarial Network (AGAN) based image augmentation is employed to generate synthetic images, improving dataset diversity and reducing class imbalance. The augmented images are then processed using Transfer Learning-based Enhanced Visual Geometry Group 16 (EVGG16) to extract local features and a Swin Transformer to capture global contextual information. The extracted features are subsequently processed using the proposed Multi-Scale Encoder Decoder Self-Attention Network (MSEDSAN) to learn comprehensive and discriminative disease representations at multiple scales. MSEDSAN, Cross-Scale Attention (CSA) establishes interactions among shallow and deep feature representations to effectively integrate fine local disease

features with high-level semantic information. Furthermore, a Spatial-Channel Attention Fusion (SCAF) adaptively emphasizes disease-relevant regions and informative feature channels while suppressing complex backgrounds and redundant information. The resulting features are further processed through a Multi-Scale Feature Enhancement (MSFE) to enhance disease-specific representations across multiple scales. Finally, Global Average Pooling (GAP) transforms the refined feature maps into a compact feature vector, which is passed through a fully connected layer with Softmax activation to classify the coconut leaf images into healthy and different disease categories. The proposed framework provides an effective multi-scale feature extraction and adaptive disease-feature fusion.

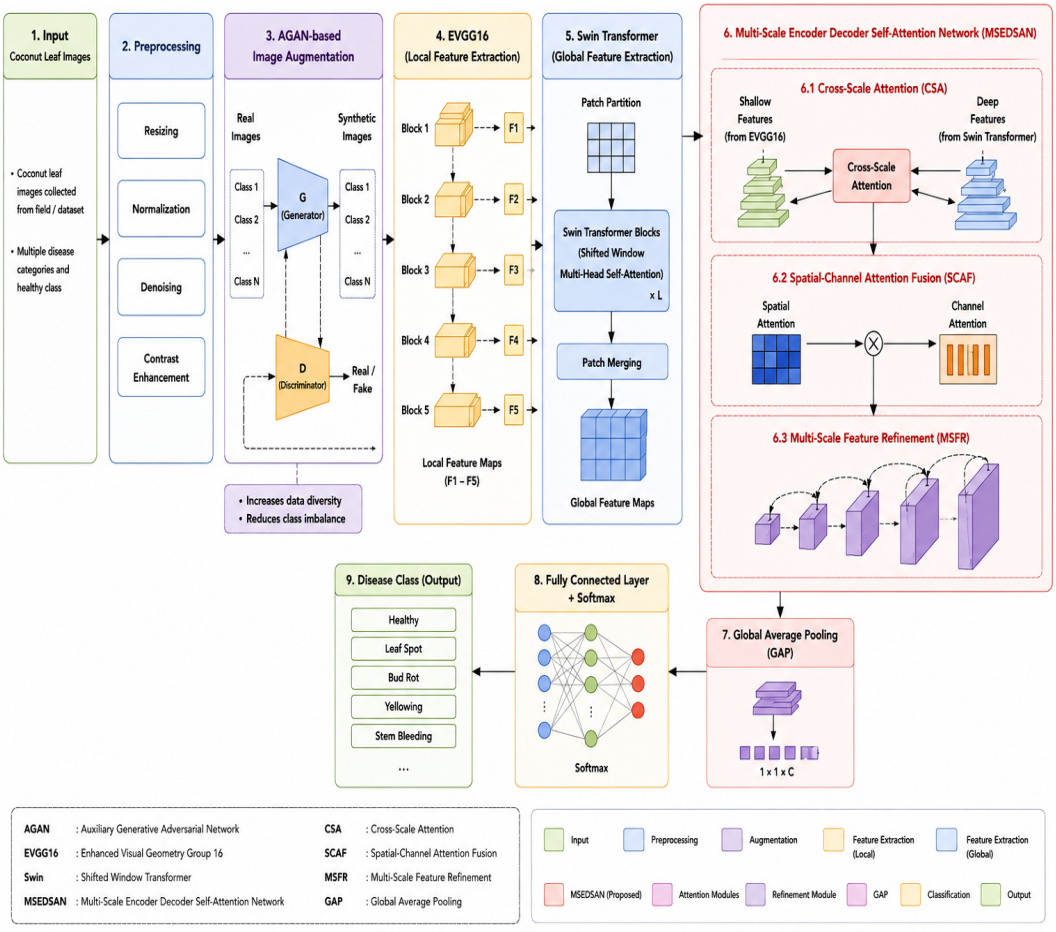


FIGURE 1. OVERALL PROCESS OF PROPOSED MODEL

3.1. Coconut (Cocos nucifera) tree disease dataset

The Coconut (Cocos nucifera) tree disease dataset comprises 5,798 images across five disease categories: Bud Root Dropping, Bud Rot, Gray Leaf Spot, Leaf Rot, and Stem Bleeding. The Coconut Tree Disease Dataset contains a collection of high-resolution images, each with dimensions of 768 pixels in width and 1024 pixels in height. The images have a resolution of 72 dots per inch (dpi), ensuring clarity and detail in the visual representation of the coconut tree diseases. Each image in the dataset is associated with one of the five disease classes, providing a class-imbalanced representation of coconut tree diseases. In addition to the existing dataset, coconut tree disease images were personally captured from coconut plantations in and around the Pollachi

region of Tamil Nadu, thereby incorporating locally collected field images into the dataset. The study of “Bud Root Dropping,” “Bud Rot,” “Gray Leaf Spot,” “Leaf Rot,” and “Stem Bleeding” in coconut trees holds paramount importance due to their collective impact on the overall health and productivity of coconut plantations. Each of these diseases presents unique challenges and potential threats to the coconut industry. “Bud Root Dropping” affects early growth stages, potentially leading to stunted development and reduced yield. “Bud Rot,” a fungal disease, can swiftly devastate entire groves if not promptly identified and managed. “Gray Leaf Spot” poses a significant threat, as it spreads rapidly and can lead to widespread defoliation. “Leaf Rot” compromises the photosynthetic capacity of the tree, directly impacting the tree's vitality. “Stem Bleeding” is a chronic condition that progressively weakens the tree’s structural integrity.Image preprocessing

Image preprocessing steps are described as follows,

Image Resizing: All images are resized to a uniform spatial dimension, such as 224 × 224 × 3 pixels, to ensure compatibility with EVGG16 and to provide standardized inputs to the proposed framework.

Noise Removal: A Median Filtering can be applied to suppress impulse noise and minor acquisition artifacts while preserving important disease boundaries, such as lesion edges, spots, and discoloration patterns. For an image I, the filtered pixel is determined by equation (1),

$$I_f(x,y) = median\{I(i,j): (i,j) \in N_{x,y}\} \quad (1)$$

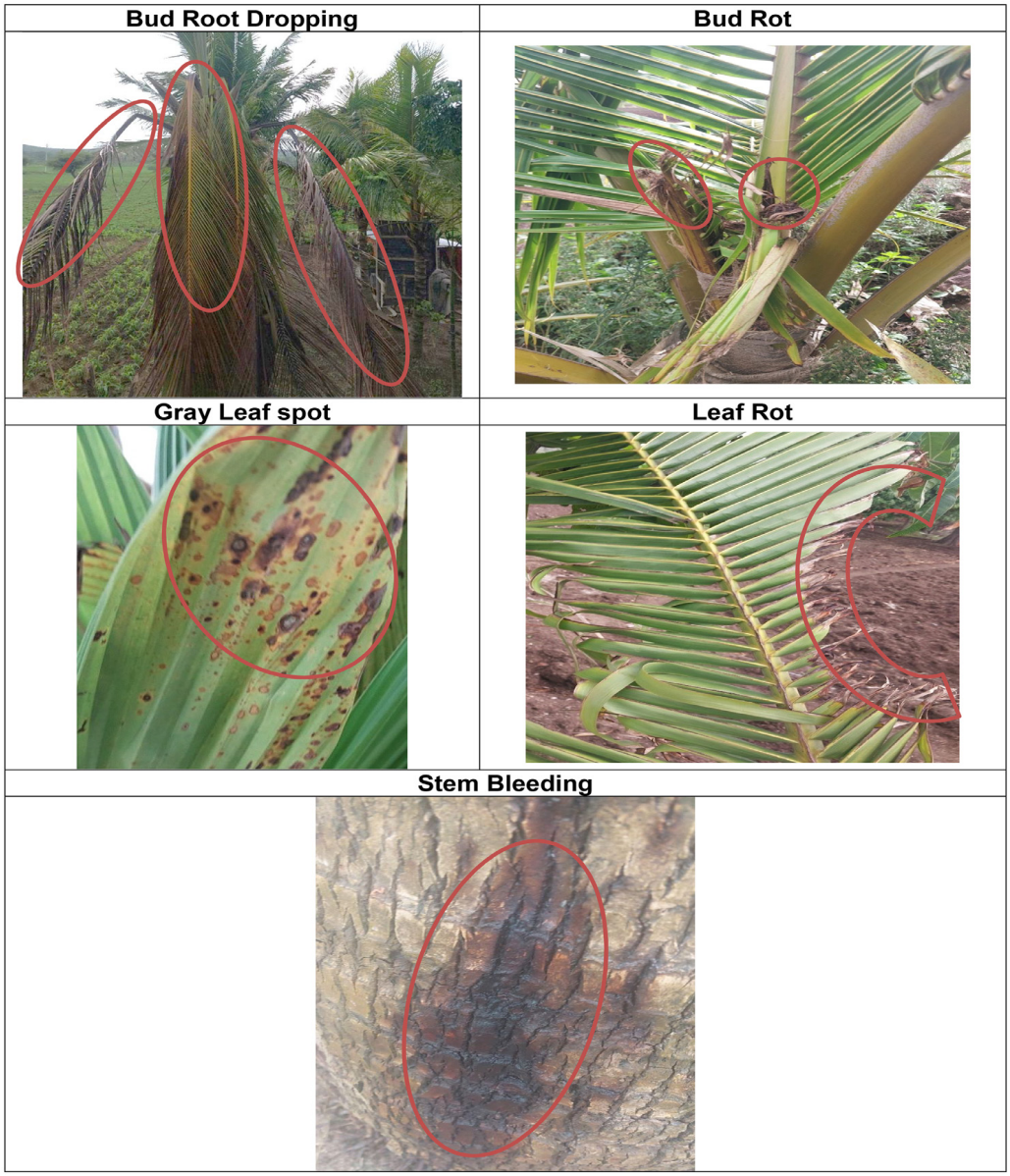


FIGURE 2. SAMPLE IMAGES OF COCONUT DISEASE DATASET

Contrast Enhancement: Contrast Limited Adaptive Histogram Equalization (CLAHE) can be employed to improve local contrast and make subtle disease symptoms more distinguishable. Unlike conventional histogram equalization, CLAHE operates on small image regions and limits excessive contrast amplification, making it suitable for coconut leaf images containing uneven illumination. CLAHE is employed to enhance the local contrast of coconut leaf images and improve the visibility of disease-related characteristics under varying illumination conditions. It divides the image into small local regions and performs histogram equalization independently within each region. A predefined clip limit restricts excessive histogram amplification, thereby preventing noise enhancement. The clipped intensity values are redistributed, and bilinear interpolation is applied between adjacent regions to obtain smooth transitions. Consequently, CLAHE highlights subtle disease symptoms such as lesions, discoloration, necrotic regions, and texture variations while preserving important image for subsequent feature extraction and classification. For each local tile (

T_k), the histogram is defined by equation (2) [26],

$$H_k(i) = \sum_{(x,y)\in T_k} \delta(I(x,y) - i) \tag{2}$$

where (I(x,y)) is denoted as the intensity of the pixel at location ((x,y)), i is denoted an intensity level, and ($I(.)$) is the indicator function. To prevent excessive amplification of noise, the local histogram is restricted by a clip limit (C),

$$H_k^c(i) = \begin{cases} H_k(i), & H_k(i) \le C \\ C, & H_k(i) > C \end{cases} \tag{3}$$

where $H_k^c(i)$ is the clipped histogram, C is the predefined **clip limit** that restricts excessive histogram amplification.. The total number of excess pixels produced by clipping is calculated as follows,

$$E_k = \sum_{i=0}^{L-1} [H_k(i) - H_k^c(i)] \tag{4}$$

where L is denoted as the total number of possible intensity levels. The clipped pixels are redistributed approximately uniformly among the histogram bins,

$$H_k^r(i) = H_k^c(i) + \frac{E_k}{L} \tag{5}$$

where $H_k^r(i)$ is denoted as the redistributed histogram. The cumulative distribution function (CDF) for each tile is then calculated as follows,

$$CDF_k(i) = \frac{\sum_{j=0}^{i} H_k^r(j)}{\sum_{j=0}^{L-1} H_k^r(j)} \tag{6}$$

where N_k is the total number of pixels in tile T_k. Based on the CDF, the enhanced intensity value is obtained as follows,

$$I_k^{CLAHE}(x,y) = (L-1)CDF_k(I_k(x,y)) \tag{7}$$

Finally, bilinear interpolation is performed between neighboring tiles to remove artificial boundaries and produce a smooth enhanced image, L is total number of possible intensity levels. Thus, CLAHE improves the visibility of subtle coconut leaf disease patterns while restricting excessive noise amplification, providing higher-quality images for subsequent AGAN-based augmentation and EVGG16–Swin Transformer feature extraction.

3.2. Auxiliary Generative Adversarial Network (AGAN) based image augmentation

GAN is composed of two neural networks: the generator G and the discriminator D. In order to trick the discriminator D, the generator must learn the image distribution of real samples and create fake images using random noise [27]. The discriminator is to the actual images from the fake images that have been generated from the coconut tree images are illustrated in figure 3.

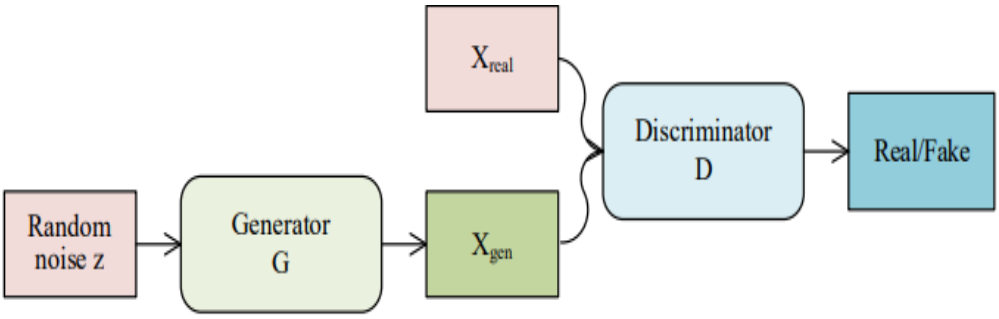


FIGURE 3. GAN ARCHITECTURE

The performance of G and D is continuously improved until Nash equilibrium is reached in these two adversarial trained neural networks. The GAN's loss function can be described as follows,

$$Loss = E_{X\sim P_{data}}[\log D(x)] + E_{Z\sim P_Z}[\log(1 - D(G(z))] \quad (8)$$

where $E_{S\sim P_{data}}$ and $E_{Z\sim P_Z}$ is denoted as the probability of x from the real image distribution as P_{data} and z sampled from the random noise prior distribution P_z, respectively. $D(x)$ is denoted as the discriminant result when the input of the discriminator is actual image x. $G(z)$ is denoted as the generated image of the generator, and $D(G(z))$ is denoted as the discriminant result when the input of the discriminator is the generated image $G(z)$. The binary minimax problem that describes the optimization procedure for D and G is represented by the equation (9),

$$Goal = \underset{G}{argmin}\,\underset{D}{maxloss} \quad (9)$$

Although conventional GANs are capable of producing realistic synthetic samples, they do not explicitly control the class of the generated image. This limitation becomes important when synthetic samples are required for specific coconut tree categories or disease classes. To address this issue, an Auxiliary Generative Adversarial Network (AGAN) incorporates class-label information into the adversarial learning process. AGAN is a modified model of GAN with the structure shown in Figure 4. Unlike GAN, AGAN can use label information to generate samples of specified types and to identify and classify the input images. Specifically, the generator G generates new samples $G(z, y)$ using random noise z and y, while the discriminator D needs not only to determine the actual or fake nature of the input images, but also to classify the input samples. During the adversarial training, the generating images capability and recognition capability of AGAN are continuously optimized. Eventually, the model has a strong ability to generate new images new with corresponding labels.

The discriminator has two major responsibilities. First, it determines the source of an input image, i.e., whether the image is real or synthetically generated. Second, it predicts the class label associated with the input image. Therefore, AGAN provides more informative supervision than a standard GAN and encourages the generator to create images that are not only visually realistic but also contain class-specific characteristics.

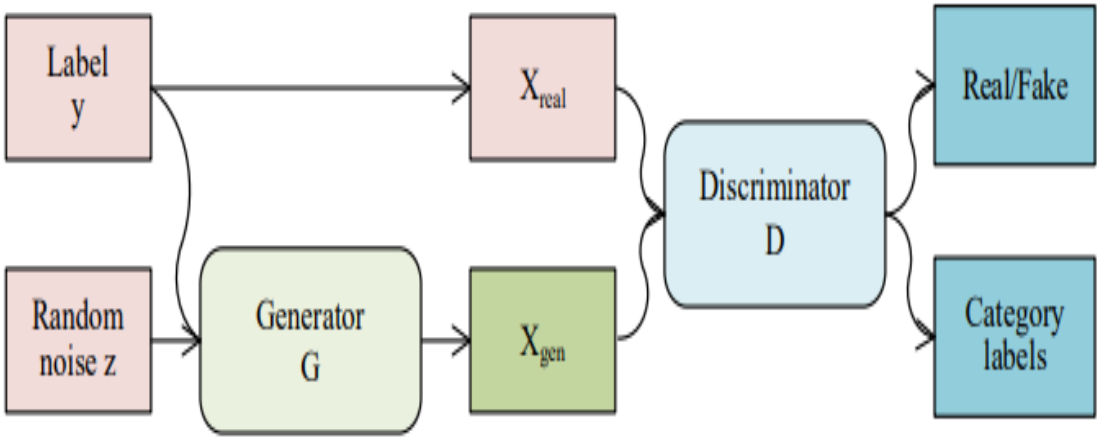


FIGURE 4. AGAN ARCHITECTURE

AGAN needs to process the source and class label information of the input images, the loss function of AGAN contains two components by equations (10-11),

$$L_{source} = E_{X\sim P_{data}}[\log D(x)] + E_{Z\sim P_Z}[\log(1 - D(G(z,y)))] \quad (10)$$

$$L_{class} = E_{X\sim P_{data}}[\log P(y = Y|X_{real})] + E_{Z\sim P_Z}[\log P(y = Y|X_{generated})] \quad (11)$$

where $G(z,y)$ is denoted as the generated image when the generator inputs are random noise z and sample label y, $P(y = Y|X_{real})$ is denoted the conditional probability distribution of the actual image, $P(y = Y|X_{generated})$ is denoted as the conditional probability distribution of the generated image. The objective function of the D is to maximize $L_{source} + L_{class}$, and the objective function of the G is to maximize $L_{source} - L_{class}$ by equations (12-13),

$$L_D = E_{X\sim P_{data}}[\log D(x)] + E_{Z\sim P_Z}[\log(1 - D(G(z,y)))] + E_{X\sim P_{data}}[\log P(y = Y|X_{real})] + E_{Z\sim P_Z}[\log P(y = Y|X_{generated})] \quad (12)$$

$$L_G = E_{X\sim P_{data}}[\log D(x)] + E_{Z\sim P_Z}[\log(1 - D(G(z,y)))] - E_{X\sim P_{data}}[\log P(y = Y|X_{real})] + E_{Z\sim P_Z}[\log P(y = Y|X_{generated})] \quad (13)$$

During training, real coconut tree images and their corresponding labels are initially supplied to the discriminator. Simultaneously, the generator receives random noise (z) together with a selected class label (y) and produces (G(z,y)). Both real and generated images are subsequently evaluated by the discriminator. The discriminator predicts whether each image is real or fake and simultaneously determines its corresponding class. The gradients obtained from these predictions are then propagated backward to update the parameters of the discriminator and generator. As adversarial training progresses, the discriminator becomes more effective at recognizing authentic images and class-specific characteristics, while the generator progressively learns to synthesize realistic images that preserve the visual properties of the requested coconut tree class.

AGAN is particularly beneficial for class-specific image augmentation. If certain coconut tree classes contain fewer training samples than others, the generator can produce additional synthetic images conditioned on those minority-class labels. These generated images can be combined with the original images to create a more balanced and diverse training dataset. Thus, AGAN can reduce class imbalance, increase intra-class variability, improve the representation of minority classes, and enhance the generalization capability of the subsequent deep learning-based coconut tree

classification or disease recognition model.

3.3. Feature extraction

Following image augmentation, the augmented coconut tree images are processed using a hybrid feature extraction framework consisting of a Transfer Learning-based Enhanced Visual Geometry Group 16 (EVGG16) network and a Swin Transformer. These two complementary feature extractors are employed to obtain both local spatial features and global contextual representations from the input images. EVGG16 primarily focuses on extracting fine-grained local characteristics, whereas the Swin Transformer captures long-range dependencies and broader contextual relationships among different image regions. The combination of these feature representations provides a more comprehensive description of coconut tree images and improves the discrimination of visually similar classes.

3.3.1. Enhanced Visual Geometry Group 16 (EVGG16) based local feature extraction

VGGNet is predicated on the repetitive stacking of smaller convolutional kernels to increase the network's depth. VGG16 comprises thirteen convolutional layers, three completely connected layers, and a SoftMax output layer. For each activation unit in the hidden layers, the Rectified Linear Unit (ReLU) function is utilized for classification. To increase the number of channels, more intricate and expressive features should be extracted and a convolution kernel with a capacity of 3×3 should be implemented. Between layers, sampling to capture finer information employs the maximum pooling approach with a 2×2 size configuration. As the number of convolution layers increases, so does the number of convolution kernels; the convolution layer depth is as follows: 64 → 128 → 256 → 512 → 512. The process consists of the following steps: initialize the basic model and load the pre-trained weights into it; freeze all the layers comprising the basic layer; construct a new fully connected layer or classifier layer atop the output of one or more basic layers; and, finally, train the newly constructed model using the new dataset. VGG16 model is performed by substituting the three fully connected layers in the original model with a fully connected layer, a batch normalization layer, and a global average pooling layer. While a batch normalization layer can accelerate convergence, a global average pooling layer can reduce the number of parameters.

Batch Normalization (BN): While neural networks exhibit greater sensitivity to data near zero, as the number of network layers increases, the characteristic data begin to diverge from the mean value of zero. Standardization can conform the data to a mean value of zero and a standard deviation distribution of one, thereby restoring the offset characteristic data to an approximate value of zero. Batch normalization transforms the input of every layer in the neural network so that it conforms to the standard normal distribution, which has a mean of zero and a standard deviation of one. Its objective is to address the gradient disappearance issue in neural networks.

Global Average Pooling (GAP): GAP layer is employed to transform the final refined feature maps into a compact one-dimensional feature vector before classification. Unlike conventional fully connected flattening operations, GAP computes the average activation of each feature channel across its entire spatial dimension. Consequently, each feature map is represented by a single scalar value. This operation preserves the overall response of each feature channel while significantly reducing the dimensionality of the feature representation. GAP is its ability to reduce

the number of model parameters. If a conventional flattening operation is employed, all spatial activations are converted into a large vector before being passed to a fully connected layer.

Optimization algorithm: The optimization algorithm updates the parameters of the filter in the process of backpropagation to reduce the error. Due to the parameter values of the randomly initialized filter, losses will occur in the forward propagation phase. It needs to be adjusted to reduce losses. Stochastic Gradient Descent is a fundamental optimization technique used for training neural networks. Instead of calculating the gradient using the complete training dataset at every iteration, SGD estimates the gradient using an individual training sample or, more commonly, a mini-batch of training samples. The network parameters are iteratively updated in the direction opposite to the gradient of the loss function.

3.3.2. Swin Transformer based global feature extraction

The Swin Transformer is a hierarchical vision transformer designed to extract both local and contextual information from images while maintaining relatively low computational complexity. In the proposed coconut tree disease classification framework, the Swin Transformer is used to complement the local features extracted by EVGG16 by learning relationships among spatially distributed disease symptoms.

Swin transformer, a patch partition module is employed to divide the disease image $I \in \mathbb{R}^{H \times W \times 3}$ into non-overlapping patches. Each patch is treated as token representing features of the corresponding locations within the original image. The patch size is set as 4×4 identical to the original implementation. Thus, dimension of feature vector for each patch is 4×4×3=48. Through the linear embedding layer, the channel dimension of each patch is projected to an arbitrary value C. Several Swin transformer blocks are applied on these patch tokens. Swin transformer block is combined with linear embedding and denoted as Swin Stage 1. Swin Stages 2–4 share similar structures consisting a patch merging layer and several Swin transformer blocks. The patch merging layer is mainly used for downsampling to increase the channel numbers and reduce the spatial resolution of feature maps. Moreover, Swin transformer blocks are responsible for effective feature extraction [28].

Figure 5 illustrates the detailed architecture of two consecutive Swin transformer blocks. Each block consists of a window-based multi-head self-attention (W-MSA) and shifted window-based multi-head self-attention (SW-MSA) followed by a 2-layer Multilayer Perceptron (MLP). The W-MSA module computes self-attention within non overlapping local windows to reduce computational complexity, while the SW-MSA module introduces a shifted window partitioning mechanism to enable cross-window connections, enhancing the ability to capture global context. Between the attention and MLP modules, Layer Norm (LN) layers are applied to stabilize training, and residual connections are added to facilitate gradient flow. Specifically, the MLP consists of two linear layers with a GELU activation function in-between, where the first layer expands the feature dimension by a factor of 4, and the second layer projects it back to the original dimension.

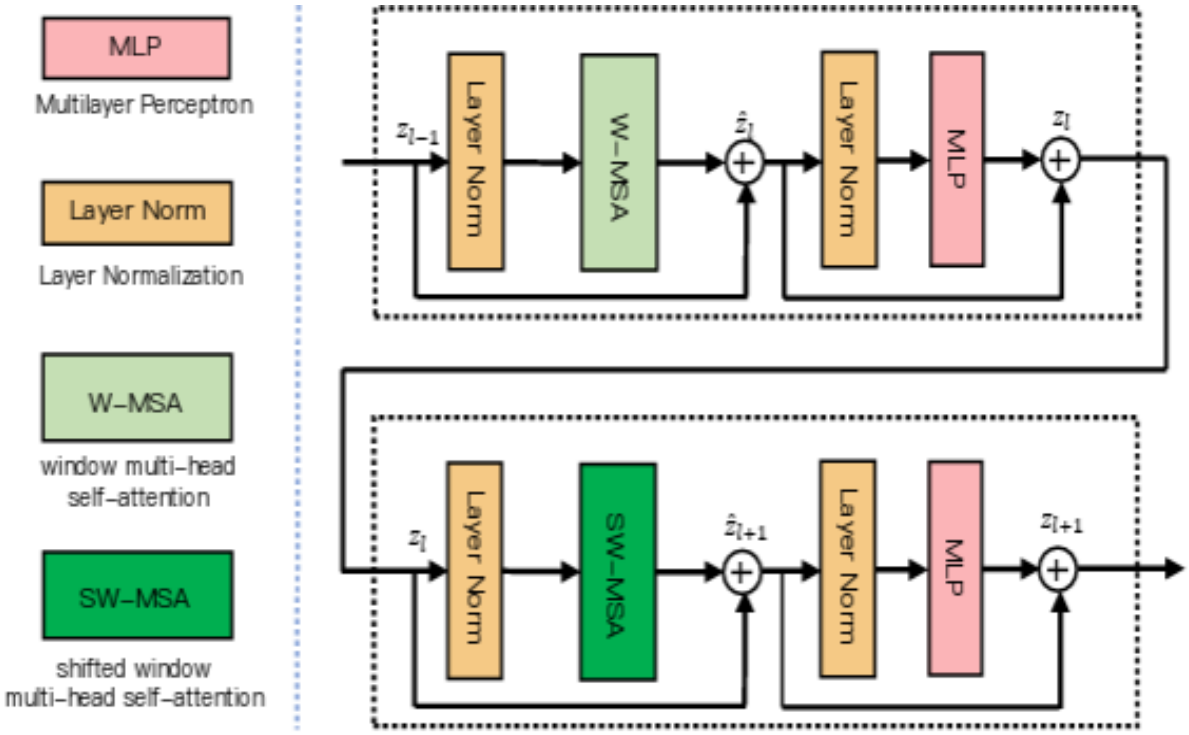


FIGURE 5. TWO CONSECUTIVE SWIN TRANSFORMER BLOCKS IN SWIN TRANSFORMER

The involved computation of consecutive Swin transformer blocks can be summarized as follows,

$$\hat{z}_l = W - MSA\left(LayerNorm\left(z_{l-1}\right)\right) + z_{l-1} \quad (14)$$

$$z_l = MLP\left(LayerNorm\left(\hat{z}_l\right)\right) + \hat{z}_l \quad (15)$$

$$\hat{z}_{l+1} = W - MSA\left(LayerNorm\left(z_l\right)\right) + z_l \quad (16)$$

$$z_{l+1} == MLP\left(LayerNorm\left(\hat{z}_{l-1}\right)\right) + \hat{z}_{l-1} \quad (17)$$

where z_l and $\hat{z}_l$ are the output features of the SW-MSA, and MLP for block l, respectively. The number of Swin transformer blocks in each stage ($\{n_1, n_2, n_3, n_4\}$) is set as {2,2,18,2} which is denoted as "Swin-S" in recent work [28].

3.4. Multi-Scale Encoder Decoder Self-Attention Network (MSEDSAN) classification

MSEDSAN is a symmetric structure which consists of an encoder, bottleneck, decoder and skip connections. Concretely, in the encoder, the multi-scale convolution is utilized to generate compact feature maps which capture the rich local detailed features of the input image. Then, the transformer is leveraged to model the long-distance dependency between high-level cotton leaf images semantics from a global space in the bottleneck. In the end, layer-by-layer upsampling and spatial attention are adopted to produce the high-resolution results. In addition, skip connections are added to connect the encoder and decoder to form the symmetric structure, and increase the spatial resolution of the semantic information.

The encoder progressively extracts hierarchical representations from the input features. Shallow encoder layers preserve high spatial resolution and therefore contain detailed local information, whereas deeper encoder layers have larger receptive fields and capture more abstract

semantic information. Cross-Scale Attention (CSA) is employed to establish explicit interactions between shallow and deep feature representations. Its main objective is to combine the exact spatial details available in shallow features with the semantic discrimination available in deeper features. The deep representation can be upsampled to match the spatial resolution of the shallow representation by equation (18),

$$\widetilde{Fe}_d = Up(Fe_d) \tag{18}$$

Channel projection can subsequently be performed using a learnable transformation in equation (19),

$$Fe'_s = W_s Fe_s, Fe'_d = W_d \widetilde{Fe}_d, Fe_s \in \mathbb{R}^{H_s \times W_s \times C_s}, Fe_d \in \mathbb{R}^{H_d \times W_d \times C_d} \tag{19}$$

where W_s and W_d is denoted as learnable projection parameters. CSA then establishes relationships between features across the two scales. A general cross-scale attention operation can be expressed as follows,

$$A_{CSA} = Softmax\left(\frac{Q_s K_d^T}{\sqrt{d}}\right) \tag{20}$$

$$Q_s = Fe'_s W_Q, K_d = Fe'_d W_K, V_d = Fe'_d W_V \tag{21}$$

where Q_s is denoted as the query generated from the shallow features, K_d and V_d is denoted as the key and value representations are generated from the deeper features. The cross-scale contextual information is obtained as follows,

$$Fe_{CSA} = A_{CSA} V_d \tag{22}$$

Thus, the shallow representation can selectively retrieve relevant semantic information from deeper feature levels rather than indiscriminately combining all deep features. A residual integration may then be performed by equation (23),

$$Fe_{CS} = Fe'_s + Fe_{CSA} \tag{23}$$

The resulting F_{CS} contains both fine spatial details and high-level disease semantics. A deeper feature representation provides broader contextual knowledge regarding the affected leaf region. CSA establishes an interaction between these representations so that the local spot information is interpreted in relation to high-level disease features. SCAF is employed to refine the cross-scale features in both the spatial and channel dimensions.

Channel attention evaluates the importance of individual feature channels. Global average pooling and global max pooling can be applied along the spatial dimensions. The resulting descriptors are passed through a shared transformation to generate channel-attention weights by equation (24),

$$M_c = \sigma\left[MLP\left(Fe^c_{avg}\right) + MLP(Fe^c_{max})\right] \tag{24}$$

where σ(·) is denoted as the sigmoid activation function. Thus, channels strongly associated with disease-specific characteristics are assigned higher weights, while redundant or less informative channels receive smaller weights.

Spatial attention determines which spatial locations of the coconut tree image contain the most discriminative disease information. Average pooling and max pooling are performed across the channel dimension. These representations are concatenated and processed using a convolutional operation in equation (25),

$$M_s = \sigma\left[Dilated\ Conv\left(Fe^s_{avg}; Fe^s_{max}\right)\right] \quad (25)$$

where $Dilated\ Conv$ is the dilated convolution is a convolution operation that enlarges the receptive field of a convolutional layer by inserting spaces between the kernel elements. Channel Enhancement is then performed by equation (26),

$$Fe_C = M_c \odot Fe_{CS} \quad (26)$$

where $\odot$ is denoted as the element-wise multiplication. The spatially refined feature representation by equation (27),

$$Fe_S = M_s \odot Fe_c \quad (27)$$

where $\odot$ is denoted element-wise multiplication. The spatially and channel-wise refined representations are subsequently integrated to obtain the final SCAF output. A general fusion operation can be represented by equation (28),

$$Fe_{SCAF} = Dilated\ Conv\left(Concat\left(Fe_c, Fe_s\right)\right) \quad (28)$$

The residual formulation preserves the original cross-scale information while incorporating the attention-refined representation.

In the decoder, several cascaded upsampling convolution operations are gradually employed to restore the original image size. Moreover, additional skip connections are added to increase the spatial resolution of the semantic information. The spatial position weights are learned through carefully designed modules and multiplied with the input feature maps; then, the results are concatenated with each other. Specifically, given a decoder feature map, max pooling and average pooling operations to aggregate the spatial information and concatenate them according to the channel dimension. Then, a 3×3 convolution and sigmoid function are utilized to obtain the spatial weights and multiply them with decoder feature map. In order to transfer the weighted information to the coarse-grained feature map, a 1×1 convolution is used to decrease the channel dimension and then interpolated to the same size as the input cotton leaf image.

Multi-Scale Feature Enhancement (MSFE): The feature map obtained from the preceding feature-fusion stage. Equation (29), capture disease-related characteristics at different spatial scales, the input feature map is processed using multiple convolutional kernels,

$$Fe_s = \sigma(we_s * Fe + b_s), s \in \{1,3,5\} \quad (29)$$

where we_s is denoted as the convolutional kernel of scale s, b_s is denoted as the corresponding bias, * is denoted as the convolution operation, and σ is denoted as the ReLU activation function. The multi-scale features are concatenated by equation (30),

$$Fe_{MS} = Concat\left(Fe_1, Fe_3, Fe_5\right) \quad (30)$$

Equation (31), the concatenated feature representation is further enhanced using a convolution operation,

$$Fe_E = \sigma(we_E * Fe_{MS} + b_E) \quad (31)$$

where we_E and b_E are the learnable weights and bias of the enhancement layer, respectively. The resulting Fe_R contains enhanced disease-specific information obtained from multiple spatial scales. After multi-scale refinement, GAP converts each two-dimensional feature map into a single representative value. For the c^{th} channel, GAP is expressed by Equation (32),

$$g_c = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} Fe_E(i,j,c) \tag{32}$$

where H and W are the spatial dimensions of the refined feature map, and $Fe_E(i,j,c)$ is denoted as the feature response at spatial position ((i,j)) of channel (c). The GAP feature vector is passed through a fully connected layer to generate the class logits by equation (33),

$$z_k = \sum_{c=1}^{C} we_{kc} g_c + b_k \tag{33}$$

where we_{kc}represents the learnable weight matrix, b_k represents the bias vector, and z_k contains the logits corresponding to the K coconut leaf classes. The gradients of the loss with respect to the weights and biases are calculated through backpropagation. AdamW then updates the parameters by equations (34-35),

$$w_{t+1} = w_t - \eta + \lambda w_t \tag{34}$$

$$b_{t+1} = b_t - \eta \tag{35}$$

where η=0.0001 is the learning rate and λ represents the weight-decay coefficient. The Softmax function transforms the logits into class probabilities by equation (36),

$$P_k = \frac{\exp(z_k)}{\sum_{j=1}^{K} \exp(z_j)} \tag{36}$$

The final leaf disease class is determined by selecting the class with the maximum predicted probability. Based on the maximum Softmax probability, the input coconut leaf image is finally classified as **healthy or one of the considered disease categories**.

The Cross-Entropy Loss (CEL) function is appropriate for optimizing the classification performance of MSEDSAN. It measures the difference between the actual disease class and the probability predicted by the Softmax layer. For N training samples and K coconut leaf classes, the loss is expressed by equation (37),

$$\ell_{CE} = -\frac{1}{N} \sum_{i=1}^{N} \sum_{k=1}^{K} y_{ik} \log(P_{ik}) \tag{37}$$

where N is denoted as the number of training images, K is denoted as the number of coconut leaf disease classes, y_{ik} is the ground-truth label for sample i and class k, and P_{ik} is denoted as the predicted probability of class k.

4. RESULTS AND DISCUSSION

The proposed framework for coconut leaf disease classification is implemented using Python 3.10 with the TensorFlow/Keras deep-learning framework. Table 1, experiments were performed on a workstation equipped with an Intel Core i7 processor, 32 GB RAM, and an NVIDIA GeForce RTX 3060. The coconut leaf images are resized to 224×224 pixels before being supplied to the proposed framework. Pixel intensities are normalized to the range ([0,1]), and image enhancement is performed during preprocessing to improve the visibility of disease symptoms. The dataset was divided into 70% training, 15% validation, and 15% testing subsets. The MSEDSAN was optimized using AdamW with a learning rate of 0.0001, batch size of 32, and 100 epochs. The model receives images of size 224×224×3 as input. Data augmentation increased the dataset from 5,798 original images to 6,793 images, resulting in 995 additional augmented images.

TABLE 1. SIMULATION PARAMETERS OF MSEDSAN MODEL

Parameters	Versions
Python version	Python 3.10
Deep-learning framework	TensorFlow/Keras
CPU	Intel Core i7
RAM	32 GB
GPU model	NVIDIA GeForce RTX 3060
GPU Memory	12 GB
CUDA version	CUDA 11.8
TensorFlow version	TensorFlow 2.13.0/Keras 2.13.1
Learning rate	**0.0001**
Optimizer	**AdamW**
Number of epochs	**100**
Batch size	**32**
Loss function	Cross-Entropy Loss (CEL)

Coconut tree disease dataset was collected through field visits to a farm located in Kendur, Taluka Shirur, District Pune, Maharashtra, India from https://data.mendeley.com/datasets/gh56wbsnj5/1. The original dataset consists of 5,798 images belonging to five disease categories: Bud Root Dropping, Bud Rot, Gray Leaf Spot, Leaf Rot, and Stem Bleeding. To ensure the dataset relevance and diversity, images were collected from various coconut plantations in the Kendur region, considering different growth stages, environmental conditions, and disease manifestations with Latitude-18°47̓06.4"N, Longitude-74 °01’19.5"E. The available dataset is divided into 70% training, 15% validation, and 15% testing sets using augmentation to maintain the class distribution. The class-wise distribution of the images in the Coconut Tree Disease Dataset before and after the augmentation process is shown in Table 2. It is clear that the original dataset contains 5,798 images that are equally divided into five diseases: Bud Root Dropping, Bud Rot, Gray Leaf Spot, Leaf Rot, and Stem Bleeding. However, the distribution shows significant class imbalance, where Gray Leaf Spot is the largest class with 2,135 images and Bud Rot class is the smallest one with 470 images. In order to enhance the presence of minority classes, the images were augmented and the number of images increased to 865 for Bud Root Dropping class, 752 for Bud Rot class, and 1,368 for Stem Bleeding class. On the other hand, there is no change in the number of images for Gray Leaf Spot and Leaf Rot classes; they still consist of 2,135 and 1,673 images, respectively. After the augmentation process, the total number of images increased to 6,793.

Table 2. Total number of images in the coconut tree disease dataset

Diseases	Total Images	After augmentation
Bud Root Dropping	514	865
Bud Rot	470	752

Gray Leaf spot	2135	2135
Leaf Rot	1673	1673
Stem Bleeding	1006	1368
Total	**5798**	**6793**

Table 3. Class-Wise Distribution of the Coconut Tree Disease Dataset after Augmentation and 70:15:15 Data Splitting

Diseases	Original	Training (70%)	Augmented Training (70%)	Validation (15%)	Testing (15%)
Bud Root Dropping	514	360	**711**	77	77
Bud Rot	470	330	**612**	70	70
Gray Leaf Spot	2,135	1,495	**1495**	320	320
Leaf Rot	1,673	1,171	**1,171**	251	251
Stem Bleeding	1,006	702	**1064**	152	152
Total	**5,798**	**4,058**	**5053**	**870**	**869**

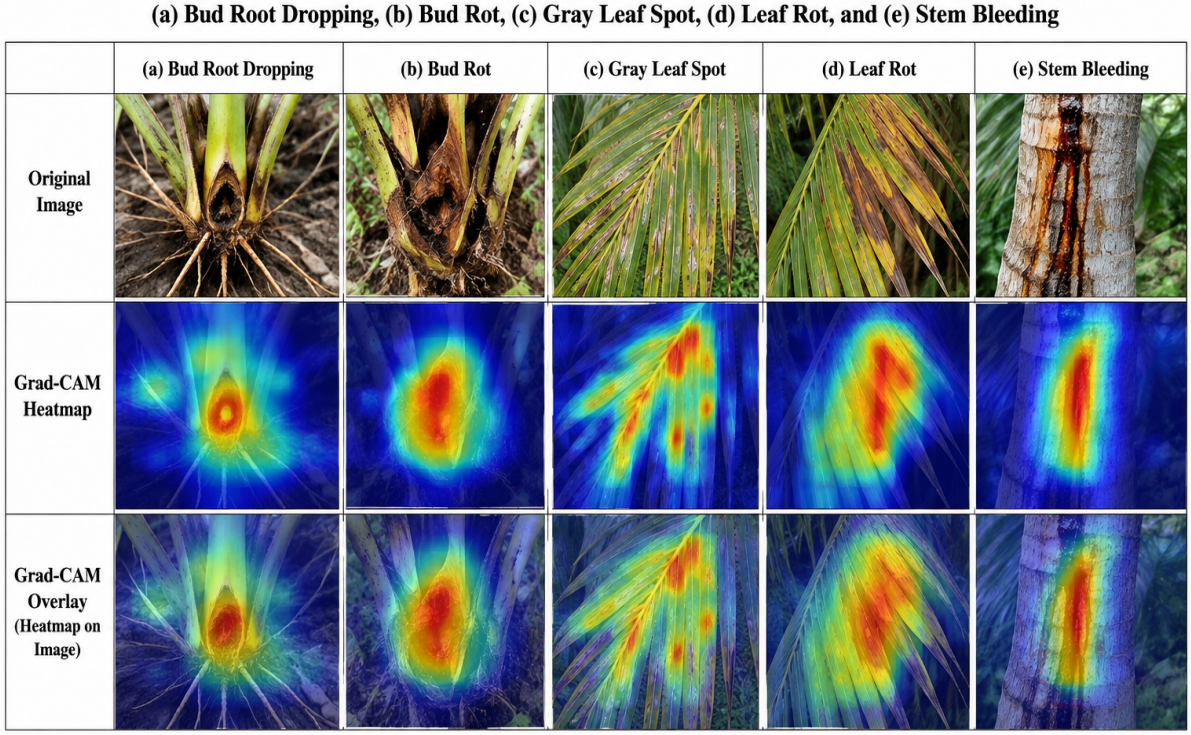
Table 3 shows the coconut disease dataset was categorized into five classes and assigned numerical labels for model training: Bud Root Dropping (Label 0), Bud Rot (Label 1), Gray Leaf Spot (Label 2), Leaf Rot (Label 3), and Stem Bleeding (Label 4). The original dataset contained 5,798 images. A class-wise 70:15:15 ratio was considered for training, validation, and testing. Initially, 4,058 images (70%) were allocated for training, while 870 images (15%) and 869 images (15%) were assigned to the validation and testing sets, respectively. To address the class imbalance, data augmentation was applied only to the training subset of the minority classes. Consequently, the number of training images increased from 4,058 to 5,053, whereas the validation and testing subsets remained unchanged. This approach ensures that augmented versions of images are used only during model training and that the validation and testing datasets contain original, unseen images for unbiased performance evaluation. The number of Bud Root Dropping training images increased from 360 to 711, while Bud Rot increased from 330 to 612 and Stem Bleeding increased from 702 to 1,064. In contrast, the Gray Leaf Spot and Leaf Rot classes were not augmented because their original training samples were comparatively higher. The effectiveness of the classifiers is evaluated using standard classification metrics such as Precision, Recall, F1-score, and Accuracy.

Table 4. Optimization and Training Parameters for Disease Detection

Methods	Optimizers	Learning Rate	Batch Size	Epochs	Training Strategy
TCNN	Adam	0.001	32	100	CNN training from scratch
MobileViT	AdamW	0.0001	32	100	Transfer learning/fine-tuning
Inception V3	Adam	0.0001	32	100	Transfer learning/fine-tuning
Mask R-	SGD +	0.001	16	100	Fine-tuning

CNN	Momentum				
MSEDSAN	AdamW	0.0001	32	100	Proposed architecture

The training and optimization parameters specific to each classifiers are shown in Table 4. Both TCNN and Inception V3 models were optimized using Adam, whereas MobileViT and MSEDSAN used AdamW optimization, and Mask R-CNN was optimized using SGD. The learning rates were determined based on the optimization needs of the models with values 0.001 and 0.0001 being chosen for different models. The number of epochs was kept at 100 for all methods. Grad-CAM is employed to provide visual interpretability by identifying the disease-relevant regions that contribute to MSEDSAN decisions, thereby supporting the explainability of the proposed model. Grad-CAM is used to understand visually how the model decides through highlighting the disease-related regions that MSEDSAN classifier uses in its decision-making process. The heatmaps show the discriminative regions that MSEDSAN classifier uses when classifying the input images into particular classes, indicating that MSEDSAN classifier has more focus to the disease-related visual features are discussed in figure 6.



Disease Names	Bud Root Dropping	Bud Rot	Gray Leaf Spot	Leaf Rot	Stem Bleeding	Total
After Augmentation (Number of Images)	865	752	2,135	1,673	1,368	6,793

FIGURE 6. Grad-CAM VISUALIZATION OF MSEDSAN CLASSIFIER

Precision: Precision represents the fraction of coconut leaf images predicted as a particular disease class that are correctly classified as disease by equation (38),

$$\text{Precision}_i = \frac{TP_i}{TP_i + FP_i} \tag{38}$$

Recall (Sensitivity): Recall measures the fraction of actual coconut leaf images belonging to a particular disease class that are correctly identified by the equation (39),

$$\text{Recall(Sensitivity)}_i = \frac{TP_i}{TP_i + FN_i} \tag{39}$$

F1-score: The F1-score is the harmonic mean of precision and recall and provides a balanced evaluation of the model by considering both false-positive and false-negative disease classifications. It is calculated using equation (40),

$$F1 - score_i = \frac{2 * Precision_i * Recall_i}{Precision_i + Recall_i} \tag{40}$$

Accuracy: Accuracy represents the overall proportion of coconut leaf images that are correctly classified into their corresponding healthy or disease categories. It is expressed using Equation (41),

$$Accuracy = \frac{TP_i + TN_i}{TP_i + TN_i + FP_i + FN_i} \tag{41}$$

Where i is the number of classes=5.

True Positive (TP): Coconut leaf images belonging to a particular disease class that are correctly classified as that disease class.

True Negative (TN): Coconut leaf images not belonging to a particular disease class that are correctly classified as other healthy or disease classes.

False Positive (FP): Coconut leaf images belonging to other classes that are incorrectly classified as the considered disease class.

False Negative (FN): Coconut leaf images belonging to the considered disease class that are incorrectly classified as another healthy or disease class.

Confusion matrix of classification techniques are described in table 5 (a-e). The TCNN classifier correctly classified 748/865 Bud Root Dropping, 642/752 Bud Rot, 1,848/2,135 Gray Leaf Spot, 1,442/1,673 Leaf Rot and 1,154/1,368 Stem Bleeding samples. MobileViT classifier correctly identified 771/865 Bud Root Dropping, 668/752 Bud Rot, 1,914/2,135 Gray Leaf Spot, 1,489/1,673 Leaf Rot, and 1,200/1,368 Stem Bleeding samples. Inception V3 classifier correctly classified 789/865 Bud Root Dropping, 682/752 Bud Rot, 1,965/2,135 Gray Leaf Spot, 1,516/1,673 Leaf Rot, and 1,235/1,368 Stem Bleeding samples The Mask R-CNN classifier correctly classified 801 of 865 Bud Root Dropping, 696 of 752 Bud Rot, 1,994 of 2,135 Gray Leaf Spot, 1,536 of 1,673 Leaf Rot and 1,256 of 1,368 Stem Bleeding samples. The proposed MSEDSAN classifier correctly classified 815 of 865 Bud Root Dropping, 707 of 752 Bud Rot, 2,018 of 2,135 Gray Leaf Spot, 1,555 of 1,673 Leaf Rot, and 1,275 of 1,368 Stem Bleeding samples.

TABLE 5. CONFUSION MATRIX OF DISEASE DETECTION METHODS

(a) Confusion Matrix of TCNN

Actual \ Predicted class	Bud Root Dropping	Bud Rot	Gray Leaf Spot	Leaf Rot	Stem Bleeding	Total
Bud Root Dropping	**748**	29	25	34	29	865
Bud Rot	28	**642**	27	29	26	752
Gray Leaf Spot	35	30	**1,848**	112	110	2,135
Leaf Rot	37	28	86	**1,442**	80	1,673
Stem Bleeding	32	27	72	83	**1,154**	1,368

Total	**880**	**756**	**2,058**	**1,700**	**1,399**	**6,793**

(b) Confusion Matrix of MobileViT

Actual \ Predicted class	**Bud Root Dropping**	**Bud Rot**	**Gray Leaf Spot**	**Leaf Rot**	**Stem Bleeding**	**Total**
Bud Root Dropping	**771**	23	20	27	24	865
Bud Rot	22	**668**	21	21	20	752
Gray Leaf Spot	26	24	**1,914**	87	84	2,135
Leaf Rot	29	21	62	**1,489**	72	1,673
Stem Bleeding	25	19	55	69	**1,200**	1,368
Total	**873**	**755**	**2,072**	**1,693**	**1,400**	**6,793**

(c) Confusion Matrix of Inception V3

Actual \ Predicted class	**Bud Root Dropping**	**Bud Rot**	**Gray Leaf Spot**	**Leaf Rot**	**Stem Bleeding**	**Total**
Bud Root Dropping	**789**	18	16	22	20	865
Bud Rot	18	**682**	16	19	17	752
Gray Leaf Spot	20	18	**1,965**	68	64	2,135
Leaf Rot	23	17	49	**1,516**	68	1,673
Stem Bleeding	20	15	43	55	**1,235**	1,368
Total	**870**	**750**	**2,089**	**1,680**	**1,404**	**6,793**

(d) Confusion Matrix of Mask R-CNN

Actual \ Predicted class	**Bud Root Dropping**	**Bud Rot**	**Gray Leaf Spot**	**Leaf Rot**	**Stem Bleeding**	**Total**
Bud Root Dropping	**801**	15	14	18	17	865
Bud Rot	15	**696**	14	14	13	752
Gray Leaf Spot	17	15	**1,994**	56	53	2,135
Leaf Rot	19	14	40	**1,536**	64	1,673
Stem Bleeding	17	12	35	48	**1,256**	1,368
Total	**869**	**752**	**2,097**	**1,672**	**1,403**	**6,793**

(e) Confusion Matrix of MSEDSAN

Actual \ Predicted class	**Bud Root Dropping**	**Bud Rot**	**Gray Leaf Spot**	**Leaf Rot**	**Stem Bleeding**	**Total**
Bud Root Dropping	**815**	12	11	14	13	865
Bud Rot	12	**707**	11	12	10	752
Gray Leaf Spot	13	12	**2,018**	47	45	2,135
Leaf Rot	15	11	34	**1,555**	58	1,673

Stem Bleeding	13	9	30	41	**1,275**	1,368
Total	**868**	**751**	**2,104**	**1,669**	**1,401**	**6,793**

The performance of the MSEDSAN model against four models such as TCNN, MobileViT, Inception V3, and Mask R–CNN has been shown in Table 6. Table 6, proposed model performed the best among all with the highest precision of 94.16%, recall of 93.72%, F1-score of 93.94%, and accuracy of 94.58%. The high precision value means that the proposed model has efficiently reduced the number of false detections of diseases, and recall showed its capability of detecting the largest possible number of diseases.

TABLE 6. PERFORMANCE COMPARISON OF DISEASE DETECTION METHODS

Methods	Precision (%)	Recall (%)	F1-score (%)	Accuracy (%)
TCNN	87.52	86.94	87.23	88.16
MobileViT	89.36	88.72	89.04	89.85
Inception V3	91.24	90.68	90.96	91.73
Mask R-CNN	92.58	92.14	92.36	93.05
MSEDSAN	**94.16**	**93.72**	**93.94**	**94.58**

MSEDSAN classifier provides the best performance among the four evaluation measures. The low standard deviation means that there is little variation between independent executions. Table 7, and Table 8 shows the MSEDSAN classifier has precision, recall, F1 score, and accuracy of 94.16±0.19%, 93.72±0.20%, 93.94±0.20%, and 94.58±0.19%, respectively. Meanwhile, TCNN provides the worst performance, whereas MobileViT, Inception V3, and Mask R-CNN offer increasing levels of performance.

TABLE 7. PERFORMANCE RESULTS OVER FIVE INDEPENDENT RUNS

Methods	Metrics (%)	Run 1	Run 2	Run 3	Run 4	Run 5	Mean ± SD
TCNN	**Precision**	87.10	87.81	87.39	87.94	87.36	**87.52 ± 0.35**
	Recall	86.48	87.21	86.77	87.36	86.88	**86.94 ± 0.34**
	F1-score	86.84	87.56	87.09	87.61	87.05	**87.23 ± 0.32**
	Accuracy	87.72	88.43	88.01	88.57	88.07	**88.16 ± 0.33**
MobileViT	**Precision**	89.01	89.62	89.21	89.71	89.25	**89.36 ± 0.29**
	Recall	88.35	88.96	88.57	89.11	88.67	**88.73 ± 0.29**
	F1-score	88.70	89.30	88.89	89.43	88.88	**89.04 ± 0.29**
	Accuracy	89.51	90.10	89.70	90.18	89.71	**89.84 ± 0.28**
Inception V3	**Precision**	90.91	91.42	91.10	91.58	91.19	**91.24 ± 0.26**
	Recall	90.34	90.87	90.52	91.01	90.66	**90.68 ± 0.25**
	F1-score	90.67	91.19	90.81	91.30	90.83	**90.96 ± 0.25**
	Accuracy	91.42	91.91	91.61	92.02	91.69	**91.73 ± 0.24**
Mask R-CNN	**Precision**	92.27	92.77	92.46	92.86	92.54	**92.58 ± 0.23**
	Recall	91.83	92.31	92.02	92.44	92.10	**92.14 ± 0.23**
	F1-score	92.11	92.54	92.25	92.72	92.28	**92.38 ± 0.23**
	Accuracy	92.72	93.21	92.94	93.38	93.00	**93.05 ± 0.25**
MSEDSAN	**Precision**	93.91	94.32	94.08	94.39	94.10	**94.16 ± 0.19**

	Recall	93.49	93.91	93.61	93.98	93.61	**93.72 ± 0.20**
	F1-score	93.71	94.12	93.85	94.21	93.81	**93.94 ± 0.20**
	Accuracy	94.35	94.73	94.51	94.82	94.49	**94.58 ± 0.19**

TABLE 8. STATISTICAL PERFORMANCE COMPARISON OVER 5 INDEPENDENT RUNS

Methods	Precision (%)	Recall (%)	F1-score (%)	Accuracy (%)
TCNN	87.52 ± 0.35	86.94 ± 0.34	87.23 ± 0.32	88.16 ± 0.33
MobileViT	89.36 ± 0.29	88.73 ± 0.29	89.04 ± 0.29	89.84 ± 0.28
Inception V3	91.24 ± 0.26	90.68 ± 0.25	90.96 ± 0.25	91.73 ± 0.24
Mask R-CNN	92.58 ± 0.23	92.14 ± 0.23	92.38 ± 0.23	93.05 ± 0.25
MSEDSAN	**94.16 ± 0.19**	**93.72 ± 0.20**	**93.94 ± 0.20**	**94.58 ± 0.19**

TABLE 9. PAIRED t-TEST RESULTS FOR ACCURACY OVER FIVE INDEPENDENT RUNS

Compared Methods	Mean Accuracy Difference (%)	t-value	df	p-value	Statistical Significance
MSEDSAN vs. TCNN	**6.42**	**93.745**	**4**	**7.76×10^{-8}**	**Significant**
MSEDSAN vs. MobileViT	**4.74**	**107.895**	**4**	**4.42×10^{-8}**	**Significant**
MSEDSAN vs. Inception V3	**2.85**	**104.768**	**4**	**4.98×10^{-8}**	**Significant**
MSEDSAN vs. Mask R-CNN	**1.53**	**46.773**	**4**	**1.25×10^{-6}**	**Significant**

Table 9 presents the paired t-test results comparing the accuracy of MSEDSAN with other methods over five independent runs. MSEDSAN achieved mean improvements of **6.42%, 4.74%, 2.85%, and 1.53%** over TCNN, MobileViT, Inception V3, and Mask R-CNN, respectively. Since all p-values were below **0.05**, the performance improvements were statistically significant and unlikely to be caused by random variation.

Table 10 presents the ablation study conducted to evaluate the individual and combined contributions of the major components of the proposed model. The baseline EVGG16 achieved an accuracy of 88.16%. The incorporation of AGAN improved the accuracy to 90.42%, while integrating the Swin module increased the accuracy to 92.17%. Combining both Swin and AGAN with EVGG16 further improved the performance to 93.46%. Finally, the full proposed model achieved the highest accuracy of 94.58%, demonstrating that each component contributes to the overall classification performance.

TABLE 10. ABLATION STUDY OF THE PROPOSED MODEL

Model	Accuracy (%)
EVGG16	88.16
EVGG16 + AGAN	90.42

EVGG16 + Swin	92.17
EVGG16 + Swin + AGAN	93.46
Proposed model	**94.58**

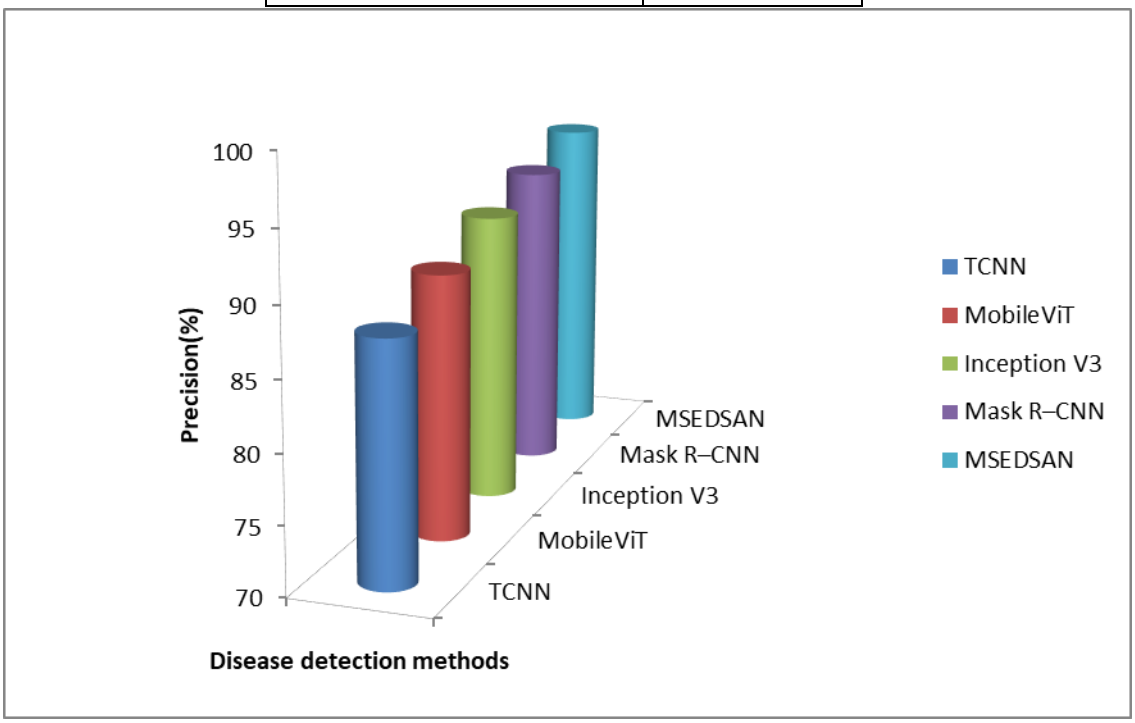


FIGURE 7. PRECISION COMPARISON OF DISEASE DETECTION METHODS

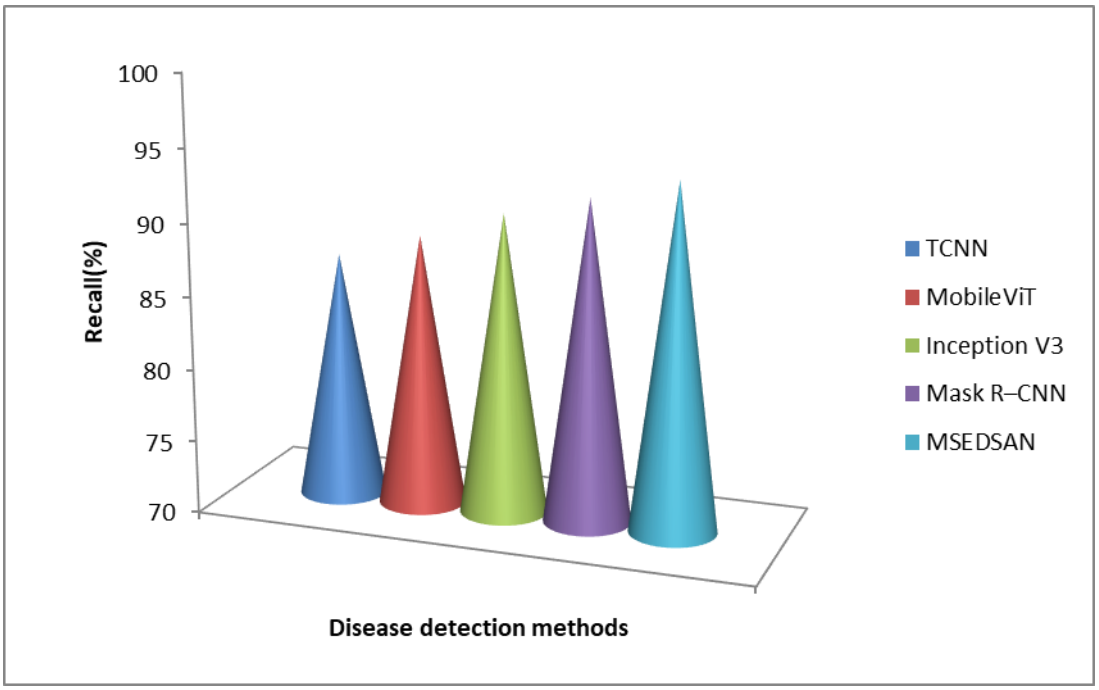


FIGURE 8. RECALL COMPARISON OF DISEASE DETECTION METHODS

Figure 7 shows the performance comparison of recall using the five approaches for disease detection methods. The precision of TCNN, MobileViT, Inception V3, Mask R–CNN, and MSEDSAN is 87.52%, 89.36%, 91.24%, 92.58%, and 94.16%, respectively. The proposed model

increases the precision of TCNN, MobileViT, Inception V3, and Mask R–CNN by 7.05%, 5.10%, 3.20%, and 1.68%, respectively. By combining hierarchical multi-scale feature extraction with attention-guided feature enhancement, MSEDSAN enhances the discrimination of disease-specific characteristics while reducing false-positive predictions, resulting in superior precision compared with the existing approaches.

The proposed model has the best recall value of 93.72%, proposed model performs better in disease detection than the TCNN, MobileViT, Inception V3, and Mask R–CNN models with their respective recall values of 86.94%, 88.72%, 90.68%, and 92.14% are illustrated in figure 8. The proposed model makes the most significant improvement in the recall value of 7.24%, 5.33%, 3.24%, and 1.69% over the four existing methods. This novel multi-scale encoder–decoder architecture integrated with a self-attention mechanism, which effectively captures both fine-grained local disease patterns and long-range contextual dependencies.

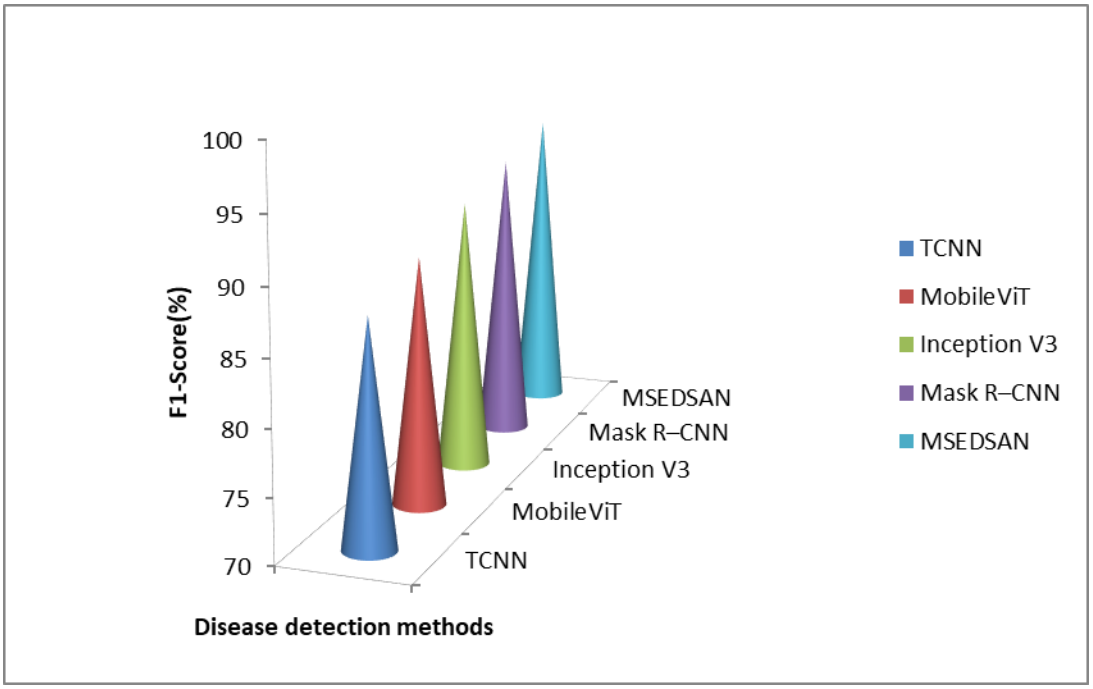


FIGURE 9. F1-SCORE COMPARISON OF DETECTION METHODS

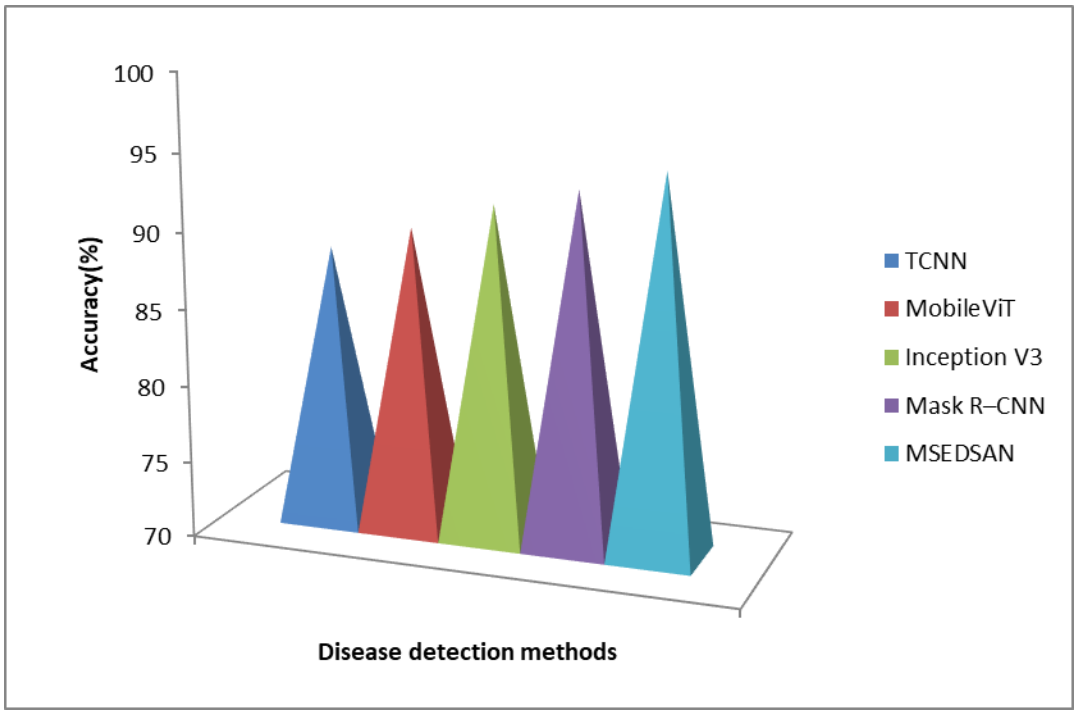


FIGURE 10. ACCURACY COMPARISON OF DETECTION METHODS

The overall classification accuracy further confirms the effectiveness of the proposed framework. achieves the highest accuracy of 94.58%, followed by TCNN (88.16%), MobileViT (89.85%), Inception V3 (91.73%), and Mask R–CNN (93.05%) are illustrated in figure 10. The classification accuracy improves by 6.78%, 5.00%, 3.01%, and 1.62% over TCNN, MobileViT, InceptionV3, and MaskR-CNN respectively. MSEDSAN model progressively extracts and reconstructs hierarchical disease-related features at different spatial resolutions. MSEDSAN is used to generate more discriminative and context-aware feature representations, resulting in accurate diffesrentiation among disease classes and demonstrating the robustness of the proposed framework.

The proposed model has the best f1-score of 93.94%, proposed model performs better in disease detection than the TCNN, MobileViT, Inception V3, and Mask R–CNN models with their respective recall values of 87.23%, 89.04%, 90.96%, and 92.36% are illustrated in figure 9. The proposed model makes the most significant improvement in the f1-score of 7.14%, 5.22%, 3.17%, and 1.68% over the four existing methods. Multi-scale feature representation enables the model to effectively identify disease patterns of varying sizes, while self-attention emphasizes the most discriminative disease-relevant regions and suppresses irrelevant features. This balanced enhancement of feature representation improves both precision and recall, thereby enabling MSEDSAN model to achieve a higher F1-score and more reliable disease detection than the existing approaches.

5. CONCLUSION AND FUTURE WORK

In this paper, Auxiliary Classifier Generative Adversarial Network (AC-GAN) is developed for conditional image generation. It extends standard GANs by adding class labels to the generator and the discriminator with an auxiliary to reconstruct those class labels, ensuring high-resolution and semantically coherent synthetic images. The augmented coconut tree images are processed using a feature extraction model consisting of a Transfer Learning-based Enhanced Visual Geometry Group

16 (EVGG16) network and a Swin Transformer (ST). EVGG16 primarily focuses on extracting fine-grained local characteristics, whereas the ST captures long-range dependencies and broader contextual relationships among different image regions. Multi-Scale Encoder Decoder Self-Attention Network (MSEDSAN) classification is developed for diseases classification with Bud Root Dropping, Bud Rot, Gray Leaf spot, Leaf Rot, and Stem Bleeding. MSEDSAN classifier, encoder progressively extracts hierarchical representations from the input features. Cross-Scale Attention (CSA) is employed to establish explicit interactions between shallow and deep feature representations. SCAF also employed to refine the cross-scale features in both the spatial and channel dimensions. Channel attention evaluates the importance of individual feature channels. Spatial attention determines which spatial locations of the coconut tree image contain the most discriminative disease information. In the decoder, several cascaded upsampling convolution operations are gradually employed to restore the original images and classify the images. Softmax probability, the coconut leaf image is finally classified as healthy or one of the considered disease categories. The performance of the proposed model is evaluated using precision, recall, F1-score, and accuracy, demonstrating its effectiveness for multiclass coconut disease classification. In the future work, present model is extended to other deep learning methods, and real time classifications of coconut leaf images. Advanced vision transformer architectures and efficient attention mechanisms are introduced to further enhance recognition speed and classification accuracy for real-time precision agriculture applications. Moreover, the proposed model can be evaluated using larger field-acquired datasets containing variations in illumination, background, disease severity, growth stage, and environmental conditions.

REFERENCES

1. Ahmad, M.S.Z.; Aziz, N.A.A.; Lim, H.S.; Ghazali, A.K.; Latiff, ‘A.A. Impact of Image Enhancement Using Contrast-Limited Adaptive Histogram Equalization (CLAHE), Anisotropic Diffusion, and Histogram Equalization on Spine X-Ray Segmentation with U-Net, Mask R-CNN, and Transfer Learning. Algorithms 2025, 18, 796, pp.1-22. https://doi.org/10.3390/a18120796
2. Annepu, V., Bagadi, K., Tolephih, M.H., Al-Ameen, J.A.I., Kumar, V.N., Mohammed, M.N., Abdullah, O.I., Nursultan, D., Abdullah, T.A. and Al-Ayash, A.A., 2025. Automated Detection of Coconut Leaf Diseases Using Deep Learning Techniques. In Tech Fusion in Business and Society: Harnessing Big Data, IoT, and Sustainability in Business: Volume 1 (pp. 63-74). Cham: Springer Nature Switzerland.
3. AnjnaSood M.Singh P.K. (2020). Hybrid system for detection and classification of plant disease using qualitative texture features analysis. Proc. Comput. Sci. 167 (2019), 1056–1065.
4. Cherian, S.J., 2025. Efficient MobileViT-Based Framework for Early Detection and Classification of Coconut Leaf Disease. International Journal of Advanced Trends in Engineering and Management (IJATEM), 4(8), pp.39–51.
5. de Luna, R.G., Alegre, J.C.M., Martinez, R.A.P., Ramos, M.R.N., Sadian, H.M.A. and Villanueva, O.G.G., 2025. Coconut Maturity Level Classification using Image Preprocessing

and Deep Transfer Learning. In 2025 RIVF International Conference on Computing and Communication Technologies (RIVF), pp. 1140-1144.

6. Duan, Y., Song, C., Zhang, Y., Cheng, P., Mei, S. STMSF: Swin Transformer with Multi-Scale Fusion for Remote Sensing Scene Classification. Remote Sensing. 2025; 17(4), pp.1-19. https://doi.org/10.3390/rs17040668.
7. Feng, W., Sun, G., & Zhang, X. (2024). Plant Disease Identification Based on Encoder–Decoder Model. Agronomy, 14(10), pp.1-22.
8. Genaev, M.A., Skolotneva, E.S., Gultyaeva, E.I., Orlova, E.A., Bechtold, N., Afonnikov, D.A. (2021). Image-based wheat fungi diseases identification by deep learning. Plants 10(8), 1–21.
9. Henrietta, H.M., Kalaiyarasi, K. and Raj, A.S., 2022. Coconut tree (Cocos nucifera) products: A review of global cultivation and its benefits. Journal of Sustainability and Environmental Management, 1(2), pp.257-264.
10. Huang, X., Alobaedy, M.M., Fazea, Y., Goyal, S.B. and Deng, Z., 2025. Disease infection classification in coconut tree based on an enhanced visual geometry group model. Processes, 13(3), pp.1-13.
11. Jaidka, P., Upadhyay, P., Anand, A., Kumar, A. and Yadav, S.P., 2024. Transforming Coconut Farming with Deep Learning Disease Detection. Evergreen, 11(4), pp.3233–3242.
12. Johri, P., Kim, S., Dixit, K., Sharma, P., Kakkar, B., Kumar, Y., Shafi, J. and Ijaz, M.F., 2024. Advanced deep transfer learning techniques for efficient detection of cotton plant diseases. Frontiers in Plant Science, 15, pp.1-22.
13. Marasović, T.; Papić, V. Symmetry-Driven Enhanced Auxiliary Classifier GAN for Data-Efficient Breast Tumor Classification. Symmetry 2026, 18, 1235, pp.1-34. https://doi.org/10.3390/sym18071235.
14. Murugavalli, S. and Gopi, R., 2025. Plant leaf disease detection using vision transformers for precision agriculture. Scientific Reports, 15, pp.1-17.
15. Parvathi, S. and Selvi, S.T., 2021. Detection of maturity stages of coconuts in complex background using Faster R-CNN model. Biosystems engineering, 202, pp.119-132.
16. Ramesh, M., Kodeeswari, K., AnithaRani, A., Maheswaran, S., Keerthi, P. and Suganth, K.M., 2024. Enhancing Detection of Nutritional Deficiencies in Coconut Trees Using Transformer-Based Convolutional Neural Networks. In 2024 15th International Conference on Computing Communication and Networking Technologies (ICCCNT), pp. 1-6.
17. Sáenz-Carbonell, L., Nguyen, Q., López-Villalobos, A. and Oropeza-Salín, C., 2020. Coconut micropropagation for worldwide replanting needs. In Coconut Biotechnology: Towards the Sustainability of the 'Tree of Life' (pp. 227-240). Cham: Springer International Publishing.
18. Santhiya, S., Krishnasamy, L., Sharmila, C., Jayadharshini, P., Dharshan, P.P. and Haripreetha, S., 2024. Early Detection of Coconut Palm Leaflet Diseases using Convolutional Neural Networks based Deep Learning Approach. In 2024 IEEE 4th International Conference on ICT in Business Industry & Government (ICTBIG), pp. 1-6.

19. Shafik, W., Tufail, A., De Silva Liyanage, C. and Apong, R.A.A.H.M., 2024. Using transfer learning-based plant disease classification and detection for sustainable agriculture. BMC Plant Biology, 24(1), pp.1-19.
20. Shanmugam, P., Moorthi, K., Swarna, S.L. and Mohamed, A., 2025. Early Identification of Coconut Leaf Diseases using Inception V3 for Precision Agriculture. In 2025 5th International Conference on Soft Computing for Security Applications (ICSCSA), pp. 925-931.
21. Singh, V., Sharma, N. and Singh, S., 2020. A review of imaging techniques for plant disease detection. Artificial Intelligence in Agriculture, 4, pp.229-242.
22. Subramanian, P., Gupta, A., Gopal, M., Selvamani, V., Mathew, J., Surekha and Indhuja, S., 2024. Coconut (Cocos nucifera L.). In Soil Health Management for Plantation Crops: Recent Advances and New Paradigms (pp. 37-109). Singapore: Springer Nature Singapore.
23. Toradmalle, D., Bhanushali, N., Dama, M. and Dhote, M., 2023. Deep Learning Framework for Early Detection of Diseases in Coconut Tree from Leaf Images. In 2023 6th International Conference on Advances in Science and Technology (ICAST), pp. 137-142.
24. Usman, U., Yunita, F. and Ridha, M.R., 2025. Improving classification accuracy of local coconut fruits with image augmentation and deep learning algorithm convolutional neural networks (CNN). Journal of Applied Data Sciences, 6(1), pp.1-19.
25. Vidhanaarachchi, S., Wijekoon, J.L., Abeysiriwardhana, W.S.P. and Wijesundara, M., 2025. Early diagnosis and severity assessment of Weligama coconut leaf wilt disease and coconut caterpillar infestation using deep learning-based image processing techniques. IEEE Access, 13, pp.24463-24477.
26. Vigneshwaran, S. and Tamburi, V.N., 2023. Identification of coconut palm trees using single shot detector deep learning model. Spatial Information Research, 31(6), pp.695-707.
27. Vishnoi, V.K., Kumar, K. and Kumar, B., 2021. Plant disease detection using computational intelligence and image processing. Journal of Plant Diseases and Protection (DPG), 128(1), pp.19-53.
28. Zhang, N., Yang, G., Pan, Y., Yang, X., Chen, L. and Zhao, C., 2020. A review of advanced technologies and development for hyperspectral-based plant disease detection in the past three decades. Remote Sensing, 12(19), pp.1-34.