[1]A. S. Devi
[2]S. Albert Antony Raj

Optimized Betel Leaf Disease Detection Using Improved CNN Model for Precision Agriculture

Abstract: - Plant diseases are a major threat to the production of betel leaves, causing large financial losses and endangering the expansion of the betel leaf market. This paper examines how deep learning methods can be used to analyze massive plant image datasets in order to tackle this problem. The processes covered in the study include feature extraction, feature selection, pre-processing, data collecting, and classification. Data collection involves gathering an extensive database of plant images depicting various diseases affecting betel plants. Pre-processing methods, like resizing,data augmentation,noise reduction, enhance image quality. Principal Component Analysis (PCA) is employed as a feature extraction technique to extract relevant data from the images. For classification, deep learning models like VGGNet and MobileNet based CNN architectures are employed, and it offers improved performance in image recognition tasks. These models use the features that have been retrieved and chosen to effectively and precisely categorize images. Through the identification of distinct patterns and symptoms linked to betel plant diseases, these algorithms provide prompt classification, thereby averting possible economic damage and preserving food security. The proposed model is implemented using Python platform.

Keywords: Betel Leaf Disease; Adam; PCA; Deep Learning; CNN; VGGNet; MobileNet.

1. INTRODUCTION

Betel leaf, scientifically known as Piper betel, is a perennial evergreen vine widely cultivated in tropical regions of Asia, Africa, and the Pacific islands. It holds immense cultural, medicinal, and economic significance, being utilized in various traditional practices, culinary preparations, and herbal remedies. However, betel leaf cultivation faces significant challenges posed by various diseases that can adversely affect its yield, quality, and overall productivity. Timely and accurate identification of these diseases is crucial for effective disease management and ensuring the sustainable production of betel leaves. Pathogens like fungi, bacteria, and viruses are the source of betel leaf diseases, which can take many different forms such as rot, wilts, blights, and leaf spots. These diseases not only diminish the aesthetic appeal of the leaves but also compromise their nutritional value and marketability. Furthermore, unchecked disease outbreaks can lead to substantial economic losses for betel leaf farmers, disrupting livelihoods and threatening food security in regions where betel leaf cultivation is a primary source of income.

Traditional methods of disease diagnosis in betel leaf cultivation rely heavily on visual inspection by experienced agronomists or plant pathologists. However, this approach is often subjective, time-consuming, and prone to errors, especially in cases where multiple diseases exhibit similar symptoms. Additionally, the increasing prevalence of betel leaf diseases necessitates more efficient and scalable disease identification methods to meet the growing demand for timely intervention and mitigation strategies. Recent advancements in artificial intelligence (AI) and machine learning (ML) offer promising solutions for automated disease detection and classification in agricultural crops, including betel leaves. Utilizing computer vision, deep learning algorithms, and data-driven approaches, these technologies analyze vast image datasets to identify disease patterns accurately. AI-driven models expedite diagnosis, improve disease surveillance, and enable targeted control measures to curb pathogen spread.

The following lists the contributions made by this paper,

[1,2]Department of Computer Applications, Faculty of Science and Humanities,
SRM Institute of Science and Technology, Kattankulathur-603 203, Tamil Nadu, India

- To contribute to the advancement of disease classification techniques, the paper used PCA as a feature extraction method. It showcases the effectiveness of PCA in extracting pertinent information from images, laying the foundation for more accurate disease classification.
- The study investigates the use of deep learning models made up of VGGNet and MobileNet architectures to solve the difficulties associated with identifying diseases in betel plants. These cutting-edge algorithms allow for the reliable and effective categorization of photos based on distinguishing patterns and symptoms linked to different plant diseases.
- The suggested approach protects the quality and productivity of betel leaf production, which helps to ensure food security. It guarantees a steady and consistent supply of betel leaves, which have substantial cultural and economic importance in many civilizations, by reducing crop losses brought on by illnesses.

The rest of this paper is organized as follows. The section II provides both related works and problem statement. The proposed methodology is explained in the section III. The result and discussion is then presented in the section IV, followed by the conclusion in the section V.

2. LITERATURE REVIEW

In 2024, Khanna *et al.* [16] proposed deep learning-based methods for identifying and classifying plant diseases from leaf images without extensive pre-processing or manual feature extraction. The PlaNet model, compared to other CNN models, demonstrates highly efficient performance across various benchmark datasets.

In 2022, Ahmed *et al.* [17] introduced a lightweight transfer learning method for tomato leaf disease detection. They optimized image quality using preprocessing techniques like illumination correction. Their system utilizes a fusion of MobileNetV2 architecture and a classifier network for feature extraction and prediction. Additionally, it incorporates runtime augmentation to prevent data leakage and tackle class imbalance issues.

In 2024, Praveena *et al.* [18] aimed to determine the most effective CNN model for classifying plant leaf species and categorizing them. The proposed basic CNN model, comprising four convolution layers, is evaluated across ten different medicinal leaf species, facilitating accurate classification and identification.

In 2021, Tiwari *et al.* [19] employed deep learning for plant disease detection and classification, utilizing leaf images of diverse resolutions. Their dense convolutional neural network was trained on a vast dataset covering multiple crops and countries across 27 categories. The model underwent thorough evaluation via five-fold cross-validation and testing on unseen data, showcasing proficient and precise classification of plant leaves.

In 2020, Zeng and Li [20] developed a Self-Attention Convolutional Neural Network (SACNN). SACNN comprises a basic network for global feature extraction and a self-attention network for local feature extraction. Extensive experiments demonstrate SACNN's high accuracy on AES-CD9214 and MK-D2 datasets, outperforming state-of-the-art methods. We also analyze the impact of self-attention network parameters, providing insights for future research on recognition mechanisms.

Yogeshwari and Thailambal [21] developed a unique deep convolutional neural network (DCNN) method in 2023. Plant leaf photos are first preprocessed using an Adaptive Mean Adjustment for enhancement and a 2D Adaptive Anisotropic Diffusion Filter for noise reduction. Adaptive Otsu thresholding and Improved Fast Fuzzy C Means Clustering are then used for segmentation. Features obtained using a grey level co-occurrence matrix are first subjected to PCA-assisted dimensionality reduction before being classified using a specially designed DCNN architecture.

A CNN method was presented by Nandhini and Ashokkumar [22] in 2021 to automatically classify four tomato leaf diseases. 6208 photos from the Plant Village database are used in the dataset. An Improved Crossover based Monarch Butterfly Optimization (ICRMBO) technique is suggested to minimize complexity and optimize parameters in order to streamline CNN architecture design. ICRMBO is used to improve the Vgg16 and Inception V3 architectures, improving classification accuracy and training efficiency. CNN, more especially "InceptionResNetV2, were used in 2021 by Krishnamoorthy *et al.* [23] to automatically diagnose rice leaf illnesses. This deep learning method helps with effective disease control by improving illness identification in rice plants through transfer learning.

In 2021, Gajjar *et al.* [24] developed a real-time system for crop disease identification through leaf images using machine learning. Deep convolutional neural network architecture is proposed for disease classification, while a single shot detector is employed for disease identification and leaf localization. This system aims to provide farmers with early detection of crop diseases, enabling timely interventions for disease management and mitigation.

A global average pooling layer was added in 2020 by Yan et al. [25] to improve convergence time and a batch normalization layer was included to minimize parameters in an improved model that leveraged VGG16 for illness diagnosis. Transfer learning is used to speed up training, guarantee effective resource use, and preserve high performance in illness identification.

2.1. Problem Statement

In the field of agriculture, particularly in the cultivation of betel leaves, the prompt detection and classification of diseases stand as pivotal tasks for upholding crop health and ensuring optimal yield. However, conventional disease identification methods heavily rely on manual inspection, rendering the process time-consuming, labor-intensive, and prone to errors [8, 14]. Moreover, the absence of early detection mechanisms may exacerbate disease spread, resulting in substantial crop losses and economic ramifications for farmers. The specific challenges encountered in early betel leaf disease classification encompass the lack of efficient and timely detection methods, limited accuracy and reliability of manual inspection techniques, scalability and resource constraints, and the complexity of disease symptoms. Tackling these hurdles necessitates the development of advanced techniques, including algorithms and deep learning models. These innovations aim to facilitate early and accurate classification of betel leaf diseases, thus mitigating risks and optimizing agricultural productivity.

3. PROPOSED METHODOLOGY

Early betel leaf disease classification involves using machine learning algorithms to identify and classify diseases affecting betel leaves at an early stage. This aids in timely intervention and management to prevent crop losses. The challenges in early betel leaf disease classification include limited labeled data availability, variability in disease symptoms, and the need for robust feature extraction methods. Furthermore, variables like leaf shape and ambient circumstances can make disease detection and classification algorithms much more difficult, necessitating the use of precise and adaptable machine learning models. This study investigated the use of models such as VGGNet and MobileNet, together with deep learning, to evaluate large-scale plant picture datasets in order to detect diseases early on. These techniques present viable paths toward precise categorization, making efficient use of sophisticated convolutional neural networks to tackle this agricultural problem. Figure 1 shows the overall suggested architecture.

3.1. Data Collection

For this study, a dataset comprising 1189 images of betel leaves was collected directly from a betel vine farm in the Kancheepuram district. The images were captured using both a camera and smartphone to ensure comprehensive coverage. The dataset is classified into three distinct classes: pest attack, leaf burn, and healthy leaf. This diverse dataset enables researchers to analyze and develop models for detecting and classifying different types of betel leaf diseases, contributing to the advancement of agricultural research and pest management practices.

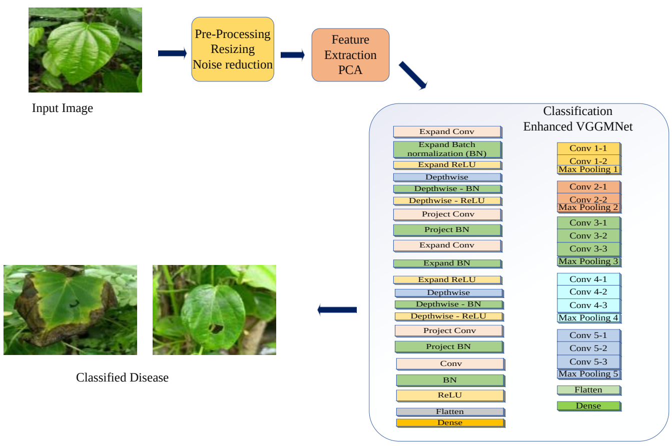


Figure 1: Overall Proposed Architecture

3.2. Pre-Processing

In this paper, pre-processing methods such as resizing and noise reduction are utilized to enhance image quality. Resizing standardizes image dimensions, while noise reduction minimizes irrelevant pixel variations, ensuring optimal input for subsequent analysis, particularly in disease classification of betel plants.

3.2.1. Resizing

Resizing involves adjusting the dimensions of images to a standardized format, which facilitates consistency in the dataset. This standardization ensures that images have the same dimensions, which simplifies subsequent processing steps and improves the efficiency of algorithms. Resizing also helps to reduce computational complexity by reducing the size of images without significantly sacrificing relevant information, thereby optimizing processing time and resource utilization.

3.2.2. Data Augmentation

Data augmentation is a way of purposely increasing the proportion of training datasets. Large datasets are required to implement a deep learning model. A version can be improved by augmenting it with available data. It is correct to assume that the comeback of artificial intelligence is primarily due to the availability of sophisticated computational resources and a large amount of data online. Scaling, twisting, cropping, rotating, padding, and translating are common augmentation techniques. Different augmentation strategies improve performance and promotes the accuracy.

3.2.3. Noise Reduction

Noise reduction is a pre-processing technique aimed at minimizing unwanted variations or disturbances in image data. It involves the application of filters or algorithms to suppress irrelevant pixel fluctuations caused by factors such as sensor imperfections, environmental conditions, or artifacts during image acquisition or transmission. By reducing noise, the clarity and fidelity of the image are enhanced, making it easier for subsequent analysis and interpretation.

3.3. Feature Extraction

In this paper, PCA is employed as a feature extraction technique to distill relevant information from images.

3.3.1. PCA

PCA is a statistical technique designed to compress and minimize the dimensionality of data. PCA preserves important information by converting high-dimensional data into a lower-dimensional space. It accomplishes this by locating major components, or the data's most variable directions. These elements are linear combinations of the basic qualities, oriented orthogonally to one another. To find these principle components, PCA computes the eigenvalues and eigenvectors of the covariance matrix of the data. Principal components are represented by eigenvectors, and the variation explained by each component is indicated by eigenvalues. The maximum degree of variability from the original dataset is preserved in the projection when PCA projects the data onto these components.

1. Covariance Matrix Calculation

   Given a dataset X with n observations and p features, calculate the covariance matrix C is expressed as per Eq. (1).

   $$C = \frac{1}{n-1}(X - \bar{X})^T(X - \bar{X}) \quad (1)$$

2. Eigenvalue Decomposition

   As per Eq. (2), compute the eigenvalues $(\lambda_1, \lambda_2, \ldots, \lambda_p)$ and corresponding eigenvectors $(v_1, v_2, \ldots, v_p)$ of the covariance matrix$C$.

   $$Cv_i = \lambda_i v_i \quad (2)$$

3. Sort Eigenvalues

   Sort the eigenvalues in descending order and rearrange the corresponding eigenvectors accordingly.

4. Choose the Main Elements

   To create the transformation matrix M, select the first k eigenvectors that correspond to the k biggest eigenvalues.

5. Project Information

   To acquire the changed dataset Y, project the original data X onto the new subspace defined by M. PCA maintains the highest variance while reducing the dataset's dimensionality. By projecting the data onto the directions along which the data fluctuates the most, it is able to efficiently capture the crucial information in a lower-dimensional space.

3.4. Classification

VGGNet and MobileNet are used in this study to carry out the categorization. An innovative convolutional neural network architecture called VGGNet, or Visual Geometry Group Network, was created by the University of Oxford's Visual Geometry Group. Its deep and consistent design helped it become well-known, especially for image classification applications. Small, 3×3 convolutional kernels are used throughout the network as the foundation of VGGNet. This homogeneity makes the design easier to understand and permits deeper networks without overly raising the number of parameters. The provided model architecture is a convolutional neural network (CNN) model with multiple layers, including convolutional layers (Conv2D), max-pooling layers (MaxPooling2D), and a dense layer (Dense). The input shape for the model is (224, 224, 3), indicating an image input with dimensions 224×224 pixels and 3 color channels (RGB). The convolutional layers, labeled as block1_conv1, block1_conv2… block5_conv3, perform feature extraction from the input images at different levels of abstraction. Max-pooling layers reduce the spatial dimensions of the feature maps while retaining important features. The flatten layer transforms the multi-dimensional feature maps into a one-dimensional vector, ready for input to the dense layer. The dense layer, labeled as dense, serves as the output layer of the model, with 3 units representing the classes or categories in the classification task.

Convolutional neural network architecture MobileNet was created for effective inference on portable and embedded platforms. Depthwise separable convolutions, which split the conventional convolution operation into two phases: depthwise convolution and pointwise convolution are used in this process.

1. Depthwise Convolution

Depthwise convolution applies a single filter to each input channel. For a feature map of size $DF \times DF$, with a kernel size of $Dk \times Dk$, M input channels, and N output channels, the computational cost $CM_{depthwise}$ can be calculated as per Eq. (6).

$$CM_{depthwise} = Dk \times Dk \times \alpha M \times \rho DF \times \rho DF \quad (6)$$

2. Pointwise Convolution

Pointwise convolution combines the filtered channels to create new features. The computational cost $CM_{pointwise}$ can be represented as per Eq. (7).

$$CM_{pointwise} = \alpha M \times \alpha N \times \rho DF \times \rho DF \quad (7)$$

3. Total Computational Cost

The total computational cost CM for the core layers of MobileNet can be obtained by adding the depthwise and pointwise convolution costs as per Eq. (8).

$$CM = CM_{depthwise} + CM_{pointwise} \quad (8)$$

4. Standard Convolution

For comparison, the computational cost CS of standard convolutions without depthwise separable convolution can be calculated as per Eq. (9).

$$CS = Dk \times Dk \times M \times N \times DF \times DF \quad (9)$$

5. Reduction in Computational Cost

The ratio of the total computational cost of conventional convolutions to that of depthwise separable convolutions, as determined by Eq. (10), may be used to compute the reduction R in computational cost attained by employing depthwise separable convolutions.

$$R = \frac{CM_{standard}}{CM_{depthwise} + CM_{pointwise}} \quad (10)$$

Where, $CM_{standard} = CS$, α represents the width multiplier, ρ represents the resolution multiplier, Dk represents the kernel size, and DF represents the size of the feature map.

A well-liked method for improving neural network models is the Adam optimizer, which was put out by Kingma and Ba. By independently adjusting learning rates for each parameter, it outperforms classical stochastic gradient descent (SGD). This adaptiveness is achieved through the computation of first and second moments of the gradients. By combining the advantages of AdaGrad and RMSProp, Adam excels in handling sparse gradients and noisy problems commonly encountered in deep learning tasks. It utilizes exponentially decaying moving averages to adjust learning rates, offering faster convergence and better performance compared to SGD, especially in scenarios with non-stationary data and varying gradient magnitudes.

The ADAM method integrates elements of gradient descent and momentum methods. It incorporates two momentum terms: the first moment m_i and the second moment s_i is given as per Eq. (11) and Eq. (12)

respectively. m_i is computed by accumulating exponentially decaying gradients, while s_i accumulates squared gradients. The method adjusts weights w_{i+1} based on these moments, using a learning rate η and two hyperparameters β_1 and β_2 and given using Eq. (13). It also employs bias correction to account for initializations. The updated weights are determined by the ratio of m_i and s_i with respect to $1-\beta_1$ and $1-\beta_2$ respectively.

$$m_i = \beta_1 m_{i-1} + (1-\beta_1)\frac{\partial c}{\partial w} \tag{11}$$

$$s_i = \beta_2 s_{i-1} + (1-\beta_2)\left(\frac{\partial c}{\partial w}\right)^2 \tag{12}$$

$$w_{i+1} = w_i - \eta\frac{\hat{m}_i}{\sqrt{\hat{s}_i+\epsilon}} \tag{13}$$

Where, $\hat{m}_i = \frac{m_i}{1-\beta_1}$ and $\hat{s}_i = \frac{s_i}{1-\beta_2}$.

4. RESULT AND DISCUSSION

The proposed model is implemented using Python platform. The proposed improved VGGNet and MobileNet models are compared against their standard models to assess performance enhancements. The enhancements aim to optimize model performance, offering superior results in tasks such as image classification or object detection. These comparisons provide insights into the effectiveness of the proposed modifications over traditional architectures.

4.1. Overall Performance of Proposed and Existing Model

To improve its performance in image classification applications, the Improved VGGNet model adds more layers and parameters to the regular VGGNet architecture. The inclusion of a flattening layer and a dense layer with a SoftMax activation function for classification are the two main changes made to this model. These modifications enhance the model's capacity to discriminate between various classes and let it to learn more intricate representations of the input data. The Improved VGGNet's greater number of trainable parameters, which enable the model to extract more complex characteristics from the input photos, is one of its primary benefits. With a total of 14,789,955 parameters, the Improved VGGNet has significantly more capacity to learn and represent diverse patterns in the data compared to the general VGGNet, which has 14,714,688 parameters. This increase in parameters enhances the model's ability to generalize well to unseen data and improves its overall performance in classification tasks. Additionally, by introducing trainable parameters in the dense layer, the Improved VGGNet can adapt its internal representations to better suit the specific characteristics of the dataset being used. Because of its adaptability, the model can perform better and attain higher accuracy on a variety of image categorization tasks. Furthermore, the Improved VGGNet reduces the number of non-trainable parameters to 14,714,688, which primarily consists of the convolutional layers and pooling layers.

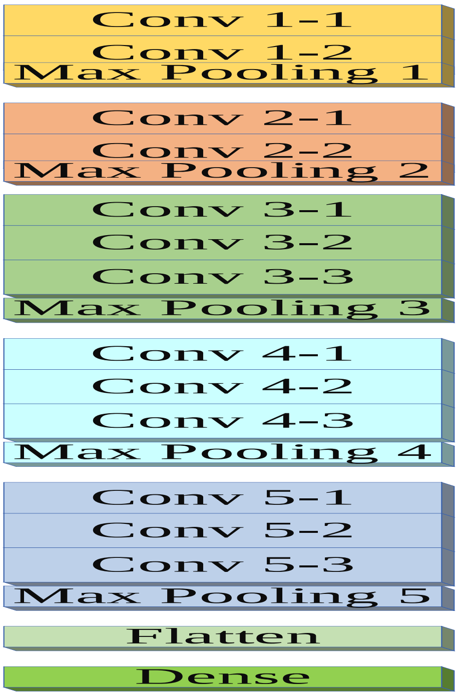
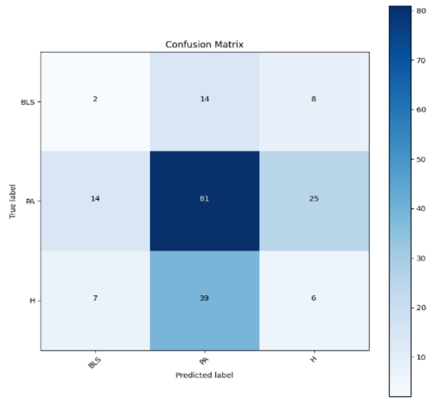


Figure 2: Improved VGGNet with Transfer Learning

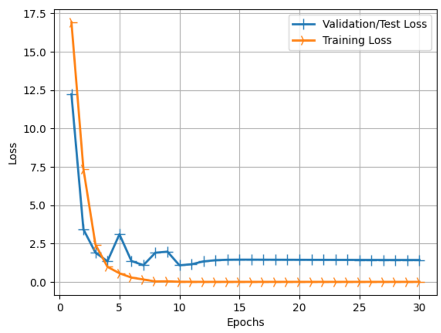
These non-trainable parameters are responsible for extracting hierarchical features from the input images and reducing their spatial dimensions, ultimately providing the dense layer with meaningful representations for classification. Overall, the Improved VGGNet demonstrates superior performance compared to the standard VGGNet by leveraging a larger number of trainable parameters and introducing additional layers for improved feature learning and classification. These enhancements result in higher accuracy and better generalization, making the Improved VGGNet a powerful model for various image classification tasks.

Fig. 3 illustrates key performance metrics of the Improved VGGNet model. Subfigure (a) presents the confusion matrix, depicting the model's classification accuracy across different classes. Subfigure (b) showcases the training and testing loss over epochs, indicating the convergence of the model during training. The training and testing accuracy are shown in Subfigure (c), which illustrates how effectively the model generalizes to previously encountered data. All things considered, these visualizations shed light on how well the model performs and how successful it is at the categorization task.

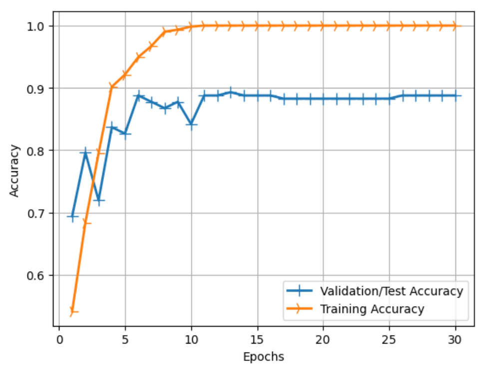
Fig. 4 presents image representations used in the analysis. Subfigure (a) displays the original image, while Subfigure (b) showcases the image with selected features. The selected features highlight specific regions or patterns identified by the model as significant for classification or analysis. This visualization aids in understanding the model's focus and the features it prioritizes in its decision-making process.



(a)



(b)

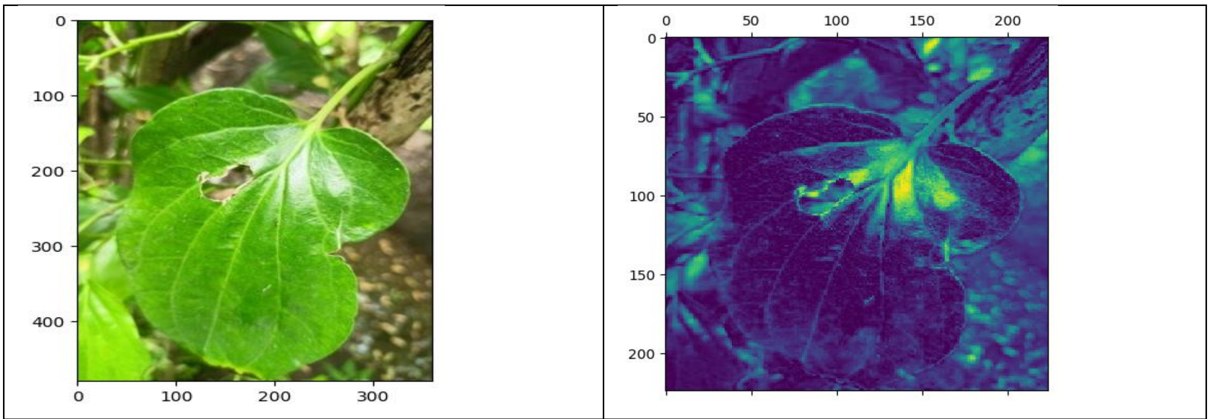


(c)

Figure 3: Graphical Representation of Improved VGGNet(a) Confusion Matrix (b) Training vs. Testing Loss (c) Training Accuracy vs. Testing Accuracy

Table 2: Performance Metrics of VGGNet

Metrics	**Leaf_burn**	**Pest_attack**	**Healthy**
Sensitivity	0.89	0.84	0.78
Precision	0.86	0.93	0.72
Specificity	0.87	0.87	0.92
Recall	0.86	0.72	0.78
Accuracy	0.90	0.92	0.94



(a) (b)

Figure 4: Image Representation (a) Original Image (b) Feature Selected Image

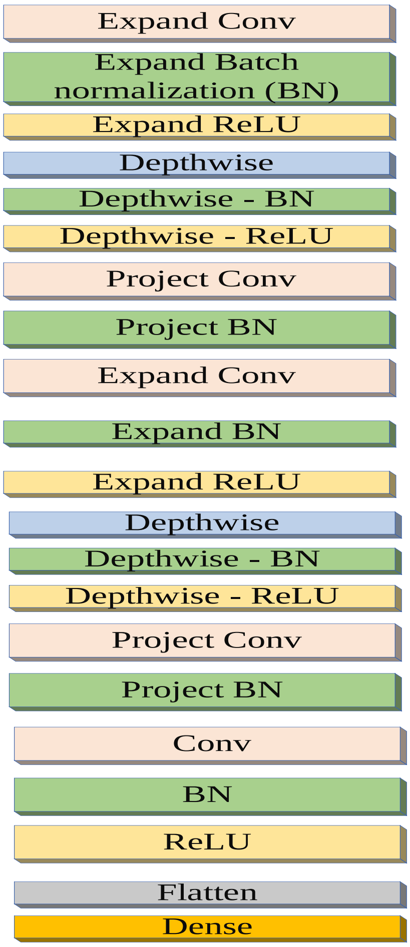


Figure 5: Improved Mobilenetv2

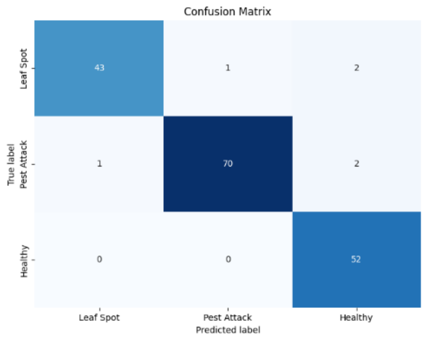
Fig. 5 depicts the improved MobileNetV2 architecture introduces modifications to enhance its performance. With fewer parameters and increased computational efficiency, it achieves comparable results to its predecessor. Notably, the addition of residual connections, as seen in block_14_add and block_15_add, facilitates gradient flow and aids in deeper network training. The expansion and depthwise convolution layers, such as block_13_expand, block_14_expand, and block_15_expand, increase the receptive field without significantly increasing the computational cost. Batch normalization layers ensure stable training by normalizing the input to each layer, preventing the vanishing gradient problem. The final Conv_1-layer projects feature onto a 1280-dimensional space, followed by a global average pooling layer, out_relu, which aggregates spatial information. The addition of a dense layer, dense, with 3 output units, allows for classification into three classes. Unlike the original MobileNetV2, the enhanced version has an additional layer called flatten that converts the output into a one-dimensional vector to make it more compatible with the next dense layer. This improves classification performance by allowing the model to discover intricate patterns and correlations in the data. Improved MobileNetV2 strikes a compromise between model complexity and efficiency with 2446147 total parameters and 188163 trainable parameters. It reduces overfitting and improves generalization to new data by decreasing the number of trainable parameters while maintaining representational capacity. The enhancements made to MobileNetV2 overall help it perform better on a variety of computer vision tasks, such as semantic segmentation, object identification, and picture categorization. It is a good option for contexts with limited resources, such mobile and embedded devices, because of its lightweight design and competitive performance.

Output: Neural network training over 10 epochs

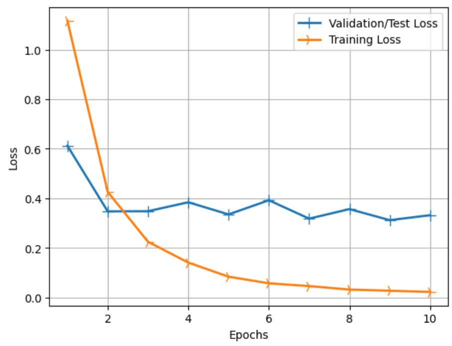
Epoch 1/10
10/10 [==============================] - 57s 5s/step - loss: 1.1164 - accuracy: 0.6098 - val_loss: 0.6123 - val_accuracy: 0.7755
Epoch 2/10
10/10 [==============================] - 46s 5s/step - loss: 0.4258 - accuracy: 0.8426 - val_loss: 0.3472 - val_accuracy: 0.8673
Epoch 3/10
10/10 [==============================] - 42s 4s/step - loss: 0.2245 - accuracy: 0.9246 - val_loss: 0.3478 - val_accuracy: 0.8878
Epoch 4/10
10/10 [==============================] - 44s 5s/step - loss: 0.1403 - accuracy: 0.9525 - val_loss: 0.3840 - val_accuracy: 0.8878
Epoch 5/10
10/10 [==============================] - 42s 4s/step - loss: 0.0833 - accuracy: 0.9770 - val_loss: 0.3350 - val_accuracy: 0.8776
Epoch 6/10
10/10 [==============================] - 42s 4s/step - loss: 0.0566 - accuracy: 0.9984 - val_loss: 0.3921 - val_accuracy: 0.8878
Epoch 7/10
10/10 [==============================] - 45s 5s/step - loss: 0.0455 - accuracy: 0.9967 - val_loss: 0.3179 - val_accuracy: 0.8980
Epoch 8/10
10/10 [==============================] - 42s 4s/step - loss: 0.0314 - accuracy: 1.0000 - val_loss: 0.3564 - val_accuracy: 0.8827
Epoch 9/10
10/10 [==============================] - 39s 4s/step - loss: 0.0267 - accuracy: 1.0000 - val_loss: 0.3117 - val_accuracy: 0.9031
Epoch 10/10
10/10 [==============================] - 43s 4s/step - loss: 0.0217 - accuracy: 1.0000 - val_loss: 0.3318 - val_accuracy: 0.9031

The resultant output displays the training process of a neural network model across 10 epochs. Each epoch consists of batches, or iterations, of training data. The model adjusts its parameters at each iteration to lower the loss function and raise accuracy. The model's accuracy at the start of the first epoch is 77.55%, with a somewhat high loss of 0.6123. The loss gradually drops as training goes on, showing that the model is improving its ability to match the training set. Accuracy rises concurrently, indicating a closer fit between the model's predictions and the real labels. By the ninth epoch, 90.31% of the data is accurate and the loss has dramatically dropped to 0.3318. The model performs better across epochs according to the validation loss and accuracy metrics, which assess the model's performance on untested data. This suggests that the model has good generalization capabilities. All things considered, the model has successfully learned to categorize the input data as evidenced by the declining loss and rising accuracy.

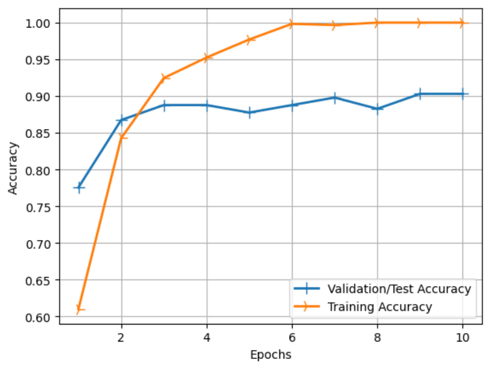
Fig. 6 illustrates key metrics of the Improved MobileNet model. (a) The Confusion Matrix displays the model's performance across different classes. (b) The graph of Training vs. Testing Loss indicates the model's convergence and generalization ability. (c) Training Accuracy vs. Testing Accuracy graph showcases how well the model learns from training data and generalizes to unseen data". These visualizations help assess the model's performance, identifying any overfitting or underfitting issues, and provide insights into its overall effectiveness in classification tasks.



(a)



(b)



(c)

Figure 6: Graphical Representation of Improved MobileNet (a) Confusion Matrix (b) Training vs. Testing Loss (c) Training Accuracy vs. Testing Accuracy

Table 1: Performance Metrics of MobileNet

Metrics	**Leaf_burn**	**Pest_attack**	**Healthy**
Sensitivity	0.93	0.77	0.66
Precision	0.82	0.93	0.76
Specificity	0.88	0.97	0.97
Recall	0.93	0.77	0.66
Accuracy	0.90	0.92	0.93

Table 1 showcases performance metrics for three distinct leaf conditions: Leaf burn, Pest attack, and Healthy. These metrics offer valuable insights into a classification model's efficacy in accurately predicting leaf conditions based on specific features. Accuracy, reflecting overall correctness across all classes, indicates strong performance, showcasing excellent leaf condition categorization. Sensitivity, or genuine positive rate, illustrates the model's ability to detect real positive cases. For instance, a sensitivity value of 0.77 for pest attack suggests the model correctly identifies around 77% of actual pest infestations. Specificity measures the model's capacity to identify genuine negative situations, with a high score of 0.97 for Healthy leaves. Precision, vital in scenarios where false positives are costly, denotes the percentage of real positive predictions among all positive ones generated by the model. For Leaf burn, the precision score of 0.82 implies approximately 82% of projected leaf burn occurrences are real positives. Despite high overall accuracy, variations in sensitivity, specificity, and precision across leaf types emphasize the necessity of analyzing multiple metrics to comprehend model performance fully. Fig. 7 offers a graphical overview of the model, aiding in visualizing essential components and connections within it, facilitating a deeper understanding of its structural dynamics and functionality.

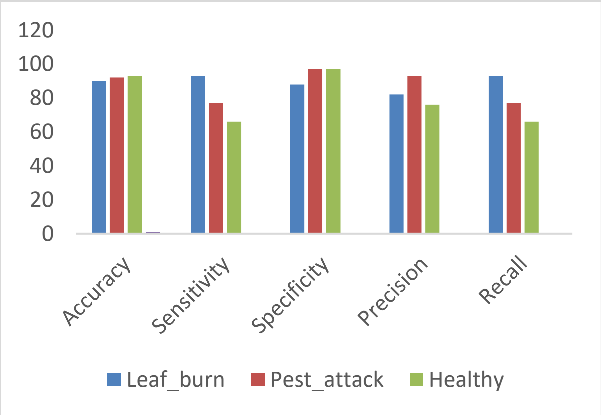


Figure 7: Overall Graphical Representation of Model

5. CONCLUSION

This research explored the possibility of analyzing large plant image datasets through the analysis of deep learning algorithms. This paper encompassed several phases including pre-processing, feature extraction, feature selection, categorization, and data collection. A comprehensive database of plant images portraying bacterial leaf spot, pest attack, and healthy conditions in betel plants was amassed during the data collection process. The image quality was improved by using pre-processing techniques including noise reduction and scaling. To retrieve pertinent information from the images, one feature extraction method that was used was PCA. For categorization, deep learning model like VGGNet and MobileNet are used to provide better performance in image recognition tests. MobileNet model has given better performance when comparing with VGGNet and classified images accurately and successfully by using the retrieved and chosen features. These algorithms allowed quick categorization by recognizing unique patterns and symptoms linked to betel plant illnesses, hence reducing possible economic harm and maintaining food security. The Python platform was used to implement the suggested model. In future betel leaf classification, emphasis is placed on segmentation to enhance accuracy. Segmentation involves partitioning the leaf image into distinct regions, enabling precise analysis of leaf features. By isolating the leaf from its background and identifying regions of interest, such as veins or lesions, segmentation facilitates more accurate classification. Through advanced segmentation techniques like semantic segmentation or instance segmentation, the model can focus on relevant leaf characteristics, leading to improved classification performance and robustness in distinguishing between different betel leaf varieties.

REFERENCES

[1] Trivedi, N.K., Gautam, V., Anand, A., Aljahdali, H.M., Villar, S.G., Anand, D., Goyal, N. and Kadry, S., 2021. Early detection and classification of tomato leaf disease using high-performance deep neural network. Sensors, 21(23), p.7987.

[2] Ramesh, S. and Vydeki, D., 2020. Recognition and classification of paddy leaf diseases using Optimized Deep Neural network with Jaya algorithm. Information processing in agriculture, 7(2), pp.249-260.

[3] Kaur, P., Harnal, S., Tiwari, R., Upadhyay, S., Bhatia, S., Mashat, A. and Alabdali, A.M., 2022. Recognition of leaf disease using hybrid convolutional neural network by applying feature reduction. Sensors, 22(2), p.575.

[4] Upadhyay, S.K. and Kumar, A., 2022. A novel approach for rice plant diseases classification with deep convolutional neural network. International Journal of Information Technology, pp.1-15.

[5] Bansal, P., Kumar, R. and Kumar, S., 2021. Disease detection in apple leaves using deep convolutional neural network. Agriculture, 11(7), p.617.

[6] Uğuz, S. and Uysal, N., 2021. Classification of olive leaf diseases using deep convolutional neural networks. Neural computing and applications, 33(9), pp.4133-4149.

[7] Bedi, P. and Gole, P., 2021. Plant disease detection using hybrid model based on convolutional autoencoder and convolutional neural network. Artificial Intelligence in Agriculture, 5, pp.90-101.

[8] Jadhav, S.B., Udupi, V.R. and Patil, S.B., 2021. Identification of plant diseases using convolutional neural networks. International Journal of Information Technology, 13(6), pp.2461-2470.

[9] Atila, Ü., Uçar, M., Akyol, K. and Uçar, E., 2021. Plant leaf disease classification using EfficientNet deep learning model. Ecological Informatics, 61, p.101182.

[10] Panigrahi, K.P., Das, H., Sahoo, A.K. and Moharana, S.C., 2020. Maize leaf disease detection and classification using machine learning algorithms. In Progress in Computing, Analytics and Networking: Proceedings of ICCAN 2019 (pp. 659-669). Springer Singapore.

[11] Li, K., Lin, J., Liu, J. and Zhao, Y., 2020. Using deep learning for Image-Based different degrees of ginkgo leaf disease classification. Information, 11(2), p.95.

[12] Pham, T.N., Van Tran, L. and Dao, S.V.T., 2020. Early disease classification of mango leaves using feed-forward neural network and hybrid metaheuristic feature selection. IEEE Access, 8, pp.189960-189973.

[13] Kotwal, J.G., Kashyap, R. and Shafi, P.M., 2023. Artificial driving based EfficientNet for automatic plant leaf disease classification. Multimedia Tools and Applications, pp.1-32.

[14] Kaur, P., Harnal, S., Gautam, V., Singh, M.P. and Singh, S.P., 2023. A novel transfer deep learning method for detection and classification of plant leaf disease. Journal of Ambient Intelligence and Humanized Computing, 14(9), pp.12407-12424.

[15] Thangaraj, R., Anandamurugan, S. and Kaliappan, V.K., 2021. Automated tomato leaf disease classification using transfer learning-based deep convolution neural network. Journal of Plant Diseases and Protection, 128(1), pp.73-86.

[16] Khanna, M., Singh, L.K., Thawkar, S. and Goyal, M., 2024. PlaNet: a robust deep convolutional neural network model for plant leaves disease recognition. Multimedia Tools and Applications, 83(2), pp.4465-4517.

[17] Ahmed, S., Hasan, M.B., Ahmed, T., Sony, M.R.K. and Kabir, M.H., 2022. Less is more: Lighter and faster deep neural architecture for tomato leaf disease classification. IEEE Access, 10, pp.68868-68884.

[18] Praveena, S., Pavithra, S.M., Kumar, A.D.V. and Veeresha, P., 2024. CNN-based Indian medicinal leaf type identification and medical use recommendation. Neural Computing and Applications, pp.1-14.

[19] Tiwari, V., Joshi, R.C. and Dutta, M.K., 2021. Dense convolutional neural networks based multiclass plant disease detection and classification using leaf images. Ecological Informatics, 63, p.101289.

[20] Zeng, W. and Li, M., 2020. Crop leaf disease recognition based on Self-Attention convolutional neural network. Computers and Electronics in Agriculture, 172, p.105341.

[21] Yogeshwari, M. and Thailambal, G., 2023. Automatic feature extraction and detection of plant leaf disease using GLCM features and convolutional neural networks. Materials Today: Proceedings, 81, pp.530-536.

[22] Nandhini, S. and Ashokkumar, K., 2021. Improved crossover-based monarch butterfly optimization for tomato leaf disease classification using convolutional neural network. Multimedia Tools and Applications, 80, pp.18583-18610.

[23] Krishnamoorthy, N., Prasad, L.N., Kumar, C.P., Subedi, B., Abraha, H.B. and Sathishkumar, V.E., 2021. Rice leaf diseases prediction using deep neural networks with transfer learning. Environmental Research, 198, p.111275.

[24] Gajjar, R., Gajjar, N., Thakor, V.J., Patel, N.P. and Ruparelia, S., 2021. Real-time detection and identification of plant leaf diseases using convolutional neural networks on an embedded platform. The Visual Computer, pp.1-16.

[25] Yan, Q., Yang, B., Wang, W., Wang, B., Chen, P. and Zhang, J., 2020. Apple leaf diseases recognition based on an improved convolutional neural network. Sensors, 20(12), p.3535.

[26] Dataset taken from: "https://www.kaggle.com/datasets/devias/beetle-data", dated 25-1-2023.