Inception-enabled Vision Transformer (ViT)-based Model for Plant Disease Identification

Sk Mahmudul Hassan

mahmudul.hassan@manipal.edu

Manipal Academy of Higher Education

Kumar Sekhar Roy

Manipal Academy of Higher Education

Ruhul Amin Hazarika

Manipal Academy of Higher Education

Mehbub Alam

Indian Institute of Information Technology Guwahati

Mithun Mukherjee

Nanjing University of Information Science and Technology

Article

Keywords: Plant disease, Machine learning, Vision Transformer, Deep learning

Posted Date: April 21st, 2025

DOI: https://doi.org/10.21203/rs.3.rs-6223674/v1

Additional Declarations: No competing interests reported.

Version of Record: A version of this preprint was published at Scientific Reports on August 23rd, 2025.
See the published version at https://doi.org/10.1038/s41598-025-16142-x.

Inception-enabled Vision Transformer (ViT)-based Model for Plant Disease Identification

Sk Mahmudul Hassan[1*], Kumar Sekhar Roy[1†], Ruhul Amin Hazarika[1†], Mehbub Alam[2], Mithun Mukherjee[3†]

[1*]Manipal Institute of Technology Bengaluru, Manipal Academy of Higher Education, 576104, Karnataka, India.
[2]Department of IT, Indian Institute of Information Technology, Guwahati, 781015, Assam, India.
[3]Nanjing University of Information Science and Technology, China.

*Corresponding author(s). E-mail(s): mahmudul.hassan@manipal.edu;
Contributing authors: sekhar.roy@manipal.edu; ruhul.hazarika@manipal.edu; mehbub@ieee.org; m.mukherjee@ieee.org;
†These authors contributed equally to this work.

Abstract

The timely and precise identification of diseases in plants is essential for efficient disease control and safeguarding of crops. Manual identification of diseases requires expert knowledge in the field, and finding people with domain knowledge is challenging. To overcome the challenge, computer vision-based machine learning techniques have been proposed by the researchers in recent years. Most of these solutions with the standard convolutional neural network (CNN) approaches use uniform background laboratory setup leaf images to identify the diseases. However, only a few works considered real-field images in their work. Therefore, there is a need for a robust CNN architecture that can identify the diseases in plants in both laboratory and real-field conditioned images. In this paper, we have proposed an Inception-enabled vision transformer (ViT) architecture to identify the diseases in plants. The proposed Inception-enabled ViT architecture extracts local as well as global features, which improves feature learning. The use of multiple filters with different kernel sizes efficiently use computing resources to extract relevant features without the need for deeper networks. The robustness of the proposed architecture is established by hyper-parameter tuning and comparison with state-of-the-art. In the experiment, we consider five datasets with both laboratory-conditioned and real-field conditioned images. From the experimental results, we see that the proposed model outperforms state-of-the-art deep

learning models with fewer parameters. The proposed model archives an accuracy rate of 99.17% for the apple leaf dataset, 99.32% for the rice dataset, 96.89% for the ibean dataset, 75.42% for the cassava leaf dataset, and 99.33% for the plantvillage dataset.

Keywords: Plant disease, Machine learning, Vision Transformer, Deep learning

1 Introduction

The global demand for food production is met with some challenges in the form of plant disease, which makes a significant threat to production of crops [1]. Plant diseases are fueled by changing climatic conditions such as temperature. Hence, accurately and timely identifying these diseases is crucial to prevent their spread [2]. Traditional identification relies on visual inspection, which is labor-intensive, lacks precision, and is prone to human error. Researchers have proposed several machine learning (ML)-based approaches to identify the diseases in plants from the leaf images and broadly classify them into traditional ML-based approaches and deep learning (DL)-based approaches. The traditional ML-based approach includes the algorithms such as K-nearest neighbour (KNN), support vector machine (SVM), random forest (RF), and decision tree (DT) [3]. The performance accuracy in this algorithm heavily depends on the extracted features from the images. Finding the set of features from the extracted features that give optimum results is an important challenge in this approach.

1.1 Motivation

CNNs have shown remarkable success in image analysis tasks, making them well-suited for the identification of visual symptoms associated with plant diseases. Their ability to automatically learn hierarchical features from images contributes to highly accurate disease classification. Several deep learning architectures such as, VGG16 [4], VGG19 [4], InceptionV3 [5], ResNet50 [6], MobileNet [7], and EfficientNet [8] are used in the identification of diseases. In recent times, deep learning with self-attention has been used in this field and gives prominent results. Despite the development of architecture and achieving high accuracy, it is still far from being implemented in real field conditions due to various reasons: (a) The number of parameters used in the state-of-the-art deep learning models is large and requires high computing devices to train the model. (b) Real-field images for the agricultural crops are unavailable. Designing a lightweight convolutional neural network (CNN) architecture that can effectively identify diseases in plants is an important area in research. In this view, Dosovitskiy et al. [9] introduced vision transformer (ViT) architecture in image processing and computer vision tasks. This architecture has achieved significant performance in classification with less memory and fewer parameters. In comparison with CNN, ViT architecture heavily relies on model regularization and data augmentation while training on smaller datasets. The main reason is that ViT architecture mainly focuses on extracting global features and long-distance features, whereas CNN architecture

focuses on local features. However, the combination of the CNN with ViT architecture will enhance the feature extraction as the model will extract both local as well as global features. Further, hybridization of both CNN and ViT may increase the performance of the model. In this point of view, a novel lightweight Inception-enabled ViT architecture is proposed to identify the diseases in plants. The model combines features extracted from both Inception CNN and ViT architecture. This model begins with two inception blocks, where we use a parallel convolution filter to extract local features, followed by a stacked transformer block.

1.2 Contribution and Organization of the paper

The main contributions of the proposed architecture are as follows

- An inception-enabled ViT architecture is proposed to identify the diseases in plants with wide and different conditioned images.
- To extract the local features, two inception blocks with parallel convolution are used, which use different filter sizes to extract the features.
- The model is lightweight and uses only 0.90M parameters, which is much less as compared to state-of-the-art deep learning models and can be feasible to implement in agriculture. The proposed model is implemented in different datasets, and the performances are compared with different state-of-art deep learning models. The proposed model outperforms several deep learning-based architectures.

The rest of the paper is organized as follows: Section 2 provides a brief discussion of several existing works. Section 3 provides the details about the proposed models. Results and performance are discussed in Section 4. Finally, the paper concludes in Section 5 with future scope.

2 Related Work

The performance of CNN in computer vision is impressive, and researchers explored and designed several deep-learning models to identify diseases in plants. In this section, we have explored and summarized different deep-learning models used in plant disease detection. Mohanty *et al.* [10] used two different deep learning architectures, AlexNet and GoogleNet, to identify the diseases in plants. In this paper, the authors used a large-scale plant dataset consisting of 54306 images of 38 different categories. Three different types of images were used, namely color, greyscale and segmented images and recorded a maximum accuracy of 99.35%. Later, Ferentinos *et al.* [11] used five different deep learning architectures to classify 58 distinct plant diseases and achieved an accuracy of 99.48% using VGG architecture. Table 1 summarizes the state-of-the-arts.

Thakur *et al.* [12] proposed a lightweight VGG-ICNN model for the identification of plant diseases in multiple plant disease datasets. In this paper, the authors used 4 convolution layers of VGG16 and three blocks of Inception v7. In their paper, the authors recorded an average accuracy rate of 99.16%. Later, a lightweight DenseNet (LWDN) proposed in [13] to identify the diseases in plants achieved an accuracy rate of 99.36% in the plantvillage dataset [10].

Too *et al.* [14] suggested a fine-tuned deep learning architecture to identify different diseases in plants. Using DenseNet architecture [14], they have recorded maximum accuracy. Further, Atila *et al.* [15] used EfficientNet architecture to identify the diseases in plants, and they have compared the performances of the model with several state-of-art deep learning models and showed that EfficientNet architectures outperform other models. Moreover, Sangeetha *et al.* [16] proposed an improved agro deep learning in the identification of panama wilts diseases in banana leaves. This technique used the arrangement of colour and shape changes in banana leaves to forecast the disease's intensity and its effects and achieved an accuracy rate of 91.56%.

CNN, with self-attention, has also gained much attention and is widely used in the identification of plant diseases. Zeng *et al.* [17] proposed a self-attention-based CNN (SACNN) model to identify different crop diseases. The SACNN model consists of the base network to extract global features and self-attention to extract the local features. In their work, different levels of noise were added in the images to evaluate the model performance and show SACNN outperform state-of-art deep learning models. Chen *et al.* [18] proposed a lightweight attention-based deep learning architecture to identify the diseases in rice plants. The authors used the MobileNet-V2 pre-trained on ImageNet as the backbone network, and to improve the learning capability for minute lesion information, they incorporated an attention mechanism. The attention mechanism helps the network understand the significance of spatial points and inter-channel relationships for input features.

Moreover, Qian *et al.* [19] proposed a deep CNN architecture to identify 4 different maize diseases. In this work, the authors divided the CNN architecture into three stages. Stage 1 extracts the image features and encodes them into a feature tokens matrix. Stage 2 is the core computation using multi-head self-attention, and finally, stage 3 is the classification stage. Deep attention dense CNN used by Pandey *et al.* [20] to identify different plant diseases. Mixed sigmoid attention learning merging with basic dense learning used in this work as in dense learning features at higher layer considering all lower layer features that provide efficient training process. Further, attention learning strengthens the learning ability of the dense block. The authors achieved an accuracy rate of 97.55% on a real-time dataset consisting of 17 plant species. Later, Bhujel *et al.* [21] proposed a lightweight self-attention CNN to identify different tomato leaf diseases. The model is proposed based on residual architecture and used 20 convolution layers, and after the 16th layer, they used an attention block.

Moreover, Ghost enlightened transformer (GET) architecture suggested by Lu *et al.* [22] to identify grape diseases and pests. The performances of GeT suppress other deep learning models, achieve an accuracy rate of 98.14%, and are also faster and lighter. To enhance the feature extraction in ViT architecture, Yu *et al.* [23] used inception convolution in ViT architecture to identify the diseases in plants. Four different datasets were used in this work, and the experimental results outperformed those of other deep learning models. Furthermore, A combination of CNN and ViT was used in the work [24] to identify different diseases in plants. Three different datasets were used to evaluate the performances and show that fusion of attention with CNN blocks compensates the speed of the architecture. Mobile device compatible, PMVT a light weight transformer based architecture used in [25] to identify the diseases in

Table 1: Summarization of the existing work.

Paper Ref.	DL Model	Dataset	Class	Accuracy (%)
Mohanty *et al.* [10]	AlexNet, GoogleNet	PlantVillage [10]	38	99.34
Ferentinos *et al.* [11]	AlexNet, VGG, Overfeat, GoogleNet	PlantVillage [10]	58	99.48
Geetharamani *et al.* [26]	AlexNetOWTBn Nine layer CNN	PlantVillage [10]	38	96.46
Chen *et al.* [27]	VGGNet with two inception layer	Maize dataset [10]	4	84.25
Sethy *et al.* [28]	11 state-of-art CNN architecture with SVM for classification	Rice dataset [28]	4	98.38
Too *et al.* [14]	Fine tune 6 different CNN models	PlantVillage [10]	38	99.76
Atila *et al.* [15]	EfficientNet	PlantVillage [10]	38	99.38
Zeng *et al.* [17]	Self-attention CNN with Residual Connection	AES-CD9214 MK-D2 dataset [29]	6	95.59
Qian *et al.* [19]	Transformer and Multi-head attention	Maize dataset [10]	4	98.7
Pandey [20]	DADCNN-5	PlantVillage [10]	38	99.93
Bhujel *et al.* [21]	CNN with Multiple attention	Tomato leaf [10]	10	99.69
Lu *et al.* [22]	GET	GLDP12k dataset [22]	11	98.14
Yu *et al.* [23]	ViT architecture	Ibean [30]	3	99.22
Borhani *et al.* [24]	ViT architecture	Wheat rust [31]	3	100

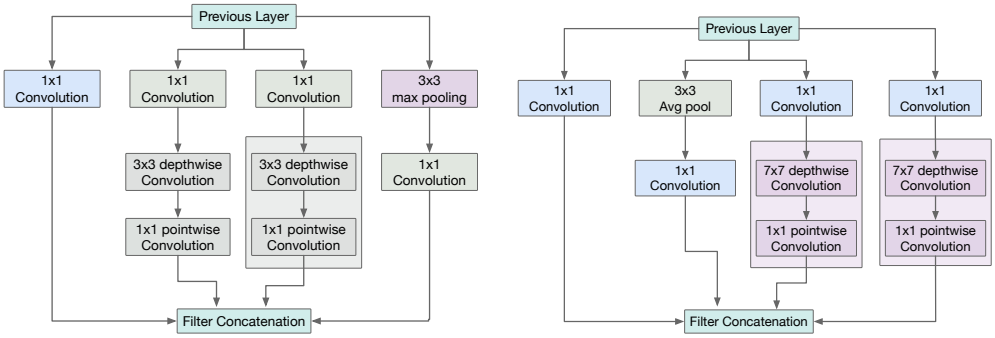
plant. In this paper, the author replaces the convolution block in MobileViT with an inverted residual structure and also incorporates CBAM into the ViT encoder. Multiple datasets were used to evaluate the performances of the model, and it achieved 1.6% higher performance than MobileNetV3 and 2.3% in Squeezenet.

3 Proposed Method

In this paper, we aim to design a lightweight deep learning architecture to identify the diseases in plants and which can also be easily deployable in agriculture. In particular, we design a fusion of Inception block and ViT transformer architecture to classify the diseases in plants.

3.1 Inception Block

Inception block in architecture first used in GoogleNet architecture by Szegedy *et al.* [5]. Increasing the layers in the DL architecture may result in overfitting in the model. In inception, it uses multiple filters of different sizes on the same layer. The outputs of all the convolution layers are concatenated together and forwarded to the next layer. The use of multiple filter sizes extracts better features, which increases the performance of the model. In this paper, we have used two inception blocks termed as



(a) InceptionA Block. (b) InceptionB Block.

Fig. 1: Block diagram of Inception architecture.

InceptionA and InceptionB block. The normal convolution used in the inception block was replaced with depthwise separable convolution. The use of depthwise separable convolution reduces the number of parameters in the model. The parameter used in depthwise separable convolution is calculated as

$$S_{\text{depthwise}} = D_K^2 \times M \times D_F^2 + M \times N \times D_K^2 \,. \tag{1}$$

We write the computation cost in standard convolution as

$$\text{Cost} = D_K^2 \times M \times N \times D_F^2 \,, \tag{2}$$

where D_F is the input image dimension, D_K is the kernel dimension, M is the number of channels and N is the number of kernel/filters. The inceptionA block consists of 1×1 convolution, 1×1 convolution followed by one 3×3 depthwise separable convolution, 1×1 convolution followed by two 3×3 depthwise separable convolution and 3×3 maxpooling followed by 1×1 convolution as shown in figure. Similarly, in inceptionB block also we have used 3×3 average pooling and 7×7 depthwise separable convolution as shown in Figure 1.

3.2 Inverted Bottleneck Block (MV2)

The inverted residual structure was first used in MobileNet architecture [32] and adopted in Inception-enabled ViT architecture, which undergoes feature enhancement and feature reduction in convolution. The block diagram of MV2 is shown in Figure 2 and it is seen that MV2 uses three separate convolutions. At first, 1×1 point-wise convolution is used to expand low-dimension to high-dimensional feature maps. In next, 3×3 depth-wise separable convolution is used followed by activation function to achieve spatial filtering of the higher-dimensional data. Finally, 1×1 point-wise convolution is used to project back to the low-dimensional subspace. The initial and final feature map is added using a residual connection.

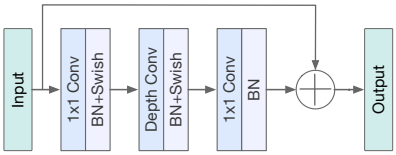


Fig. 2: Block diagram of MV2 block.

We denote X as the input tensor to the inverted bottleneck block, C_{in} represents the number of input channels and C_{out} represents the number of output channels. The first step involves expanding the number of channels by using a 1×1 convolution followed by a non-linear activation function. Let t denote the expansion factor. The output of this layer is expressed as

$$X_{\text{expanded}} = \texttt{Swish}\big(\texttt{Conv2D}(X, C_{\text{in}} \times t, 1 \times 1)\big) \tag{3}$$

Later, depthwise separable convolution is applied of kernel size $K \times K$ is applied with a depth multiplier α on each input channel. The output of this depthwise convolution layer is expressed as

$$X_{\text{depthwise}} = \texttt{DepthwiseConv2D}(X_{\texttt{ecpanded}}, K \times K, \textit{depth}\_\texttt{multiplier} = \alpha) \tag{4}$$

Following the depthwise convolution, a 1×1 pointwise convolution is applied to combine information across channels to reduce the output channel to C_{out}. The output is expressed as

$$X_{out} = \texttt{Conv2D}(X_{\texttt{depthwise}}, C_{\text{out}}, 1 \times 1)\,. \tag{5}$$

Finally, a residual connection is added between the input and output of the block. The output is expressed as

$$X_{\text{out}} = X_{\text{out}} + X\,. \tag{6}$$

3.3 Vision Transformer Block

With the success of transformer architecture in the natural language field, Dosovitskiy *et al.* [9] used transformer architecture in image recognition task and showed that it achieves the same performance accuracy as CNN. The ViT transformer architecture consists of multi-head attention (MHA) [33] layer and multi-layer perceptron (MLP) as shown in Figure 3. Instead of dividing the image into a number of patches followed by linear projection of patches, we pass the input image through the convolutional layer to extract the local features. The local features are divided into a number of patches, and the patches are forwarded to the transformer block. In transformer architecture, MHA and MLP are the main components, which are preceded by one normalization layer and followed by residual connection.

The self-attention mechanism calculates attention scores that represent the importance of each patch in the image. These attention scores are used to compute a weighted

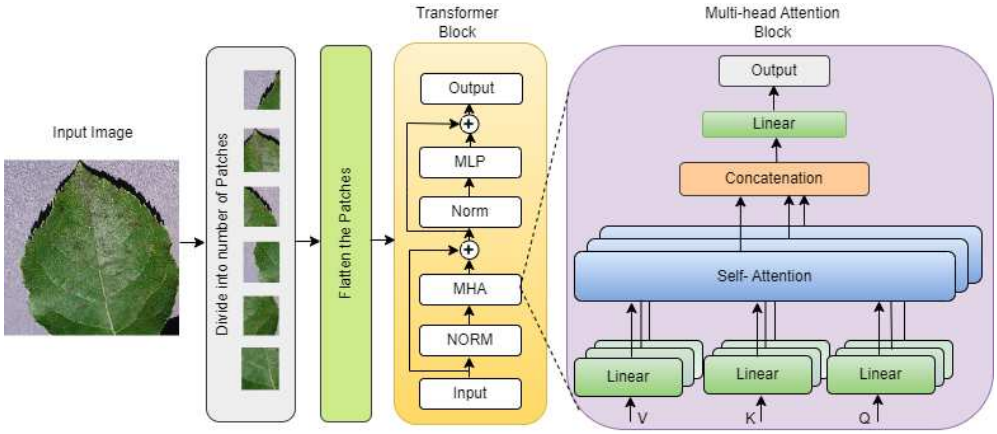


Fig. 3: Architecture of transformer block.

sum of the values (representations) of all patches, producing an attention output. Let's denote the input to the self-attention mechanism as X, where X has dimensions $N \times d$, with N being the number of tokens in the sequence and d being the dimensionality of the token embeddings. The self-attention mechanism computes attention scores as follows:

- Query, Key, and Value Matrices: Three matrices W^Q, W^K and W^V, are learned parameters mapping the input X to query, key, and value spaces, respectively. These matrices have dimensions $d \times d$.
- Query, Key, and Value Projections: Compute query $Q = X.W^Q$, key $K = X.W^K$ and value $V = X.W^V$
- Scaled Dot-Product Attention: Compute the scaled dot-product attention scores

$$\text{Attention}(Q, K) = \texttt{softmax}\left(\frac{QK^T}{\sqrt{d}}\right) \tag{7}$$

- Attention Output: Compute the attention output $A = \text{Attention}(Q, K).V$

The attention output A has dimensions $N \times d$, representing the weighted sum of values.

The ViT architecture consists of a feed forward neural network as MLP separated by a non-linear activation function. The activation function used is Swish. The output is represented as

$$MLP = \texttt{Swish}(A.W + b) \tag{8}$$

where W is the learnable weight matrices and b is the bias.

3.4 Inception-ViT Architecture

The main objective of this work is to hybridize the Inception block with ViT architecture to identify the diseases in the plants. The inception block used in the architecture extracts the local features and the ViT architecture extracts the global feature information. The inception-enabled ViT architecture consists of a Convolution block, an

Table 2: Parameters required in the proposed architecture.

Layer name	Output	Parameter
Input Layer	$256 \times 256 \times 3$	0
Conv2D	$128 \times 128 \times 16$	448
Inception A	$128 \times 128 \times 160$	9472
Inception B	$128 \times 128 \times 192$	53104
MV2	$128 \times 128 \times 16$	7264
MV2	$64 \times 64 \times 24$	1920
MV2	$64 \times 64 \times 24$	3216
MV2	$32 \times 32 \times 48$	4464
ViT	$32 \times 32 \times 80$	270048
MV2	$16 \times 16 \times 80$	28640
ViT	$16 \times 16 \times 96$	493472
Conv2D	$16 \times 16 \times 320$	31040
Global Average Pooling2D	320	0
Dense	4	1605
Total	**-**	**904,693**

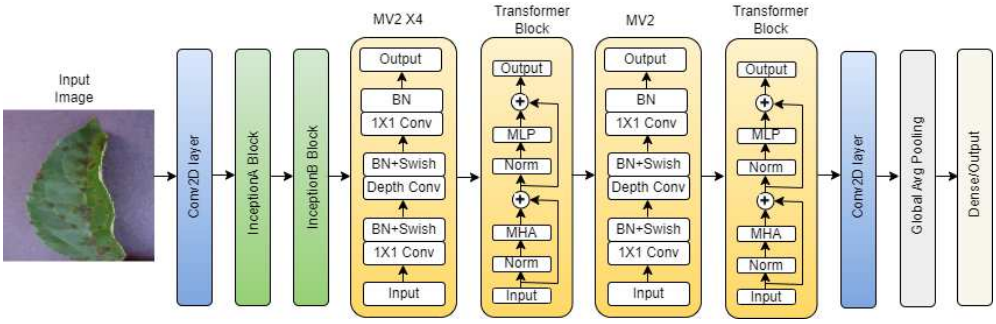


Fig. 4: Block diagram of proposed Inception-enabled ViT architecture.

InceptionA block followed by an InceptionB block, an Inverted bottleneck Block (MV2), Vision Transformer block (ViT), and Global Average Pooling.

The architecture of Inception-ViT is shown in Figure 4. The first layer is the input layer which takes the RGB images of size $256 \times 256 \times 3$ as input. The next layer used is one convolutional layer with filter size 3×3. The output generated in this layer is of size $128 \times 128 \times 16$. The output of the convolution layer is forwarded to InceptionA block followed by InceptionB block for multilevel feature extraction. InceptionA uses filter size of 1×1,3×3 depthwise convolution and 3×3 max-pooling layer as shown in figure 1. The inceptionB block consist of 1×1,7×7 depthwise convolution and 3×3 avg-pooling layer as shown in figure 1b. The output generated after the InceptionB block is of size $128 \times 128 \times 192$. The output of each layer is concatenated together and forwarded as the input of the next layer. Next, a number of MV2 blocks is used, which enhances the feature reduction as well as reduces the number of computations. The output features of the MV2 block are then divided into a number of patches of size $n \times n$.(n = 2,4,8) and passed through the transformer block for global feature extraction. The output of

Table 3: Apple [34] and Rice [28] dataset description.

Apple dataset[34]		Rice dataset[28]	
Disease name	**Number of images**	**Disease name**	**Number of images**
Healthy	515	Bacterial blight	1580
Multiple Disease	91	Blast	1440
Scab	592	Brown spot	1600
Apple Rust	622	Tungro	1308
Total	**1820**	**Total**	**5932**

the transformer block is then passed through a convolution layer and global average pooling layer, which converts the output of convolution layer into 1D vector. Finally, a dense layer is used with softmax activation function and output neuron which is equal to the number of classes in the dataset. A brief description of the parameter used along with the output size of each layer is shown in Table 2. For an instance of 5 classes, the required parameter is 904,693, which varies with regard to the number of output classes in the dataset. The size of the required parameter is 3.72MB. The activation function used in convolution block is ReLu and in MV2 and transformer block is Swish.

4 Experimental Results and Analysis

4.1 Dataset

Five open-source publicly available dataset is used to evaluate the performance of the proposed model. The dataset used are Apple leaf image dataset [34], rice disease dataset [28], ibean dataset [30], plantvillage dataset [10], Cassava dataset [35]. The images in the dataset are of different categories, such as laboratory-conditioned images, field images, images with multiple leaves and images with complex backgrounds. The images used in this work is resized into 224×224. The purpose of using multiple dataset is to evaluate the robustness of the proposed model.

Apple leaf image dataset [34] is the first dataset that is used in this work, which is a freely available dataset and consists of 1820 images. Table 3 gives the brief dataset description along with number of classes. Figure 5 shows the sample images from the dataset.

The second dataset used is the rice leaf diseases dataset [28], which consists of 5932 images of four categories of rice diseases. The images in this dataset were captured in real field-conditioned image. Figure 6 shows the sample images of the dataset and Table 3 shows the description of the dataset.

The third dataset used is ibean dataset [30], which consists of three classes of images. The images in this dataset were captured in real-time field conditioned and multiple leaves were present in single images. Table 4 and Figure 7 show the dataset description and sample images from the dataset.

The fourth dataset used is the cassava dataset [35], which consists of one healthy and four disease-class images. The images in the dataset were captured with complex backgrounds. Figure 8 shows the sample images, and Table 4 shows the detailed dataset description.

(a) Healthy. (b) Multiple Disease. (c) Apple Rust.

Fig. 5: Sample images from Apple Leaf Dataset [34].

(a) Rice blast. (b) Brown spot. (c) Tungro.

Fig. 6: Sample images from Rice Leaf dataset [28].

Table 4: Ibean [30] and Cassava [35] dataset description.

Ibean dataset[30]		**Cassava dataset[35]**	
Disease name	**Number of images**	**Disease name**	**Number of images**
Angular leaf spot	432	Cassava mosaic disease (CMD)	2658
Bean Rust	436	Cassava bacterial blight (CBB)	466
Healthy	428	Cassava green mite (CGM)	773
		Cassava brown streak disease (CBSD)	1443
		Healthy	316
Total	**1296**	**Total**	**5656**

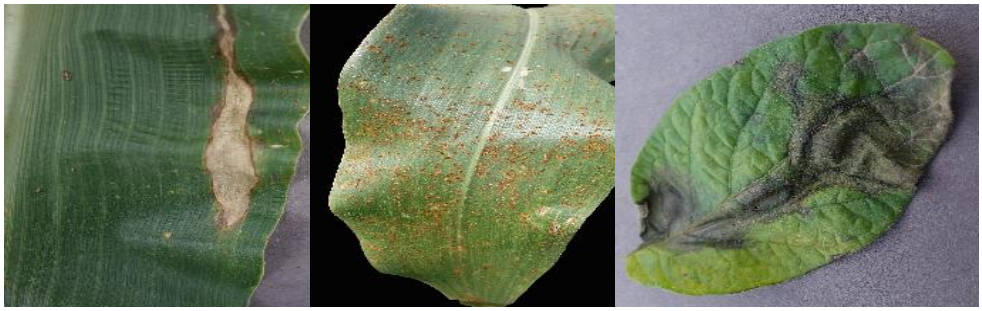
The images in the fifth dataset are from the plantvillage dataset [10]. PlantVillage dataset consists of 54305 images and 38 different categories of diseases. In this work, we have considered only 7 categories of potato and corn images. Figure 9 contains the sample images, and Table 5 summarizes the detailed dataset information.

(a) Healthy. (b) Leaf spot. (c) Ibean Rust.

Fig. 7: Sample images from ibean dataset [30].

(a) CBB. (b) CGM. (c) CMD.

Fig. 8: Sample images from Cassava dataset [35].



(a) Corn leaf blight. (b) Common rust. (c) Late blight.

Fig. 9: Sample images from PlantVillage dataset [10].

Table 5: PlantVillage (Corn & Potato) dataset [10] description.

Disease name	Number of images
Corn Gray leaf spot	513
Corn Common rust	1192
Corn healthy	1162
Corn Northern Leaf Blight	985
Potato Early blight	1000
Potato Healthy	152
Potato Late blight	1000
Total	**6004**

4.2 Evaluation Matrics

Performance evaluation matrices determine how effectively the proposed model classifies the images. In this paper, we have used several performance matrices such as accuracy, sensitivity, specificity, precision, False Positive Rate (FPR), False Negative Rate (FNR), f1-score, and Matthews Correlation Coefficient (MCC). The proportion of accurately predicted images to all images is called accuracy.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{TN} + \text{FN}} \tag{9}$$

Precision is defined as the proportion of true predictions to the total number of positive predictions of the model.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{10}$$

Sensitivity is defined as the proportion of positive classes classified as positive to the total number of positive classes.

$$\text{Sensitivity} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{11}$$

FPR is the ratio of false predictions with negative values.

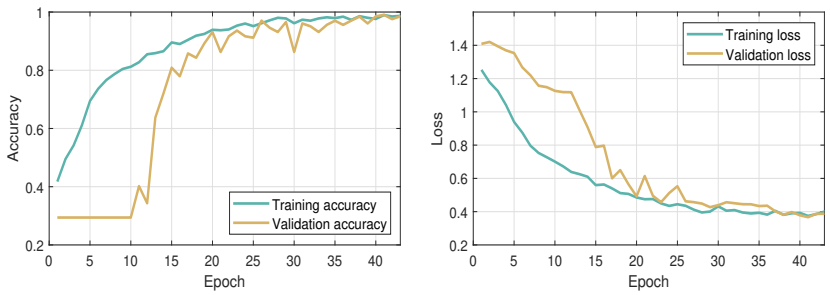
$$\text{FPR} = \frac{\text{FP}}{\text{FP} + \text{TN}} \tag{12}$$

FNR is the ratio of false negative values with positive values.

$$\text{FNR} = \frac{\text{FN}}{\text{FN} + \text{TP}} \tag{13}$$

F1-score is defined as the harmonic mean of precision and recall.

$$F1 - score = 2 \left(\frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \right) , \tag{14}$$



(a) Accuracy. (b) Loss.

Fig. 10: Performance on Apple Leaf dataset [34].

Table 6: Performance of Inception-enabled ViT architecture on different datasets.

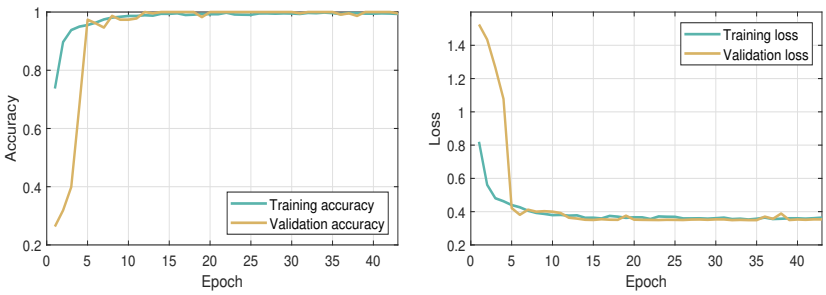
Dataset	Training Accuracy	Training Loss	Validation Accuracy	Validation Loss
Apple [34]	0.9951	0.3728	0.9917	0.3732
Rice [28]	1.0000	0.3328	0.9932	0.3528
Ibean [30]	0.9949	0.3028	0.9689	0.4028
Cassava [35]	0.9147	0.4228	0.7542	0.6328
PlantVillage [10]	0.9951	0.1347	0.9933	0.1836

where TP is true positive, which is defined as the correctly predicted positive along with the original class as positive. FP indicates false positive, which is defined as images supposed to be positive but predicted negatively. TN is true negative, indicating images are negative and predicted as negative. FN is false negative, indicating images are negative but predicted as positive.

4.3 Results and Discussion

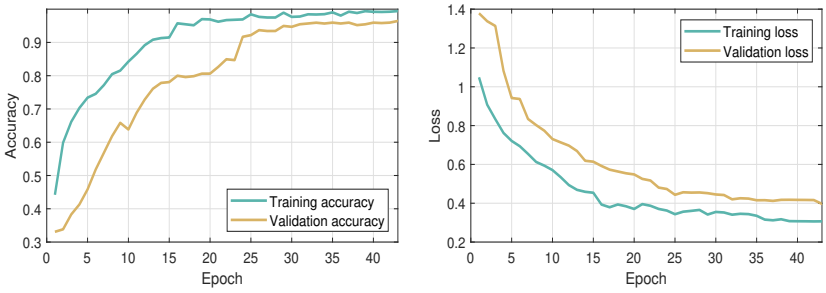
In this section, we evaluate the performance of the proposed architecture to identify diseases in plants without specifying the disease class. We consider the following five different publicly available plant disease datasets: apple leaf dataset [34], rice leaf dataset [28], ibean dataset [30], cassava dataset [35] and plantvillage dataset [10]. We set the learning rate of the proposed model as 0.001, the batch size as 32, and the training epoch as 50. Firstly, the performance of the model is evaluated in terms of accuracy and loss. Table 6 presents the training and validation accuracy, training and validation loss on five datasets with 50 epochs. Figure 10, Figure 11, Figure 12, Figure 13 and, Figure 14 shows the progression of accuracy and loss on each dataset using the Adam optimizer after 50 epochs. From the figures, we observe that the accuracy increases and loss decreases with respect to epochs. From the accuracy and loss curve, we find that that after a certain number of epochs, the accuracy and loss stabilize, thereby optimum results are achieved.

We have also investigated the model performance with other key performance matrices such as sensitivity, specificity, precision, FPR, FNR, f1-score, and mathews



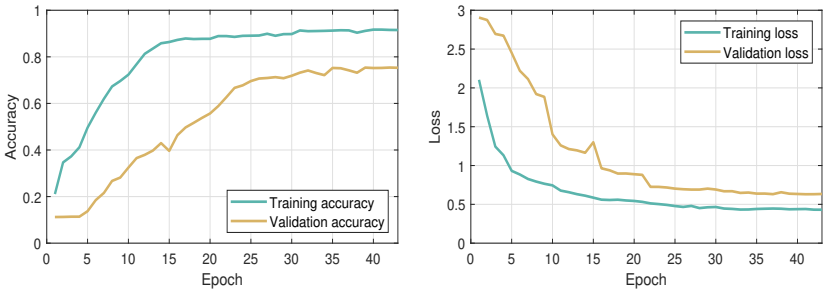
(a) Accuracy. (b) Loss.

Fig. 11: Performance on Rice dataset [28].



(a) Accuracy. (b) Loss.

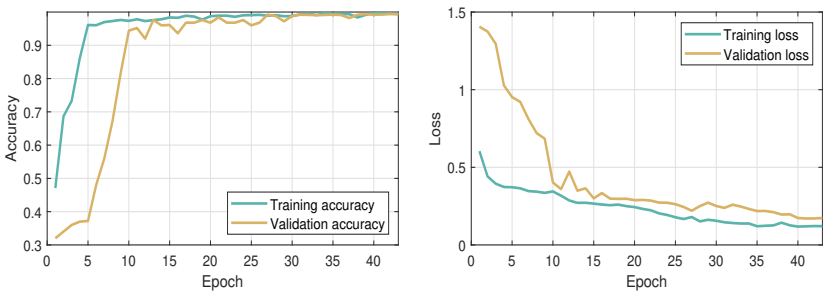
Fig. 12: Performance on Ibean leaf dataset [30].



(a) Accuracy. (b) Loss

Fig. 13: Performance on Cassava dataset [35].

correlation coefficient (MCC). Table 7 summarizes the matrices on each dataset. Moreover, the performance of the proposed model on test images is shown in terms of the confusion matrix.

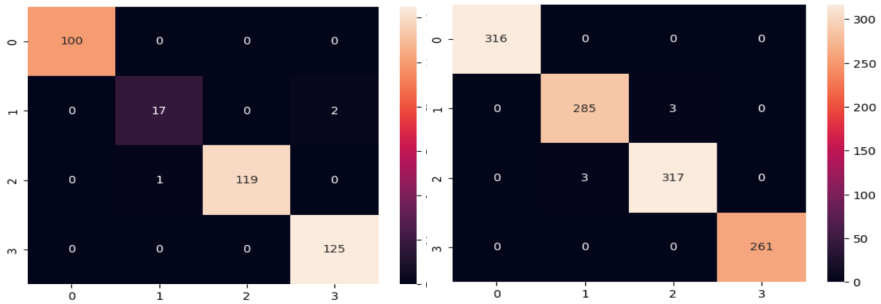


(a) Accuracy. (b) Loss

Fig. 14: Performance on PlantVillage dataset [10].

Table 7: Performance metrics on different datasets.

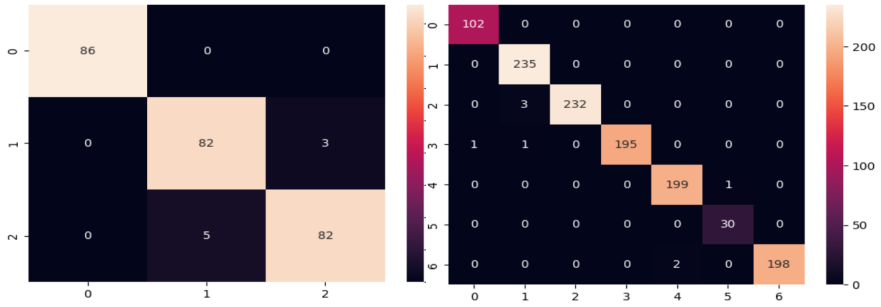
Dataset	Sensitivity	Specificity	Precision	FPR	FNR	F1-score	MCC
Apple [34]	0.9715	0.9971	0.9821	0.0028	0.0270	0.9767	0.9741
Rice [28]	0.9950	0.9983	0.9950	0.0017	0.0049	0.9950	0.9933
Ibean [30]	0.9690	0.9845	0.9690	0.0155	0.0309	0.9690	0.9536
Cassava [35]	0.6717	0.9317	0.7096	0.0682	0.3292	0.6855	0.6237
PlantVillage [10]	0.9945	0.9965	0.9901	0.0011	0.0054	0.9937	0.9912



(a) Apple dataset [34]. (b) Rice dataset [28].

Fig. 15: Confusion matrix.

The confusion matrices of apple [34], rice [28], ibean [30], cassava [35] and plantvillage datasets [10] are drawn and shown in Figure 15, 16 and 17. From the confusion matrix, it is observed that the proposed model has very few false positives and false negatives in all the datasets.



(a) Ibean dataset [30] . (b) Plant dataset [10].

Fig. 16: Confusion matrix.

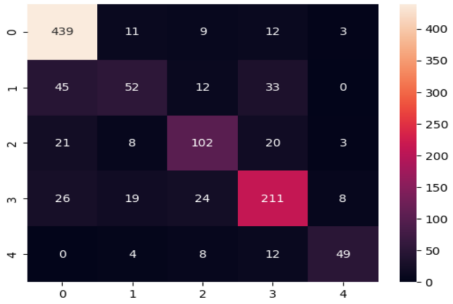


Fig. 17: Confusion matrix of cassava dataset [35].

Table 8: Performance comparison with state-of-the-arts deep learning models.

Evaluation Parameter	Deep Learning Architecture						
	Standard CNN architecture						Inception-ViT
	VGG16	VGG19	InceptionV3	ResNet50	EfficientNetB0	MobileNetV2	
Parameter	138.4M	143.7M	23.9M	25.6M	5.3M	3.5M	0.90M
Accuracy	95.65	98.87	97.27	97.98	99.62	93.52	99.31

4.3.1 Performance Comparison with Different Optimizer

We evaluate and compare the performance of the proposed Inception-ViT architecture with several optimizers to find which optimizer provides the best performance. We select the following optimizers: SGD [36], Adam [37], RMSProp [36], Adamax [36], Adadelta [36], and Ftrl [38]. From Table 9, we observe that Adam and RMSProp optimizer provide the highest performance accuracy. Moreover, we can conclude that the Adam optimizer outperforms others for all the datasets [10, 28, 30, 34, 35].

Table 9: Performance Comparison with Different Optimizers

Optimizer	Training Acc	Training loss	Validation Acc	Validation Loss
Apple Dataset [34]				
SGD	0.4638	0.9856	0.3891	1.0374
RMSProp	0.9982	0.1178	0.9968	0.1251
Adam	0.9951	0.3728	0.9917	0.3732
Adamax	0.8147	0.8556	0.7023	0.8703
Adadelta	0.8238	0.5591	0.7176	0.8390
Nadam	0.8268	0.5621	0.7248	0.8232
Ftrl	0.8562	0.5394	0.7451	0.8132
Rice Dataset [28]				
SGD	0.8692	0.5346	0.7318	0.6146
RMSProp	1.000	0.3393	0.9927	0.3567
Adam	1.000	0.3328	0.9932	0.3528
Adamax	0.9168	0.4285	0.8527	0.5128
Adadelta	0.8571	0.5348	0.8038	0.5793
Nadam	0.8179	0.5249	0.7748	0.6173
Ftrl	0.8027	0.5294	0.7819	0.6123
Ibean Dataset [30]				
SGD	0.4152	1.0348	0.3750	1.0877
RMSProp	0.9826	0.3290	0.9294	0.4241
Adam	0.9949	0.3028	0.9689	0.4028
Adamax	0.4152	1.0348	0.3750	1.0877
Adadelta	0.4152	1.0348	0.3750	1.0877
Nadam	0.4152	1.0348	0.3750	1.0877
Ftrl	0.4152	1.0348	0.3750	1.0877
Cassava Dataset [35]				
SGD	0.7682	0.6251	0.5025	0.7396
RMSProp	0.9046	0.4376	0.7381	0.6451
Adam	0.9147	0.4228	0.7542	0.6328
Adamax	0.7187	0.6429	0.4627	0.7914
Adadelta	0.7097	0.6452	0.4607	0.8014
Nadam	0.7285	0.6104	0.4821	0.7552
Ftrl	0.7047	0.6625	0.4663	0.7936
PlantVillage Dataset [10]				
SGD	0.8732	0.4753	0.8271	0.5129
RMSProp	0.9952	0.1393	0.9918	0.2178
Adam	0.9951	0.1347	0.9931	0.1836
Adamax	0.9263	0.4621	0.8575	0.4726
Adadelta	0.8357	0.4961	0.7817	0.6129
Nadam	0.9136	0.3961	0.8537	0.4327
Ftrl	0.8436	0.4853	0.7971	0.5326

4.3.2 Performance with the Number of Patches

Moreover, to show the effectiveness of patch sizes in the proposed Inception with ViT architecture for the identification of plant diseases, we provide a comparative analysis with different patch sizes. We consider the following patch sizes: 2×2, 4×4, 8×8, and 16×16. From Table 10, we can see that the patch size has less impact on the performance. However, using patch size 8×8 and 4×4 provides a superior performance as compared to patch size 2×2 and 16×16, respectively.

Table 10: Performance Comparison with Different Patch Size after 50 epochs

Patch Size	Training Loss	Training Acc	Validation Loss	Validation Acc
Apple Dataset [34]				
2	0.3741	0.9912	0.3302	0.9874
4	0.3828	0.9914	0.3722	0.9914
8	0.3728	0.9951	0.3732	0.9917
16	0.3875	0.9905	0.3826	0.9841
Rice Dataset [28]				
2	0.3354	0.9942	0.3671	0.9901
4	0.3249	1.0000	0.3521	0.9942
8	0.3228	1.0000	0.3528	0.9932
16	0.3485	0.9912	0.3662	0.9897
Ibean Dataset [30]				
2	0.3334	0.9843	0.4264	0.9579
4	0.3157	0.9904	0.4178	0.9617
8	0.3028	0.9949	0.4028	0.9689
16	0.3394	0.9808	0.4349	0.9496
Cassava Dataset [35]				
2	0.4417	0.8846	0.6579	0.7319
4	0.4259	0.9155	0.6297	0.7552
8	0.4228	0.9147	0.6328	0.7542
16	0.4491	0.8808	0.6592	0.7296
PlantVillage Dataset [10]				
2	0.1507	0.9904	0.2142	0.9902
4	0.1358	0.9947	0.1857	0.9926
8	0.1347	0.9951	0.1836	0.9933
16	0.2268	0.9923	0.1926	0.9917

Table 11: Performance comparison on PlantVillage dataset [10].

Paper Ref.	DL model used	No of class	Performance	Parameter
Mohanthy *et al.* [10]	Transfer learning (GoogleNet)	38	99.34	6.7M
Ferentinos *et al.* [11]	Transfer Learning (VGG16, Overfeat)	58	99.53	138.4M
Waheed *et al.* [39]	Dense CNN	3	98.06	NA
Pandey *et al.* [20]	DADCNN-5	38	99.93	NA
Fang *et al.* [40]	ResNet-50	10	95.61	25.6M
Thakur *et al.* [12]	VGG-ICNN	NA	99.16	6M
Dheeraj *et al.* [13]	LWDN	NA	99.37	1.5M
Proposed	**Inception- enabled ViT**	**7**	**99.31**	**0.90M**

4.4 Comparison with State-of-art Deep Learning Models

In order to verify the robustness of the proposed model, we have compared the performances of the proposed model with several state-of-the-art deep learning architectures.

The performance comparison of the proposed model with different deep learning models has been summarized in Table 8 after 50 epochs. From Table 8, it is observed that the proposed Inception-enabled ViT model outperforms the state-of-the-art deep learning architectures. Table 8 also compares the parameters required of the deep learning models, and it shows that the proposed deep learning model uses fewer parameters. The performance of the proposed model is also compared with the other deep learning models in Table 11. From Table 11, it is noted that the proposed model can successfully classify diseases in plants with higher performance accuracy as compared to the existing works. Hence, it is worthwhile to note that the proposed inception-enabled ViT outperforms state-of-the-art deep learning models. In the future, we plan to implement the proposed approach in IoT-Edge devices to manage datasets for real-time results. Additionally, this approach can be used to different benchmark datasets regardless of the geographic location.

5 Conclusion

In this paper, we proposed an inception-enabled ViT architecture to identify the diseases in plants. The fusion of inception in ViT architecture has the benefit of having both local and global feature extraction, which increases the performance of the model. In fact, the ViT blocks in the proposed model accelerate the training, and the attention model in ViT focuses on the meaningful regions in the input image. An investigation is performed with different patch sizes to achieve the optimal architecture. The results reveal that Inception-enabled ViT with 4 patch size gives the best performance accuracy. The proposed inception-enabled ViT architecture is experimented on five different datasets, which are unbalanced datasets, images in the dataset with complex backgrounds, and images with multiple leaves. The experimental result shows that the model performance achieves impressive results with an accuracy rate of 99.17%, 99.32%, 99.68%, and 99.31% on apple [34], rice [28], ibean [30], cassava [35], and plantvillage datasets [10], respectively.

The performance in the cassava dataset [35] is lower than that of the other dataset. This is because the dataset is unbalanced and the presence of multiple leaves in the single image. Moreover, the images in the dataset are highly correlated to each other. In [41], the authors recorded an accuracy rate of 52.87% and 46.24% using plain and deep residual convolutional neural networks in an imbalanced cassava dataset [35]. In comparison with this, our proposed model achieved a much higher performance accuracy rate in the imbalance dataset. The number of parameters required in the proposed model is much less and can be easily deployable in small devices like smartphones. Furthermore, the likelihood of human error and disease transmission is decreased by quick and automatic identification. Moreover, the presence of agro experts in remote areas is very few; the proposed inception-enabled ViT model provides significant benefits to the farmers to reduce crop yield loss and identify the diseases in plants in an easy manner.

Declarations

Author Contributions The authors contributed to each part of this paper equally.
Conflict of Interest: The authors have no competing interests to declare that are relevant to the content of this article.
Funding Not applicable.
Data Availability The datasets generated and/or analysed during the current study are available in Kaggle repository at:
https://www.kaggle.com/datasets/piantic/plantpathology-apple-dataset
https://www.kaggle.com/datasets/therealoise/bean-disease-dataset
https://www.kaggle.com/datasets/sinadunk23/ behzad-safari-jalal
https://www.kaggle.com/datasets/mohitsingh1804/plantvillage,
https://www.kaggle.com/datasets/nirmalsankalana/cassava-leaf-disease-classification.
Ethical Approval This article does not contain any studies with human participants or animals performed by any of the authors.

References

[1] Islam, M.M., Adil, M.A.A., Talukder, M.A., Ahamed, M.K.U., Uddin, M.A., Hasan, M.K., Sharmin, S., Rahman, M.M., Debnath, S.K.: Deepcrop: Deep learning-based crop disease prediction with web application. Journal of Agriculture and Food Research **14**, 100764 (2023)

[2] Ullah, N., Khan, J.A., Almakdi, S., Alshehri, M.S., Al Qathrady, M., El-Rashidy, N., El-Sappagh, S., Ali, F.: An effective approach for plant leaf diseases classification based on a novel deepplantnet deep learning model. Frontiers in Plant Science **14**, 1212747 (2023)

[3] Kamilaris, A., Prenafeta-Boldú, F.X.: Deep learning in agriculture: A survey. Computers and electronics in agriculture **147**, 70–90 (2018)

[4] Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)

[5] Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2818–2826 (2016)

[6] He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)

[7] Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., Adam, H.: Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861 (2017)

[8] Tan, M., Le, Q.: Efficientnet: Rethinking model scaling for convolutional neural networks. In: International Conference on Machine Learning, pp. 6105–6114 (2019). PMLR

[9] Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al.: An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929 (2020)

[10] Mohanty, S.P., Hughes, D.P., Salathé, M.: Using deep learning for image-based plant disease detection. Frontiers in plant science **7**, 1419 (2016)

[11] Ferentinos, K.P.: Deep learning models for plant disease detection and diagnosis. Computers and electronics in agriculture **145**, 311–318 (2018)

[12] Thakur, P.S., Sheorey, T., Ojha, A.: Vgg-icnn: A lightweight cnn model for crop disease identification. Multimedia Tools and Applications **82**(1), 497–520 (2023)

[13] Dheeraj, A., Chand, S.: Lwdn: lightweight densenet model for plant disease diagnosis. Journal of Plant Diseases and Protection, 1–17 (2024)

[14] Too, E.C., Yujian, L., Njuki, S., Yingchun, L.: A comparative study of fine-tuning deep learning models for plant disease identification. Computers and Electronics in Agriculture **161**, 272–279 (2019)

[15] Atila, Ü., Uçar, M., Akyol, K., Uçar, E.: Plant leaf disease classification using efficientnet deep learning model. Ecological Informatics **61**, 101182 (2021)

[16] Sangeetha, R., Logeshwaran, J., Rocher, J., Lloret, J.: An improved agro deep learning model for detection of panama wilts disease in banana leaves. AgriEngineering **5**(2), 660–679 (2023)

[17] Zeng, W., Li, M.: Crop leaf disease recognition based on self-attention convolutional neural network. Computers and Electronics in Agriculture **172**, 105341 (2020)

[18] Chen, J., Zhang, D., Zeb, A., Nanehkaran, Y.A.: Identification of rice plant diseases using lightweight attention networks. Expert Systems with Applications **169**, 114514 (2021)

[19] Qian, X., Zhang, C., Chen, L., Li, K.: Deep learning-based identification of maize leaf diseases is improved by an attention mechanism: Self-attention. Frontiers in Plant Science **13**, 864486 (2022)

[20] Pandey, A., Jain, K.: A robust deep attention dense convolutional neural network for plant leaf disease identification and classification from smart phone captured real world images. Ecological Informatics **70**, 101725 (2022)

[21] Bhujel, A., Kim, N.-E., Arulmozhi, E., Basak, J.K., Kim, H.-T.: A lightweight attention-based convolutional neural networks for tomato leaf disease classification. Agriculture **12**(2), 228 (2022)

[22] Lu, X., Yang, R., Zhou, J., Jiao, J., Liu, F., Liu, Y., Su, B., Gu, P.: A hybrid model of ghost-convolution enlightened transformer for effective diagnosis of grape leaf disease and pest. Journal of King Saud University-Computer and Information Sciences **34**(5), 1755–1767 (2022)

[23] Yu, S., Xie, L., Huang, Q.: Inception convolutional vision transformers for plant disease identification. Internet of Things **21**, 100650 (2023)

[24] Borhani, Y., Khoramdel, J., Najafi, E.: A deep learning based approach for automated plant disease classification using vision transformer. Scientific Reports **12**(1), 11554 (2022)

[25] Li, G., Wang, Y., Zhao, Q., Yuan, P., Chang, B.: Pmvt: a lightweight vision transformer for plant disease identification on mobile devices. Frontiers in Plant Science **14**, 1256773 (2023)

[26] Geetharamani, G., Pandian, A.: Identification of plant leaf diseases using a nine-layer deep convolutional neural network. Computers and Electrical Engineering **76**, 323–338 (2019)

[27] Chen, J., Chen, J., Zhang, D., Sun, Y., Nanehkaran, Y.A.: Using deep transfer learning for image-based plant disease identification. Computers and Electronics in Agriculture **173**, 105393 (2020)

[28] Sethy, P.K., Barpanda, N.K., Rath, A.K., Behera, S.K.: Deep feature based rice leaf disease identification using support vector machine. Computers and Electronics in Agriculture **175**, 105527 (2020)

[29] Lee, S.H., Chan, C.S., Mayo, S.J., Remagnino, P.: How deep learning extracts and learns leaf features for plant classification. Pattern recognition **71**, 1–13 (2017)

[30] Bean Disease dataset https://www.kaggle.com/datasets/therealoise/bean-disease-dataset. Last accessed: 03-01-2024

[31] Wheat Rust Dataset https://www.kaggle.com/datasets/sinadunk23/behzad-safari-jalal. Last accessed: 07-01-2024

[32] Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L.-C.: Mobilenetv2: Inverted residuals and linear bottlenecks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4510–4520 (2018)

[33] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. Advances in neural

information processing systems **30** (2017)

[34] Plant Pathology Dataset and Apple Leaf data https://www.kaggle.com/datasets/piantic/plantpathology-apple-dataset. Last accessed: 03-01-2024

[35] Mwebaze, E., Gebru, T., Frome, A., Nsumba, S., Tusubira, J.: iCassava 2019 Fine-Grained Visual Categorization Challenge (2019)

[36] Ruder, S.: An overview of gradient descent optimization algorithms. arXiv preprint arXiv:1609.04747 (2016)

[37] Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)

[38] Ahn, K., Zhang, Z., Kook, Y., Dai, Y.: Understanding adam optimizer via online learning of updates: Adam is ftrl in disguise. arXiv preprint arXiv:2402.01567 (2024)

[39] Waheed, A., Goyal, M., Gupta, D., Khanna, A., Hassanien, A.E., Pandey, H.M.: An optimized dense convolutional neural network model for disease recognition and classification in corn leaf. Computers and Electronics in Agriculture **175**, 105456 (2020)

[40] Fang, T., Chen, P., Zhang, J., Wang, B.: Crop leaf disease grade identification based on an improved convolutional neural network. Journal of Electronic Imaging **29**(1), 013004–013004 (2020)

[41] Oyewola, D.O., Dada, E.G., Misra, S., Damaševičius, R.: Detecting cassava mosaic disease using a deep residual convolutional neural network with distinct block processing. PeerJ Computer Science **7**, 352 (2021)