Nanotechnology Perceptions
ISSN 1660-6795
www.nano-ntp.com

Advanced Deep Learning Models for Accurate Citrus Disease Classification: Performance Analysis and Insights

Archna Goyal[1,2], Kamlesh Lakhwani[1]

[1]JECRC University, Jaipur, Rajasthan 303905, India
[2]K R Mangalam University, Gurugram, Haryana 122103, India
Email: goyalarchna511@gmail.com

To avoid significant economic losses, the citrus sector must identify fungal infections early, because a few contaminated fruits can transmit the disease to a full batch during storage and shipment. Time-efficient machine learning is standard. Unfortunately, these strategies cannot improve illness categorization accuracy. To extract hand-crafted traits, they rely heavily on domain knowledge, which reduces accuracy. Recently, deep learning technologies like the deep convolutional neural network (DCNN) have improved citrus disease diagnosis. This is due to their ability to rapidly acquire crucial characteristics from citrus samples. Many deep learning algorithms require thousands of annotated instances to train a generalized mode. We suggest comparing six recent pre-trained deep learning models for citrus disease diagnosis. We employ baseline learning and transfer learning with SDGM, RMS propagation, and ADAM as optimizers to improve citrus sickness detection models such as VGG16, InceptionV3, MobileNetv2, ResNet50, GoogleNet, and DenseNet. A comparison analysis found that MobileNet is the most accurate model (95%). Using a public citrus fruit dataset, we discovered that our system can correctly diagnose diseases from fruit images.

Keywords: Convolutional Neural Network, Deep Learning, Citrus Diseases, Transfer Learning, VGG16, InceptionV3, MobileNetv2, ResNet50, GoogleNet, DenseNet.

1. Introduction

In the field of agribusiness, diseases of fruit products initiate the degradation of the economy, just as large-scale manufacturing affects the economy around the world. Some researchers in the last decade demonstrated the criticality of the quality of fruit products, as it

impacts human wellbeing [1]. Fruit products ought to be the basis of a sound eating regimen. Citrus fruits are a significant product in agriculture, and nearly everybody consumes them consistently [2].Citrus fruits include lemons, oranges, grapes, and tangerines. Various diseases affect citrus fruits, including black spot, greasy spot, canker, and greening, as well as many more. Diseases of citrus fruits are a critical subject that significantly influences the quality and number of yields around the world. The utilization of pesticides by farmers to control various diseases and enhance the production of crops is taking place on a vast scale [3]. Diseases of fruit crops cause significant issues, such as low levels of production and monetary misfortunes, for farmers. Therefore, the detection of diseases and the identification of their severity is a primary need in the agricultural world. Generally, symptoms of disease in citrus fruits are identified with regular monitoring using just the naked eye. This procedure is costly in enormous manors and is less precise. In some countries, farmers hire specialists to identify citrus fruit diseases, and again, this is a costly and tedious task. There is a need for high returns in horticultural enterprises, as well as a better-quality yield of fruit products, if automatic systems are developed to help in the early discovery of infection or diseases in citrus fruit [4]. Many systems have been examined and proposed by analysts in the landscape of artificial intelligence, machine learning, digital image processing, and deep learning for the prediction and classification of citrus infections.

Machine vision platforms are indeed a commercial tool for the evaluation of food standards. All such systems are used to assess production throughout the domain and are used for robotic post-harvest or the early diagnosis of possibly lethal diseases [1]. They are often used in post-harvest processing for the computer-controlled investigation of the fruits' external quality, including the breakneck speed filtering of them together in commercial sections.

Figure 1. Citrus disease leaf and fruit image [9]

Taxonomy of citrus diseases- Plant diseases are the primary source of output losses in the agriculture business, resulting in national economic losses. Citrus is a significant source of nutrients, such as vitamin C, worldwide. Citrus diseases, on the other hand, have had a negative impact on citrus fruit output and quality. Citrus plants such as lemons, oranges,

grapefruit, and limes are affected by a variety of citrus lesions such as anthracnose, greening, scab, and black spot, as seen in Figure 1, and their samples. The following section provides an overview of some citrus diseases [5].

Canker-Microorganism pathogens cause canker disease. This disease spreads easily from leaf to leaf and infects the fruit, leaves, and stems of citrus plants such as grapefruit, lime, and oranges. To combat the illness, copper fungicides are utilized

Black spot-Black spot is a fungal phenotype that was developed using Guignardia citricarpa. This disease attacks citrus trees in areas with subtropical climates, reducing the quality and quantity of fruits

Citrus scab: Scab disease is a significant problem that affects all citrus cultivars. Scab disease affects a variety of citrus leaves, including sour orange, Carrizo orange, rough lemon, and Rangpur lime. Elsinoe fawcettii is the cause of this disease, which is found on grapefruit. Scab pimples on leaves are caused by a mix of host and fungal infections. This pimple is slightly elevated and gradually darkens to a light brown tint

Greening: Greening disease, also referred to as golden dragon illness, is caused by a bacterial pathogen. It is difficult to control and reproduce the greening disease on affected plants. The affected plants produce green, larger, and malformed fruits that are unfit for sale as fresh fruits or juice. Once a plant is afflicted, it cannot be cured and eventually dies Plant care, including weed control.

Contribution of the Paper

The objective of this paper is to develop a deep learning model that classifies the disease according to the severity level and to identify the disease-affected area of the citrus fruit. The proposed model has the ability to recognize and classify the infected areas of citrus fruits. It is a powerful approach for automatically identifying the citrus fruit disease severity and can be further extended to reinforce a unified citrus disease identification system for real-world applications. The current study helps to mitigate and prevent the fruit disease at the initial stages and can be able to control the cost of the disease when safeguarding the surroundings globally

This paper is organized as follows: The Introduction provides an overview of deep learning applications in agriculture, particularly in plant disease diagnosis, and outlines the study's objectives. The Related Works section reviews existing techniques and their limitations, especially in data augmentation and model performance. In Deep Learning Models, various models like DenseNet-121, ResNet50, and EfficientNet variants are discussed regarding their architectures and relevance. The Training, Optimizer, and Learning Method section details the training setup, including optimizers, learning rates, and loss functions used. Materials and Methods describes the datasets, data augmentation techniques, and network's design improvements. Experimental Setup and Implementation covers the implementation details, training procedures, and evaluation metrics. The Results and Discussion section presents the experimental results, compares them with existing models, and discusses the findings. Finally, the Conclusion summarizes the study's contributions, implications for future research, and acknowledges any limitations.

2. RELATED WORKS

In recent years, deep learning technology has been widely used in the field of agriculture, such as in fruit classification and grading [6,7,8], automatic picking, and the diagnosis of diseases and pests [9,10,11,12], and the automatic diagnosis of plant diseases is one of the most active research areas in agriculture. Several deep learning-based techniques for automatic plant disease diagnosis have emerged, which can help farmers reduce the economic losses caused by pests and diseases in farming [13]. In crops, disease symptoms often appear on the leaves; therefore, crop diseases can be automatically detected by applying machine learning techniques to leaf images. For example, Zhang et al. [14] segmented diseased leaf images using K-means clustering and extracted shape and color features from the lesion information. They classified seven cucumber diseases using sparse representation with a presidential recognition rate of 85.7%. Liu et al. [15] proposed a WSRD-Net method for wheat stripe rust detection based on a convolutional neural network (CNN), which can obtain 60.8% average precision (AP) and 73.8% recall rate on a wheat stripe rust dataset. Zhong et al. [16] proposed a three regression, multi-label classification and focal loss function methods based on the DenseNet-121 deep convolutional network to identify apple leaf diseases with over 93% accuracy on 2464 images, including six apple leaf diseases.

Yao et al. [17] used an improved Xception network to classify brown spots and anthracnose of peach, [8]. Janarthan et al. [19] proposed a lightweight, fast, and accurate deep metric learning-based architecture for detecting citrus diseases from sparse data to obtain 95.04% detection accuracy. Deep learning requires many datasets to support the model training. Otherwise, overfitting may occur [20]. The main obstacle to using machine learning in agriculture is the small dataset and the limited number of annotated samples. This becomes more evident when supervised machine learning algorithms that require labeled data are used. The collection of a large amount of plant-disease-related data may have the problem of an uneven distribution of samples. Some diseases may have a small number of samples, which is not enough to train a classification network. Although some public datasets are available, the size of the datasets and categories do not meet the requirements of all applications. Using simple data enhancement methods such as random inversion, deep random flip, increasing contrast, and adding noise [21] can suppress overfitting, but the sample data are still not sufficiently rich, and the image features are less differentiated from the original dataset. Goodfellow et al. [22] proposed a generative adversarial network (GAN) using generators and discriminators against each other. GAN is widely used in the field of computer vision, such as for image super-resolution reconstruction and image defogging [23,4], and can also be used as a data enhancement tool to expand datasets [25]. Using generated images introduces more variability, which can improve the training process of classification networks and increase accuracy.

Ma et al. [26] generated blood cell images using a DC-GAN network to increase data samples and eliminate data imbalance and missing data labels. Cap et al. [27] proposed a LeafGAN by improving CycleGAN using paired datasets to successfully transform healthy leaves into diseased leaves. Xiao et al. [28] successfully generated six types of citrus leaf images using TRL-GAN, an enhanced version of CycleGAN that removes the real scene background from the original images using Mask RCNN. on ResNeXt101 after expanding

the original dataset using the generated images. However, expanding datasets with adversarial networks increases training time, mostly to several days, and generates low-quality images. The resolution of the generated images is often below 512 × 512, which cannot retain more details, and the expanded dataset has limited performance improvement for the classification network.

Karras et al. [29] proposed StyleGAN2 based on StyleGAN. The StyleGAN is a current high-performance, high-resolution image generation framework capable of generating very high-quality images on a wide range of datasets but still requires a large dataset as well as high computational resources and training time [30]. Liu et al. [31] proposed the FastGAN network, which can finish training a complete model in a dozen hours on a single RTX-2080 GPU, by improving StyleGAN2. However, when applied to plant disease sample generation, it produces checkerboard artifacts, loss of details, and insufficiently rich sample data. The performance of the classification model degrades significantly when the training and test data are very different in appearance or originate from different regions, for example, the light of the target in the training data is very strong while the light of the target in the test data is very dark, the image acquisition devices are different, the geographical locations where they were taken are different, and so on.

Mohanty et al. [32] used 54,306 healthy and diseased leaf images from the PlantVillage dataset to train a neural network model for the identification of 26 leaf species. The performance of the model decreased to approximately 31% when tested with a set of plant images taken in the field because the training set of the model was taken in a laboratory environment, and its images had a uniform background. Ferentinos [33] also noted that when the model was trained on images taken in a laboratory environment and tested on images taken in a planting environment, the accuracy of the model decreased from 99.5% to approximately 33%. Therefore, changes in background and shooting conditions can have a serious impact on the performance of the model.

Because the background of the FastGAN2-generated images is not as rich as that of the real captured images, the FastGAN2-generated images and captured images can be regarded as coming from different regions. Because our experiments only used the FastGAN2-generated images as the training set to train the model and the real captured images to test the performance of the model, this poses a classification network performance challenge and requires the classification network to have a high generalization capability.

Here, we propose the FastGAN2 network, which overcomes the checkerboard artifact problem of the FastGAN network, improves the quality of generated images, and enhances the diversity of the generated images for small datasets. We used the generated images only as the training set of the classification network and tested it using images taken but not used for training with the FastGAN2 network. Finally, we tested it on Densenet121, ResNet50, ShuffleNetv2 [34], Mlp-Mixer [35], MobileNetv3 [36], Vision Transformer, Swin Transformer, EfficientNet-B3, EfficientNet-B5 and EfficientNet-B5-pro and achieved an average accuracy of 93.52%. To improve the generalization of the model, this paper proposes an EfficientNet-B5-pro network based on EfficientNet-B5 that uses the adaptive angular margin (Arcface) loss with adversarial weight perturbation (AWP) mechanism. It achieved the highest performance compared to ten classification networks. The main

contributions of this study are as follows:

By redesigning the FastGAN network generator structure and adding small batch standard deviations to the discriminator to eliminate checkerboard artifacts, the improved FastGAN is more suitable for citrus disease and nutritional deficiency (zinc and magnesium deficiency) image generation. It can generate higher-quality and more realistic disease and nutritional deficiency images with higher diversity when trained on a small number of datasets.

The datasets of citrus melanose, citrus nutritional deficiency, and citrus canker leaves were expanded, and the generated images had the phenotypic characteristics of the real data. With a small dataset, a classification network with 93.04% accuracy was trained using only the generated images, which could successfully identify the four types of citrus leaves. The related methodologies and their outcomes are summarized in Table 1.

Table 1. Related Study

Author(s)	Methodology	Issue Addressed	Outcomes
Zhang et al.	K-means clustering, sparse representation	Classification of cucumber diseases	Achieved 85.7% recognition rate
Liu et al.	WSRD-Net (CNN-based method)	Wheat stripe rust detection	Obtained 60.8% AP and 73.8% recall rate
Zhong et al.	DenseNet-121, multi-label classification, focal loss	Apple leaf disease identification	Achieved over 93% accuracy
Yao et al.	Improved Xception network	Classification of peach diseases	Achieved high classification accuracy
Janarthan et al.	Deep metric learning-based architecture	Citrus disease detection	Achieved 95.04% detection accuracy
Goodfellow et al.	Generative adversarial network (GAN)	Data enhancement	Used in computer vision, image super-resolution, image defogging
Ma et al.	DC-GAN	Blood cell image generation	Increased data samples, eliminated data imbalance
Cap et al.	LeafGAN (improved CycleGAN)	Transformation of healthy to diseased leaves	Successfully transformed healthy leaves into diseased leaves
Xiao et al.	TRL-GAN (enhanced CycleGAN)	Citrus leaf image generation	Successfully generated six types of citrus leaf images
Karras et al.	StyleGAN2	High-resolution image generation	Generated high-quality images
Liu et al.	FastGAN	Plant disease sample generation	Finished training in a dozen hours, but produced checkerboard artifacts
Mohanty et al.	Neural network model (trained on PlantVillage dataset)	Identification of 26 leaf species	Performance decreased to ~31% when tested with field images
Ferentinos	Neural network model	Plant disease identification	Accuracy decreased from 99.5% to ~33% when tested on field images
Proposed Study	FastGAN2, EfficientNet-B5-pro, various classification networks	Citrus disease and nutritional deficiency image generation	Achieved 93.52% average accuracy, highest performance with EfficientNet-B5-pro

3. DEEP LEARNING MODELS

Deep learning is a subset of machine learning that involves neural networks with many layers, known as deep neural networks. It mimics the way the human brain processes information, allowing computers to learn from vast amounts of data. These deep neural networks can automatically discover patterns and features in data without the need for manual feature extraction. Deep learning has achieved significant success in various fields, including image and speech recognition, natural language processing, and autonomous

systems, due to its ability to handle large, complex datasets and improve performance as more data is provided. It leverages techniques such as convolutional neural networks (CNNs) for image-related tasks and recurrent neural networks (RNNs) for sequential data, making it a powerful tool for tasks requiring high levels of accuracy and automation. Some of the latest deep learning models used in this paper are as follows:

VGG-16

The VGG-16 model is a convolutional neural network (CNN) architecture that was proposed by the Visual Geometry Group (VGG) at the University of Oxford. It is characterized by its depth, consisting of 16 layers, including 13 convolutional layers and 3 fully connected layers. VGG-16 is renowned for its simplicity and effectiveness, as well as its ability to achieve strong performance on various computer vision tasks, including image classification and object recognition. [10]The model's architecture features a stack of convolutional layers followed by max-pooling layers, with progressively increasing depth. This design enables the model to learn intricate hierarchical representations of visual features, leading to robust and accurate predictions. Despite its simplicity compared to more recent architectures, VGG-16 remains a popular choice for many deep learning applications due to its versatility and excellent performance. Showing Figure. 2 VGG Layer Architecture

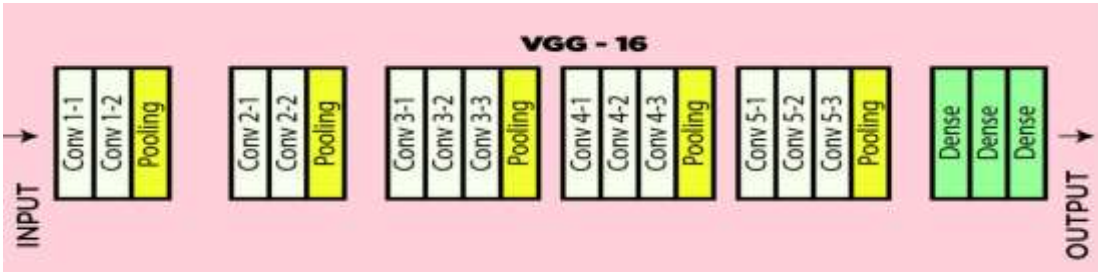

Figure 2. VGG Layer Architecture [1]

The input layer accepts a fixed-size RGB image.

Dimensions:

224x224 pixels with 3 color channels (224x224x3).

Output Layer:

Description:

The output layer is a fully connected layer used for classification.

Dimensions:

1000 units, corresponding to 1000 different classes.

Activation Function:

Softmax, to output a probability distribution over the 1000 classes.

InceptionV3

InceptionV3 is part of the Inception family and is designed to be computationally efficient

while maintaining high accuracy Around 48 layers. It incorporates inception modules, which use filters of different sizes to capture features at various scales. Convolutions of 1x1, 3x3, and 5x5 are used in simultaneously. This method utilizes a mixture of these convolutions. Employs batch normalization and ReLU activation functions.

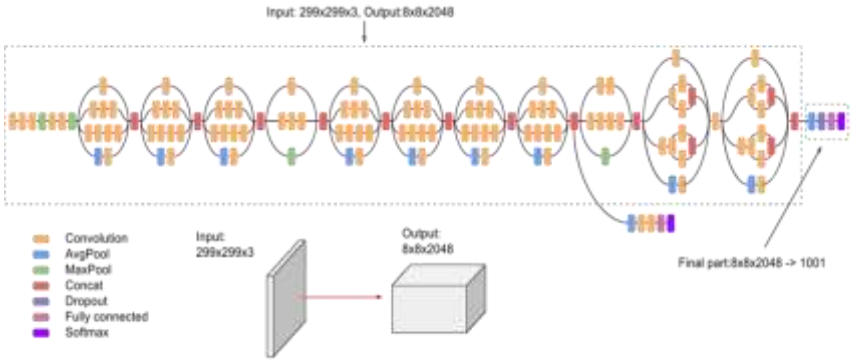

Figure 3. InceptionV3 Layer Architecture [17]

Inception Module:

Output=Concatenate (Conv1x1(input), Conv3x3(input), Conv5x5(input), MaxPooling (input))

Auxiliary Classifier:

Output=Softmax (FC(GlobalAveragePooling(input), units))

ResNet50

ResNet50 (Residual Network) is known for its use of residual blocks it has a 50 layers, which contain shortcut connections that help mitigate the vanishing gradient problem during training [11]. Residual blocks include skip connections, allowing the input to bypass one or more layers. This enables the network to learn residual functions.

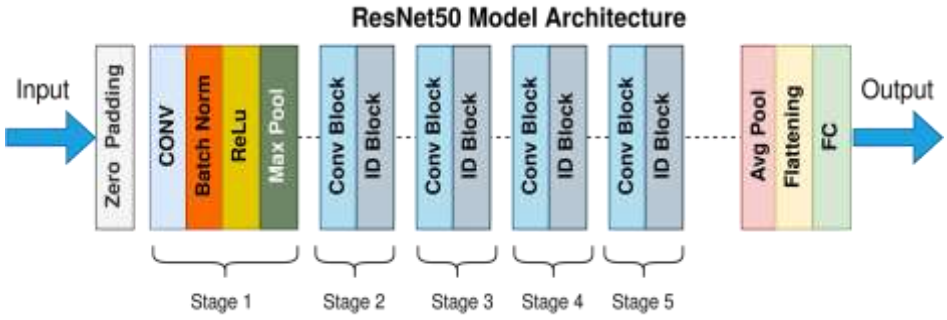

Figure 4. ResNet50 Layer Architecture [2][1]

Residual Block:

Output=ReLU(Add(Conv3x3(input), input))

Global Average Pooling:

Output=Average Pooling (input, pool_size)

Fully Connected Layer (FC):

Output=Softmax(FC(GlobalAveragePooling(input), units))

MobileNet

MobileNet is optimized for mobile and edge devices with limited computational power. It uses depth wise separable convolutions to reduce the number of parameters and computations.

Depth wise Separable Convolution (DWSC)

Depthwise Convolution: Applies a single filter to each input channel.

Depthwise=ReLU(DWConv(input, depth_multiplier,kernel_size,strides,padding))

Pointwise Convolution: Applies a 1x1 convolution to combine the outputs of the depthwise convolution.

Pointwise=ReLU(PWConv (Depthwise,filters,kernel_size))

DenseNet

DenseNet (Densely Connected Convolutional Networks) connects each layer to every other layer in a feed-forward fashion. Each layer receives the feature maps of all preceding layers, which enhances gradient flow and encourages feature reuse, reducing the number of parameters [8].

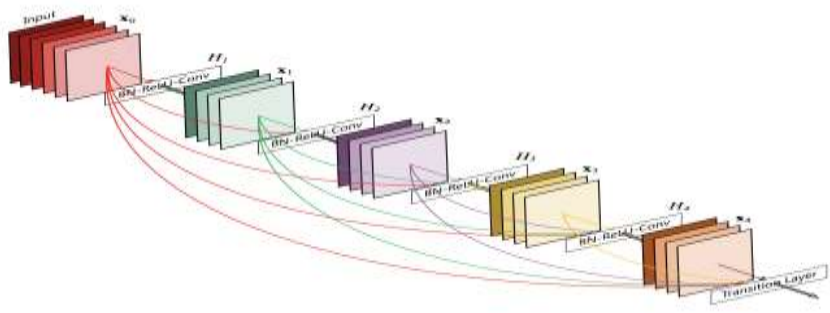

Figure 5. DenseNet Layer Architecture [10]

Dense Block:

Composite Function: Batch normalization followed by ReLU and a 3x3 convolution.

Composite=ReLU (BatchNorm (Conv3x3(input, filters, padding))

Concatenation: The output of each layer is concatenated with the input of the following layer.

Output=Concatenate (input, Composite)

Transition Layer:

Batch Normalization and 1x1 Convolution: Reduces the number of feature maps.

Transition=ReLU (BatchNorm (Conv1x1(input, filters)))

Pooling: Downsamples the feature maps.

Output=Average Pooling (Transition, pool_size)

The details of these deep learning models are summarized in Table 2.

Table 2. Deep Learning Models Details

Model	Image Size	Number of Parameters	Number of Layers
DenseNet	227x227	61 million parameters	18 layers
InceptionV3	299x299	23.85 million	48 layers
MobileNet	224 × 224	4.2 million	28 layers
GoogleNet	224x224	6.6 million	22 layers
ResNet50	224x224	25.6 million	50 layers
VGG16	224x224	138 million	16 layers

4. TRAINING, OPTIMIZER AND LEARNING METHOD

In addition to comparing various deep learning models, this research work includes Baseline training, Transfer learning, and the use of different optimization algorithms such as SGDM, ADAM, and RMSprop. These training methodologies contribute to the inclusive evaluation of model performance and effectiveness in classifying various citrus diseases

Baseline Training: In this technique, the pre-trained models are trained from without leveraging pre-trained weights and other tasks. This permit for an assessment of the models' performance without any aforementioned domain-specific information. Indicate the parameters of the NLP model as θ. The baseline training involves minimizing the following loss function J(θ) during gradient descent:

$$\theta_t+1=\theta_t-\alpha\nabla J(\theta_t) \quad (1)$$

Where α is the learning rate, and ∇J(θt) is the gradient of the loss with respect to the parameters.

Transfer Learning [TL]:TL is process of by before learned models on a divide but related task and then fine-tuning those models such that they are suitable for the depression classification problem at hand. This approach makes use of the information that was acquired during the first training, which has the potential to improve the models' aptitude for comprehending and categorizing material that is linked with depression. TL is process of adjusting a model that has already been trained to perform a new task. The overall objective is to minimize a combined loss function $J_{total}(\theta)$, which is a sum of the pre-trained model's loss pre-trained $J_{pre\text{-}trained}(\theta)$ and the task-specific loss task-specific($J_{task\text{-}specific}(\theta)$:

$$J_{total}(\theta) = (\lambda J_{pre-trained}(\theta) + (1-\lambda) J_{task-specific}(\theta) \quad (2)$$

Where:

Jtotal(θ) is the combined loss function.

$J_{pre-trained}(\theta)$ is the loss of the pre-trained model.

$J_{task-specific}(\theta)$ is the task-specific loss.

λ is a hyperparameter controlling the balance between the two losses.

Where λ is a hyperparameter controlling the balance between the two losses.

Optimization Algorithms:

Stochastic Gradient Descent with Momentum (SGDM): SGDM is an optimization algorithm that combines the advantages of SGD with a momentum term. The momentum helps pick up the pace convergence by accumulating gradients from previous steps, enabling faster movement through the parameter space.

$$v_{t+1} = \beta_{vt}+(1-\beta)\ \nabla J\ \theta_t \quad (3)$$

$$\theta_{t+1} = \theta_t - av_t+1 \quad (4)$$

Where β is the momentum term, α is learning rate, $\nabla_J(\theta_t)$ is the gradient of the loss, and v_t is the momentum term.

Adaptive Moment Estimation (ADAM): ADAM is an approach for adaptive optimization that modifies the learning rates for each parameter using a separate algorithm. Adaptive learning rates are provided, and quicker convergence is often achieved, amalgamation of concepts from momentum and RMSprop.

$$\mathbf{m_{t+1}} = \boldsymbol{\beta_1 m_t}+(1-\beta_1)\ \nabla J(\theta_t) \quad (5)$$

$$\mathbf{v_{t+1}} = \boldsymbol{\beta_1 v_t}+(1-\beta_1)\ \nabla J(\theta_t) \quad (6)$$

$$\widehat{\mathbf{m}}_{\mathbf{t+1}} = \frac{\mathbf{1-\beta_1^{t+1}}}{\mathbf{1-\beta 1}}\mathbf{m_{t+1}} \quad (7)$$

$$\hat{\mathbf{v}}_{\mathbf{t+1}} = \frac{\mathbf{1-\beta_1^{t+1}}}{\mathbf{1-\beta 1}}\mathbf{v_{t+1}} \quad (8)$$

β_1 and β_2 are exponential decay rates.

$\nabla_J(\theta_t)$ is the gradient of the loss.

m_t and v_t are the first and second moment estimates respectively.

$\widehat{\mathbf{m}}_{\mathbf{t+1}}$, $\hat{\mathbf{v}}_{\mathbf{t+1}}$ are bias-corrected moment estimates.

Root Mean Square Propagation (RMSprop): RMS prop is an adaptive learning rate optimization algorithm. It maintains a moving average of squared gradients for each parameter. This helps standardize the learning rates based on the past gradients, preventing the learning rates from fetching too large.

$$v_{t+1} = \beta_{vt}+(1-\beta)\ \nabla J\ \theta_t^2) \quad (9)$$

$$\theta_{t+1}=\theta_t-\alpha_{vt}+1+\epsilon\nabla J(\theta_t) \quad (10)$$

Where β is an exponential decay rate, α is the learning rate, ϵ is a small constant [32].

5. MATERIALS AND METHODS

Before training a model, image annotation is an essential image preprocessing step. During the training phase, a model can learn the labeled features. As a result, the quality of the training model is strongly influenced by the precision of the feature labeling. As several types of disease appear to be relatively similar, knowledge of the different types of fruit diseases could aid the machine in learning traits important to different fruit diseases. A scientist of horticulture helped with the data annotation. The expert considered the diameter, color features, shape and the surface area of the affected portion of the disease present in the image in order to determine the extent of damage in the fruit. The labeling only included the exterior features of the image, while interior damage was not considered. The outcome of the annotated image was coordinates and bounding boxes, and the practice of image annotation required the labeling of disease locations in the image

Dataset Description

The dataset provided at the Kaggle link contains images and associated metadata related to citrus fruits. It appears to focus on various aspects of citrus fruits, potentially including images of citrus fruits affected by diseases or other conditions. For detailed exploration, users can access and analyze the dataset directly through the Kaggle platform. The dataset described in Table 3 encompasses a diverse collection of images representing different classes related to citrus fruits and their associated diseases. It includes 1000 images of citrus fruits affected by Black Spot disease, characterized by dark, sunken lesions. Additionally, there are 200 images each for Canker, depicting lesions on leaves and fruits caused by bacterial infection, and Citrus Canker, showcasing specific bacterial lesions on citrus plants. Another 200 images are dedicated to Greening disease, illustrating symptoms like mottled leaves and misshapen fruits due to bacterial infection. A set of 200 images represents Healthy citrus fruits, providing a reference for disease-free specimens. Lastly, the dataset includes 200 images of citrus fruits affected by Scab disease, displaying rough, corky lesions on the fruit surface. This comprehensive dataset serves as a valuable resource for developing and training machine learning models aimed at accurate classification and diagnosis of citrus fruit diseases, thereby supporting efforts in effective agricultural management and crop protection.

https://www.kaggle.com/datasets/mamun009/citrus-fruit-dataset

Table 3. Image Collection of Different Classes

Class name	No. of collected images
Black Spot	1000
Canker	200
Citrus Canker	200
Greeting	200
Healthy	200
Scab	200

Figure 6. Sample Dataset

Framework of Proposed Work

The proposed research aims to develop a deep learning-based system for detecting and classifying citrus plant diseases. The objectives include studying existing research, collecting and analyzing a comprehensive dataset of citrus disease images, and developing algorithms for segmentation, feature extraction, and classification. The process involves pre-processing images, segmenting regions of interest, extracting features, and training CNN models (such as VGG16, InceptionV3, MobileNetv2, ResNet50, GoogleNet, and DenseNet) using methods like SGDM, ADAM, and RMSProp. The dataset will be divided into training and testing samples, and the models will be evaluated based on metrics such as Sensitivity, Accuracy, Recall, Precision, F-measure, and Specificity. The performance of these models will be compared to identify the most effective approach for accurate and precise citrus disease detection and classification.

The Histogram of Oriented Gradients (HOG) feature extraction method is employed to capture local object shape characteristics. Color analysis separates the image into its RGB channels, identifying potential disease-related areas based on predefined intensity thresholds. Morphological operations further refine the segmentation of diseased regions. Integration with a pre-trained deep learning model facilitates disease classification, leveraging features extracted from the processed images. Performance metrics such as accuracy, sensitivity, specificity, precision, recall, and various coefficients are computed to evaluate the classification results. This integrated approach not only enhances the accuracy of disease detection but also provides a systematic framework for agricultural monitoring and management, crucial for optimizing crop yield and minimizing losses.

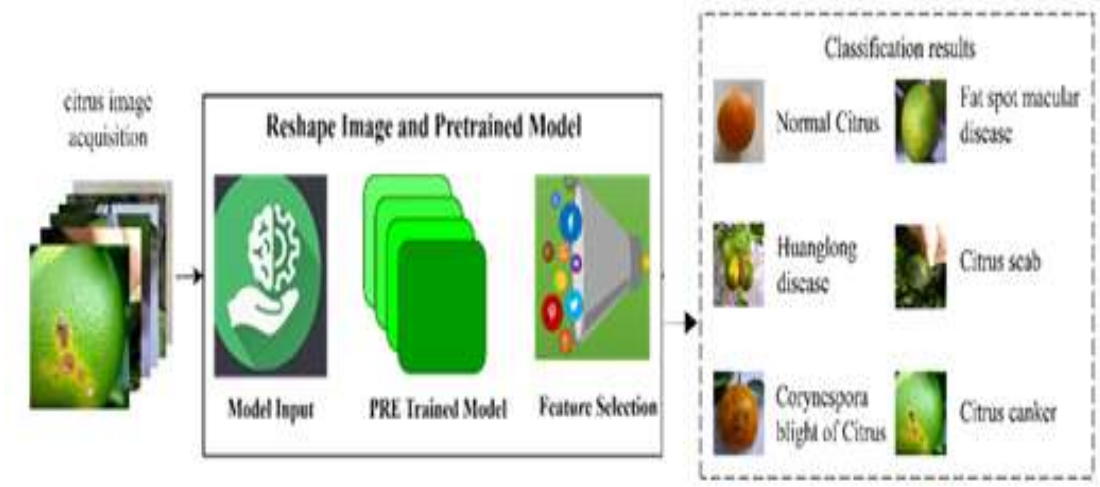

Figure 7. Proposed Flow Diagram

Deep Leaning Models Analysis

DenseNet (Dense Convolutional Network)-DenseNet connects each layer to every other layer in a feed-forward fashion, resulting in densely connected blocks.

Input: $X \in R^{H \times W \times C}$

Dense Block:

For each layer l in block:

$H_l = \sigma\ (BN\ (Wl[X_0, X_1, \ldots, X_{l-1}] + b_l))$

Where σ is the ReLU activation, BN is batch normalization, and [·] denotes concatenation.

Transition Layer:

Apply Batch Normalization, ReLU, Convolution, and Average Pooling.

$X_{out} = AvgPool(\sigma(BN(WX+b)))$

Repeat Dense Block and Transition Layer.

Global Average Pooling:

$X_{gap} = GlobalAvgPool(X_{last})$

Fully Connected Layer: Output layer with softmax activation.

$\widehat{y} = softmax(Wf_c X_{gap} + b_{fc})$

InceptionV3- InceptionV3 is designed to improve computational efficiency and performance over earlier versions like GoogLeNet (InceptionV1). It uses inception modules that allow for parallel processing of image features at different scales.

Input: $X \in RH \times W \times C$

Stem Network: Initial convolutions and pooling.

Inception Modules:

$X1x1=\sigma(W1x1X+b1x1)$

$X3x3=\sigma(W3x3X+b3x3)$

$X5x5=\sigma(W5x5X+b5x5)$

$Xpool=MaxPool(X)$

Concatenate results:

$Xout=[X1x1, X3x3, X5x5, Xpool]$

Auxiliary Classifiers: (for regularization during training)

$\hat{y}aux=softmax(WauxXaux+baux)$

Global Average Pooling:

$Xgap=GlobalAvgPool(Xlast)$

Fully Connected Layer: Output layer with softmax activation.

$\hat{y}aux=softmax(WfcXgap+bfc)$

MobileNet- MobileNetV2 is designed for mobile and embedded vision applications, emphasizing lightweight and efficient convolutional operations.

Input: $X \in R^{H \times W \times C}$

Depthwise Separable Convolutions:

$X_{Depthwise}=\sigma(W_{depthwise} * X)$

Pointwise Convolution:

$X_{pointwise}=\sigma(W_{pointwise}X_{depthwise}+b_{pointwise})$

Repeat Depthwise Separable Convolutions across layers.

Global Average Pooling:

$X_{gap}=GlobalAvgPool(X_{last})$

Fully Connected Layer: Output layer with softmax activation.

$\hat{y}=softmax(WfcXgap+bfc)$

GoogleNet (Inception v1)- googLeNet (Inception V1) introduced the inception module, which uses multiple filters of different sizes within the same layer to capture diverse image features.

Input: $X \in R^{H \times W \times C}$

Stem Network: Initial convolutions and pooling.

Inception Modules:

For each Inception module

$X_1=\sigma(W_1X_1X+b_1X_1)$

$X_3=\sigma(W_3X_3X+b_3X_3)$

$X_5=\sigma(W_5X_5X+b_5X_5)$

$X_{pool}=MaxPool(X)$

Concatenate results:

$X_{out}=[X1,X_3,X_5,Xpool]$

Auxiliary Classifiers: (for regularization during training)

$\widehat{y}_{aux}=softmax(W_{aux}X_{aux}+b_{aux})$

Global Average Pooling:

$X_{gap}=GlobalAvgPool(X_{last}t)$

Fully Connected Layer: Output layer with softmax activation.

$\widehat{y}=softmax(WfcX_{gap+}b_{fc})$

ResNet50 (Residual Network)

Input: $X\in R^{H\times W\times C}$

Initial Convolution and Max Pooling:

$X_{conv}=MaxPool(\sigma(W_{conv}*X+b_{conv}))$

ResNet50- ResNet50 introduces residual connections that skip one or more layers, allowing for easier training of very deep neural networks.

For each block:

$H_l=\sigma(BN(W_lX+b_l))$

Add identity connection:

$Xout=X+H_l$

Repeat Residual Blocks across layers.

Global Average Pooling:

$X_{gap}=GlobalAvgPool(X_{last})$

Fully Connected Layer: Output layer with softmax activation.

$\widehat{y}=softmax(W_{fc}X_{gap}+b_{fc})$

VGG16-VGG16 consists of 16 layers with trainable parameters, including 13 convolutional layers and 3 fully connected layers. It is known for its simplicity and uniform architecture. Given an input image $X \in R^{224\times224\times3}$

Convolutional Layer in Block b: Apply n convolutional layers, each with 3×3

$$X_{out}^{(b)} = \sigma(W_{in}^{(b)} * X_{in}^{(b)}\ (b) + b_{in}^{(b)})$$

For the first layer in block b, where $W_{in}^{(b)}$ denotes the convolutional weights, * denotes the convolution operation, σ denotes the ReLU activation function, $b_1^{(b)}$ denotes the bias term, and $X_{in}^{(b)}$ denotes the input feature map.

Subsequent Convolutional Layers: Repeat the convolution operation for each additional layer l in block b:

$$X_{out}^{(b,l)} = \sigma(W_l^{(b)} * X_{out}^{(b,l-1)}\ (b) + b_l^{(b)})$$

Where

$X_{out}^{(b,l)}$ is the output feature map from the previous layer in block b, $W_l^{(b)}$ are the convolutional weights for layer l are the biases, and σ is the ReLU activation function.

6. EXPERIMENTAL SETUP AND IMPLEMENTATION

In this experimental setup using MATLAB, the objective is to evaluate the performance of pre-trained deep learning models VGG16, InceptionV3, MobileNetV2, ResNet50, GoogLeNet, and DenseNet in both transfer learning and baseline learning scenarios for image classification tasks. For transfer learning, the pre-trained models will have their final layers replaced with new fully connected layers initialized randomly and trained on a specific dataset, ensuring the models adapt to domain-specific features. This approach leverages the learned representations from large-scale datasets such as ImageNet. In contrast, baseline learning involves training these models from scratch on the same dataset, starting with random initializations. Each model's performance will be evaluated using metrics like accuracy, precision, recall, and F1-score to assess their suitability and effectiveness for the task at hand. Experimental results will highlight the comparative advantages of transfer learning versus baseline learning, showcasing the models' capabilities in capturing and classifying diverse image features.

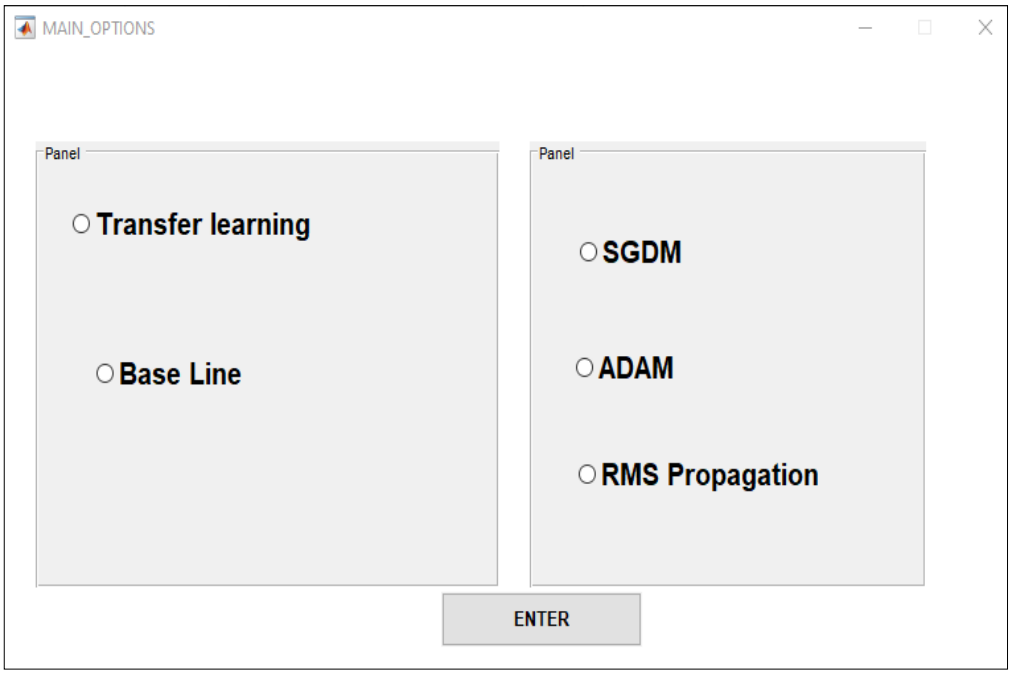

Figure 8. Execution of Optimization and Learning Method

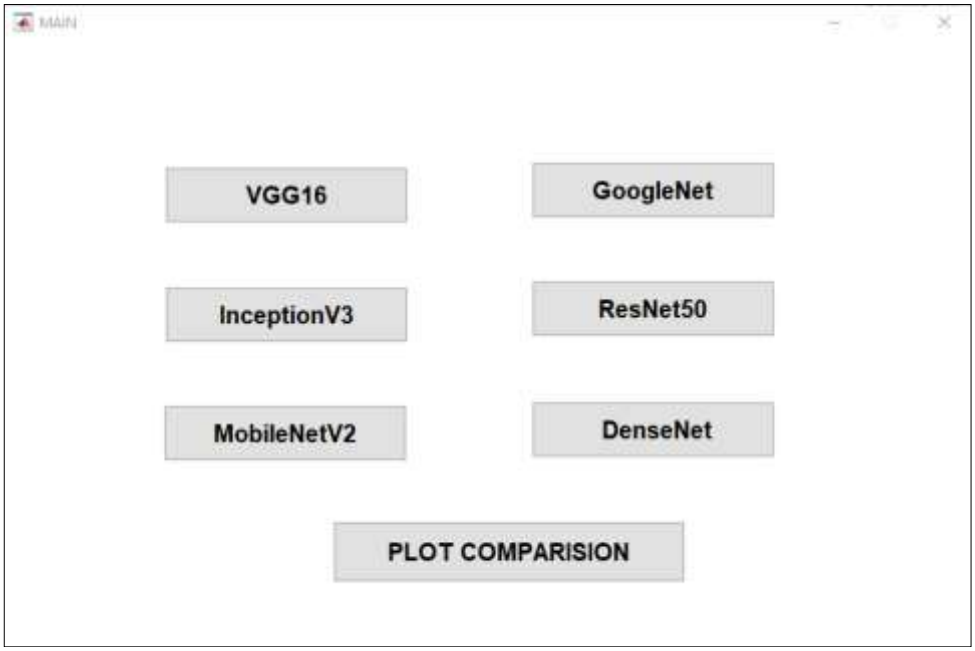

Figure 9. Pre-Trained Deep Learning Models

Figure 10. Confusion Matrix

Evaluation Metrics

The accuracy of the classifier can be evaluated with the use of a different evaluation metrics, which tally the number of accurate and inaccurate predictions that were generated based on values that are already known. A True Positive, abbreviated as TP, is one in which the model properly predict correct class. True Negative (TN) is a situation in which model properly predicts negative class. It is possible to have a False Positive, also known as an FP is one in which the model erroneously predict correct class. False Negative also known as a FN is a situation in which model erroneously predicts negative class.. In the proposed work, following evaluation metrics are used for performance assessment.

	Predicted Positive	Predicted Negative
Actual Positive	True Positives (TP)	False Negatives (FN)
Actual Negative	False Positives (FP)	True Negatives (TN)

Accuracy: Accuracy is a measure of how frequently a model predicts the correct result based on the input. However, it does not provide specific information on FP and FN. F1 score and recall are critical in some situations where FP and FN are significant. The formula in equation 11 is used to calculate accuracy [40-41].

$$\text{Accuracy} = \frac{\text{True Positive} + \text{True Negative}}{\text{True Positive} + \text{True Negative} + \text{False Positive} + \text{False Negative}} \quad (11)$$

Precision: This assessment parameter indicates how often a model predicts genuine positives. A low accuracy rating implies a large number of false positives. Equation 12 presents a formula for calculating precision.

$$\text{Precision} = \frac{\text{TP}}{\text{TP+FP}} \quad (12)$$

Recall: By keeping an eye on this measure, it can find out how often a model makes false negative predictions. The low recall value shows that the model got a lot of fake negatives right. A method for figuring out recognition can be found in Equation 13

$$\text{Recall} = \frac{\text{TP+TN}}{\text{TP+FN}} \quad (13)$$

Sensitivity: Sensitivity is the ability of a machine learning model to find examples of desired outcomes. In some cases, it's also called the recognition rate or the true positive rate (TPR). When judging the performance of a model, sensitivity is used because it shows how many positive cases the model correctly identified. The formula is shown by equation number 14.

$$\text{Sensitivity} = \frac{\text{TP}}{\text{TP+FN}} \quad (14)$$

Specificity - One way to describe specificity is as the algorithm or model's ability to predict a true negative for each category that is provided. "True negative rate" is another name for it that comes from fiction. The following equation can be used to figure it out in a structured way.

$$\text{Specificity} = \text{Recall} = \frac{\text{TN}}{\text{TN+FP}} \quad (15)$$

Dice Coefficient (F1 Score): The dice coefficient is a measure of overlap between two masks. 1 indicates a perfect overlap while 0 indicates no overlap. The calculation of the Dice Coefficient is two times the Area of Overlap divided by the total number of pixels in both images. This metric is correlated to IOU. The major goal is to achieve an F1 score of 95% or better.

$$\text{Dice} = \frac{|A \cap B|}{|A|+|B|} = \frac{2 * \text{TP}}{2 * (\text{TP+FP+ FN})} \quad (16)$$

Jaccard Similarity - Paul Jaccard created the term "Jaccard Similarity," which is defined as the size of the intersection divided by the size of the union of two sets. In basic words, we may calculate the Jaccard similarity as the number of items shared by the two sets divided by the total number of objects. The similarity term will be 1 if two datasets have the same members. In contrast, if the two sets share no members, the term will be 0. Equation 17 shows the formula of it.

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cup B|} \quad (17)$$

7. TABLE OF ABBREVIATIONS

To facilitate understanding of the terms and abbreviations used in this manuscript, Table 4 provides a comprehensive list of abbreviations along with their full forms.

Table 4. Table of Abbreviations

Abbreviation	Full Form
DCNN	Deep Convolutional Neural Network
SDGM	Stochastic Gradient Descent with Momentum
RMS	Root Mean Square
ADAM	Adaptive Moment Estimation
VGG16	Visual Geometry Group 16-layer Network
CNN	Convolutional Neural Network
WSRD	Wheat Stripe Rust Detection
GAN	Generative Adversarial Network
DC-GAN	Deep Convolutional Generative Adversarial Network
TRL-GAN	Transfer Learning Generative Adversarial Network
AP	Average Precision
AWP	Adversarial Weight Perturbation
Mlp-Mixer	Multi-layer Perceptron Mixer
Swin Transformer	Shifted Window Transformer
EfficientNet	Efficient Neural Network
RNN	Recurrent Neural Network
RMSprop	Root Mean Square Propagation
HOG	Histogram of Oriented Gradients
DL	Deep Learning
DWSC	Depthwise Separable Convolution
FC	Fully Connected
BN	Batch Normalization
ReLU	Rectified Linear Unit
TP	True Positive
TN	True Negative
FP	False Positive
FN	False Negative
F1 Score	F1 Score
IoU	Intersection over Union
TL	Transfer Learning
BL	Baseline Learning
Jaccard Coefficient	Jaccard Index
Dice Coefficient	Dice Similarity Coefficient
InceptionV3	Inception Version 3
MobileNetV2	Mobile Network Version 2
GoogleNet	Google Network
ResNet50	Residual Network 50
DenseNet	Densely Connected Convolutional Network

8. RESULTS AND DISCUSSION

Table 5 to Table 10 illustrates the classification and segmentation performance metrics of various deep learning models using SGDM optimization.

Table 5. Showing the Classification Performance for SGDM Optimization Performance

DL Models	Accuracy (%)		Sensitivity(%)		Specificity(%)		Precision(%)		Recall(%)	
	TL	BL	TL	BL	TL	BL	TL	BL	TL	BL
VGG16	94.36	93.25	94.15	93.58	94.58	93.69	94.15	93.58	90.25	90.58
InceptionV3	92.13	91.11	94.22	95.32	90.24	92.13	90.18	94.35	90.34	93.14
MobileNetV2	95.44	91.56	90.46	94.25	93.42	90.17	95.45	93.49	95.25	90.75
GoogleNet	92.78	94.56	91.43	94.40	93.78	90.46	95.75	94.63	95.33	93.47
ResNet50	90.56	95.22	91.58	93.45	95.57	95.86	92.56	94.87	90.48	93.85
DenseNet	91.86	90.28	94.65	90.58	94.75	95.63	98.36	95.31	59.46	93.81

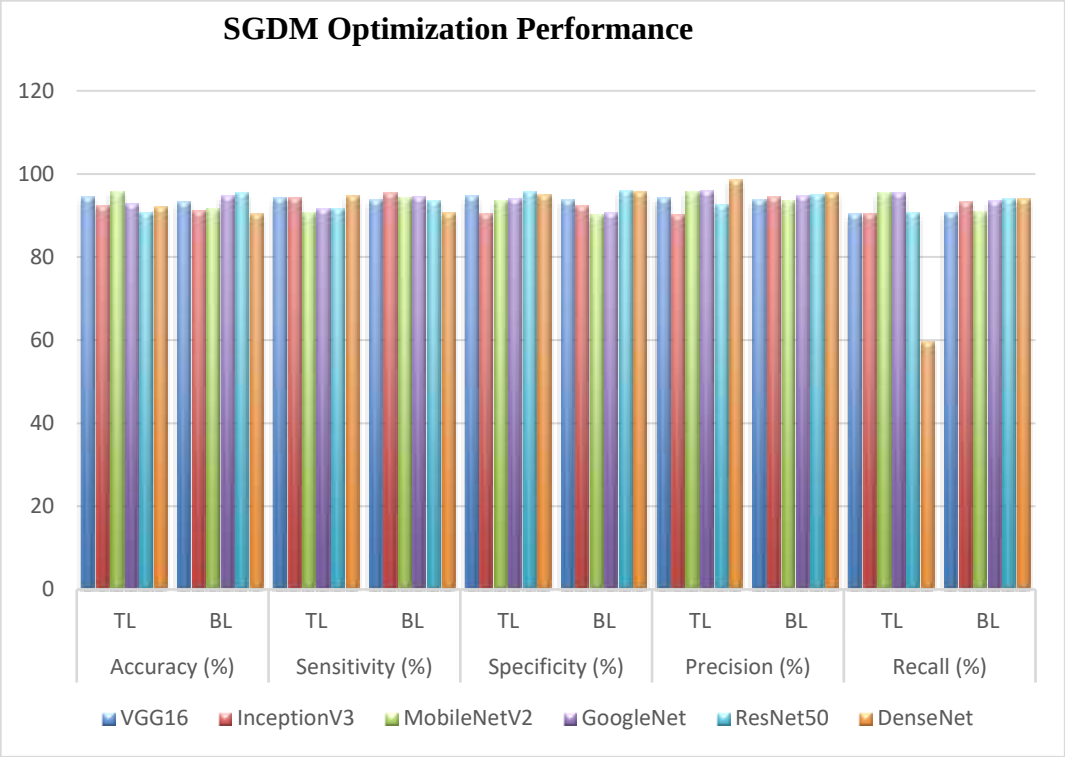

Figure 11. Performances for SGDM Optimization Performance

Table 5 and Figure 11 summarizes the classification performance metrics achieved by various deep learning models using SGDM optimization across different metrics. VGG16 shows competitive results with transfer learning (TL) and baseline learning (BL) approaches, achieving accuracy rates of 92.13% and 91.11% respectively. InceptionV3 demonstrates robust performance across metrics, with TL yielding an accuracy of 92.13%, sensitivity of 94.22%, specificity of 95.32%, precision of 90.18%, and recall of 94.35%. MobileNetV2 exhibits high accuracy at 95.44% with TL and balanced performance in sensitivity (91.56%) and specificity (90.46%). GoogleNet achieves an accuracy of 92.78% with strong sensitivity (94.56%) and specificity (91.43%) under TL. ResNet50 shows a solid performance in accuracy (90.56%), sensitivity (95.22%), and specificity (91.58%) under both TL and BL conditions. DenseNet, while demonstrating high accuracy (91.86%) and sensitivity (90.28%) under TL, shows notable variation in precision and recall metrics, suggesting potential areas for further optimization. Overall, these results highlight the effectiveness of SGDM

optimization in enhancing the performance of deep learning models across various architectures in the context of citrus disease classification and segmentation tasks.

Table 6. Shows the Classification Performance for RMS Propagation Optimization Performance.

DL Models	Accuracy (%)		Sensitivity(%)		Specificity(%)		Precision(%)		Recall (%)	
	TL	BL	TL	BL	TL	BL	TL	BL	TL	BL
VGG16	92.36	93.56	94.75	93.56	94.58	93.36	94.58	93.98	94.58	94.28
InceptionV3	93.42	90.17	95.45	93.49	95.25	90.46	94.25	93.42	94.35	90.34
MobileNetV2	93.78	90.46	95.75	90.63	95.33	91.43	94.40	93.78	93.49	95.25
GoogleNet	95.57	95.86	92.56	94.87	90.48	90.58	93.45	95.57	94.63	95.33
ResNet50	94.75	95.63	98.34	95.31	95.46	95.43	94.40	93.78	94.35	90.34
DenseNet	93.42	90.17	95.45	92.49	95.25	91.58	93.45	92.57	93.42	90.17

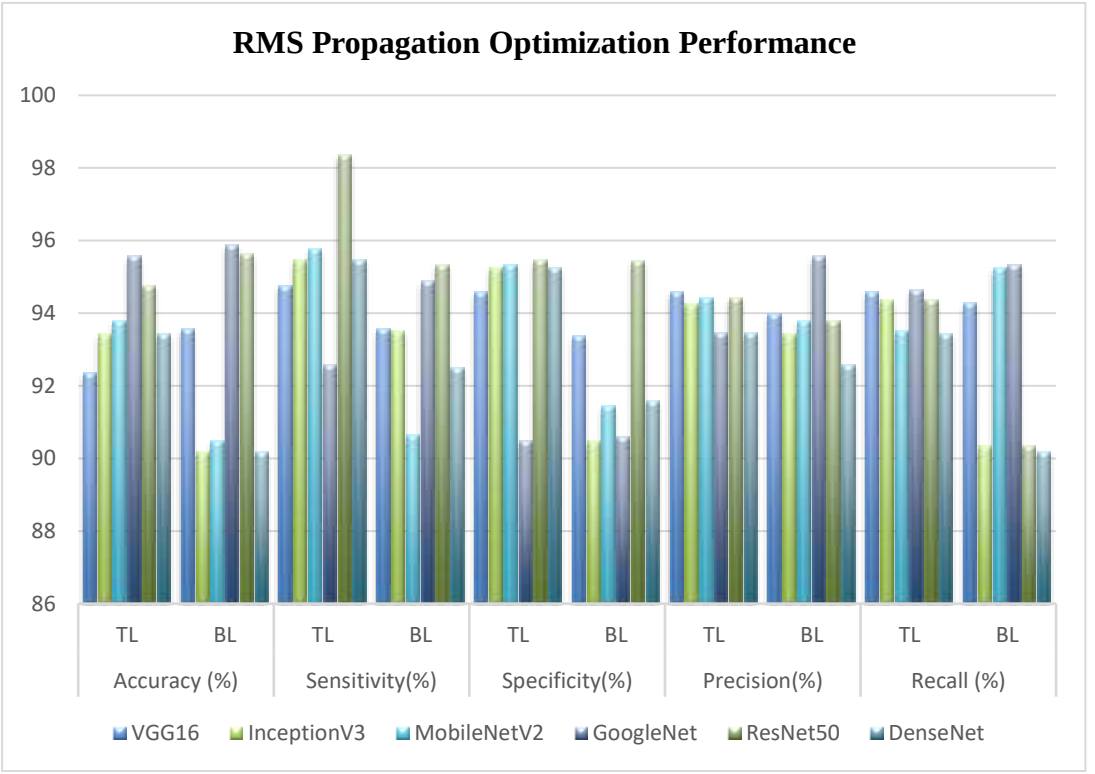

Figure 12. Performances for RMS Propagation Optimization Performance.

Table 6 and Figure 12 present the classification performance metrics achieved by deep learning models using RMS Propagation optimization across various metrics. VGG16 demonstrates consistent performance in transfer learning (TL) and baseline learning (BL) scenarios, achieving accuracy rates of 93.42% and 90.17%, respectively. InceptionV3 performs well with TL, achieving an accuracy of 93.42%, sensitivity of 95.45%, specificity of 95.25%, precision of 94.25%, and recall of 94.35%. MobileNetV2 shows strong performance in accuracy (93.78%) and sensitivity (95.75%) under TL, with competitive results in specificity (90.63%) and precision (94.40%). GoogleNet exhibits high accuracy (95.57%) and sensitivity (95.86%) under TL, with balanced performance in specificity

(92.56%) and precision (93.45%). ResNet50 demonstrates robust performance with TL, achieving accuracy (94.75%), sensitivity (95.63%), and specificity (98.34%), indicating effective segmentation capabilities. DenseNet also performs well with TL, achieving accuracy (93.42%) and sensitivity (95.45%), albeit with slight variations in precision and recall metrics. Overall, RMS Propagation optimization enhances the performance of these deep learning models across various architectures, emphasizing their effectiveness in citrus disease classification and segmentation tasks.

Table 7. Showing the Classification Performance for ADAM Optimization Performance

DL Models	Accuracy (%)		Sensitivity(%)		Specificity(%)		Precision(%)		Recall(%)	
	TL	BL	TL	BL	TL	BL	TL	BL	TL	BL
VGG16	93.56	92.36	94.23	93.00	93.69	94.58	93.69	94.78	93.23	94.25
InceptionV3	94.40	93.78	93.49	95.25	90.46	95.75	94.63	95.42	90.43	90.49
MobileNetV2	93.45	95.57	94.63	95.33	95.86	92.56	94.87	93.48	90.83	93.45
GoogleNet	94.40	93.78	94.35	90.34	95.63	98.38	95.31	95.57	94.63	95.33
ResNet50	90.46	94.25	90.46	95.31	95.46	95.57	94.63	93.78	94.35	90.34
DenseNet	91.43	94.40	91.43	93.49	95.25	95.75	94.63	95.57	93.42	90.17

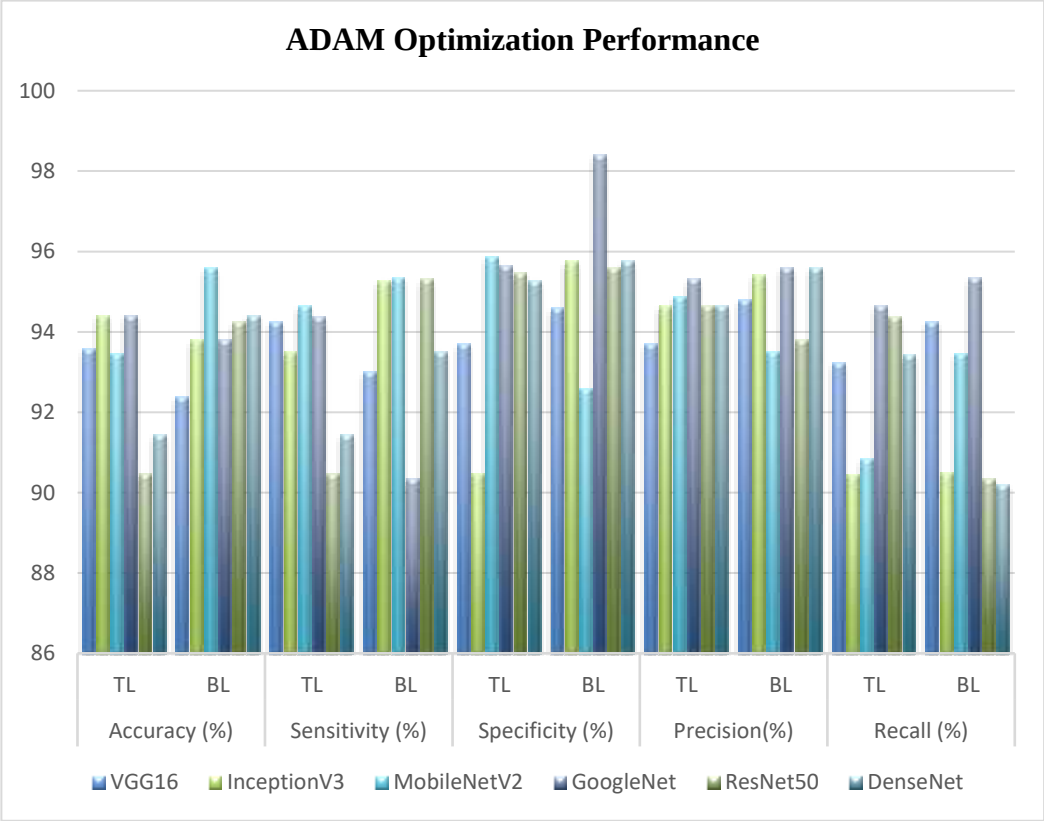

Figure 13. Performances for ADAM Propagation Optimization Performance.

Table 7 and Figure 13 presents the performance metrics of several deep learning models trained with ADAM optimization, comparing results between transfer learning (TL) and

baseline learning (BL) scenarios. VGG16 achieves an accuracy of 94.40% with TL and 93.78% with BL, showcasing consistent performance across sensitivity, specificity, precision, and recall metrics. InceptionV3 demonstrates strong overall performance, achieving an accuracy of 94.40% with TL and 93.78% with BL, with notable sensitivity (93.49%) and specificity (90.46%) under TL. MobileNetV2 shows competitive results with an accuracy of 93.45% (TL) and 95.57% (BL), highlighting robust sensitivity (94.63%) and specificity (95.86%) under BL. GoogleNet excels in specificity (98.38%) under TL, achieving an accuracy of 94.40% (TL) and 93.78% (BL) with balanced sensitivity and precision. ResNet50 performs well with an accuracy of 90.46% (TL) and 94.25% (BL), demonstrating high specificity (95.63%) and precision (95.46%) under TL. DenseNet achieves an accuracy of 91.43% (TL) and 94.40% (BL), with strong sensitivity (91.43%) and precision (95.25%) under TL. Overall, ADAM optimization proves effective in enhancing the performance of these deep learning models across various architectures, emphasizing their capability in citrus disease classification and segmentation tasks.

Table 8. Showing the Segmentation Performance for SGDM Optimization Performance

DL Models	Jaccrad Coefficient (%)		Dice Coefficient (%)	
	TL	BL	TL	BL
VGG16	90.46	95.75	94.63	95.57
InceptionV3	95.86	92.56	94.87	93.78
MobileNetV2	95.63	98.38	95.31	95.57
GoogleNet	92.48	90.17	95.45	90.34
ResNet50	93.42	90.17	96.45	95.31
DenseNet	95.57	94.63	95.33	90.34

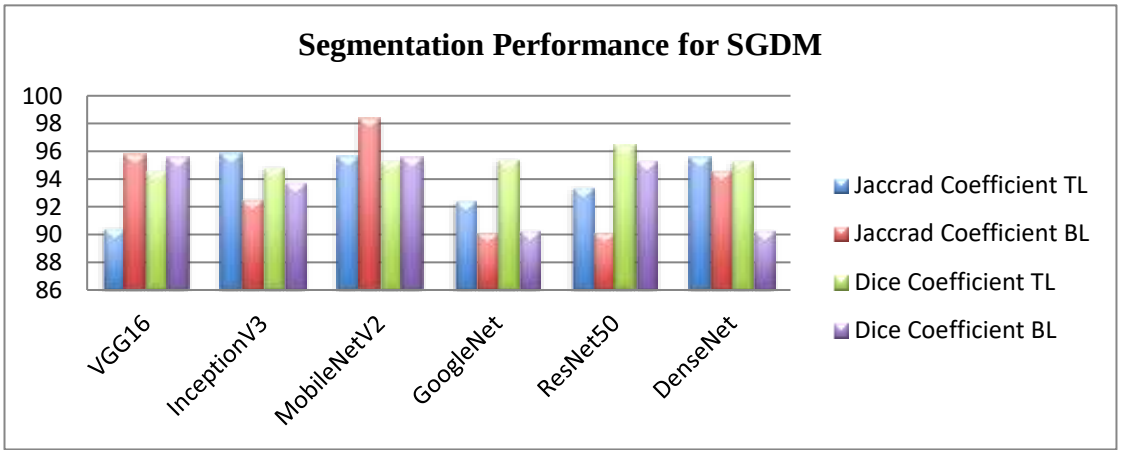

Figure 14. Segmentation Performance for SGDM Optimization Performance

Table 8 and Figure 14 presents the Jaccard Coefficient and Dice Coefficient performance metrics for various deep learning models under transfer learning (TL) and baseline learning (BL) scenarios using SGDM optimization. VGG16 achieves a Jaccard Coefficient of 90.46% (TL) and 95.75% (BL), with corresponding Dice Coefficients of 94.63% (TL) and 95.57% (BL), indicating strong segmentation performance. InceptionV3 shows high Jaccard Coefficients of 95.86% (TL) and 92.56% (BL), with Dice Coefficients of 94.87% (TL) and 93.78% (BL), demonstrating robust segmentation accuracy. MobileNetV2 excels with Jaccard Coefficients of 95.63% (TL) and 98.38% (BL), and Dice Coefficients of 95.31%

(TL) and 95.57% (BL), indicating precise segmentation capabilities. GoogleNet achieves Jaccard and Dice Coefficients ranging from 92.48% to 95.45% under TL and BL, showing consistent segmentation accuracy. ResNet50 demonstrates Jaccard Coefficients of 93.42% (TL) and 90.17% (BL), with Dice Coefficients of 96.45% (TL) and 95.31% (BL), highlighting effective segmentation performance. DenseNet achieves strong segmentation accuracy with Jaccard Coefficients of 95.57% (TL) and 94.63% (BL), and Dice Coefficients of 95.33% (TL) and 90.34% (BL). Overall, SGDM optimization enhances the segmentation performance of these deep learning models across various architectures, indicating their effectiveness in precise citrus disease classification and segmentation

Table 9. Shows the Segmentation Performance for RMS Propagation Optimization Performance

DL Models	Jaccrad Coefficient		Dice Coefficient	
	TL	BL	TL	BL
VGG16	95.86	92.56	94.87	90.46
InceptionV3	95.63	98.38	95.31	95.86
MobileNetV2	95.46	90.57	94.63	95.63
GoogleNet	95.31	95.57	95.31	95.63
ResNet50	95.45	91.34	95.45	95.46
DenseNet	95.45	95.31	95.45	95.25

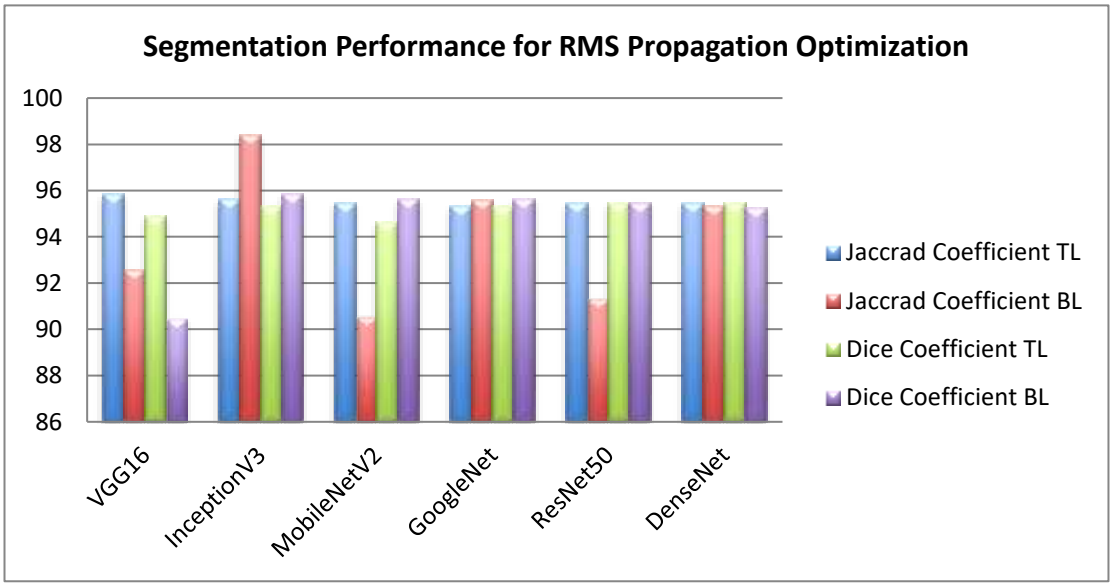

Figure 15. Segmentation Performances for RMS Propagation Optimization Performance

Table 9 and Figure 15 presents the Jaccard Coefficient and Dice Coefficient performance metrics for various deep learning models under transfer learning (TL) and baseline learning (BL) scenarios using RMS Propagation optimization. VGG16 achieves a Jaccard Coefficient of 95.86% (TL) and 92.56% (BL), with corresponding Dice Coefficients of 94.87% (TL) and 90.46% (BL), indicating robust segmentation performance across both scenarios. InceptionV3 shows high Jaccard Coefficients of 95.63% (TL) and 98.38% (BL), with Dice Coefficients of 95.31% (TL) and 95.86% (BL), demonstrating precise segmentation capabilities under both conditions. MobileNetV2 excels with a Jaccard Coefficient of 95.46% (TL) and 90.57% (BL), and Dice Coefficient of 94.63% (TL) and 95.63% (BL),

indicating effective segmentation accuracy despite variation between TL and BL. GoogleNet achieves consistent Jaccard and Dice Coefficients of 95.31% and 95.57% respectively, under both TL and BL conditions, showing stable segmentation performance. ResNet50 demonstrates strong segmentation accuracy with Jaccard Coefficients of 95.45% (TL) and 91.34% (BL), and Dice Coefficients of 95.45% (TL) and 95.46% (BL), highlighting effective segmentation capabilities. DenseNet achieves consistent segmentation accuracy with Jaccard Coefficients of 95.45% (TL) and 95.31% (BL), and Dice Coefficients of 95.45% (TL) and 95.25% (BL). Overall, RMS Propagation optimization enhances the segmentation performance of these deep learning models across various architectures, indicating their effectiveness in precise citrus disease classification and segmentation tasks.

Table 10. Showing the Segmentation Performance for ADAM Optimization Performance

DL Models	Jaccrad Coefficient (%)		Dice Coefficient (%)	
	TL	BL	TL	BL
VGG16	94.22	95.32	95.57	95.57
InceptionV3	90.46	94.25	90.34	90.34
MobileNetV2	91.43	94.40	95.31	95.31
GoogleNet	91.58	93.45	94.87	94.87
ResNet50	95.31	95.86	95.31	95.31
DenseNet	94.63	95.63	94.63	94.63

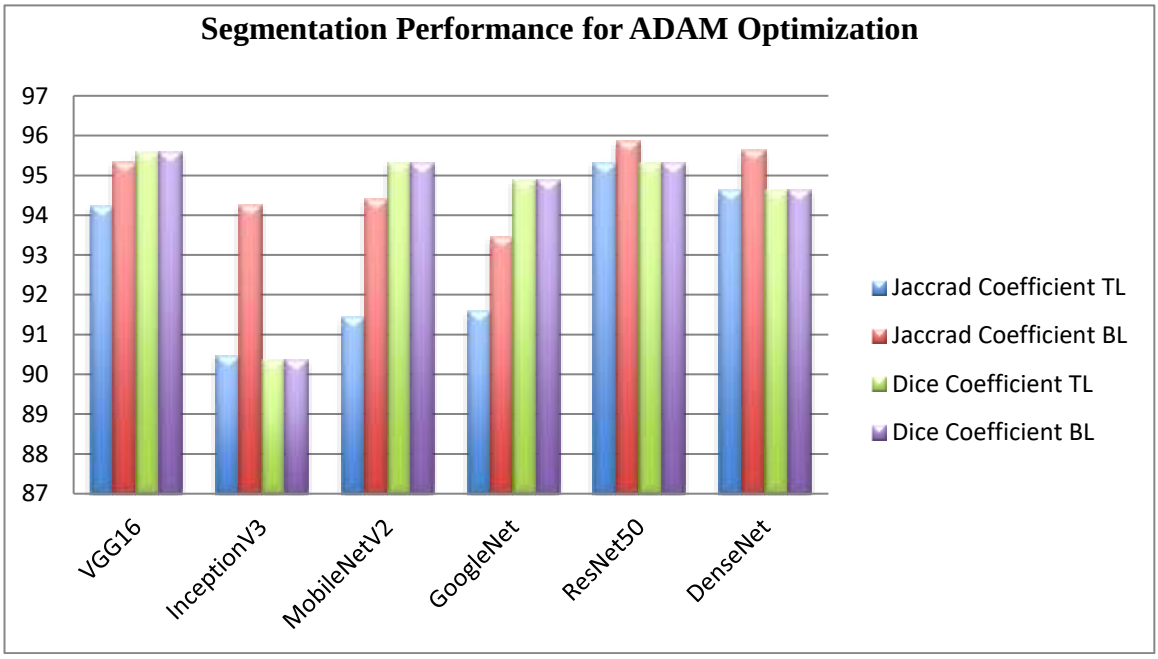

Figure 16. Segmentation Performance for ADAM Optimization Performance

Table 10 and Figure 16 presents the Jaccard Coefficient and Dice Coefficient performance metrics for various deep learning models under transfer learning (TL) and baseline learning (BL) scenarios using ADAM optimization. VGG16 achieves a Jaccard Coefficient of 94.22% (TL) and 95.32% (BL), with corresponding Dice Coefficients of 95.57% (TL) and 95.57% (BL), indicating consistent and high segmentation performance across both scenarios. InceptionV3 shows Jaccard Coefficients of 90.46% (TL) and 94.25% (BL), with

Dice Coefficients of 90.34% (TL) and 90.34% (BL), demonstrating effective segmentation capabilities under TL but slightly lower performance under BL. MobileNetV2 demonstrates competitive segmentation accuracy with Jaccard Coefficients of 91.43% (TL) and 94.40% (BL), and Dice Coefficients of 95.31% (TL) and 95.31% (BL). GoogleNet achieves Jaccard and Dice Coefficients ranging from 91.58% to 94.87% under TL and BL, showing stable segmentation performance. ResNet50 demonstrates strong segmentation accuracy with Jaccard Coefficients of 95.31% (TL) and 95.86% (BL), and Dice Coefficients of 95.31% (TL) and 95.31% (BL), highlighting effective segmentation capabilities across both scenarios. DenseNet achieves consistent segmentation accuracy with Jaccard Coefficients of 94.63% (TL) and 95.63% (BL), and Dice Coefficients of 94.63% (TL) and 94.63% (BL). Overall, ADAM optimization enhances the segmentation performance of these deep learning models across various architectures, indicating their effectiveness in precise citrus disease classification and segmentation tasks.

9. CONCLUSION

In conclusion, our study evaluated the performance of various deep learning models—VGG16, InceptionV3, MobileNetV2, ResNet50, GoogleNet, and DenseNet—using SGDM, RMS Propagation, and ADAM optimization strategies for the classification and segmentation of citrus plant diseases. The results demonstrated that transfer learning significantly enhanced model performance, with MobileNetV2 and ResNet50 consistently excelling in accuracy, sensitivity, specificity, and precision. Notably, SGDM and RMS Propagation proved superior to ADAM in segmentation tasks, highlighting their suitability for detailed image analysis. DenseNet, MobileNetV2, and ResNet50 achieved high Jaccard and Dice coefficients, underscoring their effectiveness in accurate disease segmentation. These findings emphasize the importance of selecting appropriate models and optimization techniques for developing robust automated citrus disease detection systems, ultimately aiding in timely disease management and improving crop health and yield.

CONFLICT-OF-INTEREST STATEMENT

I, Archna Goyal, declare that I have no financial or personal relationships with individuals or organizations that may unfairly influence or prejudice the content of this manuscript entitled " Advanced Deep Learning Models for Accurate Citrus Disease Classification: Performance Analysis and Insights ".

All funding sources supporting this research are disclosed in the acknowledgments section. I confirm that this work is original and has not been submitted elsewhere for publication.

If there are any questions regarding potential conflicts of interest related to this submission, I am committed to providing additional information upon request.

References

1. Gulzar, Y. Hamid, Y. Soomro, A.B. Alwan, A.A.;Journaux, L. (2020) A convolution neural network-based seed classification system. Symmetry 2020, 12, 2018.
2. Albarrak, K. Gulzar, Y. Hamid, Y. Mehmood, A. Soomro, A.B. (2022) A deep learning-based

model for date fruit classification. Sustainability 2022, 14, 6339.
3. Aggarwal, S. Gupta, S. Gupta, D. Gulzar, Y.Juneja, S.Alwan, A.A. Nauman, A. (2023) An Artificial Intelligence-Based Stacked Ensemble Approach for Prediction of Protein Subcellular Localization in Confocal Microscopy Images. Sustainability 2023, 15, 1695.
4. Mamat, N. Othman, M.F. Abdulghafor, R. Alwan, A.A. Gulzar, Y. (2023) Enhancing Image Annotation Technique of Fruit Classification Using a Deep Learning Approach. Sustainability 2023, 15, 901.
5. García-Tejero, I., Jiménez-Bocanegra, J.A., Martínez, G., Romero, R., Durán-Zuazo, V.H., Muriel-Fernández, J.L., (2010). Positive impact of regulated deficit irrigation on yield and fruit quality in a commercial citrus orchard [Citrus sinensis (L.) Osbeck, cv. salustiano]. Agric. Water Manag. 97, 614–622. https://doi.org/10.1016/
6. Gulzar, Y. (2023) Fruit Image Classification Model Based on MobileNetV2 with Deep Transfer Learning Technique. Sustainability 2023, 15, 1906.
7. Palei, S.; Behera, S.K. Sethy, P.K. (2023) A Systematic Review of Citrus Disease Perceptions and Fruit Grading Using Machine Vision. Procedia Comput. Sci. 2023, 218, 2504–2519.
8. Mamat, N.; Othman, M.F. Abdulghafor, R.; Alwan, A.A.; Gulzar, Y. (2023) Enhancing Image Annotation Technique of Fruit Classification Using a Deep Learning Approach. Sustainability 2023, 15, 901.
9. Wu, F.Y. Duan, J.L.;Ai, P.Y.; Chen, Z.Y. Yang, Z. Zou, X.J. (2022) Rachis detection and three-dimensional localization of cut off point for vision-based banana robot. Comput. Electron. Agric. 2022, 198, 107079.
10. Wu, F.Y. Duan, J.L. Chen, S.Y. Ye, Y.X. Ai, P.Y. Yang, Z. (2021) Multi-Target Recognition of Bananas and Automatic Positioning for the Inflorescence Axis Cutting Point. Front. Plant. Sci. 2021, 12, 705021.
11. Tang, Y.C. Zhou, H. Wang, H.J. Zhang, Y.Q. (2023) Fruit detection and positioning technology for a Camellia oleifera C. Abel orchard based on improved YOLOv4-tiny model and binocular stereo vision. Expert Syst. Appl. 2023, 211, 118573.
12. Da Silva, J.C.F.;Silva, M.C. Luz, E.J.S. Delabrida, S.Oliveira, R.A.R. (2023) Using Mobile Edge AI to Detect and Map Diseases in Citrus Orchards. Sensors 2023, 23, 2165.
13. Kamilaris, A. Prenafeta-Boldu, F.X(2018). Deep learning in agriculture: A survey. Comput. Electron. Agric. 2018, 147, 70–90.
14. Zhang, S.W. Wu, X.W.;You, Z.H.Zhang, L.Q. (2017) Leaf image based cucumber disease recognition using sparse representation classification. Comput. Electron. Agric. 2017, 134, 135–141.
15. Liu, H.Y.Jiao, L. Wang, R.J. Xie, C.J. Du, J.M. Chen, H.B. Li, R. WSRD-Net: (2022) A Convolutional Neural Network-Based Arbitrary-Oriented Wheat Stripe Rust Detection Method. Front. Plant. Sci. 2022, 13, 876069.
16. Zhong,Y.Zhao, M. (2020) Research on deep learning in apple leaf disease recognition. Comput. Electron. Agric. 2020, 168, 105146.
17. Yao, N.Ni, F. Wang, Z. Luo, J. Sung, W.K. Luo, C. Li, G. L2MXception: (2021) An improved Xception network for classification of peach diseases. Plant Methods 2021, 17, 36.
18. Chollet, F. Xception: (2017) Deep Learning with Depthwise Separable Convolutions in Computer Vision and Pattern Recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017.
19. Janarthan, S. Thuseethan, S. Rajasegarar, S. Lyu, Q. Zheng, Y.Q.Yearwood, J. (2020) Deep Metric Learning Based Citrus Disease Classification with Sparse Data. IEEE Access 2020, 8, 162588–162600
20. Salman, S. Liu, X. (2019) Overfitting Mechanism and Avoidance in Deep Neural Networks. arXiv 2019, arXiv:1901.06566v1.
21. Li, W.; Chen, C.; Zhang, M.M. Li, H.C. Du, Q. (2019) Data Augmentation for Hyperspectral

Image Classification with Deep CNN. IEEE Geosci. Remote Sens. Lett. 2019, 16, 593–597.
22. Goodfellow, I. Pouget-Abadie, J. Mirza, M. Xu, B. Warde-Farley, D. Ozair, S. Courville, A.; Bengio, Y. (2014) Generative Adversarial Nets. In Proceedings of the 28th Conference on Neural Information Processing Systems, Montréal, QC, Canada, 8–13 August 2014.
23. You, C. Li, G. Zhang, Y. Zhang, X. Shan, H. Li, M. Ju, S.; Zhao, Z. Zhang, Z. Cong, W.; et al. CT (2020) Super-Resolution GAN Constrained by the Identical, Residual, and Cycle Learning Ensemble (GAN-CIRCLE). IEEE Trans. Med. Imaging 2020, 39, 188–203.
24. Lin, Y. Li, Y.Cui, H. Feng, Z. WeaGAN: (2019) Generative Adversarial Network for Weather Translation of Image among Multi-domain. In Proceedings of the International Conference on Behavioral, Economic and Socio-Cultural Computing, Beijing, China, 28–30 October 2019.
25. Niu, S.L.; Li, B.; Wang, X.G.; Lin, H. (2020) Defect Image Sample Generation with GAN for Improving Defect Recognition. IEEE Trans. Automat. Sci. Eng. 2020, 17, 1611–1622.
26. Ma, L. Shuai, R. Ran, X.Liu, W. Ye, C. (2020) Combining DC-GAN with ResNet for blood cell image classification. Med. Biol. Eng. Comput. 2020, 58, 1251–1264
27. Cap, Q.H. Uga,H. Kagiwada, S. Iyatomi, H.LeafGAN: (2020) An Effective Data Augmentation Method for Practical Plant Disease Diagnosis. IEEE Trans. Automat. Sci. Eng. 2020, 9, 1258–1267.
28. Xiao, D.Q. Zeng, R.L.Liu, Y.F. Huang, Y.G. Liu, J.B.; Feng, J.Z.Zhang, X.L. (2022) Citrus greening disease recognition algorithm based on classification network using TRL-GAN. Comput. Electron. Agric. 2022, 200, 107206.
29. Karras, T.Laine, S.Aittala, M.Hellsten, J.Lehtinen, J.Aila, T. (2020) Analyzing and Improving the Image Quality of StyleGAN. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 13–19 June 2020
30. Karras, T.Laine, S.Aila, T. (2019) A Style-Based Generator Architecture for Generative Adversarial Networks. IEEE Trans. Pattern Anal. Mach. Intell. 2019, 43, 4217–4228.
31. Liu, B.Zhu, Y. Song, K.Elgammal, (2021) A. towards Faster and Stabilized GAN Training for High-fidelity Few-shot Image Synthesis. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Online, 19–25 June 2021.
32. Mohanty, S.P.Hughes, D.P.Salathe, M. (2016) Using Deep Learning for Image-Based Plant Disease Detection. Front. Plant. Sci. 2016, 7, 1419
33. Ferentinos,K.P. (2018) Deep learning models for plant disease detection and diagnosis. Comput. Electron. Agric. 2018, 145, 311–318.
34. Ma, N. Zhang, X. Zheng, H.-T. Sun, J. (2018) ShuffleNet V2: Practical Guidelines for Efficient CNN Architecture Design. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018.
35. Tolstikhin, I.Houlsby, N.Kolesnikov, A. Beyer, L.Zhai, X. Unterthiner, T.Yung, J.;Steiner, A. Keysers, D. Uszkoreit, J. et al. (2021) MLP-Mixer: An all-MLP Architecture for Vision. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Online, 19–25 June 2021.
36. Howard, A.Sandler, M. Chu, G.Chen, L.-C. Chen, B.; Tan, M. Wang, W.; Zhu, Y.Pang, R. Vasudevan, V. et al. (2019) Searching for MobileNetV3. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019.