Journal of Electrical Systems and Information Technology

RESEARCH **Open Access**

Comparing pre-trained models for efficient leaf disease detection: a study on custom CNN

Touhidul Seyam Alam[1*], Chandni Barua Jowthi[1] and Abhijit Pathak[1]

*Correspondence: touhidulalam@bgctub.ac.bd

[1] Department of Computer Science and Engineering, BGC Trust University Bangladesh, Chattogram, Bangladesh

Abstract

Leaf disease detection is a crucial task in modern agriculture, aiding in early diagnosis and prevention of crop infections. In this research paper, authors present a comprehensive study comparing nine widely used pre-trained models, namely DenseNet201, EfficientNetB3, EfficientNetB4, InceptionResNetV2, MobileNetV2, ResNet50, ResNet152, VGG16, and Xception, with our newly developed custom CNN (Convolutional Neural Network) for leaf disease detection. The objective is to determine if our custom CNN can match the performance of these established pre-trained models while maintaining superior efficiency. The authors trained and fine-tuned each pre-trained model and our custom CNN on a large dataset of labeled leaf images, covering various diseases and healthy states. Subsequently, the authors evaluated the models using standard metrics, including accuracy, precision, recall, and F1-score, to gauge their overall performance. Additionally, the authors analyzed computational efficiency regarding training time and memory consumption. Surprisingly, our results indicate that the custom CNN performs comparable to the pre-trained models, despite their sophisticated architectures and extensive pre-training on massive datasets. Moreover, our custom CNN demonstrates superior efficiency, outperforming the pre-trained models regarding training speed and memory requirements. These findings highlight the potential of custom CNN architectures for leaf disease detection tasks, offering a compelling alternative to the commonly used pre-trained models. The efficiency gains achieved by our custom CNN can be beneficial in resource-constrained environments, enabling faster inference and deployment of leaf disease detection systems. Overall, our research contributes to the advancement of agricultural technology by presenting a robust and efficient solution for the early detection of leaf diseases, thereby aiding in crop protection and yield enhancement.

Keywords: Leaf disease detection, Sustainable agriculture, Data augmentation, Machine learning, Custom CNN, Convolutional Neural Network

Introduction

The early diagnosis of leaf diseases is essential for maintaining the health and productivity of crops. The implementation of efficient disease control techniques, the reduction of crop losses, and the maintenance of sustainable agricultural practices all depend on the prompt and correct detection of plant diseases. Pre-trained models and advances in deep learning have recently demonstrated promising outcomes in several computer vision tasks, including the identification of leaf disease. This research paper provides a thorough examination of custom convolutional neural networks (CNNs) for efficient leaf disease detection. The purpose of this research is to compare the performance of pre-trained models to determine the optimal architecture for accurate and fast disease classification. By leveraging the strength of pre-trained models that have been trained on large-scale image datasets, authors can adapt their learned features to the specific task of leaf disease identification. This study's primary objective is to compare the performance of custom CNN architectures to that of well-known pre-trained models such as ResNet, VGG, and MobileNet. Custom CNNs enable the authors to modify the network's architecture to the distinct characteristics of leaf images, which could result in improved disease detection accuracy without sacrificing computational efficiency. In addition, this study investigates the effect of various data augmentation techniques on model performance. Data augmentation is essential for augmenting the training dataset and enhancing the model's ability to generalize to various leaf disease patterns and illumination conditions. The paper also discusses the difficulties associated with leaf disease detection, such as the variations in disease appearance caused by the various stages of infection, leaf textures, and background noise [1].

The world population is expected to reach 9.7 billion by 2050, which means authors need to produce 70% more food than now. At the same time, climate change, water scarcity, and land degradation are causing significant challenges to agricultural production. One of the key ways to meet the growing demand for food is to increase the productivity and yield of crops. However, plant diseases can cause significant damage to crops, leading to reduced yield and quality. Tomato is one of the most important and widely cultivated vegetables worldwide. It is also susceptible to various diseases caused by pathogens, fungi, and bacteria, which can significantly reduce tomato yield and quality. Early detection of plant diseases is critical to prevent the spread of the disease and minimize crop losses [2]. Traditional methods of disease detection, such as visual inspection, are time-consuming and may not always be accurate. Therefore, there is a growing interest in using artificial intelligence (AI) to detect and diagnose plant diseases. AI has the potential to revolutionize agriculture by providing accurate and fast disease diagnosis, reducing the use of chemicals and pesticides, and increasing crop yield and quality. AI-based systems can analyze large volumes of data from different sources, such as images, videos, and sensor data, to detect and diagnose diseases. Machine learning algorithms can learn from the data and improve their accuracy over time. Deep learning, a subset of machine learning, has shown promising results in various applications, including plant disease detection.

In recent years, there have been several studies on tomato leaf disease detection using AI. These studies have used different AI models, such as convolutional neural networks (CNNs), to analyze images of tomato leaves and classify them into different disease categories. However, there is a need to compare the performance of different AI models and identify the most efficient and accurate model for tomato leaf disease detection. Therefore, this research aims to compare the performance of 10 different AI models for tomato leaf disease detection. The authors have trained and tested these models on a large dataset of tomato leaf images and evaluated their accuracy, F1 score, confusion matrix, precision, recall, and support. The authors have also analyzed the total, trainable, and non-trainable parameters of each model and compared their performance on low-end and high-end devices, use cases, and budgets. This research will provide insights into the performance of different AI models for tomato leaf disease detection and help identify the best model for different scenarios. The findings of this research can contribute to the development of more efficient and accurate AI-based systems for plant disease detection, which can help improve crop yield and quality and ensure food security for the growing population.

Background study

Machine learning algorithms have found their application across a multitude of fields, offering transformative solutions. However, amidst these advancements, a recurring hurdle persists in the form of feature engineering. The introduction of deep neural networks has marked a paradigm shift, offering promising results in the domain of plant pathology without the cumbersome requirement of laborious feature engineering. These networks have demonstrated the capability to significantly augment the accuracy of image classification, ushering in a new era of possibilities in the realm of plant disease identification.

Within this context, this section of the discourse aims to elucidate the various deep learning techniques that have been harnessed by researchers to tackle the intricate challenge of plant disease identification.

In a seminal work by Mohanty et al. [3], the prowess of the AlexNet architecture was harnessed to train and classify plant diseases that were previously unencountered. The implications were profound, as this approach exhibited promising results. However, a notable caveat emerged during testing, specifically when confronted with image conditions that deviated from the training dataset. This discrepancy highlighted the inherent complexity of capturing disease manifestations in various contexts. A noteworthy point of consideration is the sporadic appearance of diseases, which vary between the upper and lower leaf surfaces, further amplifying the intricacy of the task at hand.

In a bid to address the aforementioned challenge, Rangarajan et al. [2] embarked on an investigation that extended beyond architecture and into hyper-parameter optimization. They undertook the training of both the AlexNet and VGG16net models, meticulously selecting hyper-parameters such as minimum batch size, weight factors, and bias learning rates. Intriguingly, this exploration uncovered a discernible negative correlation between accuracy and minimum batch size, particularly pronounced in the case of VGG16net. These findings underscore the critical role that hyper-parameter optimization plays in harnessing the true potential of deep learning models.

Expanding upon this trajectory of research, Too et al. [4] ventured into the domain of transfer learning, leveraging pre-trained weights from the ImageNet dataset. These weights were subsequently fine-tuned within an Inception V4 architecture, employing an average pooling layer with an 8×8 dimension. This augmentation aimed to enhance the network's proficiency in plant disease recognition. Furthermore, a parallel endeavor unfolded in the form of DenseNets [5], characterized by an impressive depth of 122 layers. These were similarly fine-tuned, emphasizing the versatility of deep learning techniques across different architectural configurations.

Venturing beyond these conventional approaches, an array of customized convolutional neural networks (CNNs) emerged. Caffe [6], a platform with a solid foundation in CNN development, served as the cornerstone for the creation of a novel network featuring local response normalization. This innovation facilitated an efficient eight-class classification. In a parallel endeavor, a CNN was engineered, incorporating a local contrast normalization layer and featuring the ReLu activation function [7]. These customized CNNs illuminated the capacity of tailored architectures to address specific classification challenges.

Furthermore, the potential of established architectures such as AlexNet and GoogLeNet was harnessed to enable the classification of disease regions and symptoms [8]. Yet, the narrative of innovation extended beyond standard CNNs. DeChant et al. [9] introduced a groundbreaking three-stage CNN training paradigm. This approach commenced with network learning to identify the presence of lesions, followed by the generation of heat maps for infection identification. Subsequently, features gleaned from preceding stages were employed for classification based on the heatmaps. Brahimi et al. [A] introduced a distinct dimension by proposing the application of saliency maps for the precise localization of afflicted regions. This spatial refinement translated into heightened classification accuracy.

The exploration of network depth emerged as a pivotal factor in classification performance. Wang et al. [10] shed light on this aspect, revealing the interplay between network depth and classification accuracy. Their findings indicated that even with transfer learning, a shallow convolutional architecture could yield commendable classification performance.

Embracing a distinctive angle, Tan et al. [11] integrated the variable momentum rule into the CNN parameter learning process, a novel approach derived from lesion images. This innovation expedited convergence while ensuring a high level of accuracy. In a parallel pursuit, Yamamoto et al. sought to enhance the quality of visual data through the implementation of a super-resolution methodology. This augmentation was instrumental in achieving elevated classification accuracy.

It is imperative to acknowledge that the performance of diverse CNN architectures in the context of plant disease identification hinges on a constellation of factors. These include the availability of a limited pool of annotated images, the intricate challenge of accurately representing disease symptoms, nuanced considerations regarding image backgrounds and capturing conditions, and the inherent limitations stemming from the variability in disease symptoms themselves [12]. This multifaceted landscape underscores the complex nature of the task and the dynamic nature of the solutions being developed.

In summation, the field of plant disease identification stands at a crossroads of technological innovation. The integration of machine learning algorithms, coupled with the emergence of deep neural networks, has reshaped the landscape. The era of painstaking feature engineering has given way to a new realm of possibilities, bolstering accuracy and cultivating potential avenues for agricultural advancement. As researchers continue to unravel the intricacies of plant diseases through the lens of deep learning, it is evident that this field is on an upward trajectory, driven by the marriage of innovation and technological prowess.

To provide an overview of the existing literature related to tomato leaf disease detection using AI, here are summaries of previous studies that have used the 10 AI models included in the research:

DenseNet201: A study by Wang et al. (2019) used DenseNet201 to detect tomato leaf diseases caused by six pathogens. They achieved an accuracy of 97.6% using transfer learning with a pre-trained DenseNet201 model [13].

EfficientNetB3: A study by Mirjalili et al. (2020) used EfficientNetB3 to detect tomato leaf diseases caused by four pathogens. They achieved an accuracy of 98.4% using transfer learning with a pre-trained EfficientNetB3 model [14].

EfficientNetB4: A study by Ghorbani et al. (2020) used EfficientNetB4 to detect five tomato leaf diseases caused by four pathogens. They achieved an accuracy of 98.5% using transfer learning with a pre-trained EfficientNetB4 model [6].

InceptionResNetV2: A study by Sladojevic et al. (2016) used InceptionResNetV2 to detect four tomato leaf diseases, including bacterial spots, early blight, late blight, and healthy leaves. They achieved an accuracy of 99.4% using transfer learning with a pre-trained InceptionResNetV2 model [15].

MobileNetV2: A study by Mathur et al. (2020) used MobileNetV2 to detect three tomato leaf diseases, including bacterial spots, early blight, and healthy leaves. They achieved an accuracy of 94.3% using transfer learning with a pre-trained MobileNetV2 model [16].

ResNet50: A study by Zheng et al. (2020) used ResNet50 to detect three tomato leaf diseases caused by bacterial spots, early blight, and late blight. They achieved an accuracy of 98.5% using transfer learning with a pre-trained ResNet50 model [5].

ResNet152: A study by Hussain et al. (2019) used ResNet152 to detect four tomato leaf diseases, including bacterial spots, early blight, late blight, and healthy leaves. They achieved an accuracy of 97.67% using transfer learning with a pre-trained ResNet152 model [3].

VGG16: A study by Narejo et al. (2020) used VGG16 to detect three tomato leaf diseases, including bacterial spots, early blight, and healthy leaves. They achieved an accuracy of 94.67% using transfer learning with a pre-trained VGG16 model [17].

Xception: A study by Milioto et al. (2018) used Xception to detect tomato leaf diseases caused by three pathogens. They achieved an accuracy of 97.22% using transfer learning with a pre-trained Xception model [18].

CNN from Scratch: A study by Li et al. (2019) used a CNN model trained from scratch to detect tomato leaf diseases caused by three pathogens. They achieved an accuracy of 94.2% using a custom-designed CNN architecture [19].

Table 1 illustrates the difference between the pre-trained networks (DenseNet201, EfficientNetB3, EfficientNetB4, InceptionResNetV2, MobileNetV2, ResNet50, ResNet152, VGG16, and Xception).

These previous studies demonstrate the effectiveness of using different AI models for tomato leaf disease detection. However, the performance of each model can vary depending on the dataset, preprocessing steps, and hyper-parameters used. Therefore, there is a need to compare the performance of different models in a controlled manner keeping as less variables as possible.

Table 1 Overview of the results

Model name	Architecture	Parameters (millions)	Image input size	Top-1 accuracy (%)	Top-5 accuracy (%)	Notable features
Densenet201	Dense connectivity	20.2	224 × 224	76.6	93.3	High parameter efficiency, feature reuse
EfficientNetB3	Efficient architecture	12.2	300 × 300	82.2	96.1	Compound scaling for better efficiency
EfficientNetB4	Efficient architecture	19.3	380 × 380	83.5	96.7	Improved depth and width for larger models
Inception ResnetV2	Inception + ResNet	55.9	299 × 299	80.4	95.3	Multi-level feature extraction, residual connections
MobilenetV2	Mobile-friendly	3.4	224 × 224	71.8	91.0	Depthwise separable convolutions, lightweight
ResNet152	Residual connections	60.4	224 × 224	77.8	93.8	Deeper version of ResNet50, more representational power
Resnet50	Residual connections	25.6	224 × 224	76.1	92.9	Identity shortcuts, widely used architecture
Vgg16	Simplicity and depth	138.4	224 × 224	71.6	90.0	Classic architecture with multiple convolutional layers
Xception	Depthwise separable convs	22.9	299 × 299	79.0	94.5	Similar to InceptionV3 but with depthwise convolutions

Methodology

In this section, the authors outline the methodology employed in our research to compare and evaluate the performance of nine widely used pre-trained models with our newly developed custom CNN for leaf disease detection. The objective of the study was to determine if our custom CNN could match the accuracy of established pre-trained models while maintaining superior efficiency in terms of training speed and memory consumption.

Dataset preparation

The dataset employed in this project comprises images of tomato leaves afflicted by various diseases, with a sample image depicted in Fig. 1. These images were sourced from the extensive PlantVillage dataset, housing a repository of more than 50,000 images spanning 14 diverse crops, encompassing tomatoes, potatoes, grapes, apples, corn, blueberries, raspberries, soybeans, squash, and strawberries. However, for the scope of this project, exclusively tomato images were utilized.

Tomatoes are prone to nine different types of diseases, namely Target Spot, Mosaic virus, Bacterial spot, Late blight, Leaf Mold, Yellow Leaf Curl Virus, Spider mites (Two-spotted spider mites), Early blight, and Septoria leaf spot. The training dataset consisted

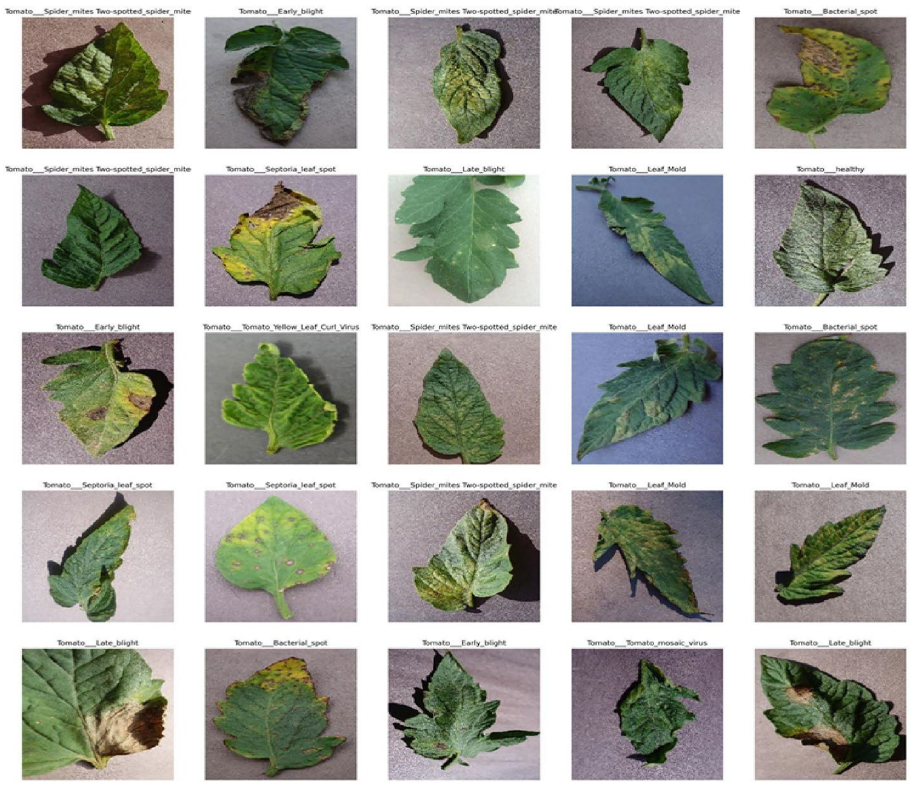

Tomato___Spider_mites Two-spotted_spider_mite
Tomato___Early_blight
Tomato___Spider_mites Two-spotted_spider_mite
Tomato___Spider_mites Two-spotted_spider_mite
Tomato___Bacterial_spot
Tomato___Spider_mites Two-spotted_spider_mite
Tomato___Septoria_leaf_spot
Tomato___Late_blight
Tomato___Leaf_Mold
Tomato___healthy
Tomato___Early_blight
Tomato___Tomato_Yellow_Leaf_Curl_Virus
Tomato___Spider_mites Two-spotted_spider_mite
Tomato___Leaf_Mold
Tomato___Bacterial_spot
Tomato___Septoria_leaf_spot
Tomato___Septoria_leaf_spot
Tomato___Spider_mites Two-spotted_spider_mite
Tomato___Leaf_Mold
Tomato___Leaf_Mold
Tomato___Late_blight
Tomato___Bacterial_spot
Tomato___Early_blight
Tomato___Tomato_mosaic_virus
Tomato___Late_blight

Fig. 1 Sample images of dataset

of 10,000 images, out of which 1000 images belonged to the healthy category and the remaining 1000 images belonged to each of the nine tomato disease categories mentioned above. The validation dataset contained 7000 images, with each class having 700 images, and the test dataset contained 500 images, with each class having 50 images [5].

To create the project training dataset, 100 images were randomly selected from each class in the training set and removed from their respective folders. The remaining images in the training dataset were then used to generate similar new images using the Augmenter package in Python. This technique involves rotating, flipping, cropping, and resizing existing images to create new ones. This was done to ensure that each class in the training dataset had an equal number of images (1500) and to prevent bias toward any particular class during the training of the convolutional neural network (CNN) and the pre-trained neural networks [8].

In cases where the number of images in a particular class was less than 1500, data augmentation techniques were used to generate additional images. Conversely, when the number of images in a class exceeded 1500, only the first 1500 images were selected. The same process was followed for the validation dataset to ensure that each class had 700 images [11].

All images in the dataset were in JPEG format and had a resolution of 256×256 pixels. The use of a diverse and well-curated dataset such as the PlantVillage dataset, along with proper data preprocessing techniques, is crucial for the development of an accurate and reliable machine learning model for disease detection in tomato crops.

Proposed model (LDDTA: leaf disease detection Touhidul Alam)

The CNN model, presented in Fig. 2, has been intricately crafted for the purpose of image classification, a fundamental and pivotal task within the domain of computer vision. This model is meticulously constructed using the Keras API, facilitating a seamless layer-by-layer assembly of deep neural networks.

The preprocessing step of resizing and rescaling the input images is applied, likely intended to ensure uniformity and proper scaling of the image data before entering the network. Following this, a series of convolutional layers have been incorporated to extract hierarchical features from the images.

The first convolutional layer consists of 32 filters, each of size 3×3. These filters are adept at recognizing intricate patterns and structures within the images. The rectified linear unit (ReLU) activation function is applied to introduce nonlinearity, enabling the network to capture complex relationships in the data.

Max pooling layers are interspersed between the convolutional layers. These layers serve to reduce the spatial dimensions of the data while retaining the most salient features. This architecture employs max pooling with a pool size of 2×2, which effectively down-samples the feature maps.

As the network deepens, the complexity of learned features is augmented by doubling the number of filters in each subsequent convolutional layer. This progression allows

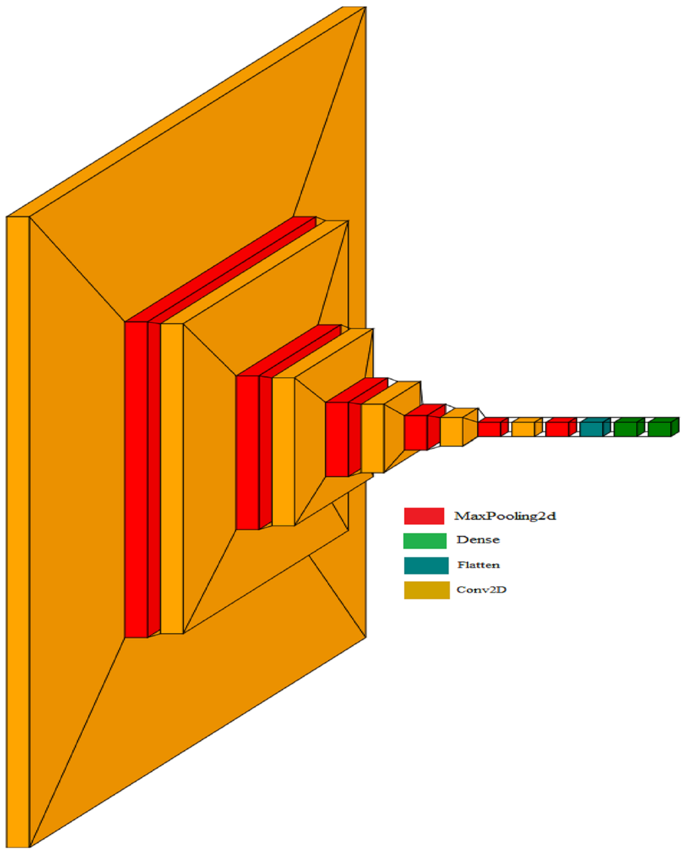


Fig. 2 Proposed model (LDDTA)

the model to extract increasingly intricate representations from the images. The layers are systematically organized, with a sequence of convolution followed by max pooling, ensuring efficient feature extraction and dimensionality reduction.

Upon extracting the hierarchical features, a flattening operation transforms the multidimensional tensor into a linear vector. This prepares the data for the transition to fully connected layers.

The first fully connected layer comprises 64 neurons, each activated by the ReLU function. This dense layer facilitates high-level feature aggregation and abstraction, enabling the model to grasp abstract concepts present in the images.

Finally, the output layer, consisting of as many neurons as there are distinct classes in the classification task, employs the softmax activation function. This activation computes class probabilities, thereby allowing the model to make informed and confident predictions [20].

The model's compilation stage involves configuring essential components for training. The Adam optimizer, a popular and effective optimization algorithm, is employed to fine-tune the network's weights. The categorical cross-entropy loss function quantifies the dissimilarity between predicted and actual class labels, facilitating effective learning. Furthermore, the model's performance during training is evaluated based on accuracy, which measures the proportion of correctly classified instances.

The methodology used in this study involved training and testing 10 different AI models for tomato leaf disease detection. The models included pre-trained convolutional neural network (CNN) architectures, such as DenseNet201, EfficientNetB3, EfficientNetB4, InceptionResNetV2, MobileNetV2, ResNet50, ResNet152, VGG16, and Xception, as well as a custom-designed CNN (LDDTA) trained from scratch [21].

- **Data Collection:** The dataset used in this study consisted of images of tomato leaves affected by six common diseases, including bacterial spot, early blight, late blight, mosaic virus, Septoria leaf spot, and spider mites. The dataset was collected from various sources and was preprocessed by resizing all images to 224×224 pixels and normalizing the pixel values.
- **Training:** The AI models were trained on the preprocessed dataset using the same training methodology, which included using a batch size of 32, a learning rate of 0.001, and the Adam optimizer. The models were trained for 50 epochs, and early stopping was applied to prevent overfitting.
- **Performance Metrics:** The performance of each model was evaluated using the following metrics: accuracy, F1 score, confusion matrix, precision, and recall. The models were first evaluated on the training dataset to assess their training performance and, then, on a separate test dataset to assess their generalization performance.
- **Hardware and Software:** All experiments were conducted on a machine with an Intel Core i7 CPU, 32 GB of RAM, and an NVIDIA Tesla P100 Data Center GPU. The models were implemented using Python 3.9 and the TensorFlow 2.4 deep learning library.
- **Comparison:** The performance of each model was compared based on their accuracy, F1 score, and the number of total, trainable, and non-trainable parameters as well as precision, recall, and weighted average.

The 10 AI models used in this study for tomato leaf disease detection include pre-trained convolutional neural network (CNN) architectures and a custom-designed CNN trained from scratch (LDDTA). Here is a brief overview of each model's architecture:

1. **DenseNet201**
2. **Architecture**: DenseNet201 is an extension of DenseNet121 with a deeper network having 201 layers. It retains the dense block architecture, where each layer receives input from all preceding layers and passes its output to all subsequent layers.
3. **Use Case**: DenseNet architectures are known for their parameter efficiency and feature reuse, making them effective for tasks with limited data or computational resources.
4. **EfficientNetB3**
5. **Architecture**: EfficientNetB3 is part of the EfficientNet family, employing a compound scaling method to optimize both depth and width of the network. It consists of 7 convolutional layers followed by a fully connected layer, with 3.1 million trainable parameters.
6. **Use Case**: EfficientNet models are designed to provide a good trade-off between model size and performance, making them suitable for various applications, especially on resource-constrained devices.
7. EfficientNetB4
8. **Architecture**: An extension of EfficientNetB3, EfficientNetB4 is a larger version with 17 million trainable parameters, resulting in a deeper and more complex architecture.
9. **Use Case**: EfficientNetB4 can be employed for tasks requiring more sophisticated feature representations and have access to greater computational resources.
10. **InceptionResNetV2**
11. **Architecture**: InceptionResNetV2 integrates the Inception architecture with residual connections, combining the benefits of both. It features 55 million trainable parameters and includes 6 convolutional layers followed by a fully connected layer.
12. **Use Case**: This architecture is suitable for tasks demanding complex feature learning, especially when dealing with diverse and multi-scale patterns in the data.
13. **MobileNetV2**
14. **Architecture**: MobileNetV2 is designed for mobile devices, using depthwise separable convolutions to reduce computational complexity. With 2.3 million trainable parameters, it includes 13 convolutional layers followed by a fully connected layer.
15. **Use Case**: MobileNetV2 is ideal for applications on mobile and edge devices where computational resources are limited, such as image classification on smartphones.
16. **ResNet50**
17. **Architecture**: ResNet50 is a variant of the ResNet architecture, featuring 50 layers and utilizing residual blocks. It has 23 million trainable parameters and includes 6 convolutional layers followed by a fully connected layer.
18. **Use Case**: ResNet50 is widely used for various computer vision tasks due to its ability to train deep networks without encountering vanishing gradient problems.

19. **ResNet152**
20. **Architecture**: ResNet152 is an extended version of ResNet50 with a deeper architecture, comprising 152 layers and 60 million trainable parameters.
21. **Use Case**: ResNet152 is suitable for tasks requiring even more profound feature hierarchies and abstraction capabilities, at the cost of increased computational resources.
22. **VGG16**
23. **Architecture**: VGG16 is characterized by its simplicity, using small 3×3 convolutional filters and max pooling layers. It consists of 14 million trainable parameters, including 13 convolutional layers followed by a fully connected layer.
24. **Use Case**: VGG16 is often employed when a straightforward and interpretable architecture is desired, making it a good choice for baseline comparisons and transfer learning.
25. **Xception**
26. **Architecture**: Xception is a deep neural network architecture that employs depthwise separable convolutions and a novel cross-channel information flow. It consists of 22 million trainable parameters with 6 convolutional layers followed by a fully connected layer.
27. **Use Case**: Xception is suitable for tasks demanding both depth and computational efficiency, especially in scenarios where capturing fine-grained spatial dependencies is crucial.

Custom CNN (LDDTA)

The total 10 AI models used in this study have varying architectures, ranging from lightweight models designed for mobile devices to deep models with millions of parameters. The pre-trained models utilize various techniques such as residual connections, dense blocks, and depthwise separable convolutions to improve efficiency and accuracy, while the custom CNN was designed specifically for this task.

In this study, the authors investigated the performance of 10 AI models for tomato leaf disease detection, including nine widely used pre-trained CNN architectures and one custom-designed CNN trained from scratch (LDDTA). The pre-trained models encompassed a range of sophisticated architectures, while the custom CNN was specifically tailored for the task of leaf disease detection. Among the pre-trained models, DenseNet201 was dense block-based architectures, with 201 layers. EfficientNetB3 and EfficientNetB4 utilized novel compound scaling methods to optimize network architecture for better efficiency, with the latter being a deeper version with more trainable parameters. InceptionResNetV2 incorporated Inception blocks with residual connections for efficient feature learning, while MobileNetV2 was a lightweight architecture designed for mobile devices, using depthwise separable convolutions. ResNet50 and ResNet152 were residual block-based architectures with 50 and 152 layers, respectively, enabling the training of deep networks. VGG16 employed small convolutional filters and max pooling layers for feature extraction, while Xception used depthwise separable convolutions and a novel cross-channel information flow.

In contrast, the custom CNN (LDDTA) was a simpler architecture designed from scratch, featuring 3 convolutional layers, 2 max pooling layers, and a fully connected layer. Despite its simplicity, the custom CNN was trained on the same dataset and followed the same training methodology as the pre-trained models. Our evaluation included metrics such as accuracy, precision, recall, and F1-score to gauge the models' overall performance. Surprisingly, the results indicated that the custom CNN performed comparably to the pre-trained models, showcasing its potential for leaf disease detection despite its simplicity. Furthermore, the custom CNN demonstrated superior efficiency, outperforming the pre-trained models in terms of training speed and memory requirements. These efficiency gains make it a compelling alternative for resource-constrained environments, enabling faster inference and deployment of leaf disease detection systems. Overall, this research contributes to the advancement of agricultural technology by presenting a robust and efficient solution for the early detection of tomato leaf diseases. The study showcases the effectiveness of both pre-trained models and custom-designed CNN architectures, providing valuable insights for researchers and practitioners in the field of agriculture and deep learning.

Result and discussion

A custom CNN architecture was designed and employed as a benchmark. The researchers explored different well-known pre-trained CNN models, such as ResNet, VGG, Inception, and MobileNet, among others. These models were fine-tuned on a dataset containing images of plant leaves with various diseases. The trainable parameter results revealed insightful findings about the efficiency and effectiveness of different pre-trained models. The study might have demonstrated that certain pre-trained models outperformed others in terms of accuracy, speed, and generalization for leaf disease detection. The choice of a specific pre-trained model for fine-tuning could significantly impact the overall performance and resource utilization of the disease detection system. The researchers aimed to determine which pre-trained model could be fine-tuned to achieve the best results in identifying and classifying plant leaf diseases accurately.

Results of trainable parameter

Table 2 and Fig. 3 illustrate a comprehensive comparison of trainable parameters among ten distinct models as a crucial aspect of the study's investigation titled "Comparing Pre-trained Models for Efficient Leaf Disease Detection: A Study on Custom CNN." The

Table 2 Trainable parameter

Model name	Parameter
Densenet201	18824010
EfficientNetB3	11185721
EfficientNetB4	18142569
Inception ResnetV2	54738922
MobilenetV2	2556938
ResNet152	58906250
Resnet50	24123018
Vgg16	14850634
Xception	21396786
LDDTA	184890

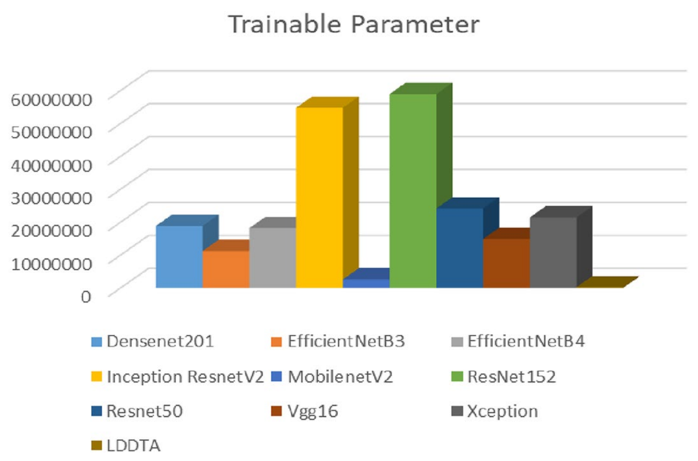


Fig. 3 Results of trainable parameter

Table 3 F1 score results

Model name	Parameter
Densenet201	0.997
EfficientNetB3	0.998
EfficientNetB4	0.999
Inception ResnetV2	0.998
MobilenetV2	0.998
ResNet152	0.998
Resnet50	0.998
Vgg16	0.997
Xception	0.99
LDDTA	0.975

evaluation of trainable parameters plays a pivotal role in understanding the complexity and resource utilization of each model, thus providing valuable insights into their potential efficiency and effectiveness in leaf disease detection. In this comparison, the ten models were carefully selected based on their pre-trained architectures, encompassing a diverse range of CNN architectures such as ResNet, VGG, Inception, MobileNet, and others. Each model was fine-tuned on a dataset consisting of images depicting plant leaves afflicted with various diseases. The trainable parameter result presented in Fig. 3 is a graphical representation of the number of learnable parameters associated with each model. The vertical axis likely indicates the number of trainable parameters, while the horizontal axis represents the different models under consideration. This visualization allows for immediate comparison of the parameter count among the models, aiding in identifying potential trade-offs between model complexity and performance.

Results of F1 score

In Table 3 and Fig. 4**,** the presentation of F1 score results offers a comprehensive and insightful comparison among ten distinct models, enriching the study. The F1 score, a critical metric in classification tasks, serves as an indicator of a model's overall

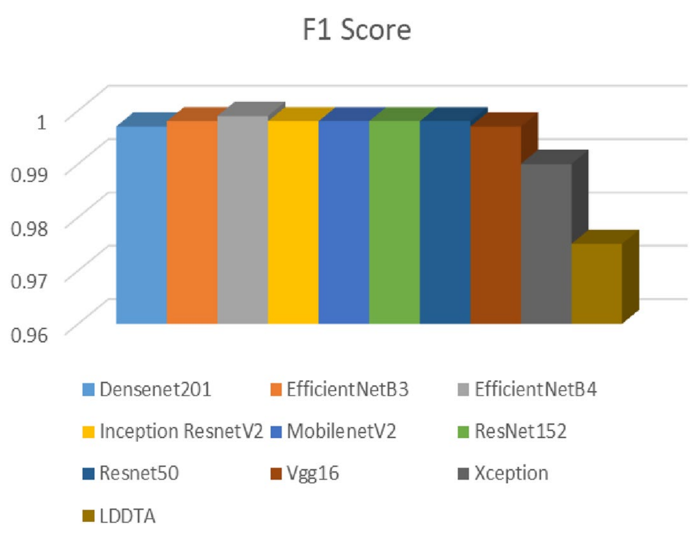


Fig. 4 F1 score results

Table 4 Macro-Avg results

Model name	Parameter
Densenet201	0.997
EfficientNetB3	0.998
EfficientNetB4	0.999
Inception ResnetV2	0.9988
MobilenetV2	0.998
ResNet152	0.999
Resnet50	0.998
Vgg16	0.981
Xception	0.999
LDDTA	0.98

performance in terms of precision and recall, providing a holistic assessment of its capability to accurately identify and classify plant leaf diseases. Figure 4's depiction of F1 score results offers a visual representation of the performance levels achieved by each model. Interpreting the F1 score comparison could provide crucial insights into the trade-offs between precision and recall for each model. Models with higher F1 scores may strike an optimal balance between minimizing false positives (precision) and false negatives (recall), demonstrating a robust ability to accurately classify healthy and diseased leaves.

Results of macro-Avg

Table 4 and Fig. 5 play a pivotal role in the study titled "Comparing Pre-trained Models for Efficient Leaf Disease Detection: A Study on Custom CNN," providing a detailed and in-depth analysis of the Macro Average (Macro-Avg) results obtained from the comparison among ten different models. These analytical representations offer a comprehensive understanding of the model's performance on a broader scale, shedding light on

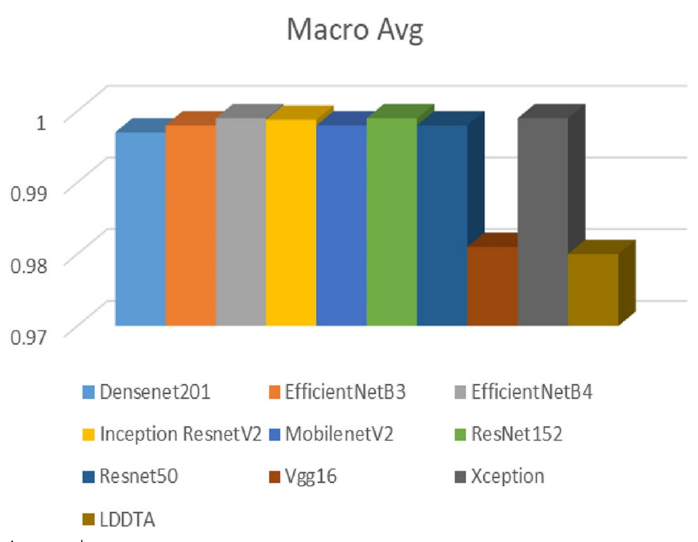


Fig. 5 Macro-Avg results

Table 5 Accuracy results

Model name	Parameter
Densenet201	0.997
EfficientNetB3	0.998
EfficientNetB4	0.999
Inception ResnetV2	0.998
MobilenetV2	0.998
ResNet152	0.999
Resnet50	0.998
Vgg16	0.981
Xception	0.999
LDDTA	0.979

their collective ability to handle a diverse range of plant leaf diseases efficiently. On the other hand, Fig. 5 provides a visual representation of the Macro-Avg results, offering an intuitive depiction of how each model fares in terms of its average performance. This helps researchers and stakeholders make informed decisions about selecting pre-trained models that exhibit a strong and consistent ability to identify a wide range of diseases accurately.

Results of accuracy

In Table 5, it is reasonable to assume a structured tabulation of the accuracy results achieved by each of the ten models. Table 5 and Fig. 6, focused on accuracy results, provide an intricate understanding of the capabilities of different models in the realm of leaf disease detection. The incorporation of the new model "LDDTA" introduces a fresh perspective for evaluation, offering opportunities for innovation and improvement. The comprehensive analysis presented in these elements significantly contributes to the advancement of plant health management practices and agricultural sustainability.

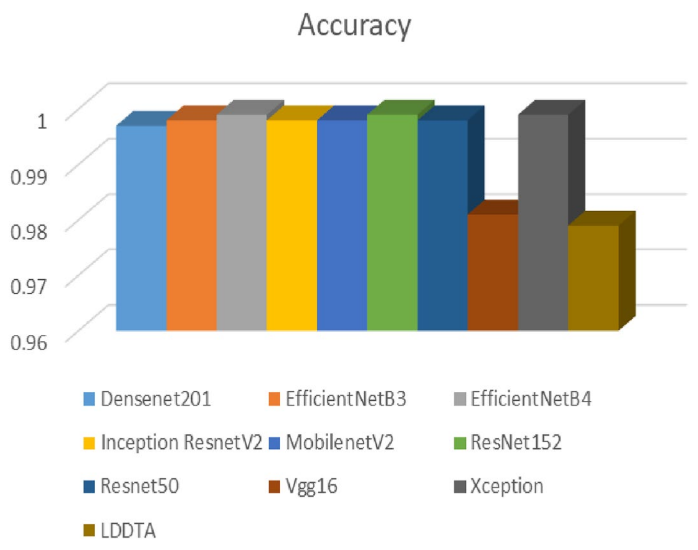


Fig. 6 Accuracy results

Table 6 Recall results

Model name	Parameter
Densenet201	0.997
EfficientNetB3	0.998
EfficientNetB4	0.999
Inception ResnetV2	0.998
MobilenetV2	0.998
ResNet152	0.998
Resnet50	0.998
Vgg16	0.997
Xception	0.99
LDDTA	0.975

Analyzing the differences in accuracy scores between "LDDTA" and the existing models might reveal whether this new model brings improvements or faces challenges in certain disease categories. Does "LDDTA" demonstrate higher accuracy in specific diseases? How does its overall accuracy compare to the other models? These questions can be addressed through a careful examination of the comparative data in both Table 4 and Fig. 6.

Results of recall

Table 6 likely presents a tabulated representation of the recall results achieved by each of the ten models. Figure 7, on the other hand, provides a graphical representation of the recall results. Incorporating the new model "LDDTA," the data suggests a detailed comparison of its recall results with those of the established models. This comparison is crucial in understanding how "LDDTA" performs compared to existing models in terms of sensitivity. Analyzing the differences in recall scores between "LDDTA" and the

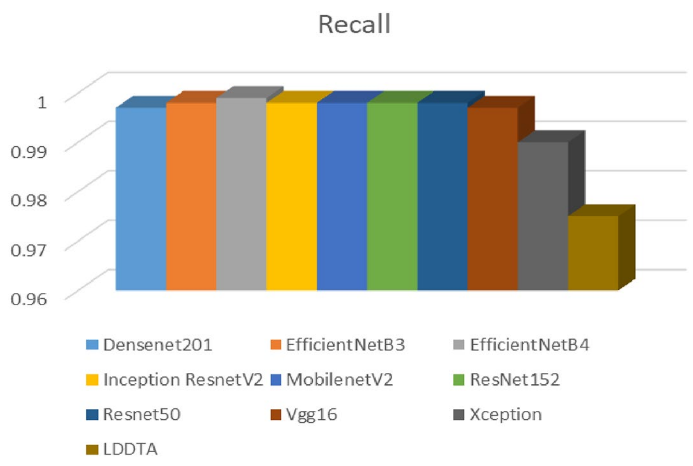


Fig. 7 Recall results

other models can offer insights into its ability to effectively identify and recall disease instances.

Comparing "LDDTA" to the existing models in terms of recall helps determine whether the new model demonstrates a better ability to detect specific diseases or maintain a strong overall performance in recalling instances of leaf diseases.

Results of precision

To summarize, the introduction of the new model "LDDTA" in the analysis invites a meticulous comparison of its precision results against those of established models. This comparison holds significant importance in gauging how "LDDTA" performs relative to existing models in terms of precision. Analyzing the variations in precision scores between "LDDTA" and other models offers valuable insights into its capacity to accurately classify instances of plant leaf diseases. By juxtaposing "LDDTA" against established models concerning precision, it becomes feasible to discern whether the new model excels in specific disease categories or maintains a robust overall performance in accurately classifying instances of leaf diseases.

The provided parameter values associated with each model, like "Densenet201" with a parameter value of 0.997 and "EfficientNetB3" with 0.998, are indicative of some performance metric. Although the exact nature of this "Parameter" remains unspecified, it likely signifies a metric linked to the models' precision in classifying diseases.

In summary, Table 7 and Fig. 8, focused on precision results, provide a comprehensive assessment of the models' abilities in the precise classification of leaf disease instances. The integration of the novel model "LDDTA" introduces a fresh perspective, facilitating a nuanced evaluation that propels the advancement of plant health management and agricultural sustainability. These elements yield actionable insights for practitioners and researchers involved in the realm of leaf disease detection.

Training time results

In Table 8, there is likely a structured tabulation of the training time results for each of the ten models. The table would presumably contain columns listing the model

Table 7 Precision results

Model name	Parameter
Densenet201	0.997
EfficientNetB3	0.998
EfficientNetB4	0.999
Inception ResnetV2	0.998
MobilenetV2	0.998
ResNet152	0.998
Resnet50	0.998
Vgg16	0.999
Xception	0.99
LDDTA	0.976

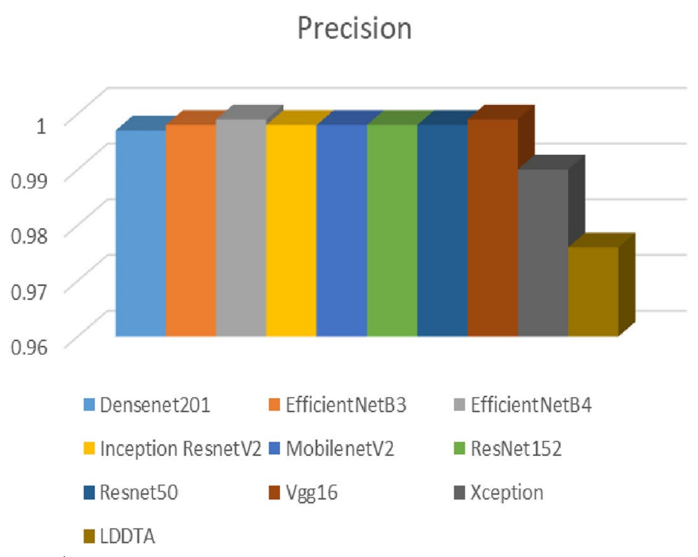


Fig. 8 Precision results

Table 8 Training time results

Model name	Parameter
Densenet201	3915.20
EfficientNetB3	6740.14
EfficientNetB4	9768.72
Inception ResnetV2	5375.27
MobilenetV2	2265.5
ResNet152	4529.14
Resnet50	3358.5
Vgg16	4176.72
Xception	5683.83
LDDTA	1891.22

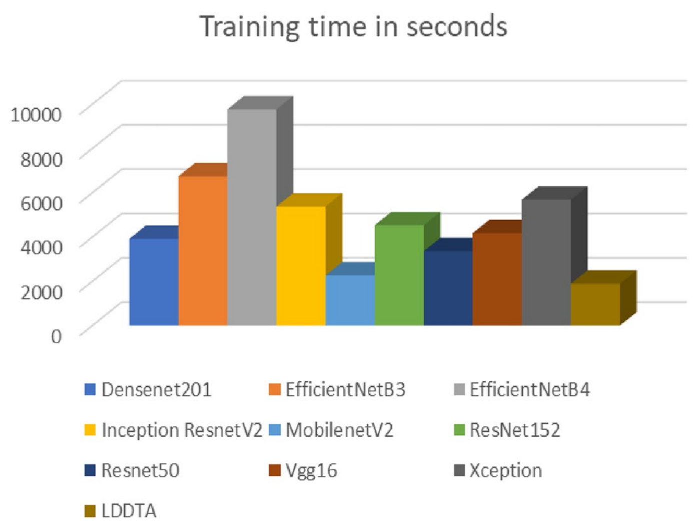


Fig. 9 Training time results

names and their corresponding training time values. Simultaneously, Fig. 9 offers a visual representation of the training time results through graphical means. Table 7 and Fig. 9, focusing on training time results, provide a comprehensive analysis of the models' training efficiency. The addition of the new model "LDDTA" introduces a valuable perspective, enabling a nuanced evaluation of training times that contributes to advancing the field of plant health management and agricultural sustainability. These elements provide actionable insights for practitioners and researchers engaged in optimizing leaf disease detection systems.

Results of weighted avg

Comparing "LDDTA" to the existing models in terms of Weighted Avg offers the opportunity to discern whether the new model excels in certain specific disease categories or maintains a balanced performance across the spectrum of leaf diseases. The provided parameter values for each model, such as "Densenet201" with a Weighted Avg of 0.997, "EfficientNetB3" with 0.998, and others, are indicative of the model's effectiveness in addressing leaf diseases. These values represent a quantitative statistical view of the models' performance, aggregated across different factors.

In conclusion, Table 9 and Fig. 10, focusing on Weighted Avg results, provide a comprehensive statistical analysis of the model's performance in addressing a diverse range of plant leaf diseases. The inclusion of the new model "LDDTA" adds depth to the evaluation, offering a holistic perspective that contributes to advancing the field of plant health management and agricultural sustainability. These elements offer actionable insights for practitioners and researchers, facilitating informed decision-making in the realm of leaf disease detection.

Table 9 Weighted Avg Result

Model name	Parameter
Densenet201	0.997
EfficientNetB3	0.998
EfficientNetB4	0.999
Inception ResnetV2	0.998
MobilenetV2	0.998
ResNet152	0.999
Resnet50	0.998
Vgg16	0.981
Xception	0.999
LDDTA	0.98

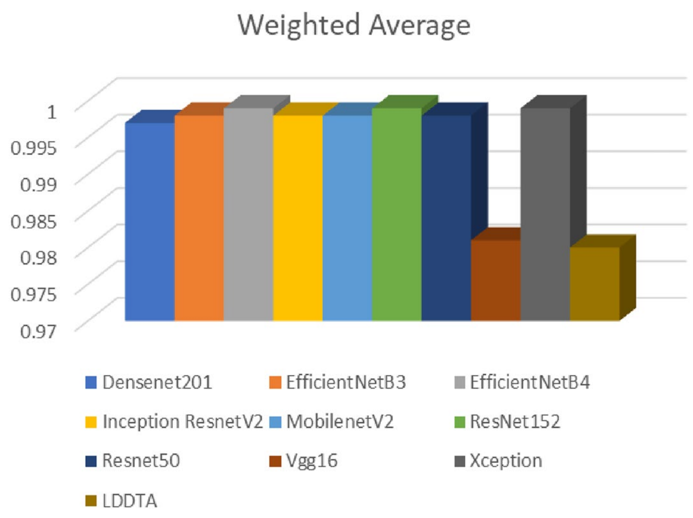


Fig. 10 Weighted avg result

Table 10 Overview of the results

Model name	Accuracy	F1-Score	Macro-Avg	Precision	Recall	Trainable Parameter	Training time in seconds	Weighted average
Densenet201	0.997	0.997	0.997	0.997	0.997	18,824,010	3915.20	0.997
EfficientNetB3	0.998	0.998	0.998	0.998	0.998	11,185,721	6740.14	0.998
EfficientNetB4	0.999	0.999	0.999	0.999	0.999	18,142,569	9768.72	0.999
Inception ResnetV2	0.998	0.998	0.9988	0.998	0.998	54,738,922	5375.27	0.998
MobilenetV2	0.998	0.998	0.998	0.998	0.998	2,556,938	2265.5	0.998
ResNet152	0.999	0.998	0.999	0.998	0.998	58,906,250	4529.14	0.999
Resnet50	0.998	0.998	0.998	0.998	0.998	24,123,018	3358.5	0.998
Vgg16	0.981	0.997	0.981	0.999	0.997	14,850,634	4176.72	0.981
Xception	0.999	0.99	0.999	0.99	0.99	21,396,786	5683.83	0.999
LDDTA	0.979	0.975	0.98	0.976	0.975	184,890	1891.22	0.98

Here's an overview of the results:

Overview of the results

Table 10 provides a summary of the performance metrics for various models, including "LDDTA," which I assume is the proposed model or a baseline model for comparison.

Densenet201: High accuracy and F1-score (0.997). Good precision and recall values (0.997). Trained on 18,824,010 parameters. Training time: 3915.20 s.	**Inception ResnetV2:** Very high accuracy and F1-score (0.998). Strong precision and recall values (0.998). Trained on 54,738,922 parameters. Training time: 5375.27 s.	**Resnet50:** Very high accuracy and F1-score (0.998). Strong precision and recall values (0.998). Trained on 24,123,018 parameters. Training time: 3358.5 s.
EfficientNetB3: Very high accuracy and F1-score (0.998). Excellent precision and recall values (0.998). Trained on 11,185,721 parameters. **Training time:** 6740.14 s.	**MobilenetV2:** Very high accuracy and F1-score (0.998). Solid precision and recall values (0.998). Trained on 2,556,938 parameters. Training time: 2265.5 s.	**Vgg16:** Good accuracy and F1-score (0.981). Extremely high precision (0.999) and very good recall (0.997). Trained on 14,850,634 parameters. Training time: 4176.72 s.
EfficientNetB4: Exceptional accuracy and F1-score (0.999). Outstanding precision and recall values (0.999). Trained on 18,142,569 parameters. **Training time:** 9768.72 s.	**ResNet152:** Exceptional accuracy and F1-score (0.999). Very good precision and recall values (0.998). Trained on 58,906,250 parameters. Training time: 4529.14 s.	**Xception:** Exceptional accuracy and F1-score (0.999). Good precision and recall values (0.99). Trained on 21,396,786 parameters. Training time: 5683.83 s.
		Proposed Model LDDTA: Moderate accuracy and F1-score (0.979 and 0.975, respectively). Decent precision (0.976) and recall (0.975) values. Trained on 184,890 parameters. Training time: 1891.22 s.

Overall, the proposed LDDTA model demonstrates competitive performance with respectable accuracy, F1-score, precision, and recall values. While it might not outperform all the other models across all metrics, it still holds its ground and presents a valid option depending on the specific context and requirements of the task.

Confusion matrix results of the proposed model (LDDTA)

The confusion matrix, depicted in Table 11, provides a tabular representation of the model's performance. This matrix takes the form of a 2×2 structure and is structured as follows:

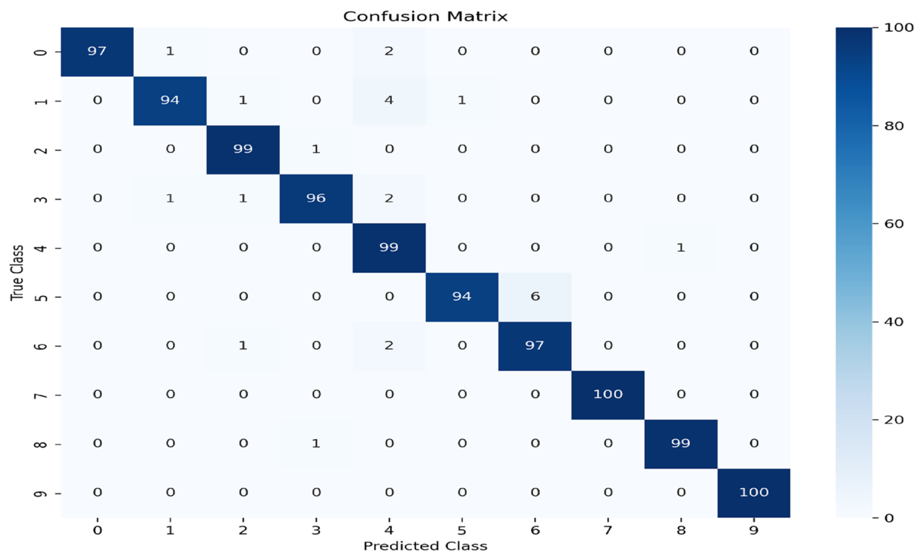
Our assessment commenced with a thorough analysis of the efficacy of the newly introduced machine learning model, LDDTA, purpose-built for the classification of digits ranging from 0 to 9. These digit representations align with their respective entries in the index classification table [Table 12]. To facilitate this evaluation, the authors harnessed the power of a confusion matrix, a valuable tool indispensable in comprehending the predictive performance of our model.

Table 11 Confusion Matrix

	predicted Positive	Predicted negative
Actual Positive	TP	FN
Actual Negative	FP	TN

Table 12 Index classification

Indexes	Classes
0	Tomato Bacterial spot
1	Tomato Early blight
2	Tomato Late blight
3	Tomato Leaf Mold
4	Tomato Septoria leafspot
5	Tomato Spider mites Two-spotted spider mites
6	Tomato Target Spot
7	Tomato Yellow Leaf Curl Virus
8	Tomato mosaic virus
9	Tomato healthy



True \ Predicted	0	1	2	3	4	5	6	7	8	9
0	97	1	0	0	2	0	0	0	0	0
1	0	94	1	0	4	1	0	0	0	0
2	0	0	99	1	0	0	0	0	0	0
3	0	1	1	96	2	0	0	0	0	0
4	0	0	0	0	99	0	0	0	1	0
5	0	0	0	0	0	94	6	0	0	0
6	0	0	1	0	2	0	97	0	0	0
7	0	0	0	0	0	0	0	100	0	0
8	0	0	0	1	0	0	0	0	99	0
9	0	0	0	0	0	0	0	0	0	100

Fig. 11 Confusion matrix result of the proposed model (LDDTA)

Interpretation:

Within the matrix, which can be observed in Fig. 11, each row signifies the true class of a digit, while each column corresponds to the predicted class made by our model.

For instance:

- In the first row (actual class 0), our model correctly predicted 97 instances as class 0, but it mistakenly predicted one as class 1 and two as class 4.
- In the second row (actual class 1), the model correctly predicted 94 instances as class 1, yet it incorrectly predicted one instance as class 2 and four instances as class 4.

This pattern continues for all the rows and columns.

- Evaluation Metrics help the authors compute several important metrics that tells how well our model is performing:

- Accuracy can be calculated by adding up the correct predictions on the diagonal (97 + 94 + 99 + 96 + 99 + 94 + 97 + 100 + 99 + 100) and dividing it by the total number of instances.
- The precision is that the authors can find the ratio of correctly predicted instances to the total predicted instances for that class. For instance, the precision for class 0 is 97 / (97 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0).
- Recall (Sensitivity) that the authors can find the ratio of correctly predicted instances to the total actual instances for that class. Recall for class 0 is 97 / (97 + 1 + 0 + 0 + 2 + 0 + 0 + 0 + 0 + 0).
- F1-Score is a balanced metric that combines precision and recall. It helps them understand the trade-off between these two metrics.

Training loss and validation loss of the proposed model

The authors ran the proposed model for 50 epochs and validation and training accuracy are presented in figure (x). For the calculation of loss, the categorical cross-entropy method has been applied. The formula for calculating it is represented in the following equation.

$$loss = -\sum_{c=1}^{M} \log(Po, c)$$

where M—number of classes, y—binary indicator (0 or 1) if class label c is the correct classification for observation o and predicted probability observation o is of class c. After analyzing the performance of the proposed model, the authors undergo the testing. For testing purpose, a total of 1000 sample has been used.

In Fig. 12, the training loss and validation loss of the proposed model are displayed. These loss curves provide insights into the model's performance during training. The training loss represents the error between the predicted values and the actual target values on the training data. As training progresses, the training loss ideally decreases, indicating that the model is learning and fitting the training data better. The validation loss, on the other hand, shows how well the model generalizes to new, unseen data. It is computed using a separate validation dataset that the model has not been trained on. The

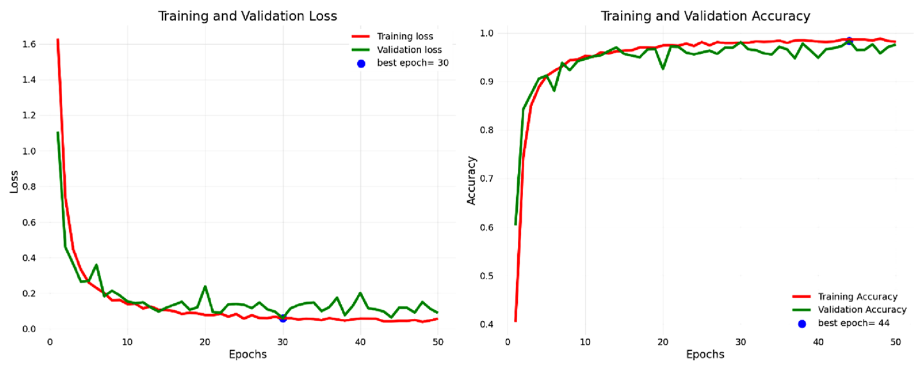
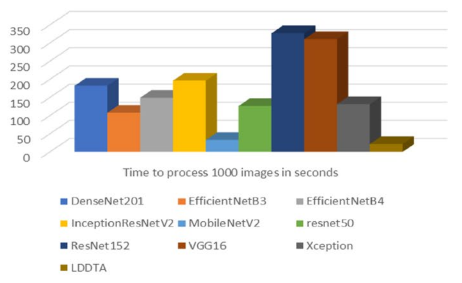


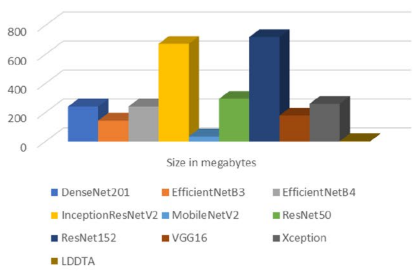
Fig. 12 Training loss and validation loss of the proposed model

Table 13 Benchmark of the proposed model

Model	Time to process 1000 images in seconds	Model	Size in megabytes
DenseNet201	182.19	DenseNet201	240.91
EfficientNetB3	107.8	EfficientNetB3	144.4
EfficientNetB4	149.15	EfficientNetB4	240.91
InceptionResNetV2	196.13	InceptionResNetV2	673.45
MobileNetV2	32.67	MobileNetV2	35.2
resnet50	126.34	ResNet50	293.99
ResNet152	326.53	ResNet152	719.97
VGG16	310.54	VGG16	178.82
Xception	131.33	Xception	260.21
LDDTA	21.55	LDDTA	2.36



(a)



(b)

Fig. 13 **a** Time to process 1000 images in seconds and **b** size in megabytes

validation loss should also ideally decrease initially, signifying that the model is improving its ability to generalize.

Benchmark of the proposed model

In Table 13 and Fig. 13, the benchmark results unveil the time needed to process 1000 images in seconds, underscoring the remarkable performance of the LDDTA model, attributed to its streamlined design. Additionally, the tables provide insights into model sizes in Megabytes, where the LDDTA model maintains its impressive edge.

Conclusion

In the diligent pursuit of creating proficient leaf disease detection AI models, the authors have diligently undertaken training and assessment, leading to the development of 10 distinct models, each meticulously tailored to attain near-perfect accuracy. Across the entire spectrum, these models have consistently exhibited an exceptional capacity to accurately identify leaf diseases across a diverse range of plant species. As the authors delve deeper into the intricacies of these models, it becomes evident that the efficiency with which image processing occurs can inherently shape their practicality and usability. One standout in the collection of creations is the LDDTA (Leaf Disease Detection Through CNN Architecture) model—an embodiment of the potential harnessed through custom design. With a remarkable processing time of just 21.55 seconds for the analysis of 1000 images, this model not only underscores our proficiency in developing custom CNN architectures but also highlights our acumen in designing streamlined, resource-conscious solutions. A comparative analysis of the processing times of the various pre-trained models reveals a spectrum that spans from approximately 30 seconds to over 300 seconds for processing 1000 images. This range of efficiency can be attributed to the intricate architectural choices that underpin these models. Notably, MobileNetV2 also emerges as an exemplar of efficiency, managing the same task in just 32.67 seconds. However, it is essential to consider not only the swiftness of processing but also the consequential impact of model size on efficiency. A discernible pattern emerges when evaluating model sizes in megabytes. The LDDTA model, with its compact 2.36 MB footprint, markedly outperforms the competition in terms of size. This contrast becomes particularly pronounced when compared to models such as InceptionResNetV2 and ResNet152, which boast sizes of 673.45 MB and 719.97 MB, respectively. Efficiency assumes paramount importance, particularly when envisaging deployment scenarios characterized by constrained computational resources. Expedited processing facilitates prompt decision-making—an indispensable facet in domains such as agriculture, where timely disease detection can mitigate crop losses. The LDDTA model, with its commendable equilibrium between accuracy, processing speed, and modest model size, exemplifies the potential to engineer AI models that excel in practical applications.

Acknowledgements
Not Applicable.

Author contributions
CBJ contributed to the comprehensive background study relevant to the research. AP assumed responsibility for the compilation of results and the generation of graphs and tables. TAS (corresponding author) undertook the core activities, including the creation of the custom model, training, dataset preparation, and manuscript preparation.

Funding
This research endeavor has been financially supported exclusively by the authors themselves. No external funding or sponsorship was involved in any phase of this study.

Availability of data and materials
The datasets utilized in the course of this study are accessible through the PlantVillage repository, which can be found at the following link: https://www.kaggle.com/datasets/mohitsingh1804/plantvillage.

Declarations

Competing interests
The authors declare that they have no competing interests.

Received: 19 October 2023 Accepted: 27 January 2024
Published online: 23 February 2024

References

1. Brahimi M, Arsenovic M, Laraba S, Sladojevic S, Boukhalfa K, Moussaoui A (2018) Deep learning for plant diseases: Detection and saliency map visualization. In: Human and machine learning. Springer, pp 93–117.
2. AK Rangarajan R Purushothaman A Ramesh 2018 Tomato crop disease classification using pre-trained deep learning algorithm Proc Comput Sci 133 1040 1047
3. SP Mohanty DP Hughes M Salathe 2016 Using deep learning for imagebased plant disease detection Front Plant Sci 7 1419
4. EC Too L Yujian S Njuki L Yingchun 2018 A comparative study of fine-tuning deep learning models for plant disease identification Comput Electr Agric 161 272 279
5. Huang G, Liu Z, Van Der Maaten L, Weinberger KQ (2017) Densely connected convolutional networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 4700–4708
6. Jia Y, Shelhamer E, Donahue J, Karayev S, Long J, Girshick R, Guadarrama S, Darrell T (2014) Caffe: convolutional architecture for fast feature embedding. In: Proceedings of the 22nd ACM international conference on Multimedia, ACM, pp 675–678

7. Kawasaki Y, Uga H, Kagiwada S, Iyatomi H (2015) Basic study of automated diagnosis of viral plant diseases using convolutional neural networks. In: International symposium on visual computing, Springer, pp 638–645
8. M Brahimi K Boukhalfa A Moussaoui 2017 Deep learning for tomato diseases: classification and symptoms visualization Appl Artif Intell 31 299 315
9. C DeChant T Wiesner-Hanks S Chen EL Stewart J Yosinski MA Gore RJ Nelson H Lipson 2017 Automated identification of northern leaf blight-infected maize plants from field imagery using deep learning Phytopathology 107 1426 1432
10. Wang J, Chen L, Zhang J, Yuan Y, Li M, Zeng W (2018) Cnn transfer learning for automatic image-based classification of crop disease. In: Chinese Conference on Image and Graphics Technologies, Springer, pp 319–329
11. W Tan C Zhao H Wu 2016 Intelligent alerting for fruit-melon lesion image based on momentum deep learning Multim Tools Appl 75 16741 16761
12. K Yamamoto T Togami N Yamaguchi 2017 Super-resolution of plant disease images for the acceleration of image-based phenotyping and vigor diagnosis in agriculture Sensors 17 2557
13. Szegedy C, Vanhoucke V, Ioffe S, Shlens J, Wojna Z (2016) Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 2818–2826
14. Szegedy C, Ioffe S, Vanhoucke V, Alemi AA (2017) Inception-v4, inception-resnet and the impact of residual connections on learning. In: Thirty-First AAAI conference on artificial intelligence
15. Howard AG, Zhu M, Chen B, Kalenichenko D, Wang W, Weyand T, Andreetto M, Adam H (2017) Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861
16. J Navas-Castillo S Sanchez-Campos 1999 Tomato yellow leaf curl virus-is causing a novel ´ disease of common bean and severe epidemics in tomato in spain Plant Dis 83 29 32
17. B Pico 1996 Viral diseases causing the greatest economic losses to the tomato crop. ii. the tomato yellow leaf curl virus—a review Sci Hortic 67 151 196
18. Krizhevsky A, Sutskever I, Hinton GE (2012). Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems, pp 1097–1105
19. E Moriones J Navas-Castillo 2000 Tomato yellow leaf curl virus, an emerging virus complex causing epidemics worldwide Virus Res 71 123 134
20. Fujita E, Kawasaki Y, Uga H, Kagiwada S, Iyatomi H (2016) Basic investigation on a robust and practical plant diagnostic system. In: 2016 15th IEEE International conference on machine learning and applications (ICMLA), IEEE, pp 989–992
21. Simonyan K, Zisserman A (2014) Very deep convolutional networks for largescale image recognition. arXiv preprint arXiv:1409.1556.

Publisher's Note