A HYBRID MODEL FOR PLANT DISEASE DETECTION BASED ON DEEP LEARNING

Zainab Fadhıl Abbas ISHKAYYIR

Computer Engineering Department

Master's Degree Thesis

Supervisor: Assoc. Prof. Dr. Tolga AYDIN

May 2025

T.C.
ATATURK UNIVERSITY
GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCE
DEPARTMENT OF COMPUTER ENGINEERING

A HYBRID MODEL FOR PLANT DISEASE DETECTION BASED ON DEEP LEARNING

(Derin Öğrenmeye Dayalı Bitki Hastalıkları Tespiti İçin Hibrit Bir Model)

Master's Degree Thesis

Zainab Fadhıl Abbas ISHKAYYIR

Supervisor: Assoc. Prof. Dr. Tolga AYDIN

Erzurum

May, 2025

KABUL VE ONAY TUTANAĞI

Zainab Fadhıl ISHKAYYIR tarafından hazırlanan Derin Öğrenmeye Dayalı Bitki Hastalıkları Tespiti için Hibrit Bir Model başlıklı çalışması 02 /05 / 2025 tarihinde yapılan tez savunma sınavı sonucunda başarılı bulunarak jürimiz tarafından Bilgisayar Mühendisliği Ana Bilim Dalında Yüksek Lisans tezi olarak oybirliği / oy çokluğu (…/…) ile kabul edilmiştir.

Jüri Başkanı:	Doç. Dr. Ferhat BOZKURT Atatürk Üniversitesi	………..................
Danışman:	Doç. Dr. Tolga AYDIN Atatürk Üniversitesi	………..................
Jüri Üyesi:	Dr. Saffet Gökçen ŞEN Erzurum Teknik Üniversitesi	………..................

Bu tezin Atatürk Üniversitesi Lisansüstü Eğitim ve Öğretim Yönetmeliği'nin ilgili maddelerinde belirtilen şartları yerine getirdiğini onaylarım.

Prof. Dr. Saltuk Buğrahan CEYHUN

Enstitü Müdürü

Not: Bu tezde kullanılan özgün ve başka kaynaklardan yapılan bildiriş, çizelge, şekil ve fotoğrafların kaynak olarak kullanımı, 5846 sayılı Fikir ve Sanat Eserleri Kanunundaki hükümlere tabidir.

ETİK BİLDİRİM VE İNTİHAL BEYAN FORMU

Yüksek Lisans Tezi olarak Doç. Dr. Tolga AYDIN danışmanlığında sunulan Derin Öğrenmeye Dayalı Bitki Hastalıkları Tespiti İçin Hibrit Bir Model başlıklı çalışmanın tarafımızdan bilimsel etik ilkelere uyularak yazıldığını, yararlanılan eserlerin kaynakçada gösterildiğini, Fen Bilimleri Enstitüsü tarafından belirlenmiş olan Turnitin Programı benzerlik oranlarının aşılmadığını ve aşağıdaki oranlarda olduğunu beyan ederiz.

Tez Bölümleri	Tezin Benzerlik Oranı (%)	Maksimum Oran (%)
Giriş	*11*	**30**
Kuramsal Temeller	*7*	**30**
Materyal ve Metot	*20*	**35**
Sonuçlar ve Tartışma	*9*	**20**
Karar ve Gelecek Çalışmalar	*2*	**20**
Tezin Geneli	*18*	**25**

Not: *Yedi kelimeye kadar benzerlikler ile Başlık, Kaynakça, İçindekiler, Teşekkür, Dizin ve Ekler kısımları tarama dışı bırakılabilir. Yukarıdaki azami benzerlik oranları yanında tek bir kaynaktan olan benzerlik oranlarının* ***%5'den*** *büyük olmaması gerekir.*

Beyan edilen bilgilerin doğru olduğunu, aksi halde doğacak hukuki sorumlulukları kabul ve beyan ederiz.

Tez Yazarı (Öğrenci)	**Tez Danışmanı**
Zainab Fadhıl ISHKAYYIR	Doç. Dr. Tolga AYDIN
02.05.2025	**02.05.2025**
İmza:	İmza:

* Tez ile ilgili YÖKTEZ'de yayınlamasına ilişkin bir engelleme var ise aşağıdaki alanı doldurunuz.

☐ Tezle ilgili patent başvurusu yapılması / patent alma sürecinin devam etmesi sebebiyle Enstitü Yönetim Kurulunun …./…/…. tarih ve …………. sayılı kararı ile teze erişim 2 (iki) yıl süreyle engellenmiştir.

☐ Enstitü Yönetim Kurulunun …./…/…. tarih ve …………. sayılı kararı ile teze erişim 6 (altı) ay süreyle engellenmiştir.

ACKNOWLEDGEMENTS

I want to express my heartfelt thanks to my supervisor, Assoc. Prof. Dr. Tolga AYDIN, for his constant assistance, mentorship, and insightful perspectives during this research work.

Additionally, I express my gratitude to Assoc. Prof. Dr. Ferhat BOZKURT, thank you for your exceptional contribution. Furthermore, I thank Research Assistant Ebru ERDEM for her support and comments on enhancing my thesis.

Zainab Fadhıl ISHKAYYIR

ÖZET

YÜKSEK LİSANS TEZİ

DERİN ÖĞRENMEYE DAYALI BİTKİ HASTALIKLARI TESPİTİ İÇİN HİBRİT BİR MODEL

Zainab Fadhıl ISHKAYYIR

Danışman: Doç. Dr. Tolga AYDIN

Amaç: Son yıllarda, bitki hastalıklarını erken tespit etmek ve tarım ürünlerinin kalitesini artırmak için makine öğrenimi tekniklerini kullanmaya olan ilgi artmıştır. Çiftçilerin makine öğrenimi tekniklerinden memnuniyetini değerlendirmeye yönelik artan bir ihtiyaç vardır. PlantVillage veri seti, bitki hastalıklarını tanımlamak için bir araç olarak seçilmiştir.

Yöntem: Bitki hastalıklarının sınıflandırılması amacıyla Plant Village veri seti kullanılmıştır. Sınıflandırma için ResNet50, VGG16, VGG19, MobileNet, DenseNet121, InceptionV3, ConvNext gibi çeşitli derin öğrenme tabanlı evrişimli sinir ağı modelleri kullanılmıştır. Bu modeller, ImageNet veri kümesi üzerinde önceden eğitilmiş ağırlıklarla başlatılmış ve PlantVillage veri seti üzerinde transfer öğrenme yöntemiyle yeniden eğitilmiştir. Sonrasında, CNN tabanlı özellik çıkarımı sonrasında elde edilen öznitelikler kullanılarak SVM, KNN, Random forest, Logistic regression, Decision tree gibi klasik makine öğrenmesi sınıflandırıcıları eğitilerek hibrit modeller geliştirilmiştir.

Bulgular: İnce ayar yapılmamış en iyi model MobileNet modellerinin metrikleri %97,70 doğruluk, %97,50 kesinlik ve %97,30 geri çağırma, %97,40 F1 puanı ve %99,98 Roc-AUC' dir. Bu sonuçlar modelin yedi modelden daha iyi performans gösterdiği anlamına gelir. Önerdiğimiz hibrit model metrikleri: Doğruluk: %98,46, Kesinlik, geri çağırma ve F1 puanı: %98, Roc-Auc puanı %99,97'dir.

Sonuç: Önerilen hibrit model üstün doğruluk sağladı ve ayrıca tarımsal izleme için iyi donanımlı, birden fazla bitki hastalığının etkili bir şekilde tespit edildiğini gösterdi. Bu çalışma, gelecekte tarımsal hastalıkların izlenmesi için çok umut verici bir yol açıyor.

Anahtar Kelimeler: Bitki hastalığı, CNN, Derin öğrenme, Önceden eğitilmiş modeller, Hibrid modeller, MobileNet, SVM

May 2025, 75 sayfa

ABSTRACT

MASTER'S THESIS

A HYBRID MODEL FOR PLANT DISEASE DETECTION BASED ON DEEP LEARNING

Zainab Fadhıl ISHKAYYIR

Supervisor: Assoc. Prof. Dr. Tolga AYDIN

Purpose: In recent years, interest in using machine learning techniques to detect plant diseases early and improve the quality of agricultural products has grown. There is a growing need to assess farmers' satisfaction with machine learning techniques. The PlantVillage dataset was chosen as a tool for identifying plant diseases.

Method: PlantVillage dataset was used for the classification of plant diseases. Various deep learning based convolutional neural network models, such as ResNet50, VGG16, VGG19, MobileNet, DenseNet121, InceptionV3 and ConvNeXt, were used for classification. These models were initialized with pre-trained weights on ImageNet dataset and retrained on PlantVillage dataset using the transfer learning method. Then, hybrid models were developed by training classical machine learning classifiers such as SVM, KNN, Random Forest, Logistic Regression, and Decision Tree using the features obtained after CNN-based feature extraction.

Findings: The best model that was not fine-tuned is the MobileNet model, with metrics of accuracy of 97.70%, precision of 97.50%, recall of 97.30 %, an F1 score of 97.40%, and ROC-AUC score of 99.98%. These results mean that the model outperformed the seven models. Our proposed hybrid model metrics are: Accuracy: 98.46%; Precision, recall, and F1 score: 98%; ROC-AUC score is 99.97%.

Results: This hybrid model delivered superior accuracy and effectively detected multiple plant diseases, making it well-equipped for agricultural monitoring. This study opens a promising avenue for monitoring agricultural diseases in the future.

Keywords: Plant disease, CNN, Deep learning, Pre-trained models, Hybrid models, MobileNet, SVM

May 2025, 75 pages

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS AND ABBREVIATIONS

CNN	: Convolutional Neural Network
DL	: Deep Learning
ML	: Machine Learning
Relu	: Rectified Linear Unit
PL	: Pooling Layer
FC	: Fully Connected Layers
LF	: Loss Function
SGD	: Stochastic Gradient Descent
BN	: Batch Normalization
ES	: Early Stop
TSWV	: Tomato Spotted Wilt Virus
EWADEL	: Efficient Weighted Average Deep Ensemble Learning

INTRODUCTION

Background

Plant diseases are a significant challenge in agriculture, affecting both the quality and quantity of crop yields. Like humans, plants undergo specific development stages, making them more vulnerable to various diseases. These diseases can lead to a substantial reduction in crop output and net profit for farmers. Therefore, the Early detection of plant diseases is crucial in reducing the economic losses associated with agriculture and safeguarding food security. Food security is where all individuals consistently enjoy physical, social, and economic access to sufficient, safe, and nutritious food that meets their dietary needs and personal preferences, enabling an active and healthy lifestyle (FAO, 2002). The early identification of plant diseases allows farmers to implement timely preventive measures, thereby reducing crop losses and promoting sustainable agricultural practices.

The effects of plant pathogens can vary from mild symptoms to devastating occurrences, destroying extensive areas of food crops. Catastrophic plant diseases exacerbate food supply gaps where at least 800 million people are undernourished (Strange, 2005). Traditionally, experts in agriculture diagnosed plant diseases through ocular examination. Manual diagnostics have several limitations, the most important of which is that they are prone to human error. Moreover, skilled experts cannot be made accessible with complete assurance to remote areas where agriculture is more feasible. Thus, the implementation of manual diagnostic processes would not be straightforward. These challenges depict a need that necessitates automated plant disease detection systems to be accurate and efficient to help farmers in decision-making on a real-time basis.

The most recent developments in machine learning, intense learning techniques, have shown considerable potential in automating the identification of plant diseases through image classification. It has been demonstrated that Convolutional Neural Networks (CNNs), which are highly effective for image processing tasks, play a crucial role in this process and have demonstrated extraordinary effectiveness in identifying and categorizing plant diseases. Artificial neural networks (CNNs) can automatically learn complicated information from photos, enabling them to accurately differentiate between healthy and diseased plant leaves. CNNs constitute one of the most potent techniques for modeling complex processes and performing pattern recognition in applications with a large amount of data, like pattern recognition in images. However, many of the deep learning models that are now in use have many drawbacks, including the fact that they require a large number of training parameters, that they require a high amount of computational power, and that they provide a restricted level of classification accuracy when applied to the detection of plant diseases.

The performance of several traditional pattern recognition techniques was compared with the performance of CNN models in plant identification, and it was concluded that CNNs significantly outperformed the traditional methods (Pawara et al, 2017). This research investigates applying seven cutting-edge pre-trained CNN models to classify plant diseases. These models are ResNet-50, VGG-19, VGG-16, MobileNet, DenseNet121, ConvNext and Inception V3. Pre-trained models are beneficial since they enable better generalization even when trained on relatively smaller datasets like PlantVillage. This is because they utilize features learnt from large datasets, which allows them to leverage the features learnt from large datasets. By fine-tuning these models for plant disease classification, the objective is to get a high level of accuracy while simultaneously reducing the time needed for training and the amount of computer resources needed.

Every one of the pre-trained models utilized in this investigation possesses distinctive qualities that render them appropriate for endeavors. For example, ResNet-50 solves the vanishing gradients problem through residual learning. It enables going deeper without losing network performance. On the one hand, VGG-19 and VGG-16 were called very user-friendly and efficient models for extracting features hierarchically from an image. Conversely, Inception V3 combined a big set of features using different filter sizes with parallel processing. Also, the MobileNet ConvNext and DenseNet models are good to use. Because of this diversity, these designs are particularly well-suited for combination, as each model might bring different key strengths to bear in the classification task.

This further ascertains the potential of hybrid models, considering the encouraging outcomes these different individual models have produced. The best-performing model, MobileNet, was combined with Logistic Regression, Decision Tree, KNN, SVM, and Random Forest to create five hybrid architectures, aiming to identify the most effective model. Hybrid model development is driven by the need to realize better classification performance if the characteristics of diverse architectures can be combined. A hybrid approach tries to collect a combined set of feature vectors salvaged from multiple models to concatenate them, further improving plant disease classification's accuracy and robustness.

Objective

The project's main objective will be a flexible, efficient, and scalable identification of plant diseases by achieving the following goals:

1. This work will be achieved by building an efficient hybrid model combining the power of different CNN architectures. This will contribute to developing agricultural technologies and assist farmers, especially in low-input areas, by providing them with a user-friendly tool for early disease diagnosis.
2. The long-term objective is to enhance agricultural productivity with minimal economic loss due to plant diseases, helping ensure global food supply security.
3. The widespread use of deep learning algorithms in various fields of life to estimate and develop agricultural technologies and assist farmers.
4. Improving the performance of agricultural material analysis techniques by developing a hybrid machine learning model.

Problem Statement

Plant diseases vary from place to place and are the leading cause of crop failure. However, another challenge may arise in identifying and treating these diseases. While numerous studies have explored hybrid deep learning models for plant disease identification, the present study introduces a novel framework that integrates the strengths of both machine learning and deep learning techniques. (Chug, 2022).

This study introduces a novel approach to machine learning and deep learning techniques that help farmers identify plant diseases and treat agricultural pests that affect plants. Machine learning and deep learning provide a comprehensive view of plant diseases. Similarly, machine learning and deep learning techniques collect and distinguish between plant images, thus identifying plant diseases early, allowing for prompt diagnosis and effective treatment.

Methodology

The proposed method starts by training individual CNN models and generating hybrid models from the best-performing model. We used deep learning techniques and deep learning models to analyze plant diseases.

We used a 20638 plant leaf images dataset to train and evaluate the models. This method combines advanced deep learning features. In addition, we developed hybrid models such as MobileNet, ResNet-50, VGG-16, and VGG-19 with weights pre-trained on ImageNet to assist in the early diagnosis of plant diseases.

The Contribution of the Thesis

This study analyzes plant diseases to determine the satisfaction of agricultural field owners with machine learning technologies. Our work can be summarized in the following subpoints:

1. This thesis explores deep learning techniques (machine learning techniques) to enable farmers to benefit from deep learning technologies.
2. This thesis evaluates hybrid models of MobileNet, Resnet-50, VGG-16, VGG-19, DenseNet121, ConvNext, and Inception V3.
3. This research is based on a dataset collected from Plant Village.
4. This work provides valuable insights into plant disease knowledge, enabling farmers to better interact with deep learning techniques. The ultimate beneficiaries of this research will be farming owners and shop owners, as the research will provide them with helpful information.

THEORETICAL FOUNDATION

This chapter will examine the standards utilized for categorizing plant disease. Furthermore, it comprehensively examines deep learning technology, specifically focusing on two convolutional neural network (CNN) models while addressing network training concerns. Subsequently, we will delve into software testing and metrics and present a methodology based on design models.

Deep Learning

Artificial Intelligence is a field dedicated to automating cognitive processes traditionally performed by humans. Within this domain, Machine Learning (ML) and Deep Learning (DL) represent key methodologies for achieving this goal (Choi and Coyner, 2020). Deep learning is a branch of machine learning that aims to mimic the activity of neurons in the human brain. Machine learning focuses on learning new things by analyzing existing data and finding patterns.

Deep learning may automatically extract data, leading to very accurate results. Feature extraction from a vast collection of dimensions, regardless of their immediate visibility, is within deep learning capabilities. One day, clinical decision-making powered by deep learning could be just as good as human experts. Computer vision and image processing are only one of many uses for deep learning systems, which find use in fields like Medical X-ray detection. Optimal result prediction, illness progression probability, and the efficient creation of trustworthy medical images have all benefited from deep learning's application in the medical field. Machine Learning (ML) algorithms construct models or derive rules by analyzing training datasets that consist of input-output data pairs and their corresponding outcomes (Balali et al, 2020).

Deep learning, a subfield of machine learning, seeks to emulate the neural architecture of the human brain to enable hierarchical learning and representation. Deep Learning algorithms use hierarchical learning architecture inspired by artificial intelligence's attempt to mimic the brain’s neocortex, where the primary sensory areas learn in deep, layered stages automatically extracting features and abstractions from raw data (Arel et al,2010).

Deep learning algorithms mimic technological processes and adhere to data mining concepts. Machine learning is primarily concerned with recognizing patterns and learning from data. Analytical thinking and data collection are integral to the machine learning area of study. Automatic feature extraction and reliable findings are both possible with deep learning. Deep learning can extract characteristics from several dimensions regardless of their appearance. The unification of ML features is an effective way to address these issues.

Convolutional Neural Network Concept

Image classification and segmentation are broad computer vision applications in which Convolutional Neural Networks (CNNs) are extensively used. Convolutional neural networks (CNNs) perform very well on grid-like pictures. They are a special type of multilayer neural network or deep learning architecture inspired by the visual system of living beings (Ghosh et al., 2019).

Convolution is at the center of a convolutional neural network. In convolution, a set of filters (more often called kernels or feature detectors) processes the input data. These filters are small matrices multiplied and added element by element as they are convolved with the input data. The convolution process produces a feature map focusing on the input data's specific characteristics.

Deep Convolutional Neural Networks (CNNs) are a type of representation learning algorithm that automatically learn and extracts meaningful information from raw images, eliminating the need to craft feature descriptors manually (Khan et al, 2020)

CNNs are constructed using convolutional layers, which generally involve a number of these filters. Be it edges, corners, or textures, each filter is set to detect some particular feature or pattern in the arriving data at a specific spatial position. By convolving over the input data, each of the various filters will produce feature maps sensitive to features of the same kind but will tolerate small changes in the neighborhood. By employing many such filters, a CNN learns the feature representation step by step in a hierarchically complex manner. Following the convolutional layers, CNNs frequently employ pooling layers to reduce the spatial dimensionality of feature maps while retaining essential information. Typically, only the highest or average value within a particular region is maintained when data from a feature map is pooled, and all other values are discarded. Reducing the network's computational load and improving translation invariance are both achieved by down-sampling. Translation invariance is the capacity to detect patterns regardless of their position in the input. In recent years, convolutional neural networks have aimed to deliver greater efficiency and accuracy across various domains, notably object detection, digit recognition, and image recognition (Ajit et al, 2020).

A CNN's typical architecture resembles a standard neural network, beginning with convolutional and pooling layers and having one or more fully connected layers. The data from those fully linked layers, which have been trained on the more abstract features, may be used to make predictions. This encourages researchers to extract discriminative features with minimal human effort and domain knowledge.

Activation functions like SoftMax are commonly used in the final layer of Convolutional Neural Networks (CNNs) to provide the desired output: class probabilities in image classification. During training, a Convolutional Neural Network (CNN) minimizes the discrepancy between its predictions as well as the actual labels in the training data, adjusting its parameters (weights and biases) accordingly. This procedure is known as back-propagation. The network may learn from its experiences and apply that knowledge to novel contexts, eventually outperforming its training data. This is achieved by repeatedly adjusting the settings until the loss function minimizes. The activation function enhances the neural network's expressive power, enabling it to model complex patterns and effectively contribute to the realization of artificial intelligence (Wang et al, 2020).

Convolution is capable of processing multi-dimensional inputs. In a typical neural network, all input units communicate with all the output units. Look at Figure 1. Overall, Convolutional Neural Networks (CNNs) have dramatically improved computer vision. Equipped to glean vital details from photos automatically, they have made state-of-the-art performance on various jobs possible. Convolutional neural networks (CNNs) can effectively recognize complex visual patterns because of their hierarchical design, which incorporates convolution as well as pooling processes. Since the advent of Convolutional Neural Networks (CNNs), there has been a significant surge in deep learning research, which persists today. (Ketkar and Moolayil, 2021).

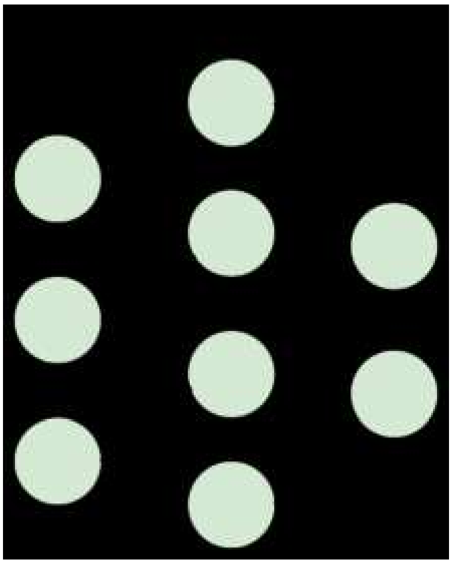

Figure 1. A concrete illustration of an artificial neural network featuring a concealed layer

With each input node representing an input, the neural network's hidden layers and non-linear activation functions subtly alter the input picture or feature bus. A network of neurons ensures that all cells below a given sublayer are in constant contact. Finally, A fully connected neural network exhibits its learned classification capabilities at the output layer. Many factors, including picture resolution, input amount, weight quantity, number of hidden layers, and total nodes, make it challenging to scale a neural network correctly. Figure 2 shows several computer vision applications that have recently shown promise for CNNs using a forward propagation architecture.

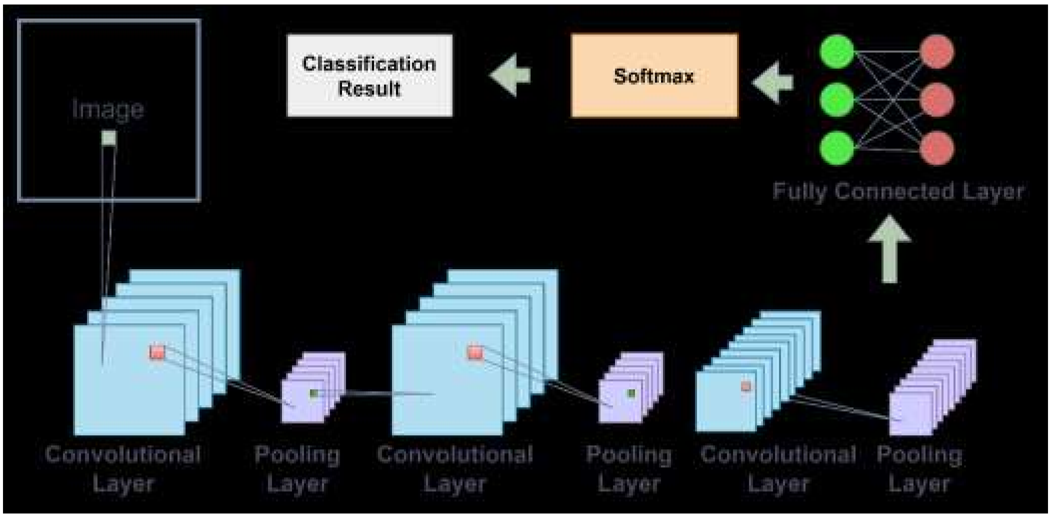

Figure 2. CNN, in its most basic form of conception

The composition and organization of a Convolutional Neural Network (CNN) are as follows:

Layers for Convolution

A CNN's processing power is in the convolutional layer. A coupled convolutional layer consisting of multiple parallel convolution operations using mutually constrained filters is suggested (Uchida et al, 2018). This crucial network component processes data with a filter as well as a feature map. Envision the input as a fragmented, 3D color picture with the exact dimensions as a standard RGB picture. One way to determine if an image has a specific feature is to use a feature detector, which may be called a kernel or filter.

The feature detector uses a two-dimensional weighted array to characterize a specific portion of the picture. A 3x3 matrix is the default but adjustable receptive field size, which is determined by the filter size. After applying the filter to a selected picture region, the output is computed by taking the dot product of the input pixels as well as the filter. The dot product is input to the output array. The image is then iterated over as the filter moves to the next level.

Convolved features, activation maps, and feature maps are all examples of such maps. Many Convolutional Neural Network (CNN) models incorporate non-linearity through ReLU modifications to the feature map after each convolution operation, like in Figure 3. This change happens after the first convolution layer. If there are more convolution layers, the information from the receptive fields of the previous levels may be accessed by the succeeding ones. CNN's hierarchical features make it resemble a tree. Take the challenge of identifying bicycles in pictures as an example. The four main parts of a bicycle (the frame, the handlebars, the wheels, and the pedals) are seen as simple structures. These parts are classified as lower-level patterns in the CNN's feature hierarchy, whereas the complete bicycle is classified as higher-level. Convolutional layer serves as the initial level for measurement extraction from input photos (Sawane et al, 2024).

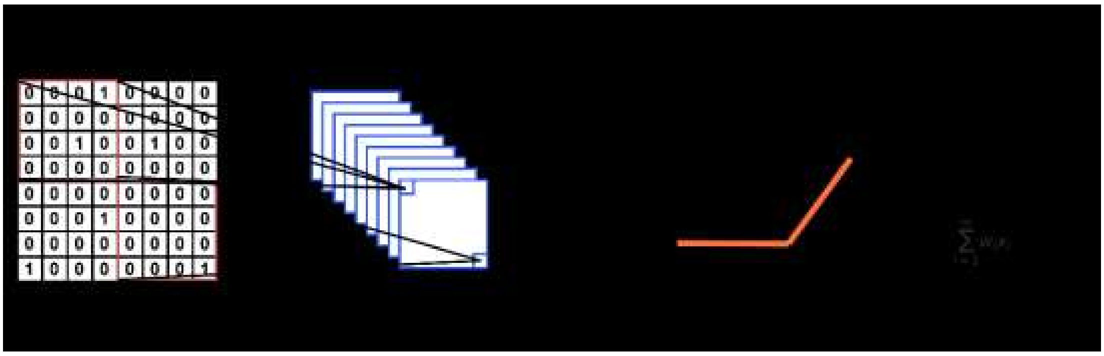

Figure 3. Rectifier linear unit

A non-linear function could manage non-linear issues efficiently via maintaining as well as representing the activation function of a triggered neuron. The output value in neural network models is calculated using an activation function that considers the weighted sum of the neuron inputs. Plant traits are often presumed to have a linear relationship with environmental gradients like grazing intensity. However, to extend findings beyond individual study sites, it is crucial to understand the degree to which environment-trait relationships deviate from linearity (Saatkamp et al, 2010).

Activation Functions

The selection of activation functions in deep neural networks plays a pivotal role in shaping the training dynamics and significantly influences overall task performance.

Activation functions enhance the model's capacity to capture complex patterns, underscoring neural networks' expressive power. In Convolutional Neural Networks (CNNs), each trainable layer, convolutional or fully connected, is typically followed by a nonlinear activation layer, which introduces nonlinearity and enables the network to model intricate relationships within the data.

The activation function plays a crucial role in building a deep neural network. Since convolution is a linear process, a non-linear activation function is applied afterward to introduce non-linearity and forward the result. In the past, mathematical representations of the body's neural activity were based on smooth, non-linear functions.

Many current network topologies use the Rectified Linear Unit (ReLU) as their activation function. Convolutional Neural Networks (CNNs) can achieve greater efficiency under certain conditions. One such condition arises when a few neurons are simultaneously affected by linear behavior in the positive domain and inactivity in the negative domain, resulting in a highly efficient and uniformly distributed network. This behavior is primarily attributed to the asymmetry of the activation function, which contributes to its non-linearity. Unlike sigmoid functions exhibiting gradient saturation at both extremes, the Rectified Linear Unit (ReLU) does not saturate in the positive region. However, it has a zero gradient in the negative direction.

$$f(x)_{ReLU=max(0,x)} \tag{1}$$

The sigmoid activation function has been enhanced with the SoftMax function for applications requiring classification across several classes. The Convolutional Neural Network (CNN) uses the SoftMax function after the final, completely connected layer. The function's curve has a point of inflection. Here is a mathematical representation of the sigmoid function:

$$\sigma(x) = \frac{\mathbf{1}}{1+e^{-x}} \tag{2}$$

The Tanh activation function restricts the input values (real numbers) from -1 to 1. It is represented by the formula (3) in decimal notation.

$$f(x)_{\tanh} = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{3}$$

Pooling Layer (PL)

To reduce the dimensionality of feature maps generated by convolutional layers, Convolutional Neural Networks (CNNs) incorporate pooling layers. Techniques such as max pooling and average pooling are applied within local receptive fields to downsample the spatial dimensions of the input, thereby decreasing computational complexity and promoting spatial invariance. Using pooling for dimensionality reduction makes pattern identification translation invariant, meaning it works regardless of orientation. For computer vision tasks, it is usual practice to place pooling layers after convolutional layers to improve feature extraction and maximize processing efficiency (Gholamalinezhad and Khosravi, 2020).

The filter processes the values within the receptive field as well as transmits their average to the output array while scanning the input. Although pooling layers introduce some data loss, they provide significant advantages to convolutional neural networks (CNNs) by reducing the risk of overfitting, optimizing computational processes, and enhancing overall efficiency. Pooling operations are commonly categorized, with max-pooling being a widely used type.

Fully Connected Layers (FC)

A key component of neural networks, such as CNNs, is thick layers that are fully connected. Each neuron is fully connected to the preceding and succeeding layers ' neurons. These layers serve as intermediaries between the extracted feature maps and the final output predictions, facilitating the interpretation of complex patterns and relationships within the data. Activation functions incorporate non-linearity, and the network parameters are fine-tuned throughout the training process. In the last stage of the network, particularly in the fully linked layers, the learnt high-level characteristics are used to generate predictions. Shown in Figure 4. is the pattern of connections between neurons, which illustrates the architectural architecture of completely linked layers Convolutional Neural Networks (CNNs), particularly in fields like computer vision, have significantly minimized the reliance on handcrafted features by learning task-specific features directly from raw input data (Basha et al, 2020).

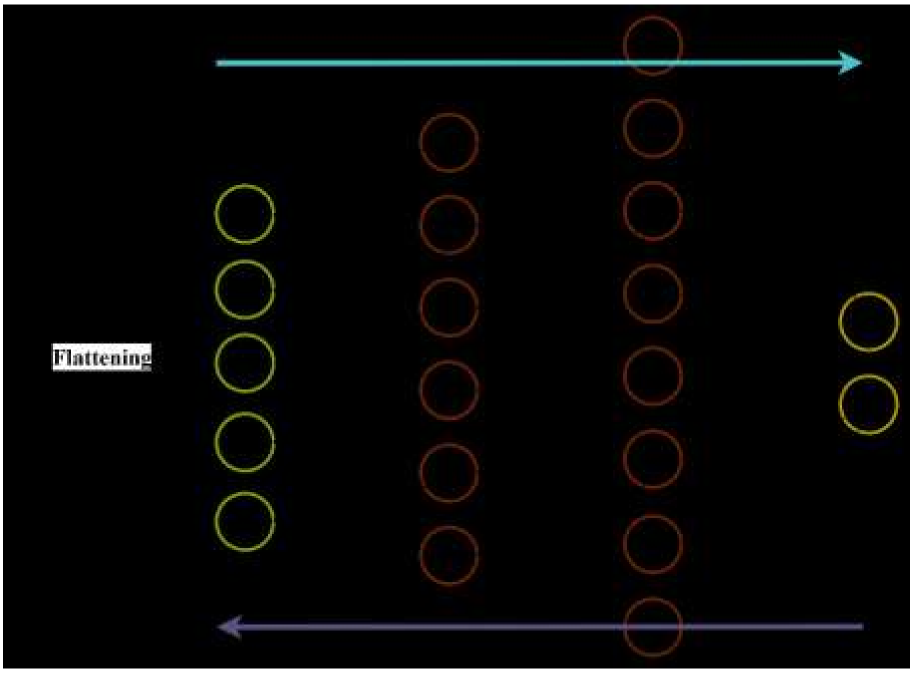

Figure 4. Architecture of fully connected layers

Hyperparameters And Model Parameters

A model's parameters are its changeable parts that, when changed, can provide different results. Hyperparameters are predefined settings that may be used to structure the training process. There is no automated way to learn the hyperparameters during training; they must be specified directly. After the hyperparameters have been fine-tuned, the improvements in output and efficiency become obvious.

Loss Function (LF)

The loss function plays a pivotal role in shaping and guiding the development of machine learning algorithms (Wang et al, 2022). It is an essential component in training machine learning and deep learning models and is often called the cost or objective function. The loss function quantitatively measures the discrepancy between the model's predicted output and the actual or expected outcome. During training, the model iteratively adjusts its parameters to minimize this discrepancy. For each input instance, the loss function takes the predicted output (commonly denoted as ŷ) as well as the actual output (denoted as y) as inputs and returns a scalar value that reflects the error between them.

Loss functions should be employed to optimize during the training procedure. They assist the learning algorithm in modifying the model's parameters to minimize overall loss by assessing its performance. The optimal model parameters are determined by minimizing the loss function, resulting in the best fit for the training data.

Obstacles and Educational Networks

One method for training neural networks involves integrating the weights of fully connected layers with the kernels used in convolutional layers. Backpropagation remains a fundamental technique in this process, as it effectively reduces the error between predicted outputs and actual labels within the training dataset. Constructing such empirical relationships presents significant challenges, requiring sophisticated modeling strategies, domain expertise, and informed intuition (Goh, 1995).

Optimization fundamentally depends on the loss function and the gradient descent algorithm. After the kernel parameters and weights are initialized, the model employs a loss function alongside forward propagation to produce the final output based on the training dataset. Learnable parameters, including kernels as well as weights, are subsequently refined through minimizing the loss function using gradient descent and backpropagation methods.

Training a convolutional neural network model enhances its predictive capabilities. If the loss on the training set exceeds an acceptable threshold, which is argued should not happen, then the model exhibits underfitting of the data (Hinton, 2006). Recently, a method was proposed that uses learning at each stage of a deep network.

Overfitting arises when a network exhibits remarkable performance on the training data but struggles to generalize to a new dataset, like the validation data. This leads to a decline in the network's capacity for generalization. Overfitting occurs when a significant disparity exists between the error rates seen during training and validation. Figure 5. Illustrates the potential range of models for underfitting and overfitting:

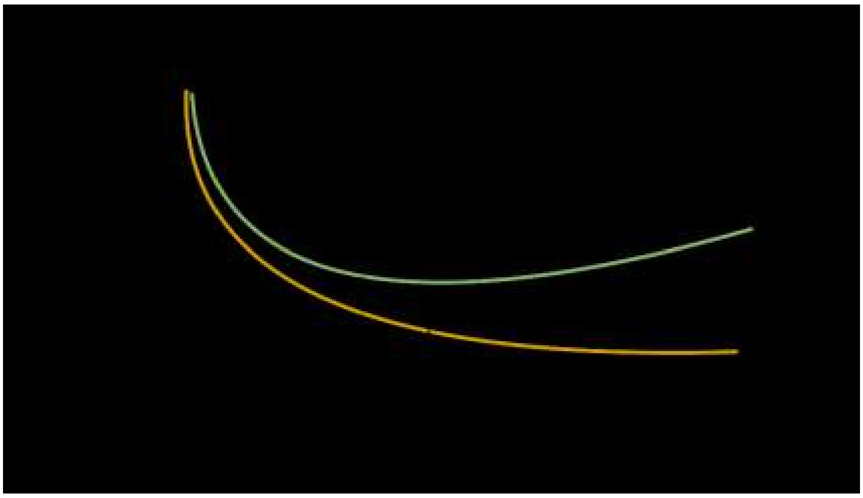

Figure 5. Avoiding overfitting and underfitting

Non-homogeneous characteristics, such as the severity, variety, and tissue diversity of distinct diseases, further complicate the already challenging task of diagnosing any illness. Data separation from a single source into training, validation, and testing sets is another difficult aspect.

The trustworthiness of the training data is paramount, especially when the model is to be applied in a context with significant consequences, such as medicine. Due to the large number of parameters that must be updated, learned, as well as specified, as well as the scarcity of readily available training data, overfitting is a common problem in deep neural networks. According to popular belief, neural networks can better generalize when there is more variation in the shape, size, as well as location of the affected, as well as in the intensity and variance. Deep learning requires more data than standard deep learning.

Representative training data sets are significant for performing all classification methods (Kavzoglu ,2009). Most challenges can be overcome, and a realistic model can be constructed by training neural networks with many inputs. Using a validation set, the hyperparameters as well as effectiveness of the trained neural network are fine-tuned as well as assessed. Using a test set that is bigger than the training data set is negative when training a neural network, since the training data is scarce.

Using higher-quality data for training and validation increases the data quantity fourfold. To achieve this, you can either increase or decrease the image's size, flip it horizontally or vertically, crop it, or rotate it. Consequently, training is necessary for a machine learning model to achieve success.

- Spend as little time as possible training.
- This will speed up the transition from training to testing.

In deep learning, it is a common practice to use overparameterized networks and continue training them for extended periods to achieve optimal performance (Chen et al, 2022). By adding more depth, neurons, removing regularization, etc., a network's optimal capacity can be surpassed. As the model's performance improves, the training error (loss) and validation error (Accuracy) start to seem more different. The objective is to minimize the disparity while preserving the model's generalizability. If this inconsistency is not fixed, the Overfitting component will be allowed to enter. Here, we can expect the training loss to remain constant or even go down, but the validation loss will either remain unchanged or go up. Accumulating validation mistakes over time strongly indicates overfitting. There are typically two approaches to Overfitting:

- Because it contains fewer layers and neurons, a shallow network is ideal for reducing the model's complexity.

- Apply some method of regularization.

Although neural networks may operate with fewer datasets, they maintain their superiority. Appropriate strategies are also required to prevent model inadequacies and increase generalization ability. Regularization techniques such as weight decay, dropout, and data augmentation should be employed for this. Regularization methods are generally more favorable than huge network sizes to check overfitting. The vanishing gradient problem may occur when training a deep CNN with many layers (exceeding 1500). The weights of each neuron in a layer must be updated by calculating the loss gradients concerning their respective weights. Due to the diminishing gradient effect during the network's backward propagation, the neurons in the initial layers will experience minimal updates to their weights. They will acquire knowledge of these tiers gradually and effectively.

One commonly used approach is the Rectified Linear Unit (ReLU) or its variant, leaky ReLU, which differ from traditional activation functions such as sigmoid and tanh. Additionally, incorporating batch normalization layers has been suggested as an effective technique to mitigate this issue. The buildup of substantial error gradients in backpropagation results in substantial modifications to network weights, causing the model to become unstable and impeding further training or enhancement.

This scenario presents an issue of explosive behavior, which is the antithesis of a problem involving a diminishing gradient. The two possible ways to solve this problem are changing the network's architecture or applying weight regularization methods. The entire training process mainly comprises four steps: preparation of the data, augmentation of the data, initialization of parameters, CNN regularization, and selection of an optimizer.

Initialization of Parameters

In recent years, neural networks have delivered outstanding performance across various applications in machine learning and computer vision (Narkhede et al, 2022). A deep CNN has many parameters, which may reach millions or even billions. Initialization is one of the keys to CNN's success; hence, it must be done before training starts. One of the possible initial steps for the network is to randomize the parameters of the CNN. A deep CNN usually contains many parameters, up to millions or even billions. Before training a CNN model, careful tuning and optimization should be performed in advance. Using a stochastic seed is one common way to initialize parameters in CNNs. Notable variants of this approach involve the Gaussian distribution, the uniform distribution, and the orthogonal distribution. However, one of the key disadvantages of randomization is that it can shrink or increase gradients.

Choosing an Optimizer

In the era of artificial intelligence, efficiently managing large volumes of data presents a highly motivating yet complex challenge. Stochastic gradient descent (SGD) stands out for its simplicity and effectiveness among machine learning techniques. CNNs use inherent optimizers to monitor and enhance convergence, enhancing classification accuracy. RMSProp, Adam, and SGD utilization as optimizers have recently increased in the scientific community. Gradient descent, a commonly employed optimization technique, is often utilized to modify the adjustable parameters of the network, such as the kernel and weight configurations. Minimizing the expected and actual production disparity is imperative to optimize efficiency. The modification of observable parameters is achieved by utilizing a hyperparameter known as the learning rate, which adjusts the parameters in the opposite direction as the gradient through the process of gradient back-propagation. The learning rate determines the magnitude of the stochastic increments during the learning process. Figure 6. depicts a visual representation of this method being implemented. An illustration of a configuration modification could be outlined in (4):

$$w := w - \alpha * \partial L/\partial w \tag{4}$$

The symbol w represents the training parameters, the loss function is denoted by L, as well as the learning rate is denoted by $\alpha.$ Establishing the learning rate, a significant hyperparameter, is crucial before commencing any training. A model aims to replicate a particular decision-making process, similar to how a team of experts analyzes and evaluates all available data.

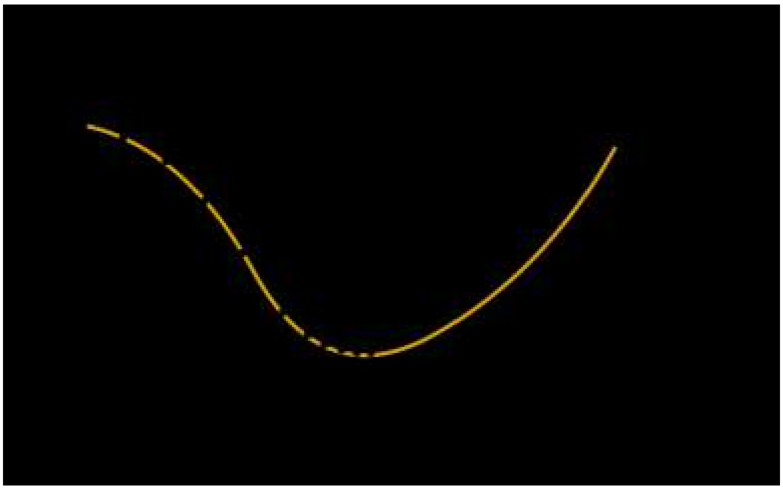

Figure 6. Gradient descent (GD)

Applying Regularization Techniques to Convolutional Neural Networks (CNNs)

Dropout, data augmentation, early stopping, and batch normalization are some of the regularization techniques often employed to mitigate the impact of overfitting. The subsequent sections will offer additional elaboration on specific concepts. Convolutional Neural Networks (CNNs) have greatly enhanced the performance of various image processing tasks, including image classification and object detection.

Batch normalization (BN)

Batch normalization is a recently introduced technique that has gained popularity for accelerating the training of deep feed-forward neural networks (Schilling and Fabian, 2016). As the parameters of the previous layers are modified, the distribution of the input layers is also altered. Consequently, instructing deep neural networks is a challenge. This presents a significant difficulty in training models, requiring lower learning rates and accurate parameterization. The issue at hand is commonly called internal covariate shift. It can be addressed by standardizing the inputs of each layer. Integrate strong normalization techniques into the model's architecture, particularly for processing individual mini-batches of training data. Batch normalization greatly surpasses the baseline model by eliminating the need for dropout in some situations and enabling higher learning rates with less concern for initialization.

Dropout

By implementing the dropout regularization technique in the fully connected layers, one can effectively merge multiple neural network topologies to prevent overfitting. Dropout refers to the removal of both overt and covert nodes in a neural network. When a node is randomly removed from the network, all its outward and incoming connections are temporarily severed. Overfitting poses a significant challenge in such networks. Additionally, large networks tend to be slow during inference, making it impractical to mitigate overfitting by aggregating predictions from multiple large neural networks at test time.

Early stop (ES)

Early stopping is a common practice when gradient-based optimization trains a too-expressive model to avoid poor generalization performance. A standard approach to determining the optimizer's optimal stopping point is to split the data into a training as well as, typically, a much smaller validation set, which allows obtaining a quasi-continuous estimate of the generalization performance.

Pre-trained models of convolutional neural networks (CNN)

Pre-trained CNN models have significantly improved performance in computer vision. These models are trained on giant picture data sets such as ImageNet, comprising millions of images labelled under different categories. The pre-trained models are shared, enabling the Research and Development community to tap into the representations and hierarchical features acquired by the model. These might be used for newer applications for which labeled data is scarce.

Literature Review

Recently, numerous successful early diagnostic uses of intelligent plant leaf disease identification methods have been made. Researchers have devised a few strategies for the automated diagnosis and classification of plant diseases.

This paper provides plant leaf detection and disease identification using deep learning. A camera was used to detect multiple diseases in plant species. The system was designed to detect and identify plant species, particularly apples, corn, grapes, potatoes, sugarcane, and tomatoes. It consists of 3,500 images of healthy and diseased plant leaves. The model achieved a high accuracy rate of 96.5% (Militante et al,2019).

In this work, images of tomato leaves were used to classify diseases. This model showed an accuracy rate of 95.90%, and the best model was obtained with good accuracy and prevented overfitting (Meeradevi et al.,2020).

Many plant diseases, including mango leaves, were studied using feed-forward neural networks and hybrid trait selection, and the accuracy rate was 97.49% (Pham et al, 2020).

A hybrid model for sunflower disease classification using deep learning was proposed. The hybrid models employed in this study include MobileNet and VGG-16, which achieved an accuracy of 89.2% (Sirohi et al, 2021).

A novel approach was used to classify olive leaf diseases based on images using a deep hybrid model. Six convolutional neural structures and dough models were used to detect diseases in olive leaves, and an impressive accuracy of 96.14% was achieved (Akhal et al., 2023).

This review paper provides a comprehensive overview of image-based crop disease detection using machine learning algorithms. It highlights the effectiveness of different imaging platforms and their applicability across various crop types and environmental conditions, bridging the gap between research and practice (Dolatabadian et al., 2024).

The study uses a hand-held Raman instrument and Machine Learning to detect TSWV infection in tomato plants. The method, which uses a portable Raman spectrometer, achieves an average 90-95% accuracy in detecting infected plants within 3-7 days after inoculation (Orecchio et al, 2025).

This paper presents a classification-based heuristic-assisted adaptive segmentation model and a deep detection framework for rice leaf disease. The model uses a combination of MRCNN, HOGCO, and a hybrid multi-scale residual attention network to classify and treat diseased rice leaves, improving the crop's economic value (Mehzabeen and Gayathri, 2025).

The study examines the impact of unexpected weather on agricultural output and the effectiveness of AI-based machine learning and deep learning algorithms in identifying apple leaf diseases, using bibliometric analysis and 109 publications from 2011-2022 (Bonkra et al, 2024).

This study proposes a machine learning model for identifying diseases in mango leaves, overcoming time-consuming and error-prone methods. The model uses DVNet, a deep mutual learning model, and Particle Swarm Optimization for hyperparameter optimization (Vijay and Pushpalatha, 2024).

The study analyzes a classification system for five sweet potato leaf illnesses, demonstrating impressive performance metrics and accuracy. The model's efficacy is highlighted, potentially improving agricultural productivity and food security (Goyal et al, 2024).

This study uses a convolutional neural network and advanced image preprocessing techniques to detect and segment tea leaf diseases. The method achieves an accuracy of 95.06%, outperforming existing models. The dataset includes six categories (Balasundaram et al, 2025).

This research uses a deep model-based approach to create automated systems for recognizing and classifying plant illnesses. The model uses Adaptive and Attention-aided Mask Region-based Convolutional Neural Network, BRP-GTBO, Hybrid Convolution Two-Dimension/One-Dimension, and HC-MDEB7 (Patil et al, 2025). Researchers are developing AI-assisted techniques for early detection, surveillance, and treatment of plant diseases, particularly in rice leaf disease detection.

This study uses EfficientNet and YOLO neural network architectures to develop a prediction model for identifying cotton plant diseases. The model's accuracy varies based on environmental parameters and monitoring, highlighting the importance of holistic approaches in plant protection (Pavate et al., 2025).

The research developed a deep learning model for classifying Haas avocado ripeness using enhanced spectral data, hyperspectral images, and fruit images, achieving high accuracy rates (Nuanmeesri, 2025). A CNN-based architecture classifying plant leaf diseases in Precision Agriculture (PA). It integrates multi-contextual features using RL-block and PL-blocks 1 and 2, combining different model streams trained on heterogeneous data. The model achieves an impressive 94.43% accuracy in classifying avocado leaf diseases, providing an innovative solution for early detection and classification.

This research proposes an ensemble deep learning-based technique for early detection and classification of plant leaf diseases in apple and maize plants. The technique uses two pre-trained models and a hybrid approach, with a 90% accuracy rate. This approach contributes to food security and sustainability by reducing the negative impact on crop productivity and quality. The US leads in corn production, but corn leaf deficiency affects yield manufacturing. A hybrid model, Whale Optimization Algorithm with Joint Search Mechanisms (JSWOA), is proposed for corn leaf disease prediction. The model outperforms existing methods in predicting maize leaf classes (Ashwini and Sellam, 2024).

To summarize the above studies, Table 1 provides an overview of the main machine learning-based approaches for plant disease diagnosis, identifying the data used, methodologies applied, performance metrics reported, and notable contributions and limitations for each study.

Table 1. Comparison of the Related Work with the Proposed Study

Study (Author, Year)	Dataset Used	Methods Applied	Performance Metrics	Notes on Limitations/Strengths
Militante et al, 2019	35,000 images of healthy plant leaves	Plant Village Dataset, Image Acquisition, Image pre-processing, Classification	Accuracy: 96.5%	The model was perfect
Meeradevi et al, 2020	5 types of diseased leaf images and healthy	VGG-16 inputs an image of size 180×180 with three filters	Accuracy: 95.90%	The model is good, but it leads to overfitting.

	leaf images			
Pham et al,2020	Occur in mango leaves	CNN	Accuracy: 97.53%	Neural networks, such as ResNet-50 and others, were used. These are robust networks that produce excellent results.
Sirohi et al, 2021	Sunflower leaves	Stacking ensemble learning methods and combining two models	Accuracy: 89.2%	Using only two models for training
El Akhal et al, 2023	4138 images of olive leaves	CNN + SVM and KNN	Accuracy: 99.53%	This paper uses hybrid models and six pre-trained neural network structures.
Dolatabadian et al, 2024	Pictures of different types of crop diseases	Multi-layer neural networks for learning features and patterns	Accuracy:	This paper has achieved a very low level of accuracy, and more techniques could be used to improve it.
Orecchio et al,2025	Tomato wilt virus infection detection	Application of a hand-held Raman spectrometer in conjunction with machine learning techniques.	Accuracy: 85%	Sample models were used to build a robust training and testing model that helps combat tomato leaf spot infections.
Mehzabeen et al,2025	5932 images by Rice diseases	By collecting and dividing images	Accuracy: 91%	Hybrid models such as VGG-16 and Mobile Net-V2 have been used for rice disease detection and have a significant application in all countries. However, large amounts of input parameters were introduced, reducing learning ability.
Bonkra et al,	apple leaf	Plyometric techniques were used.	Accuracy:-	Few models were used in the research.

Vijay and Pushpalatha, 2024	Mango leaf	Particle Swarm optimization	Accuracy: 94.72%	VGG-16 and Dense Net were used to detect leaf diseases, and although the accuracy varied, the accuracy rate was excellent
Goyal et al, 2024	Five images of the sweet potato leaf	A CNN-SVM Hybrid Approach	Accuracy: 97%	Five diseases affecting sweet potatoes were identified and measured, but more diseases would be detected if more measures were used.
Balasundaram et al, 2025	Tea leaf	Approach to Image Segmentation	Accuracy: 95.06 %	Tea is highly culturally important in many societies, and detecting diseases affecting tea leaves is important.
Patil and Borse, 2025	Plant Disease	EfficientNetB7	Accuracy: 90%	Despite the economic benefits of crop cultivation, several challenges persist, including identifying and classifying plant leaf types.
Pavate et al, 2025	Cotton plant	An efficient model for cotton plant health monitoring using YOLO based disease prediction	Accuracy: 99.2%	It has achieved very high accuracy
Nuanmeesri, 2025	1000 Avocado leaf	visible and near-infrared spectral data	Accuracy: 94.43%	Working with avocado leaves is different, and spectral data has also been used.
Ashwini and Sellam, 2024	corn leaf	hybrid 3D-CNN and LSTM	Accuracy: 90%	The shortage of corn contributes to rising crop prices. While deep learning techniques offer potential solutions, developing and applying additional models could enhance prediction accuracy
Proposed work	Plant Village	Hybrid MobileNet + SVM	Accuracy: 98.46%	Among various widely used CNN architectures, MobileNet yielded the most favorable results.

	Consequently, a hybrid model combining MobileNet with a Support Vector Machine (SVM) was developed.

Based on the number of studies examined previously, some gaps remain:

1. Focusing solely on the quantity of data is insufficient, as some studies employ limited datasets, compromising accuracy and efficiency. Additionally, reliance on a single methodological approach or a few hybrid models further restricts performance.
2. Machine learning techniques are limitedly used despite their potential, especially with their current development and increased use to solve problems.
3. In some studies, a single plant leaf is used, while different types of plants can be used and compared to find the accuracy and efficiency ratio between them, making the work more developed.

Summary

We live in a world dominated by technology. The chapter briefly overviews the theoretical basis for plant disease identification and explains some techniques, like convolutional neural networks as well as deep learning. The following section then delves further into the implementation as well as layout of the system. Several measures are used to test the performance of the algorithms and the suggested model. Some measures include accuracy, retrieval accuracy, as well as an F1 score.

Relationship Between the Topic of the Dissertation Proposal and the Priority Research Areas in the Institute and Department

AI, ML, Deep Learning Techniques, CNN, and pre-trained models in the medical field, especially in disease detection, including plant diseases, have greatly interested our department. These technologies enable us to analyze big data effectively to identify diseases, which contributes to developing prevention and control strategies. Therefore, these tools are vital in supporting advanced scientific research and exploration within the department.

MATERIAL AND METHOD

The proposed methodology has two major phases: training individual CNN models and generating hybrid models from the best-performing one.

Classical Machine Learning Models

This section presents a machine learning-based classification pipeline designed to train and evaluate conventional machine learning algorithms. Five widely used models—Random Forest, Logistic Regression, SVM, KNN, and Decision Tree- are employed to analyze plant disease images. The pipeline comprises several key stages: data preparation, image preprocessing, feature extraction, model training, and performance evaluation.

The model is trained using a classification method that can be applied to other models, such as tree-based and margin-based. The model's effectiveness is evaluated using multiple criteria, including accuracy, precision, recall, and overall performance score. Visual performance analysis, such as complexity matrix, ROC curve, and loss-acc plots, provides deeper insights into the model's performance.

The script is adaptable to other models, such as tree-based ensembles and margin classifiers, to improve classification accuracy, robustness, and generalization. The feature extraction process remains unchanged, making it easy to integrate different classification models into the same pipeline.

Non-Hybrid Model Training

Pre-trained models, MobileNet, ResNet-50, DenseNet121, ConvNext, VGG-19, VGG-16, and Inception V3, were trained on the PlantVillage dataset, which contains images of plant leaves with various diseases. The training methodology followed a standard process of loading and labeling the dataset, augmenting the data, and fine-tuning each pre-trained model to fit the dataset. Specifically, the steps were as follows:

- The dataset was loaded from a specified directory, and labels were assigned based on folder names.
- The data was split into a 70:20:10 ratio for training, validation, and testing.
- Image augmentation techniques were applied to enhance data diversity. These included resizing images to 224x224x3 or the model's input size, converting grayscale to RGB, and other preprocessing methods.

- Each pre-trained model was fine-tuned by freezing the lower layers and replacing the final fully connected layers with custom layers suitable for the number of classes in the PlantVillage dataset.
- The training parameters in each model, which are learning rate, batch size, and number of epochs, are optimized to converge efficiently. The starting learning rate was 0.001, with a batch size of 32, while the maximum number of epochs was 20 and 50. Each of these seven pre-trained models is further trained separately using the above methodology: MobileNet, ResNet-50, VGG-19, VGG-16, ConvNext, DenseNet121, and InceptionV3.
- This ensured the realization of the distinctive capabilities of each architecture in classifying plants.

Hybrid Model Development

Hybrid modeling has gained growing interest because it leverages additional data to enhance process understanding and improve model accuracy.

Since MobileNet showed the highest performance among the independent models in the preliminary examination, hybrid models were created independently by combining MobileNet with other models. The methodology for developing hybrid models is as follows:

A model was created by taking all layers except the last layer of MobileNet. This model was queried with validation data, and features were taken from the previous layer. Classical ML methods were trained with these feature vectors.

Standard performance metrics, including accuracy, precision, recall, F1 score, AUC, and the confusion matrix, were used to evaluate the performance of both individual and hybrid models. The flow chart for the proposed models is presented in Figure 7.

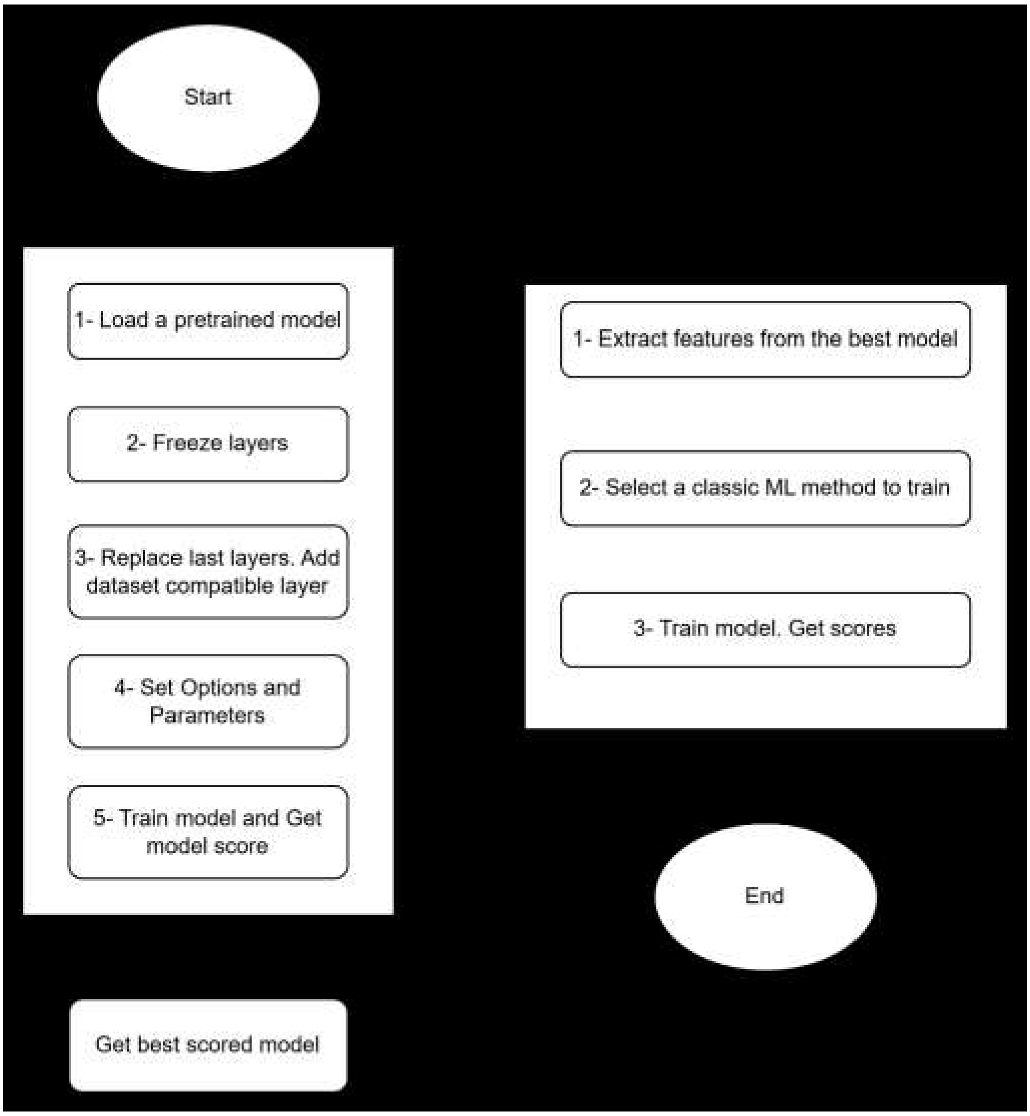

Figure 7. The flowchart for the proposed models

Integrating Features of Pre-trained Models with Machine Learning Models

We present our study on utilizing the widely used PlantVillage dataset (Noyan, 2022) to train deep learning-based models for plant disease detection. This research is based on deep learning from the image representations of the PlantVillage dataset, segmenting the plant disease classification approach. The methodologies used to develop this approach include data preprocessing, feature extraction using Resnet, VGG, and Inception as the pre-trained model, training a classifier, model evaluation, and computing some performance metrics for the results.

1. Data Preprocessing and Collection: This study utilizes the PlantVillage dataset, which comprises categorized images of various plant diseases. Each image is organized into directories that correspond to its associated disease label. First, the dataset is loaded from the specified directory, and proper labeling is ensured using the folder names as labels. These images are then resized to the input size of the chosen model, for example, InceptionV3, that

is, to 299x299 pixels with three color channels, hence RGB. The dataset has been split into two subsets: 70% of the images for training, 20% for validation, and %10 testing.

The grayscale images were first transformed into RGB format, and both the training and testing sets went through data augmentation to further improve model generalization.

2. Model Selection and Feature Extraction: Resnet, VGG, MobileNet, or Inception is a pre-trained CNN selected for feature extraction. The model is loaded with pre-trained weights trained on the ImageNet dataset. Feature extraction involves passing the augmented training and testing images through the network to extract the feature vectors from the fully connected layer 'fc7'. These feature vectors are used as inputs to the classifier.

The network activations are feature representations of images that hold the views pertinent to classification.

3. Classifier Training: The classification task is performed with a classifier trained on the features from the augmented training set.

4. Model Evaluation and Metrics Calculation: Using the already trained classifier, the performance of the same on the test set is evaluated. These predictions from the classifier are matched with the actual labels to calculate a confusion matrix from which the key performance metrics can be calculated: accuracy, precision, recall, and F1 score.

Where precision or recall measure the classifier's performance characteristics to identify a priori the correctly positive samples from each class, accuracy gives information about the overall percentage of correct predictions. The F1 score measures performance based on precision and recall combined as a single score, providing a better-balanced measure of model performance.

5. AUC-ROC Curve: Analysis. Besides the traditional measures of classification, the AUC of the ROC curve will be calculated for each class. AUC quantifies the classifier's ability to distinguish between two classes: the higher the values, the better. In the end, ROC curves for each class will be plotted, and the mean AUC across classes will be computed as an overall measure of discriminative power.

6. Visualizing Results: Finally, visualize the results by plotting. A confusion matrix is drawn to intuitively understand the classifier's performance for each class. The bar plots of evaluation metrics like accuracy, precision, recall, and F1 score are drawn to emphasize the strengths and weaknesses of the model. Also, ROC curves for each class are plotted, which show the trade-off between the actual positive rate and false positive rate at different thresholds.

This methodology ensures that the model is well-trained, its performance is rigorously evaluated, and various classification metrics are correctly estimated to ensure high-quality and predictable results on plant disease detection.

Dataset

The dataset to be used in this study is the PlantVillage dataset (Noyan, 2022).

It is one of the known sources on plant diseases. The dataset consists of over 20.6k images of healthy and diseased plant leaves, representing 14 disease categories and one category of healthy ones. The images are 256* 256 in size. Figure 8 shows the class distribution for the dataset.

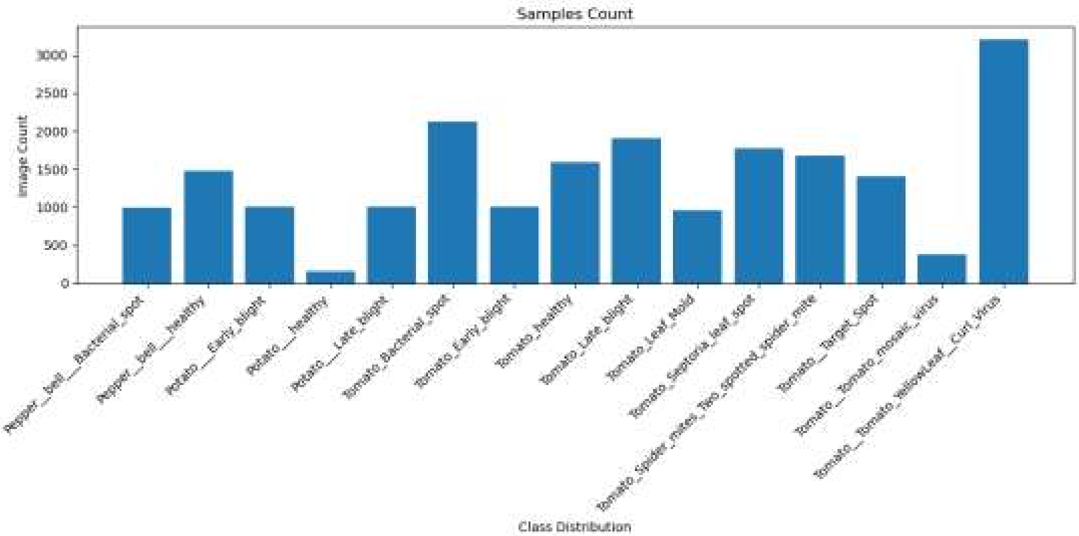

Figure 8. Class distribution of the dataset

The dataset covers various conditions, including different disease manifestations across multiple plant species, making it a valuable resource for deep learning-based classification tasks. Images were collected under control conditions to ensure consistency, yet they reflect significant color, shape, and texture diversity. This diversity presents realistic challenges that help improve the robustness of trained models.

Figure 9 shows some healthy and diseased images of a representative plant leaf, completely representing common plant diseases. As such, each image is titled with the specific type of plant and its disease; hence, this dataset best suits supervised learning model training applications.

The PlantVillage dataset is the foundational input for training individual and hybrid models, allowing the network to learn distinguishing features between healthy and diseased leaves. Its diversity and comprehensiveness make it suitable for developing robust models for real-world plant disease detection. Figure 9. shows some samples of the dataset used in the study.

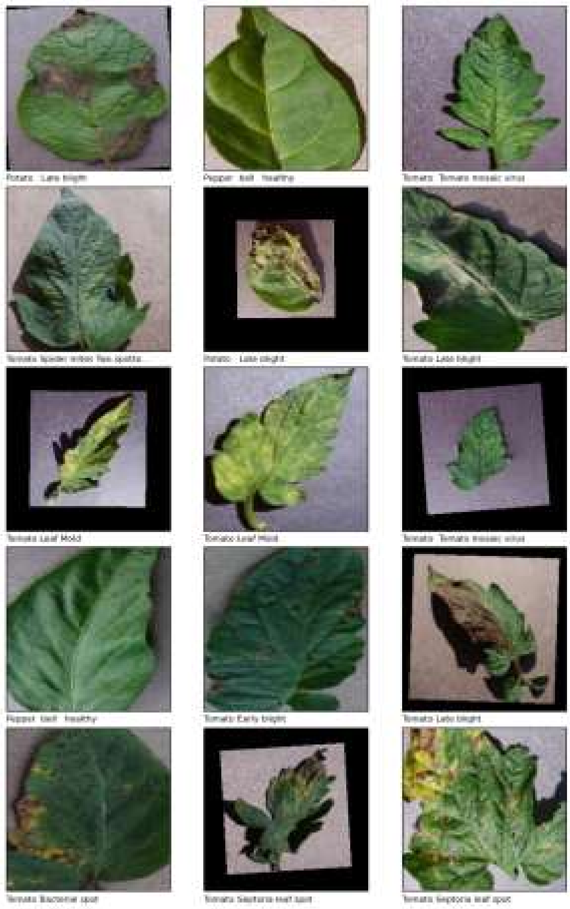

Figure 9. Some samples of the dataset used in the study

Fine-Tuning Technique

One of the most essential techniques employed in pre-trained models for this study is fine-tuning, which describes the sensitive dependencies of facts or properties on the values of specific parameters. Fine-tuning in CNN takes a model pre-trained on a vast dataset, say ImageNet, and restructures it to suit the dataset used in this study. In this regard, the process of fine-tuning is summarized as follows:

- It retains a pre-trained network's first few layers, which capture general features such as edges and textures. These are often frozen during training, as the learned weights are not updated.
- New layers replace the model's last, more task-specific layers with several classes equal to those in the PlantVillage dataset.

These added layers are then trained, and the whole model is fine-tuned by using a smaller learning rate to adjust weights throughout the network without significantly altering the earlier learned features.

Fine-tuning allows the model to leverage features learned from the large-scale general dataset more precisely and effectively in this specific case of plant disease classification problems.

Transfer Learning

Transfer Learning networks utilize the acquired characteristics from training on extensive databases to accomplish a comparable task.

In this scenario, the fundamental idea is to utilize the general characteristics acquired by the layer nearest to the input while adjusting the deeper layers to provide optimum features suitable for the specific target domain. This is particularly advantageous for tasks with limited data about the specific field.

Optimizing only the last few layers yields satisfactory outcomes when the difficulties are similar. Plant Village pictures differ from the common objects used to train most pre-trained algorithms. Hence, it was necessary to train at all levels for this study. The initial weights expedite the process and enhance the learning, reducing the training time and overcoming the data restriction. We examined MobileNet, ResNet-50, InceptionV3, VGG-19, DenseNet121, ConvNext and VGG-16 as comparable networks.

Data Augmentation Technology

It could be used during network training to increase the dataset size by changing the format of the original data and obtaining additional hidden data for training. This is achieved using a generator to progressively enlarge each image before sending it through the network. Various techniques are employed for data augmentation, including rotation, flipping, scaling, zooming, and adding Gaussian noise. Figure 10. shows data-augmented images.

This is especially helpful when there is a small amount of available training data related to critical real-world applications, such as medical datasets. It helps reduce the model's overfitting during training and ultimately improves the performance of the resulting model. This is one of the reasons for the significant differences among several machine learning algorithms.

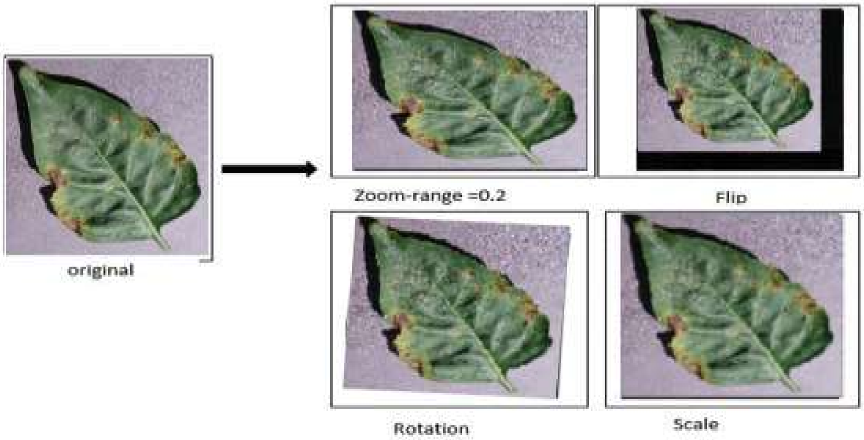

Figure 10. Data augmentation technology

Models Training Stage

Fine-tuning is one way to improve a pre-trained CNN model by optimizing the already acquired weight coefficients. In training, the first couple of steps are done with the weights from a pre-trained big dataset by initializing the weights in the network. The tasks have learned these weights through seeing a large dataset and hence can detect the prevalent visual components. It gives the model more data to increase performance during training.

The optimization approach iteratively adjusts the weights using backpropagation and gradient descent. It aims to minimize the difference between the actual labels and the predicted outputs. Therefore, this would involve using the accuracy, precision, recall, or loss functions to measure the dependability of the model's predictions during the training approach.

The gradient will tell you in which direction and with what magnitude to alter the weights so that performance improvement may be enhanced. They have been used to update the model weights at each iteration.

The training has been done in several iterations, sometimes called epochs, where a single iteration is one complete cycle through the training dataset.

The model's performance will improve progressively during training, during which the weights are optimized to minimize some error measure. Two possible metrics may be the mean square error or the categorical cross-entropy. Each of the pre-trained CNN models may require different editing of weights and optimization parameters. Although the training structure as well as method will differ for each of them, updating the current weights of the model on some particular task or dataset is always included in fine-tuning its performance.

Pre-Trained CNN Models

CNN models consist of several layers, including convolutional, pooling, and fully connected layers, among others. Convolutional layers are responsible for the process of feature extraction. This is accomplished by applying filters to the input images, which helps to recognize localized patterns and spatial linkages. Pooling layers reduce the spatial dimensions of the features, which in turn reduces the complexity of the computations while maintaining the integrity of the essential information. Fully connected layers establish linkages between the extracted features and the final output, which enables predictions to be made based on the representations that have been gathered. Seven pre-trained models were used in the research project. These models were MobileNet, DenseNet121, ConvNext, ResNet-50, VGG-19, VGG-16, and Inception V3. These models were then hybridized in order to construct a unified model.

ResNet-50 Model

ResNet-50 is a deep convolutional neural network and part of the Residual Network family. This is a 50-layer deep architecture devised to solve the vanishing gradient problem that occurs in deep networks and, in turn, permits the practical training of deep models. Resnet architecture is shown in Table 2.

ResNet's novelty is the residual blocks, which explicitly employ skip connections, in other words, shortcuts. These allow the network to learn residual mappings rather than direct mappings.

For ResNet-50, which is 50 layers deep, the convolutional layers are combined with pooling and fully connected layers. ResNet-50 uses a bottleneck architecture for its residual blocks.

Table 2. ResNet Layer Architecture

Stage	Filters	Layers (Blocks)	Output Size
Conv1	7×7, 64, stride 2	1	112×112×64
Pool	33 max pooling, stride 2	-	56×56×64
Conv2	[1×1, 64], [3×3, 64], [1×1, 256] ×3	3	56×56×256
Conv3	[1×1, 128], [3×3, 128], [1×1, 512] ×4	4	28×28×512
Conv4	[1×1, 256], [3×3, 256], [1×1, 1024] ×6	6	14×14×1024
Conv5	[1×1, 512], [3×3, 512], [1×1, 2048] ×3	3	7×7×2048
Pool	Global average pooling	-	1×1×2048
FC	Fully connected (SoftMax)	-	1000 (classes)

VGG-19 Model

The VGG-19 model is a deep convolutional neural network proposed by the Visual Geometry Group (VGG) at the University of Oxford. It was introduced in the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) 2014 and is renowned for its simple and elegant architecture.

VGG-19, with 19 layers with learnable weights, comprises 16 convolutional layers and three fully connected layers. The depth of a network like this allows the system to learn complicated and hierarchical features of images.

At Convolutional Layers, Small convolutional 3 × 3 features, with step 1 and padding 1, are used. Fine details captured- the filters preserve spatial information. Following each convolutional layer is an activation using ReLU, allowing non-linearity into the model. Pooling is used with a filter size of 2×2 and a stride of 2. Using pooling layers reduces spatial dimensions and computational complexity, retaining only the essential features. At Fully Connected Layers, the rest of the network comprises three FCs. The first two have 4096 neurons, and the last is a SoftMax to make the classification. The model uses small filters (3×3) to reduce the number of parameters compared to networks with larger filters.

It is a variant of the VGG model, which consists of 19 layers. These layers are segmented as follows: (16 convolutional layers, three fully connected layers, five max pool layers, and 1 SoftMax layer).

The structure of the convolutional layers is in the form of pooling blocks, and the pooling process reduces the dimensions to the maximum possible extent. The structure can be summarized in Table 3.

Table 3. VGG-19 Layer Architecture

Layer Type	**Configuration**	**Output Size Image 224×224 Input**
Input	224×224×3	224×224×3
Conv3-64, Conv3-64	2× (3×3 conv, 64)	224×224×64
Max Pooling	2×2	112×112×64
Conv3-128, Conv3-128	2× (3×3 conv, 128)	112×112×128
Max Pooling	2×2	56×56×128
Conv3-256 (×4)	4× (3×3 conv, 256)	56×56×256
Max Pooling	2×2	28×28×256
Conv3-512 (×4)	4× (3×3 conv, 512)	28×28×512

Max Pooling	2×2	14×14×512
Conv3-512 (×4)	4× (3×3 conv, 512)	14×14×512
Max Pooling	2×2	7×7×512
Fully Connected	2× (FC, 4096 neurons)	-
Output Layer	SoftMax (1000 classes)	-

VGG-19 is very straightforward and modular for architecture implementation. It captures rich features, which makes it suitable for transfer learning and feature extraction.

However, VGG-19 has many parameters (~ 144 million) and is computationally expensive. This will require considerable GPU memory, which might limit its applicability in resource-constrained environments. Training is very time-consuming, especially with more recent architectures such as ResNet.

VGG-16 Model

VGG-16 is a CNN proposed by VGG, Visual Geometry Group at the University of Oxford. It is a straightforward and highly modular network architecture proposed in ILSVRC 2014, where it performed very well.

The 16 in VGG-16 refers to the number of learnable layers in the network: 13 convolutional layers and three fully connected layers. The VGG-16 architecture is shown in Table 4.

VGG-16 has 16 layers with learnable parameters that can capture rich hierarchical features. It uses small 3×3 convolutional filters throughout the network. With a small filter size, the overall model remains thin and is free to capture outstanding details of images. All convolutional layers use 3×3 filters, stride = 1, padding = 1 to preserve the spatial dimensions between layers before pooling. After every set of convolutional layers, max pooling with a 2×2 Spatial dimension is reduced by applying a 2×2 filter with a stride of 2 for computational efficiency in the network.

Table 4. VGG-16 Layer Architecture

Stage	**Layers**	**Output Size**
Input	RGB Image	224×224×3
Conv Block 1	2 × Conv 3×3, 64 filters	224×224×64
Max Pooling 1	MaxPool 2×2, stride 2	112×112×64
Conv Block 2	2 × Conv 3×3, 128 filters	112×112×128

Stage	Layers	Output Size
Max Pooling 2	MaxPool 2×2, stride 2	56×56×128
Conv Block 3	3 × Conv 3×3, 256 filters	56×56×256
Max Pooling 3	MaxPool 2×2, stride 2	28×28×256
Conv Block 4	3 × Conv 3×3, 512 filters	28×28×512
Max Pooling 4	MaxPool 2×2, stride 2	14×14×512
Conv Block 5	3 × Conv 3×3, 512 filters	14×14×512
Max Pooling 5	MaxPool 2×2, stride 2	7×7×512
Flatten	Flatten	25,088
Fully Connected 1	Dense (FC), 4096 units	4096
Fully Connected 2	Dense (FC), 4096 units	4096
Output	Dense + Softmax (1000 classes)	1000 classes

Fully Connected Layers: It contains an end of 3 fully connected layers followed by a SoftMax layer. High Parameter Count: Despite its simplicity, VGG-16 contains about 138 million parameters and is, therefore, computationally expensive.

The constant use of 3×3 filters: The architecture is easy to implement because of the use of 3×3 filters. Extracted by VGG-16 rich hierarchical features means being ideal for transfer learning. Results are pretty good from pre-trained VGG-16 models across many image tasks outside of those from ImageNet.

The model has nearly 138 million parameters, so it is computationally intensive and requires a lot of memory. At the same time, training VGG-16, with its depth and count of parameters, is very resource-intensive and can be time-consuming even for state-of-the-art GPUs. The large number of parameters increases the risk of overfitting because of the small-sized datasets.

Initially designed for the ImageNet challenge, it can classify images into 1000 categories. Pre-trained VGG-16 is often used to extract features as the backbone architecture for transfer learning in object detection and segmentation, among other applications. Medical Imaging: The model can be applied to detect diseases or abnormalities in medical images.

InceptionV3 Model

Inception V3 is a deep convolutional neural network proposed by Google and part of the large family of Inception models. It is an improved version of earlier architectures, InceptionV1-GoogLeNet and InceptionV2, introduced in the paper Rethinking the Inception Architecture for Computer Vision, by (Szegedy et al, 2016).

InceptionV3 is designed to perform well while maintaining computational efficiency. It has adopted new ideas, such as factorized convolutions and auxiliary classifiers.

The architecture is based on modular building blocks called Inception modules. These modules perform parallel convolutions of different kernel sizes, such as (1×1, 3×3, 5×5) on the same input, enabling the network to capture features at multiple scales. Instead of large kernels like 5×5, InceptionV3 factorizes them into smaller 3×3 convolutions. For example, the 5×5 convolution can be replaced by two 3×3 convolutions. They have fewer parameters and a lower computation cost. Asymmetric Convolutions: The 3 × 3 convolutions are further factorized into 1×3, and it also uses 3×1 convolutions for a reduction in complexity and thus computational efficiency. This network downsamples early using strided convolutions and max pooling layers to reduce spatial dimensions and computation. InceptionV3 uses auxiliary classifiers in the training phase. This has helped improve gradient flow in deeper layers, and it acts as a kind of regularizer. They are removed during the time of inference. Heavy use of batch normalization increases convergence speed and helps avoid overfitting. The Inception V3 architecture is shown in Table 5.

It does this through a few stages in the processing of input images. It performs initial convolutions and pooling to downsample the image and extract basic features. It consists of several parallel convolutional layers with different filter sizes, pooling operations, and 1×1 convolutions to reduce dimensionality. Intermediate layers provide additional supervision to improve learning in earlier parts of the network. It uses a global average pooling layer instead of fully connected layers to reduce the parameters; this output is fed through a SoftMax layer for classification.

Table 5. Inception V3 Layer Architecture

Stage	**Layers**	**Output Size**
Input	RGB Image	299×299×3
Convolution 1	Conv 3×3, stride 2	149×149×32
Convolution 2	Conv 3×3, stride 1	147×147×32

Stage	Layers	Output Size
Convolution 3	Conv 3×3, stride 1	147×147×64
Max Pooling	MaxPool 3×3, stride 2	73×73×64
Inception Modules	Multiple blocks with 1×1, 3×3, 5×5, pooling	Varies (e.g. 35×35×288)
Reduction Modules	Strided Inception blocks	17×17×768 → 8×8×1280
Inception Modules	High-depth Inception blocks	8×8×2048
Global Avg Pool	Global Average Pooling	1×1×2048
Fully Connected	Dense + Softmax (1000 classes)	1000 classes

Inception reduces computational cost and the number of parameters by factorized convolutions and dimensionality reduction. It can easily be scaled up by adding more modules or filters to the network.

This architecture is more complex than simple models like VGG, making it challenging to implement and modify.

Inception is designed to facilitate high-accuracy classification on ImageNet. Pre-trained models of InceptionV3 are widely used as backbone pretraining for fine-tuning to perform object detection, semantic segmentation, and so on. It is used in healthcare applications that analyze X-rays, MRIs, and other forms of medical imagery. It is also used in most generative tasks, like style transfer.

MobileNet Model

A neural network is considered an architectural model. It is used in various applications and use cases, including fine-grained classification, facial features, and large-scale geolocation (Howard et al., 2017).

MobileNet architecture is simplified and uses deep discrete convolutions to build lightweight, low-latency deep neural networks for mobile and embedded devices.

DenseNet Model

Convolutional networks can be more accurate and efficient in training. Dense networks connect layers. They have several compelling advantages: they help mitigate the vanishing gradient problem, enhance feature propagation, promote feature reuse, and significantly reduce the number of parameters (Huang et al.,2018).

DenseNet architecture consists of two layers. Each layer has direct connections to all subsequent layers, resulting in L(L+1)/2 connections.

ConvNext Model

These networks have shown strong performance in various scenarios. These models are designed to learn using ImageNet labels. ConvNext has undergone several improvements in terms of accuracy and efficiency. Innovations were discovered during the use of supervised training on the ImageNet dataset. ConvNext has demonstrated exceptional performance, especially when lower computational complexity is essential. (Woo et al, 2013). The ConvNext architecture adheres to the general principle of progressively applying convolutional layers to the input.

The Importance of Pre-Trained Hybrid Models

Pre-trained hybrid models are essential for detecting plant diseases using deep learning techniques. Identifying a disease is time-consuming, so farmers rely on machine learning techniques. Machine learning helps quickly identify diseases, saving time and combating them. These features and these techniques have helped many farmers protect their farms from agricultural pests and potential financial losses. Extensive experiments demonstrate that the plant disease pre-trained model achieves higher accuracy than existing pre-trained models while requiring less training time, thereby enhancing the effectiveness of plant disease diagnosis (Dong et al,2023).

Transfer Learning benefits from the knowledge acquired by training on extensive and diverse datasets by utilizing pre-trained Convolutional Neural Network (CNN) models like VGG-19. If you possess a restricted quantity of data, this knowledge might assist you in gaining an initial advantage in preparing for your particular undertaking.

Feature extraction involves utilizing the convolutional layers of pre-trained models to extract intricate hierarchical information from input pictures effectively. Consequently, engineers and designers no longer need to create features manually. Transfer learning can significantly decrease training time and the computer resources required. It enables efficient and straightforward network adaptation without beginning from scratch with intricate CNN designs.

By using their prior knowledge of generalizable characteristics acquired from other datasets. Consequently, they can boost performance on your unique dataset even with fewer samples.

Regularization involves fine-tuning pre-trained models using your dataset to prevent overfitting. It achieves this by constraining the model's learning ability.

Nevertheless, the performance of the pre-trained model may be suboptimal with your data due to a lack of compatibility between the domains. The model may require significant adjustments to succeed in your specific circumstances. The model is susceptible to catastrophic forgetting, a phenomenon where previously learned information is entirely lost during subsequent training. Due to the fixed nature of many architectural designs, which may not align optimally with specific applications, careful consideration is essential before adding new layers or altering the existing structure.

Adaptive Fine-Tuning, which is gradual adaptation, will help mitigate the domain mismatch problem. Some layers can be frozen during training to allow for smoother model adjustment before being unlocked later. To avoid overfitting, fine-tuning can use various regularization techniques, such as weight decay, data augmentation, and dropout.

Hyperparameter tuning involves tuning things like learning rates, batch sizes, and optimization techniques to provide a balance that would give learning a new task a nice trade-off between forgetting too much previously learned information and not forgetting it.

Pooling predictions from many pre-trained models with different topologies can improve robustness and performance. Transfer learning strategies include the feature extraction approach, which selectively freezes most of the model's layers while allowing only the last few layers to be learned for a particular task. These strategies will be essential in catching catastrophic forgetting and overfitting. Such problems could have been detected by regularly checking the model's performance on a validation set during training.

A pre-trained model has considerable advantages regarding time, resources, and generalization. However, careful fine-tuning and monitoring are necessary to ensure the model learns the task well without losing its acquired skills.

The Diagnosis of Plant Diseases Based on the Importance of Machine Learning Models

Machine learning models are important for efficiently and adequately diagnosing diseases in plantations from image databases. Here is why they are necessary:

Automation and Speed: With large-scale image databases, ML models can analyze them quickly, allowing for real-time diagnosis without requiring human experts.

High Accuracy: Advanced models, like CNNs, assure good performance concerning pattern and visual feature identification; thus, the diseases can be identified precisely even in very complex situations.

Early Detection: ML models identify diseases at an early stage and help reduce damage to the crop, thereby increasing agricultural yield.

Cost-Effectiveness: It reduces dependence on costly laboratory tests and expertise for diagnosing disease outbreaks and makes this accessible to farmers.

Scalability: ML models can handle heterogeneous datasets, making them relevant for various crop types and regions. Continuous Improvement: Models improve over time because they are subjected to more diverse and annotated image databases, enhancing their robustness and accuracy. Integrating ML models into plant disease diagnosis has redefined agriculture to cater to decision-making regarding pest and disease management in a more environmentally friendly and information-driven way.

RESULTS AND DISCUSSION

Performance tests of classical machine learning models, SVM, KNN, Logistic Regression, Decision Tree, and Random Forest, were performed. Then, tests of both hybrid and non-hybrid deep learning models were performed.

Measures for Assessment

Several distinct classification metrics are employed to assess the study's results as well as the model's efficacy in carrying out the desired task on an undisclosed test dataset. Frequently utilized metrics include classification precision, accuracy, as well as sensitivity, which are frequently associated with loss evaluation matrices.

Accuracy

The accuracy of a deep learning classification algorithm can be used to assess its efficiency. It is calculated by comparing the ratio of accurate diagnoses to the total ratio. A high accuracy score signifies that the model could effectively predict positive situations while minimizing false positives.

The provided accuracy function calculates accuracy via multiplying the proportion of correct predictions by 100%.

$$Acc = \frac{Number\ of\ Correct\ Predictions}{Total\ Number\ of\ Predictions} * 100 \tag{5}$$

F1 Score

It is commonly employed to assess the performance of a classification system, particularly in scenarios with data asymmetry. The term is characterized as the multiplication of the model's accuracy and recall, as well as the formula determines its calculation:

$$\mathrm{F1} = \frac{2\ (\mathrm{Precision} * \mathrm{recall})}{\mathrm{Precision} + recall} \tag{6}$$

Precision quantifies the model's ability to accurately distinguish true positives from false positives.

Recall quantifies the number of true positives detected by the model.

Higher F1 values, ranging from 0 to 1, indicate improved model performance regarding precision and recall.

Confusion Matrix (CM)

This table type provides complete insight into the model's classification performance by presenting correct and incorrect classifications. This tool facilitates the calculation of various performance metrics as well as provides valuable insights into the model's accuracy and efficiency.

AUC-ROC

The metric known as AUC-ROC, or Area Under the Receiver Operating Characteristic curve, assesses performance in binary classification. It is one applied method for evaluating the performance of a classifier under various circumstances. A higher AUC-ROC value means better performance, ranging from 0 to 1. One use of this metric is to compare different models. This becomes very useful if datasets are relatively balanced or the cost of being wrong in either direction differs.

Batch size

The batch size is the number of training samples used in a single iteration to train a model. It is a critical parameter in machine learning, directly affecting the model's performance, decision-making, accuracy, and efficiency. Beyond performance, understanding how batch size works involves several key aspects that influence training dynamics and resource usage.

Training Efficiency: Smaller batch sizes result in more frequent parameter updates, which can accelerate convergence but may introduce greater variability (noise) in the training process. In contrast, larger batch sizes produce more stable updates, making them preferable when aiming to reduce noise during training.

Model Performance: Batch size also influences the model's generalization ability. Small batches can help the model escape local minima, potentially improving generalization. Larger batches, while speeding up convergence, may increase the risk of overfitting.

Optimal Selection: Selecting the ideal batch size depends on factors such as the specific task, model architecture, and available computational resources. A typical strategy is experimenting with various batch sizes to evaluate their impact on training stability and model performance.

Benefits of Tweaking: Tweaking the batch size can help fine-tune the training process and improve efficiency. For example, a large batch size may benefit stable datasets, while a smaller one may be more effective for noisy data. Therefore, tweaking is very important.

Classical machine learning models

The test results show that classical machine learning models perform poorly compared to deep learning models, both hybrid and non-hybrid. One primary reason is that classical models rely on manually extracted features using techniques such as HOG and LBP before training. As a result, the model's accuracy heavily depends on the quality of the selected features. In contrast, convolutional neural networks (CNNs) learn features automatically from raw data, allowing them to discover complex details that cannot be extracted manually. Table 6 shows the results of testing SVM, KNN, and Random Forest machine learning models.

Table 6. Performance Comparison of Classic Machine Learning Methods

Model	Accuracy	Precision	Recall	F1 Score	Roc-Auc Score
SVM	64.00%	61.00%	58.00%	59.00%	94.82%
KNN	36.00%	47.00%	34.00%	32.00%	74.71%
Random Forest	56.00%	49.00%	43.00%	42.00%	88.47%
Logistic Regression	63.00%	59.00%	55.00%	55.00%	93.38%
Decision Tree	38.00%	30.00%	30.00%	30.00%	62.62%

Non-Hybrid Model Results

Each of the seven pre-trained CNN models was examined individually in plant disease classification. All models underwent the same training methodology for training and fine-tuning on the PlantVillage dataset. Each model was trained and fine-tuned using the same method on the same dataset. The following key metrics, accuracy, Precision, Recall, F1 Score, and AUC, were implemented to investigate the performance of these strategies.

Among all the evaluated hybrid models, MobileNet demonstrated the best performance, achieving an accuracy of 97.70%, a precision of 97.50%, and a recall of 97.30%. This resulted in an F1 Score of 97.40% and a ROC-AUC of 99.98%.

ResNet-50 achieved an accuracy of 96.20%, with a precision of 96.60% and a recall of 95.70%, leading to an F1 Score of 95.90% and a ROC-AUC of 99.00%.

The second model, Inception V3, obtained an accuracy of 91.20%, a precision of 89.90%, and a recall of 89.60%, resulting in an F1 Score of 89.60% and a ROC-AUC of 99.57%.

VGG-16 achieved an accuracy of 76.10%, with a precision of 76.20%, a recall of 74.80%, an F1 Score of 72.40%, and a ROC-AUC of 97.99%.

Similarly, VGG-19 obtained an accuracy of 77.30%, a precision of 77.60%, a recall of 74.90%, an F1 Score of 73.90%, and a ROC-AUC of 98.05%.

The DenseNet121 model showed strong performance with an accuracy of 97.10%, a precision of 97.20%, a recall of 96.60%, an F1 Score of 96.80%, and a ROC-AUC of 99.93%.

Finally, ConvNext achieved an accuracy of 95.90%, precision of 95.60%, recall of 95.70%, F1 Score of 95.60%, and ROC-AUC of 99.94%. Table 7 shows the result.

Table 7. Performance Comparison of Non-Hybrid Models

Model	Accuracy	Precision	Recall	F1 Score	Roc-Auc
ResNet-50	96.20%	96.60%	95.70%	95.90%	99.00%
Inception V3	91.20%	89.90%	89.60%	89.60%	99.57%
VGG-16	76.10%	76.20%	74.80%	72.40%	97.99%
VGG-19	77.30%	77.60%	74.90%	73.90%	98.05%
MobileNet	97.70%	97.50%	97.30%	97.40%	99.98%
DenseNet121	97.10%	97.20%	96.60%	96.80%	99.93%
ConvNext	95.90%	95.60%	95.70%	95.60%	99.94%

The ROC-AUC graphics of the ResNet-50 model are shown in Figure 11.

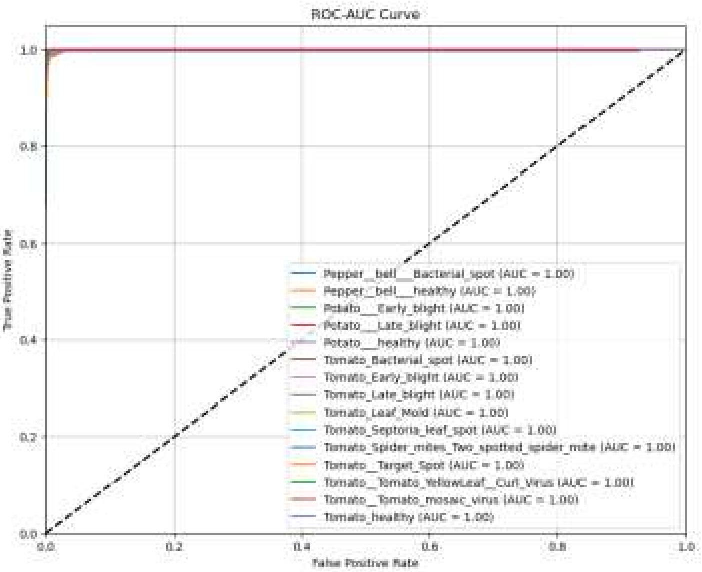

Figure 11. Resnet-50 ROC-AUC curve

The confusion matrix for ResNet-50 is as shown in Figure 12.

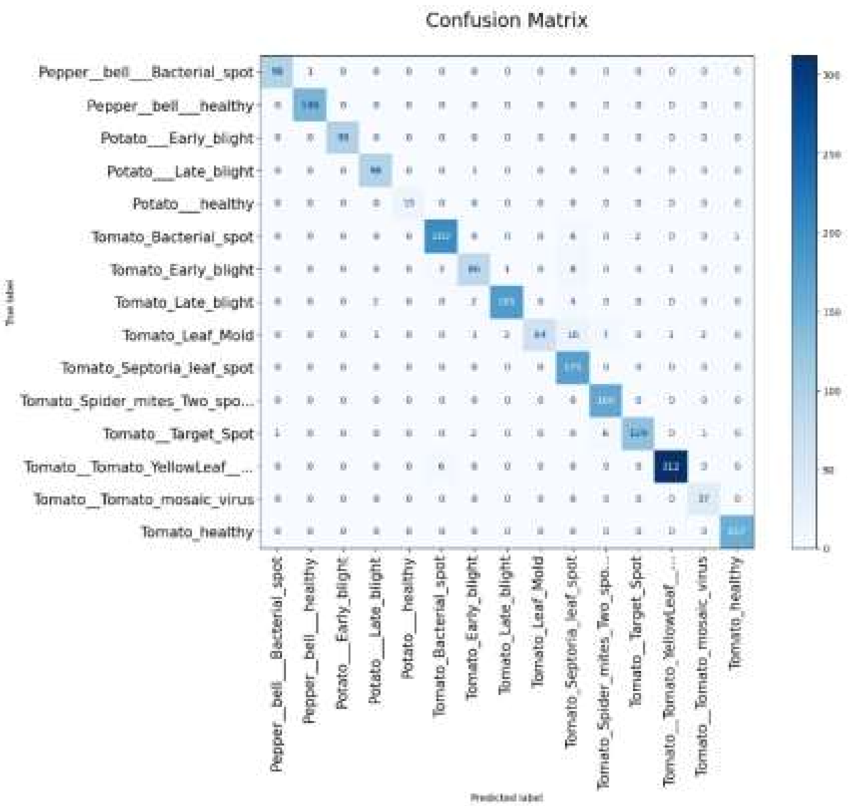

Figure 12. ResNet-50 confusion matrix

Choosing a value between 10 and 50 for the epoch number is sufficient for transfer learning. Since the epoch number is 20 in this ResNet model, the Acc-Epoch graphs do not provide a sufficient view of the model. The Acc-Epoch and Loss-Epoch graphics for ResNet is shown in Figure 13.

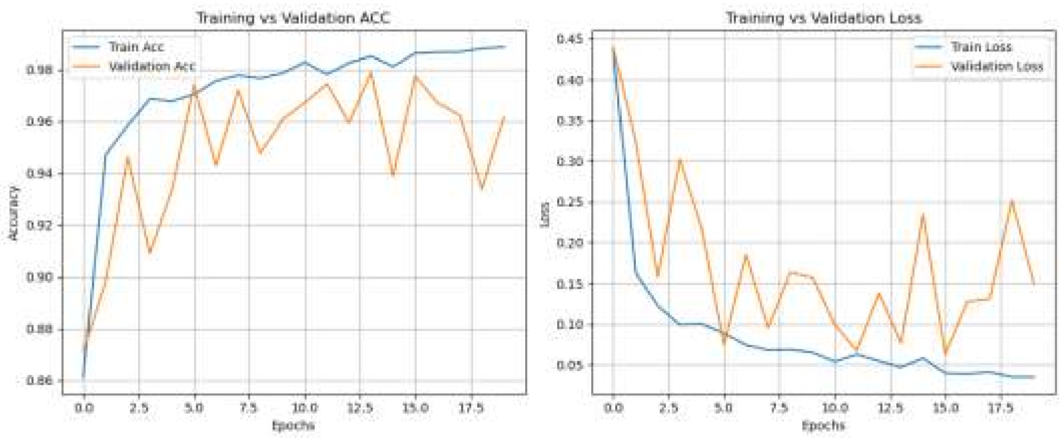

Figure 13. Resnet-50 Acc-Epoch and Loss-Epoch graphic

The ROC-AUC graphic of the InceptionV3 model is shown in Figure 14.

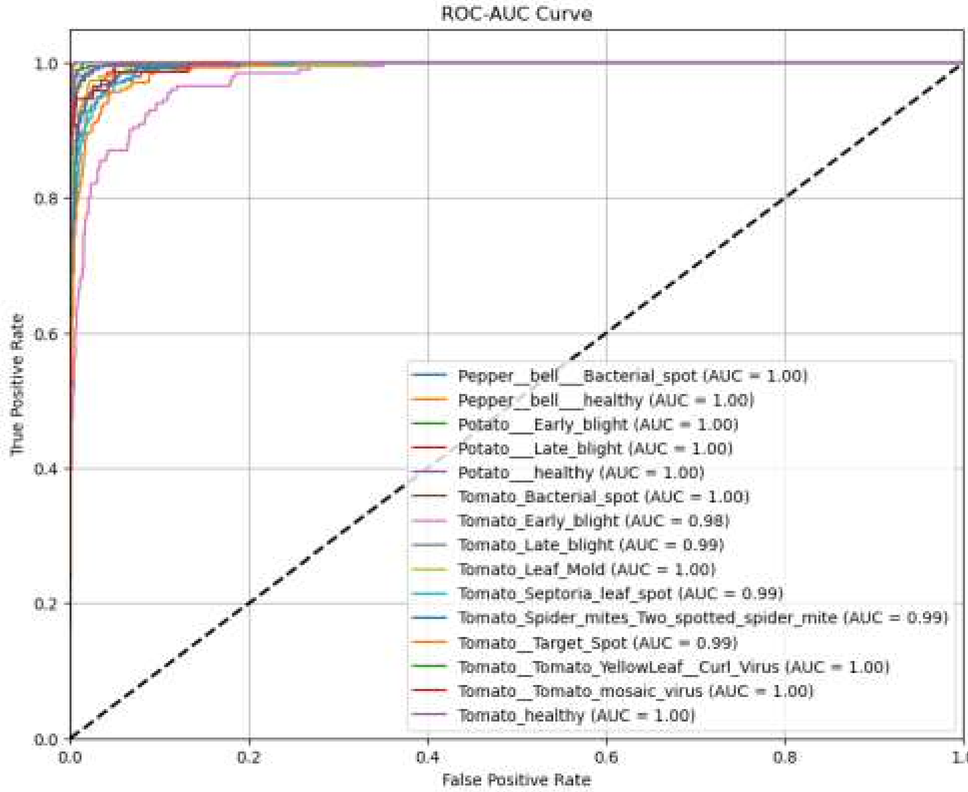

Figure14. Inception V3 ROC-AUC curve

The confusion matrix for Inception V3 is as shown in Figure 15.

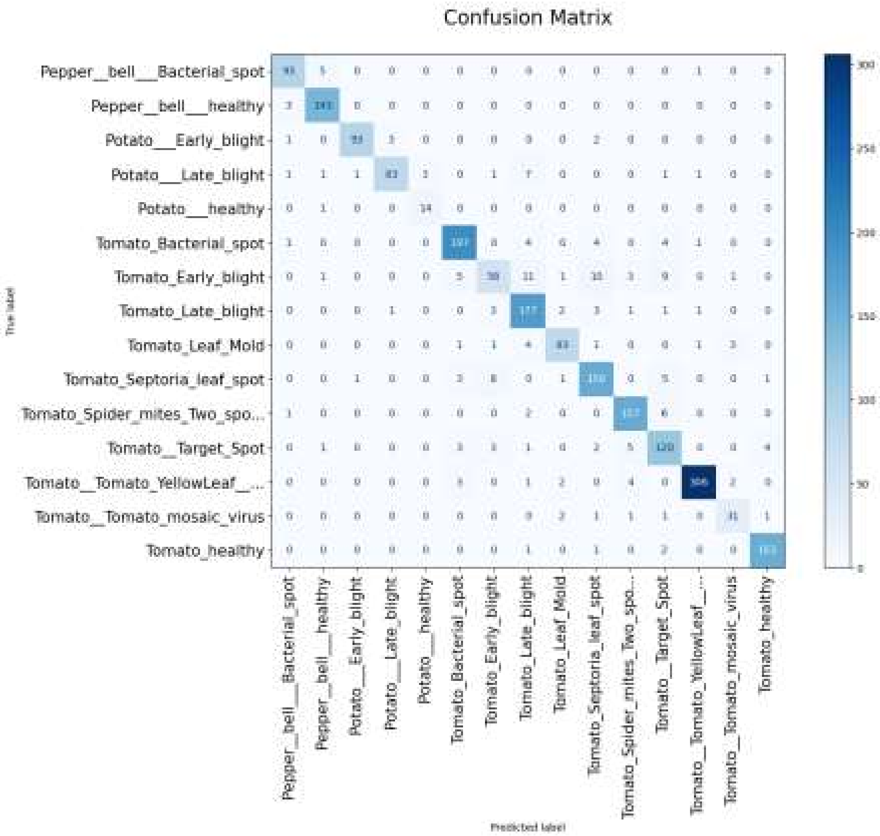

Figure 15. Inception V3 confusion matrix

The Acc-Epoch and Loss-Epoch graphics for Inception V3 are shown in Figure 16.

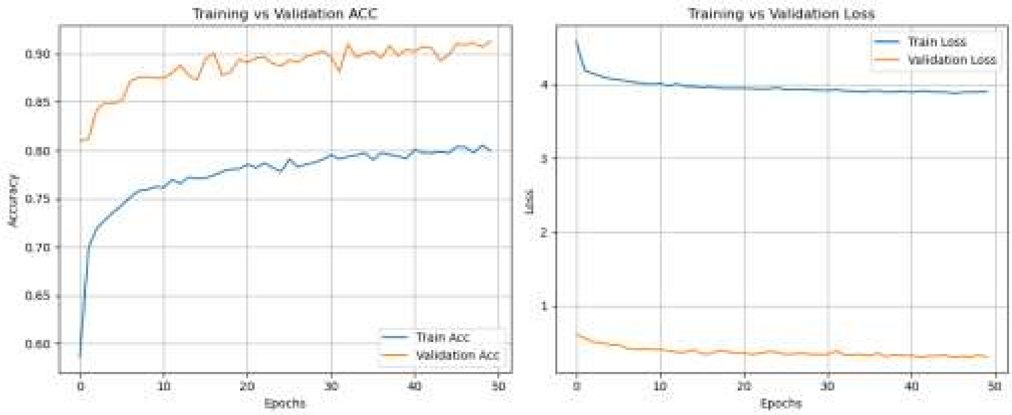

Figure 16. Inception V3 model Acc-Epoch and Loss-Epoch graphics

Notably, the validation results significantly improved compared to the training performance. This outcome can be attributed to the model not being trained from scratch. Instead, we leveraged pre-trained weights obtained from the ImageNet dataset, followed by a fine-tuning process. This approach enables the model to benefit from previously learned features, accelerating convergence and enhancing generalization capabilities on the target task.

The ROC-AUC graphic of the VGG-16 model is shown in Figure 17.

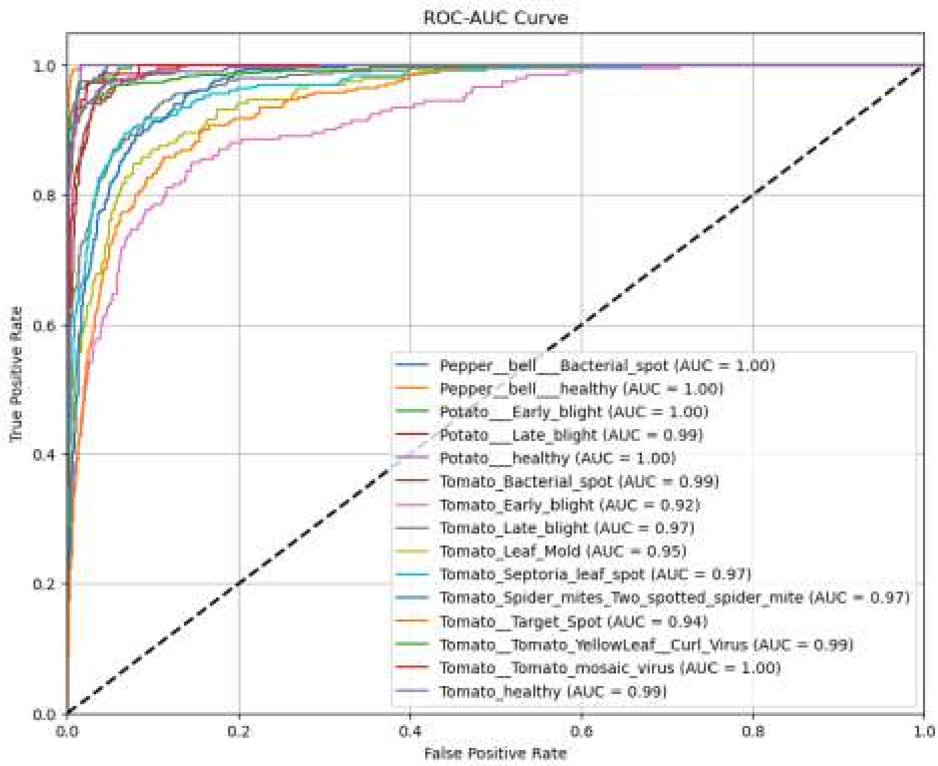

Figure 17. VGG-16 model ROC-AUC curve

The confusion matrix for VGG-16 is as shown in Figure 18.

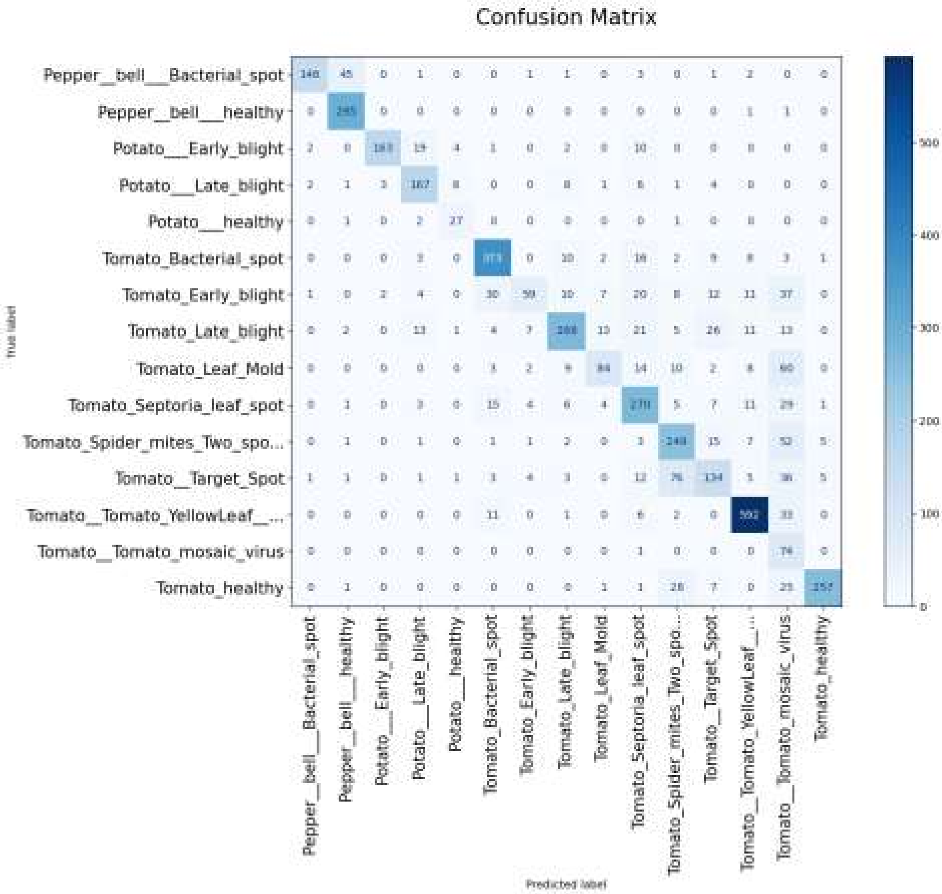

Figure 18. VGG-16 model confusion matrix

The Acc-Epoch and Loss-Epoch graphics for VGG-16 is shown in Figure 19.

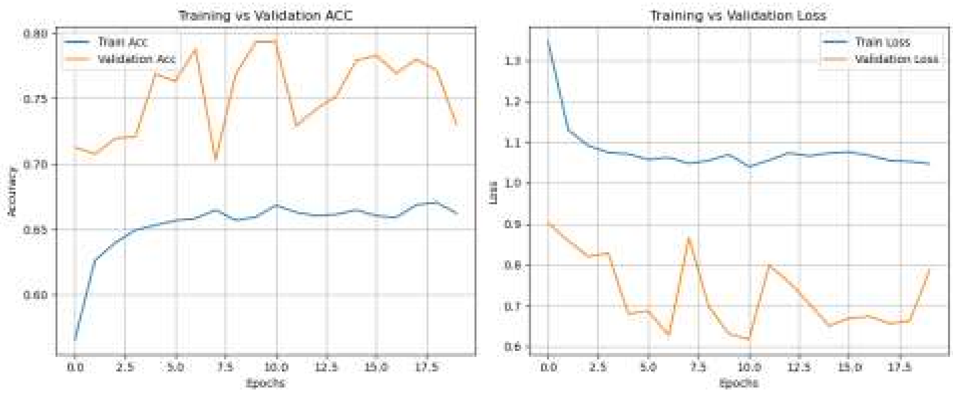

Figure 19. VGG-16 model Acc-Epoch and Loss-Epoch graphics

The ROC-AUC graphic of the VGG-19 model is shown in Figure 20.

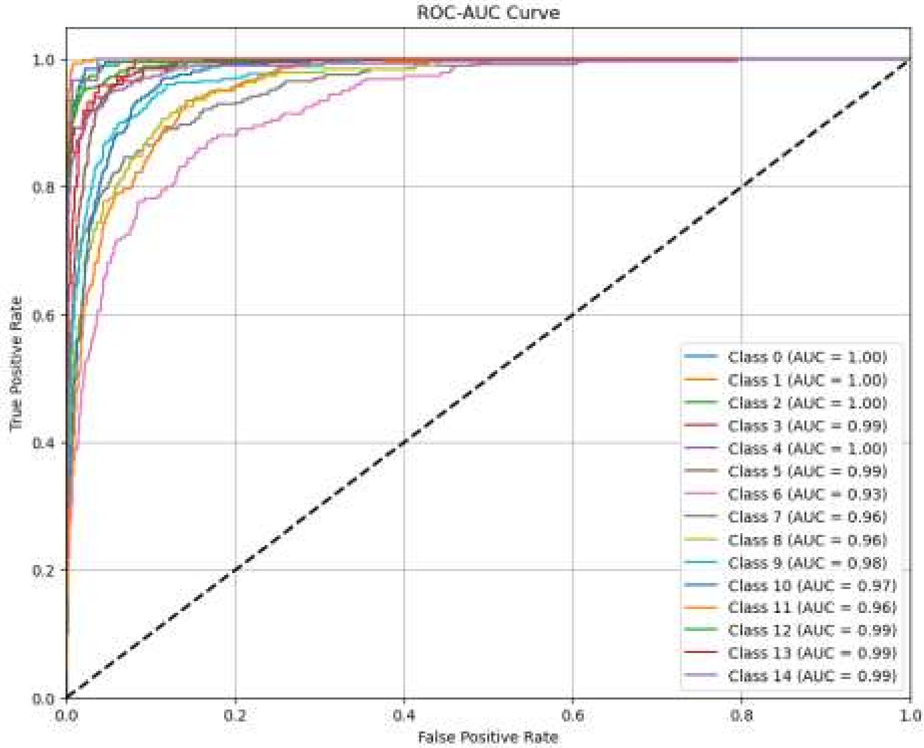

Figure 20. VGG-19 model ROC-AUC curve

The confusion matrix for VGG-19 is as shown in Figure 21.

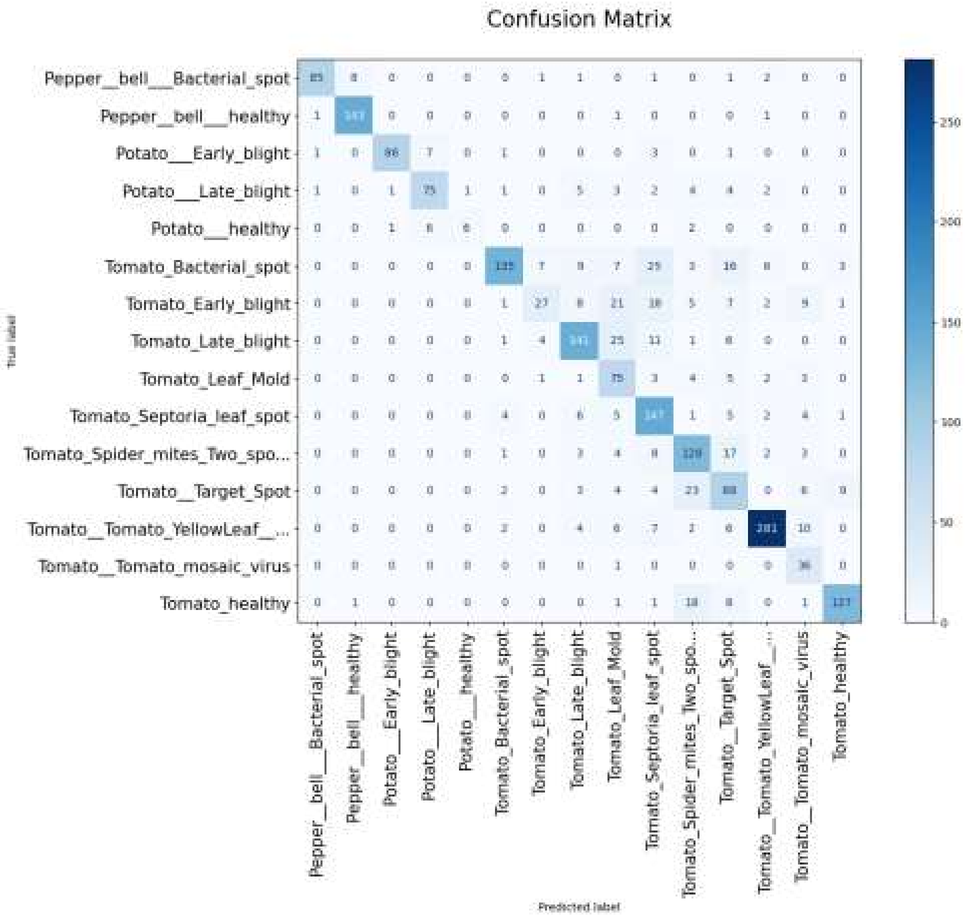

Figure 21. VGG-19 model confusion matrix

The Acc-Epoch and Loss-Epoch graphics for VGG-19 are shown in Figure 22.

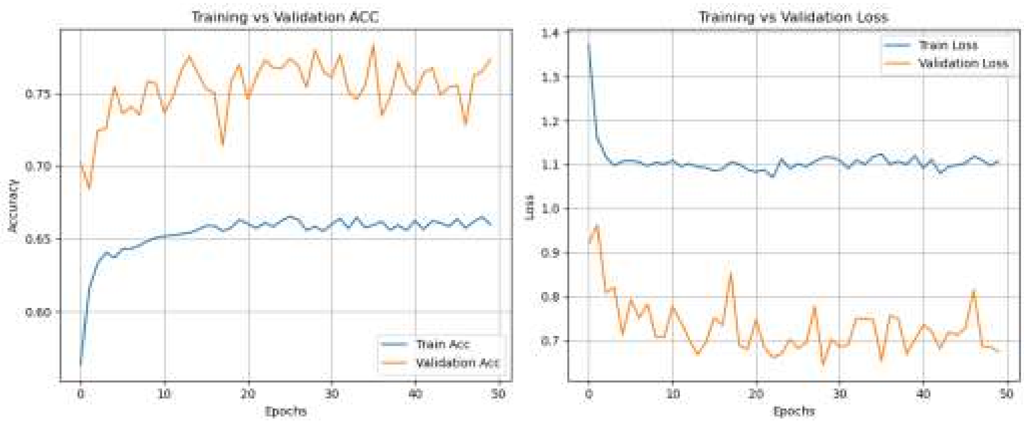

Figure 22. VGG-19 model Acc-Epoch and Loss-Epoch graphics

The ROC-AUC graphic of the MobileNet model is shown in Figure 23.

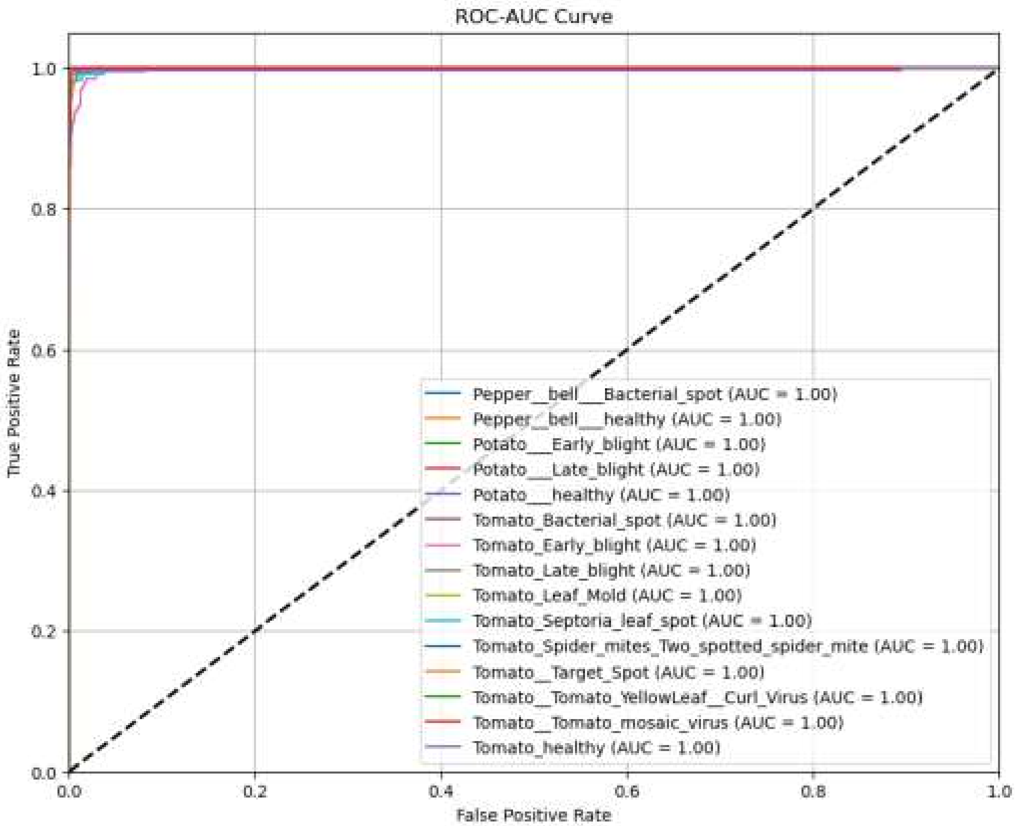

Figure 23. MobileNet model ROC-AUC curve

The confusion matrix for MobileNet is as shown in Figure 24.

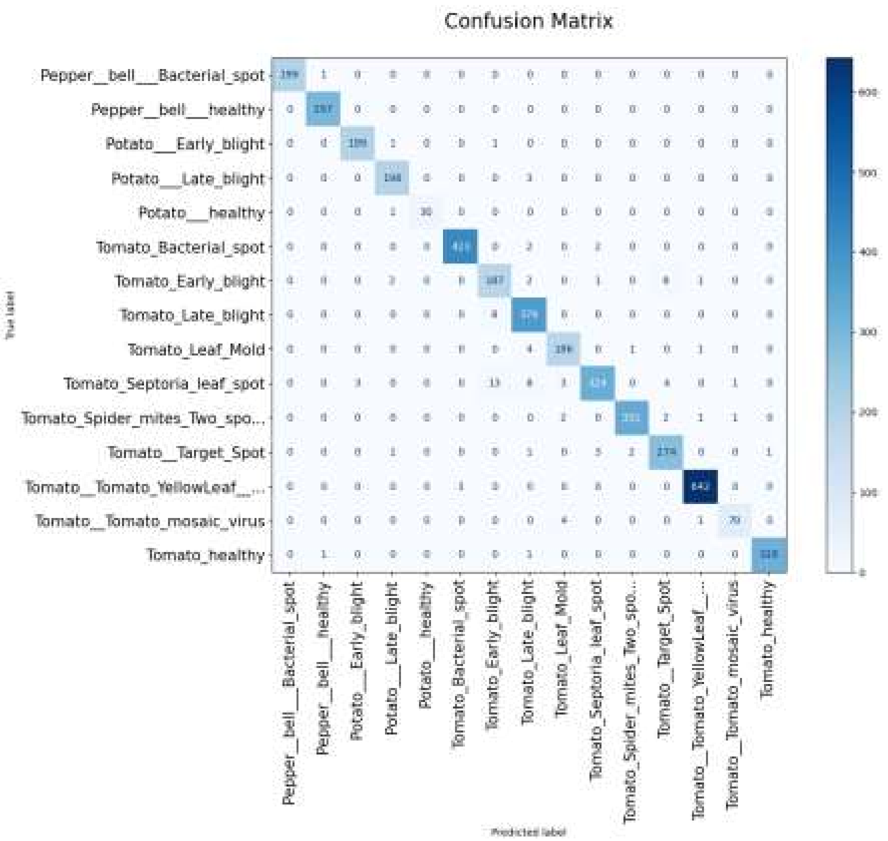

Figure 24. MobileNet model confusion matrix

The Acc-Epoch and Loss-Epoch graphics for MobileNet are shown in Figure 25.

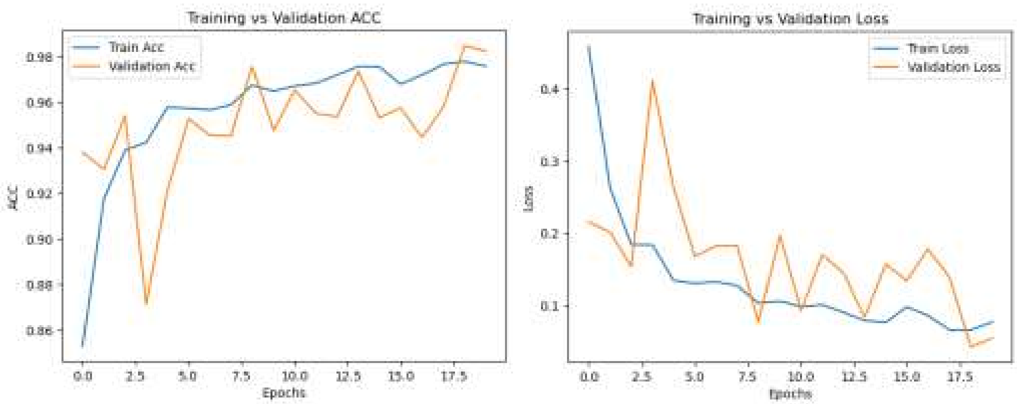

Figure 25. MobileNet model Acc-Epoch and Loss-Epoch graphics

The ROC-AUC graphic of the DenseNet121 model is shown in Figure 26.

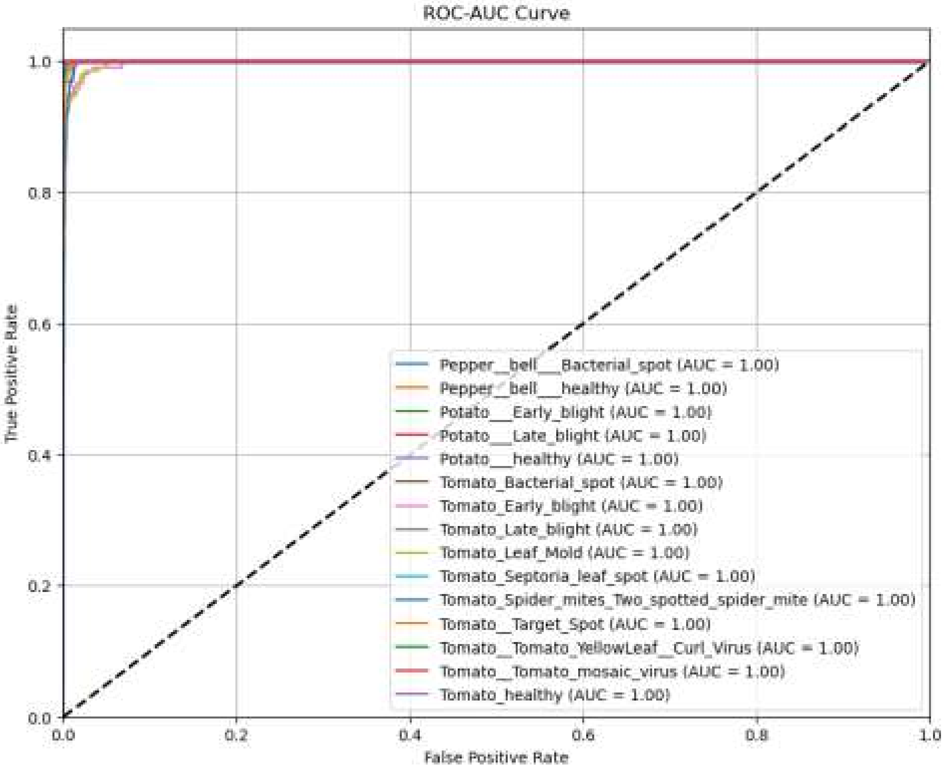

Figure 26. DenseNet121 model ROC-AUC curve

The confusion matrix for DenseNet121 is as shown in Figure 27.

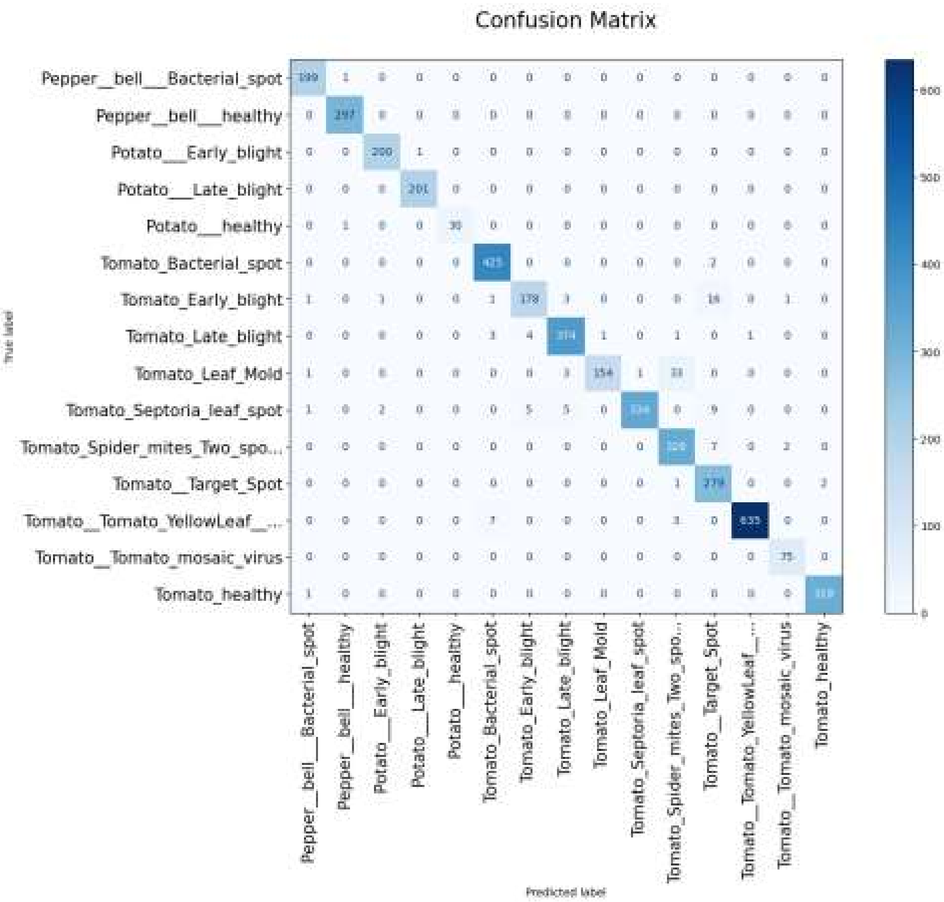

Figure 27. DenseNet model confusion matrix

The Acc-Epoch and Loss-Epoch graphics for DenseNet121 are shown in Figure 28.

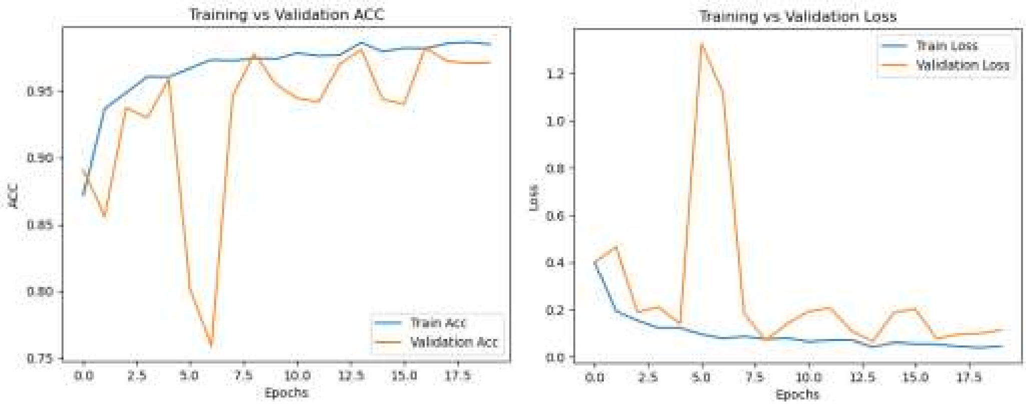

Figure 28. DenseNet121 model Acc-Epoch and Loss-Epoch graphics

The ROC-AUC graphic of the ConvNext model is shown in Figure 29.

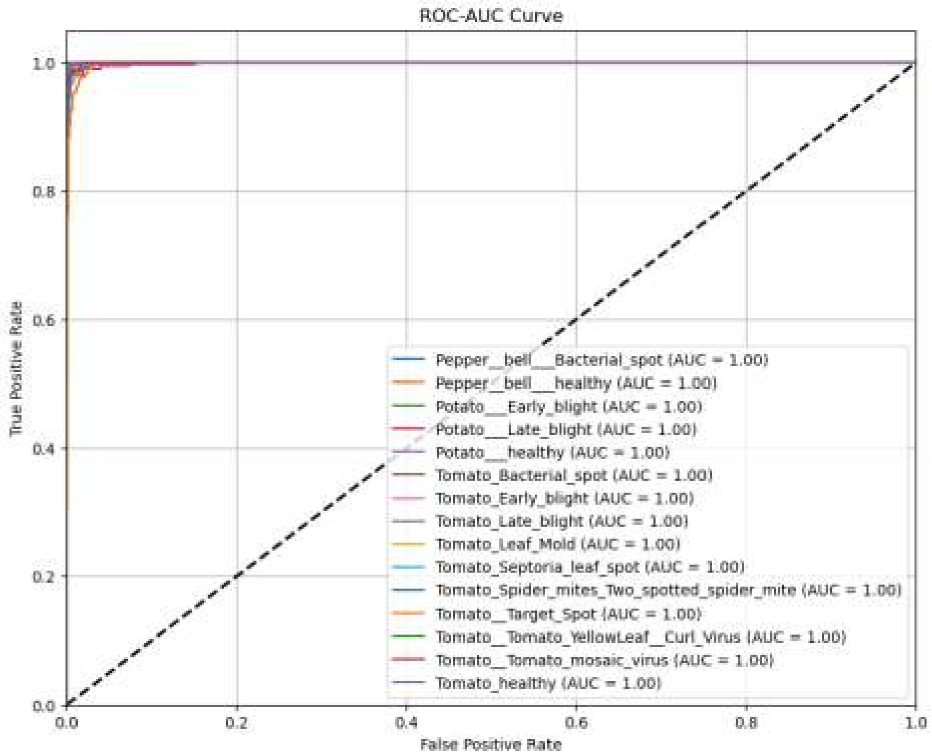

Figure 29. ConvNext model ROC-AUC curve

The confusion matrix for ConvNext is as shown in Figure 30.

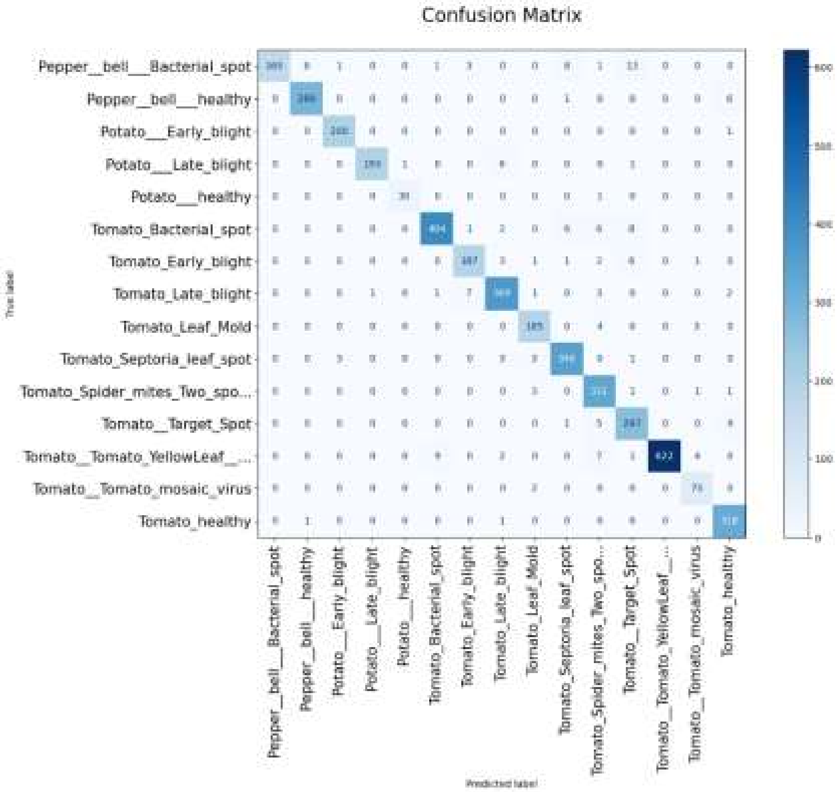

Figure 30. ConvNext model confusion matrix

The Acc-Epoch and Loss-Epoch graphics for ConvNext are shown in Figure 31.

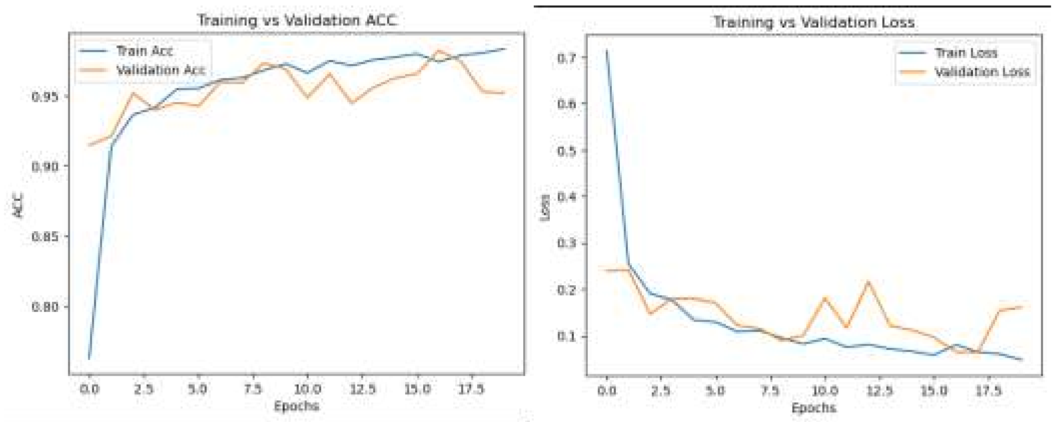

Figure 31. ConvNext model Acc-Epoch and Loss-Epoch graphics

Hybrid Model Results

The latter step involved developing hybrid models. MobileNet, the best model, was combined with other classic machine learning methods. Each hybrid model was re-evaluated based on the metrics discussed above. We conducted two separate tests to see the hybrid models before and after Fine Tune.

Table 8. Shows the results of the hybrid models created with the original MobileNet model taken before fine-tuning.

Table 8. Performance Comparison of Hybrid Models (Models before fine-tuning)

Model	Accuracy	Precision	Recall	F1 Score	Roc-Auc Score
MobileNet & KNN	69.37%	69.00%	64.00%	64.00%	91.48%
MobileNet & SVM	90.00%	89.00%	88.00%	88.00%	99.46%
MobileNet & Random Forest	67.00%	60.00%	54.00%	54.00%	95.15%
MobileNet & Logistic Regression	86.00%	86.00%	84.00%	85.00%	99.15%
MobileNet & Decision Tree	43.00%	36.00%	36.00%	36.00%	65.75%

Table 9 enumerates each hybrid model's performance results regarding accuracy, precision, recall, and F1 score, showing that hybrid models outperform non-hybrid models.

Table 9. Performance Comparison of Hybrid Models (MobileNet fine-tuned with the dataset)

Model	Accuracy	Precision	Recall	F1 Score	Roc-Auc Score
MobileNet & KNN	96.29%	96.00%	96.00%	96.00%	99.26%
MobileNet & SVM	**98.46%**	**98.00%**	**98.00%**	**98.00%**	**99.97%**
MobileNet & Random Forest	97.98%	98.00%	97.00%	97.00%	99.95%
MobileNet & Logistic Regression	98.39%	98.00%	98.00%	98.00%	99.97%
MobileNet & Decision Tree	86.02%	82.00%	82.00%	82.00%	90.69%

Figure 32. Shows the confusion matrix of the best hybrid model created with MobileNet and SVM.

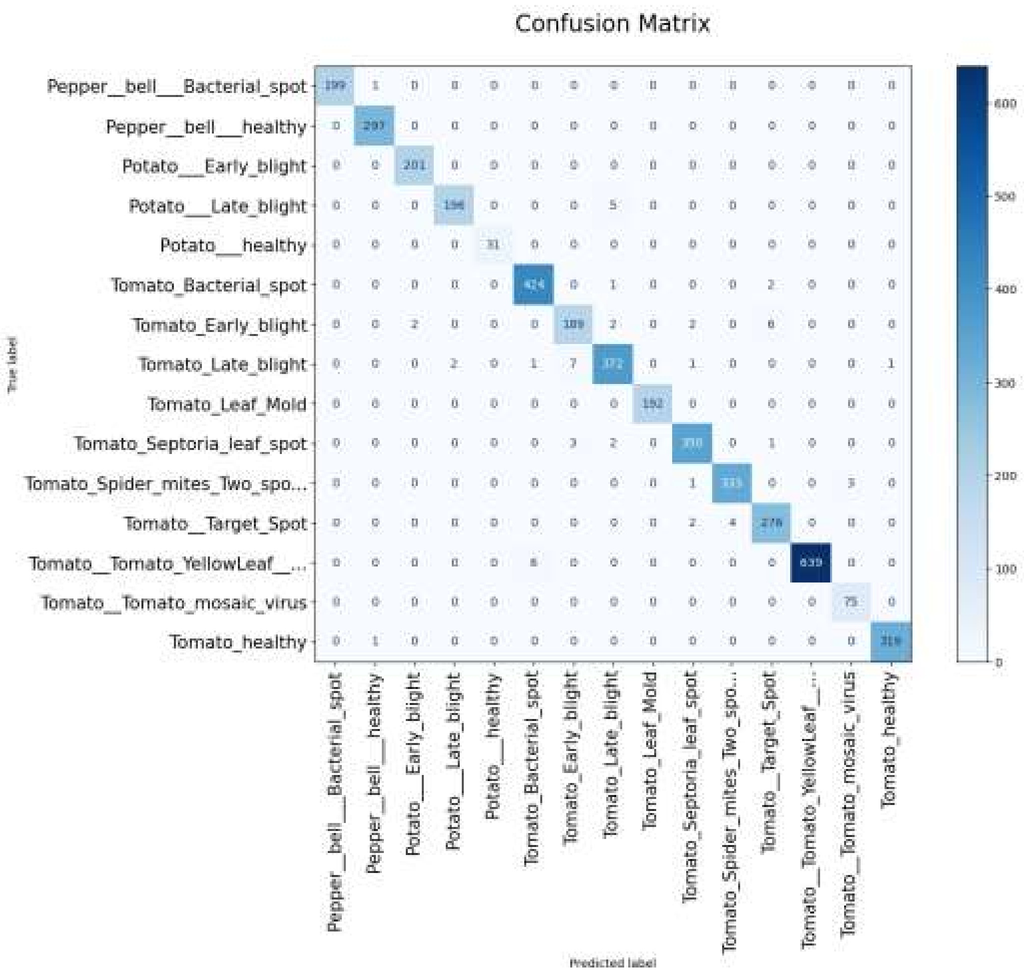

Figure 32. MobileNet & SVM models' confusion matrix

Mobile Net& KNN: Results with Mobile Net combined with the KNN model returned an accuracy of 96.29% on the test set. Precision was 96%, while Recall was 96%, thus showing that this model can recognize all positive instances without any false negatives. Regarding F1, which indicates the balance between precision and recall, it reached as much as 96%, and the AUC value was 99.26%, meaning a very excellent overall performance.

Mobile Net& SVM: The accuracy in the second model, with Mobile Net and SVM, is 98.46%. This model is the best. The precision is 98%, and the Recall is 98%, which gives an F1 Score of 98%, which can mean that the model's capability in distinguishing classes is highly effective, though the performance metrics went down by a margin.

Mobile Net & Random Forest: The model that married Mobile Net and Random Forest architectures combined returned an accuracy of 97.98%. Precision was 98%, Recall was 97%, thus yielding the value of F1 Score to 97% and the AUC 99.95%.

MobileNet & Logistic Regression: The fourth model, which combined the MobileNet and Logistic Regression architectures, returned an accuracy of 98.39%. Precision was 98%, Recall was 98%, thus yielding the F1 Score of 98% and the AUC of 99.97%.

MobileNet & Decision Tree: The final model, which combined the MobileNet and Decision Tree architectures, returned an accuracy of 86.02%. Precision was 82%, and Recall was 82%, thus yielding the value of F1 Score to 82% and the AUC 90.69%

The results depicted that SVM and Logistic Regression work best for the classification tasks in terms of accuracy and AUC, i.e., distinguishing between classes. Model 3 (Mobile Net &Random Forest) performed almost equally in accuracy and F1 Score, but Model 1 slightly outperformed the other regarding precision and recall. Model 4, on the other hand, lost more momentum than the other five models, with a clear drop in performance, especially at Precision.

CONCLUSION

This thesis studies hybrid CNN models for plant disease detection by training hybrid models. The research evaluated the performance of several machine learning-based convolutional neural network (CNN) models, such as ConvNext and VGG19. These models are initialized using weights previously trained on the ImageNet dataset and retrained on the PlantVillage dataset, and we obtained promising results.

The best model was MobileNet, which achieved encouraging results and very high accuracy compared to other models: Accuracy 97.70%, Precision 97.50%, Recall 97.30%, F1 Score 97.40%, and ROC-AUC 99.98%.

Second, we developed hybrid models by training classical classifiers such as SVM and KNN using the features we obtained after extracting CNN-based features. The best model was MobileNet, with SVM achieving an accuracy of 98.46%, Precision of 98%, Recall of 98%, F1 score of 98%, and ROC AUC score of 99.97%.

The hybrid mobile net model was the most successful, achieving the best accuracy, ROC-AUC, precision, and recall scores. This demonstrates the effectiveness of deep learning convolutional neural network models for detecting plant diseases.

Furthermore, we validated the mobile net model using classical training classifiers and achieved excellent accuracy.

Our research using these models has yielded a detailed understanding of plant diseases, significantly benefiting farmers. This information has the potential to change everything. By detecting and identifying diseases early, they can be treated and potentially prevented from occurring. Hybridization can expand robustness and efficiency in the plant disease classification system.

This hybrid model delivered superior accuracy and effectively detected multiple plant diseases, making it well-equipped for agricultural monitoring. This study opens a promising avenue for monitoring agricultural diseases in the future. Farmers and agricultural experts will pursue practical applications in the real world. Further work may include additional network combinations, for which, after testing these hybrid approaches on their datasets, the described hybrid approach can be better revealed in its effectiveness.

FUTURE WORKS

The proposed hybrid technology demonstrates significant potential for scalability and future advancements. Expanding the dataset to encompass a broader range of diseases from diverse geographical regions is essential to address plant diseases effectively. Although machine learning (ML) and deep learning (DL) models do not make decisions in the human sense, their predictive outputs can effectively support decision-making in critical domains such as medical diagnostics, financial forecasting, and autonomous systems. However, the predominantly correlational nature of traditional ML and DL models limits their reliability in high-stakes or complex environments, where causal understanding is crucial. Integrating causal inference into these models is increasingly recognized as a necessary step toward developing interpretable, robust, and trustworthy decision-support systems, particularly in dynamic agricultural contexts, thereby enhancing stakeholder confidence and satisfaction. The future steps for improvement can be outlined in the following points:

1. The research may offer the potential for transferring models to various fields of plant disease analysis, such as using these models in multiple areas of life to help improve accuracy and efficiency.
2. Further exploration may lead to fine-tuning the results or developing advanced or new models, which may improve model performance.
3. Real-time model analysis may enable the detection of various types and forms of plant diseases in the field.
4. Expanding the scope of this research to include disease analysis in multiple countries may lead to the development of more transparent and interpretable disease analysis systems.

Overall, this thesis has enhanced our understanding of plant disease analysis across all agricultural fields, laying the foundation for future studies on addressing complex plant diseases.

REFERENCES

Ajit, A., Acharya, K., & Samanta, A. (2020). *A Review of Convolutional Neural Networks.* IEEE.

Akhal, H. E., Yahya, A. B., Moussa, N., & Alaoui, A. E. (2023). *A novel approach for image-based olive leaf diseases classification using a deep hybrid model.* Ecological Informatics.

Arel, I., Rose, D. C., & Karnowski, T. P. (2010). *Deep Machine Learning - A New Frontier in Artificial Intelligence Research [Research Frontier].* IEEE.

Ashwini, C., & V.Sellam. (2024). *An optimal model for identification and classification of corn leaf disease using hybrid 3D-CNN and LSTM.* Biomedical Signal Processing and Control.

Balali, F., Nouri, J., Nasiri, A., & Zhao, T. (2020). *Machine Learning Principles.* Data Intensive Industrial Asset Management.

Balasundaram, A., & Sundaresan, P. (2025). *Tea leaf disease detection using segment anything model and deep convolutional neural networks.* Results in Engineering.

Basha, S. S., Dubey, S. R., Pulabaigari, V., & Mukherjee, S. (2020). *Impact of fully connected layers on performance of convolutional neural networks for image classification.* Neurocomputing.

Bonkra, A., Pathak, S., Kaur, A., & Shah, M. A. (2024). *Exploring the trend of recognizing apple leaf disease detection through machine learning: a comprehensive analysis using bibliometric techniques.* Artificial Intelligence Review.

Chen, J., Cheng, Y., Gan, Z., Gu, Q., & Liu, J. (2022). *Efficient Robust Training via Backward Smoothing.* Thirty-Sixth AAAI Conference on Artificial Intelligence.

Choi, R. Y., Coyner, A. S., Kalpathy-Cramer, J., Chiang, M. F., & Campbell, J. P. (2020). *Introduction to Machine Learning, Neural Networks, and Deep Learning.* . Trans. Vis. Sci. Tech.

Dolatabadian, A., Neik, T. X., Danilevicz, M. F., Upadhyaya, S. R., Batley, J., & Edwards, D. (2024). *Image-based crop disease detection using machine learning.* Plant Pathology.

Dong, X., Wang, Q., Ge, Q. H., Zhao, K., Wu, X., Wu, X., . . . Hao, G. (2023). *PDDD-PreTrain: A Series of Commonly Used Pre-Trained Models Support Image-Based Plant Disease Diagnosis.* Plant Phenomics.

FAO. (2002). *EVALUATION OF CERTAIN FOOD ADDITIVES AND CONTAMINANTS.* Evaluation of Certain Food Additives and Contaminants: Fifty-seventh Report ...

Gholamalinezhad, H., & Khosravi, H. (2020). *Pooling Methods in Deep Neural Networks, a Review.* arxiv.

Ghosh, A., Sufian, A., Sultana, F., Chakrabarti, A., & De, D. (2019). *Fundamental Concepts of Convolutional Neural Network.* Recent Trends and Advances in Artificial Intelligence and Internet of Things.

Goh, A. (1995). *Back-propagation neural networks for modeling complex systems.* Artificial Intelligence in Engineering.

Goyal, D., Banerjee, D., Chauhan, R., Devliyal, S., & Gill, K. S. (2024). *Advanced Techniques for Sweet Potato Leaf Disease Detection: A CNN-SVM Hybrid Approach.* IEEE.

Hinton, G. E. (2006). *Reducing the Dimensionality of Data with Neural Networks.* Science.

Howard, A., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., & Marco Andreetto, H. A. (2017). *MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications.* arxiv.

Huang, G., Zhuang Liu, Maaten, L. v., & Weinberge, K. Q. (2018). *Densely Connected Convolutional Networks.* arxiv.

Kavzoglu, T. (2009). *Increasing the accuracy of neural network classification using refined training data.* Environmental Modelling & Software.

Ketkar, N., & Moolayil, J. (2021). *Convolutional Neural Networks.* Deep Learning with Python.

Khan, A., Sohail, A., Zahoora, U., & Qureshi, A. S. (2020). *A survey of the recent architectures of deep convolutional neural networks.* Artificial Intelligence Review.

Meeradevi, V, R., Mundada, M. R., Sawkar, S. P., Bellad, R. S., & Keerthi, P. S. (2020). *Design and Development of Efficient Techniques for Leaf Disease Detection using Deep Convolutional Neural Networks.* IEEE.

Mehzabeen, S., & Gayathri, R. (2025). *Heuristically Improvised rice disease classification framework based on adaptive segmentation with the fusion of LSTM layer into Multi-Scale Residual attention Network.* Biomedical Signal Processing and Control.

Militante, S., Gerardo, B. D., & Dionisio, N. V. (2019). *Plant Leaf Detection and Disease Recognition using Deep Learning.* IEEE.

Narkhede, M. V., Bartakke, P. P., & Sutaone, M. S. (2022). *A review on weight initialization strategies for neural networks.* Artificial Intelligence Review.

Noyan, M. A. (2022). *Uncovering bias in the PlantVillage dataset.* arxiv.

Nuanmeesri, S. (2025). *Spectrum-Based Hybrid Deep Learning for Intact Prediction of Postharvest Avocado Ripeness.* IEEE.

Orecchio, C., Botto, C. S., Alladio, E., & Vincenti, M. (2025). *Non-invasive and early detection of tomato spotted wilt virus infection in tomato plants using a hand-held Raman spectrometer and machine learning modelling.* Plant Stress.

Patil, M. P., & Borse, I. S. (2025). *A Novel Hybrid Convolution and Multiscale Dilated EfficientNetB7-Based Plant Disease Detection and Classification with Adaptive Segmentation Procedures.* World Scientific.

Pavate, A., Kukreja, S., Janrao, S., Bankar, S., & Patil, R. (2025). *Efficient model for cotton plant health monitoring via YOLObased disease prediction.* Indonesian Journal of Electrical Engineering and Computer Science.

Pawar, A., Amrutkar, R., Somani, S., & Mundada, G. S. (2024). *Plant leaf disease detection using CNN .* AIP.

Pham, T. N. (2020). *Early Disease Classification of Mango Leaves Using Feed-Forward Neural Network and Hybrid Metaheuristic Feature Selection.* IEEE.

Saatkamp, A., Römermann, C., & Dutoit, T. (2010). *Plant Functional Traits Show Non-Linear Response to Grazing.* Folia Geobotanica.

Sawane, A., Bora, S., Vyawahare, A., Badri, U., More, P., & Patil, G. (2024). *Plant Disease Detection and Diagnosis.* Sant Gadge Baba Amravati University, Amravati.

Schilling, & Fabian. (2016). *The Effect of Batch Normalization on Deep Convolutional Neural Networks.* DiVA.

Sirohi, A., & Malik, A. (2021). *A Hybrid Model for the Classification of Sunflower Diseases Using Deep Learning.* IEEE.

Strange, R. N., & Scott, P. R. (2005). *Plant Disease: A Threat to Global Food Security.* ANNUAL REVIEWS.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2016). *Rethinking the Inception Architecture for Computer Vision.* CVPR.

Uchida, K., Tanaka, M., & Okutomi, M. (2018). *Coupled convolution layer for convolutional neural network.* Neural Networks.

Vijay, C., & Pushpalatha, K. (2024). *DV-PSO-Net: A novel deep mutual learning model with Heuristic search using Particle Swarm optimization for Mango leaf disease detection.* Journal of Integrated Science and Technology.

Wang, Q., & Cai, C. (2022). *Machine Learning Fundamentals.* Machine Learning in Chemical Safety and Health: Fundamentals with Applications.

Wang, Q., Ma, Y., Zhao, K., & Tian, Y. (2020). *A Comprehensive Survey of Loss Functions in Machine Learning.* Annals of Data Science.

Woo, S., Debnath, S., Hu, R., Chen, X., Liu, Z., Kweon, I. S., & Xie, S. (2023). *ConvNeXt V2: Co-designing and Scaling ConvNets with Masked Autoencoders.* arxir.

Personal Information	
Name Surname:	Zainab FADHIL ABBAS ISHKAYYIR
Birth Date:	
Birth Place:	
Country:	IRAK
Adress:	
Tel:	
E-mail:	Zainabfadel51@gmail.com
Education	
High school:	El-Tâli'aa Hazırlık Okulu
Bachelor:	Divaniye Üniversitesi
Master:	Atatürk Üniversitesi, Fen Bilimleri Enstitüsü, Bilgisayar Mühendisliği Anabilim Dalı
Information of Foreign Language	
English:	
Türkçe:	
Tezden Üretilmiş Yayınlar	
ICMUSTED Conference (2024) A HYBRID MODEL FOR PLANT DISEASE DETECTION BASED ON DEEP LEARNING.	