Deep Learning approach to Plant Disease Detection

A dissertation submitted in partial fulfillment of the academic requirement for the award of the degree of

BACHELOR OF ENGINEERING

In

ELECTRONICS AND COMMUNICATION ENGINEERING

By

NENAVATH DEEPAK 1602-20-735-128

MOHAMMED RIZWAN 1602-20-735-139

MOHAMMED SAMEER 1602-20-735-140

Under the guidance of

Dr. S. ARUNA DEEPTHI

Assistant Professor, E.C.E

Department of Electronics and Communication Engineering Vasavi College of Engineering (Autonomous)

ACCREDITED BY NAAC WITH 'A+' GRADE

IBRAHIMBAGH, HYDERABAD-500031

2020-2024

Department of Electronics and Communication Engineering

Vasavi College of Engineering (Autonomous)

ACCREDITED BY NAAC WITH 'A+' GRADE

IBRAHIMBAGH, HYDERABAD-500031

Date:

CERTIFICATE

This is to certify that the project work titled " **Deep Learning approach to Plant Disease Detection** " submitted by

NENAVATH DEEPAK 1602-20-735-128

MOHAMMED RIZWAN 1602-20-735-139

MOHAMMED SAMEER 1602-20-735-140

students of Electronics and Communication Engineering Department, Vasavi College of Engineering in partial fulfilment of the requirement for the award of the degree of Bachelor of Engineering in Electronics and Communication Engineering is a record of the bonafide work carried out by them during the academic year 2023-2024. The result embodied in this project report has not been submitted to any other university or institute for the award of any degree.

SUPERVISOR

Dr. S. ARUNA DEEPTHI

Assistant Professor

E.C.E Department

Head of the Department

Dr.E. SREENIVASA RAO

Professor and HOD

E.C.E Department

External Examiner

ACKNOWLEDGEMENTS

Without mentioning the people whose constant support and encouragement made it possible, the joy and elation that come with completing any endeavour successfully would be lacking. We are pleased to propose to you our project, which is the outcome of a well-considered synthesis of expertise and study.

It is our privilege to express our earnest gratitude and venerable regards to our internal guide **Dr. S. ARUNA DEEPTHI**, Assistant Professor, E.C.E. Department, Vasavi College of Engineering, Ibrahim Bagh, for the abundant and knowledgeable help during the planning and implementation of the project work. We appreciate his assistance and his insightful advice.

We record with pleasure our deep sense of gratitude to **Dr. E. SREENIVASA RAO**, Head of the Department, E.C.E. for the helpful advice and generous support we received during the project.

We also thank our Project coordinators **Dr.K.Veera Swamy**,Professor, Department of ECE and **Dr.K. Deepthi**, Associate Professor, Department of ECE, for their continuous guidance and valuable suggestions throughout our project.

Our sincere thanks to the **Principal** and **Management**, Vasavi College of Engineering, Hyderabad for providing all the facilities in carrying out the project successfully.

Also we acknowledge with thanks for the support extended by all the staff members and technical staff in shaping up our project. We are thankful to one and all who co-operated us during the course of our project work.

NENAVATH DEEPAK 1602-20-735-128

MOHAMMED RIZWAN 1602-20-735-139

MOHAMMED SAMEER 1602-20-735-140

DECLARATION

This is to state that the work presented in this thesis titled "**Deep Learning approach to Plant Disease Detection**" is a record of work done by us in the Department of Electronics and Communication Engineering, Vasavi College of Engineering,Hyderabad.No part of the thesis is copied from books/journals/internet and wherever the portion is taken, the same has been duly referred in the text. The report is based on the project work done entirely by us and not copied from any other source. I hereby declare that the matter embedded in this thesis has not been submitted by me in full or partial thereof for the award of any degree/diploma of any other institution or university previously.

Signature of the students

NENAVATH DEEPAK 1602-20-735-128

MOHAMMED RIZWAN 1602-20-735-139

MOHAMMED SAMEER 1602-20-735-140

ABSTRACT

Agriculture is a cornerstone of human civilization, contributing significantly to the economy and providing sustenance through food production. However, the growth of crops can be hindered by diseases, which pose a threat to agricultural productivity. Early and accurate detection and classification of these diseases are crucial to minimizing damage to plants and preventing further spread.

Farmers' typical way of predicting and classifying plant leaf diseases can be boring and erroneous. Problems may arise when attempting to predict the types of diseases manually. The inability to detect and classify plant diseases quickly may result in the destruction of crop plants, resulting in a signifcant decrease in products. Farmers that use computerized image processing methods in their felds can reduce losses and increase productivity.

The timely identification of plant diseases is paramount as they can significantly impact the growth of crops. While various Machine Learning (ML) models have been employed for disease detection and classification, the emergence of Deep Learning (DL) has revolutionized this area of research, offering promising potential for increased accuracy.

Deep Learning, particularly through Convolutional Neural Networks (CNNs), has demonstrated notable success in advanced image detection and classification tasks, especially when ample data and computational resources are available. However, the effectiveness of DL models may vary across different datasets, and performance evaluation using various metrics is essential to assess their efficacy.

Our work aims to provide a comprehensive overview of DL models utilized for visualizing different plant diseases, offering insights into their applications and performance in disease detection and classification tasks.

Table of Contents

Name of Content	Page No
1.Introduction	**9-14**
1.1 Motivation	11
1.2 Problem Statement	12
1.3 Objectives	13
1.4 Performance Metrics	14
1.4.1 Confusion Matrix	14
1.4.2 Accuracy	14
1.4.3 Precision	14
1.4.4 Recall	14
1.4.5 F1-Score	14
2. Literature Survey	**15-17**
3. About DataSet	**18-20**
4. Proposed Method	**21-37**
4.1 Support Vector Machine (SVM)	21-23
4.2 Convolutional Neural Network (CNN)	24-26
4.2.1 CNN Overview	24
4.2.2 CNN Architecture	25-26
4.3 Transfer Learning with Pre-trained Models	27-37
4.3.1 About Transfer Learning	27-29
4.3.1.1 Architecture and Workflow	27
4.3.1.2 Traditional (vs) Transfer Learning	28
4.3.1.3 Applications & Advantages	29
4.3.2 Inception-V3	30-31
4.3.3 ResNet-50	32-33
4.3.4 VGG-16	34-35
4.3.5 VGG-19	36-37
5. Results	**38-49**
6. Conclusion	**50**
7. Future Scope	**51**
8. References	**52**
9. Plagarism Report	**53-59**

LIST OF FIGURES

Figure No.	Figure Name	Page No.
1	Summary of the evolution of deep learning from 1943–2006	9
2	Flow diagram of DL implementation	10
3	Data-Set: Plant-Village	18
4-6	Data-Splitting	19
7	Image Samples of Plant-Village DataSet	20
8	System Block Diagram	21
9	System Architecture	22
10	CNN Architecture	24
11	Computer vision-based techniques for plant disease detection and classification	26
12	Transfer Learning Architecture	27
13	Fine-Tuning VS Feature-Extraction	28
14	Inception-V3 Model	31
15	ResNet-50 Model	32
16	VGG-16 Model	34
17	VGG-19 Model	36
18-19	SVM Results	38
20-24	CNN Results	39-40
25-29	Inception-V3 Results	41-42
30-34	ResNet-50 Results	43-44
35-39	VGG-16 Results	45-46
40-44	VGG-19 Results	47-48

LIST OF TABLES

Table No.	Table Name	Page No.
1	Traditional Learning vs Transfer Learning	20
2	Comparison of Various Models	41

CHAPTER 1 : INTRODUCTION

The development of Deep Learning (DL) has progressed through various stages, each marked by significant advancements and innovations. Originating in 1943, DL's roots can be traced back to early research on threshold logic, which laid the foundation for creating computational systems inspired by the human brain's biological processes. This initial phase introduced key concepts such as backpropagation, the chain rule, and the Neocognitron, paving the way for further exploration into neural networks and their practical applications.

However, DL truly came into its own during the modern era, which began around 2012. This period witnessed a resurgence of DL research driven by breakthroughs in algorithms and architectures. A significant milestone was the success of AlexNet in the ImageNet challenge, demonstrating DL's capabilities in large-scale visual recognition tasks. This achievement acted as a catalyst for a wave of innovation, leading to the development of new architectures and techniques that expanded the scope of DL's capabilities.

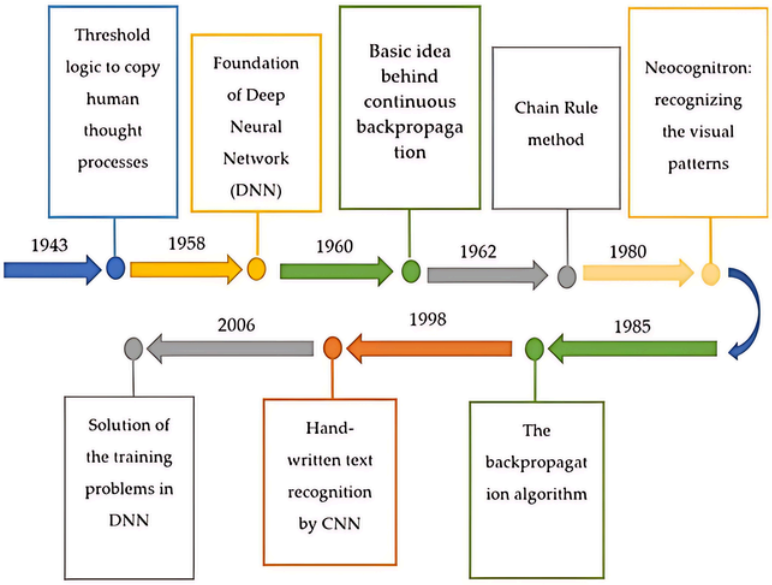


Figure 1 Summary of the evolution of deep learning from 1943–2006.

DL's impact has been felt across various industries, revolutionizing sectors such as self-driving vehicles, healthcare, and finance. With advancements in text recognition, earthquake prediction, and image recognition, DL has become a fundamental technology driving a wide range of applications by extracting insights from complex datasets.

Researchers and practitioners utilize a variety of performance metrics to evaluate the effectiveness of DL algorithms. These metrics include error rates, precision, recall, and classification accuracy, which serve as benchmarks for assessing the performance and reliability of DL models across different tasks and datasets. By leveraging these metrics, researchers can refine and optimize their approaches, ensuring the development of robust DL models.

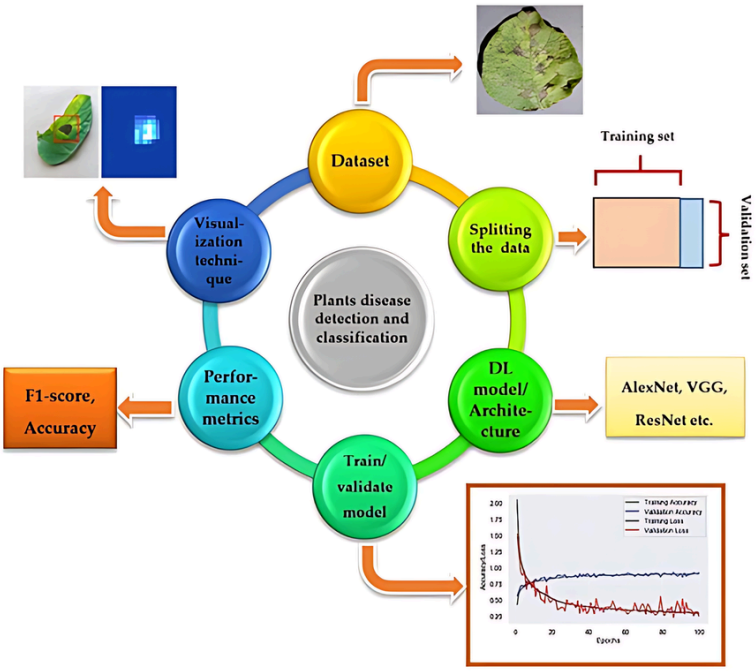


Figure 2 Flow diagram of DL implementation

As DL continues to progress, researchers and practitioners are dedicated to pushing the boundaries of innovation within the field. Each breakthrough contributes to strengthening DL's position as a transformative force in technology, driving advancements and innovation across various industries. This ongoing evolution reshapes our understanding of the world and transforms how we interact with it.

Figure 2 depicts the streamlined process of implementing Deep Learning (DL) for plant disease detection and classification. It begins with dataset collection "**Plant-Village**", followed by a split into 70% for training , 20% for testing and 10% for validation. DL models are then trained, either from scratch or using transfer learning. Performance is evaluated with training/validation plots, and metrics are used for image classification.

1.1 Motivation

The motivation behind this project stems from the critical need to address the challenges posed by plant diseases in agriculture. Plant diseases significantly impact crop yield, quality, and overall agricultural productivity, thereby posing a threat to food security and livelihoods worldwide. Traditional methods of disease detection and classification are often labor-intensive, time-consuming, and prone to errors, leading to delays in diagnosis and ineffective management strategies.

To address these challenges, the project aims to harness the advancements in Deep Learning (DL) techniques to revolutionize plant disease detection and classification. DL, as a subset of Machine Learning (ML), has exhibited remarkable prowess in tasks such as image recognition and classification. By employing DL models to analyze images of diseased plants, the project endeavors to achieve early and precise identification of plant diseases, facilitating prompt interventions to mitigate their impact.

Furthermore, the project is motivated by the potential of DL to enhance the efficiency and scalability of disease detection procedures. Through the automation of detection and classification tasks using DL algorithms, researchers aim to streamline agricultural processes, reduce dependence on manual labor, and augment overall productivity. This automation holds particular promise for large-scale farming operations, where timely disease detection is pivotal in averting widespread crop damage.

Moreover, the project seeks to address prevailing research gaps and constraints in the field of plant disease detection. By conducting a comprehensive review of DL architectures, visualization techniques, and performance metrics, the project aims to identify promising methodologies for enhancing disease detection accuracy. This research endeavor could lay the groundwork for the development of more robust and dependable tools for plant disease management.

Ultimately, the overarching objective of the project is to contribute to sustainable agriculture and global food security by harnessing the potential of DL to effectively combat plant diseases. By advancing the state-of-the-art in disease detection and classification, the project strives to empower farmers, researchers, and agricultural stakeholders with innovative solutions to mitigate the impact of plant diseases and ensure the resilience and productivity of the agricultural sector.

1.2 Problem Statement

The agriculture sector in India stands as a cornerstone of the economy, providing employment to nearly 50% of the workforce. With India being the world's largest producer of essential crops like pulses, rice, wheat, and spices, the economic well-being of farmers hinges on the quality and yield of their produce. However, the growth and yield of crops are significantly threatened by various plant diseases, which can disrupt ecological balance and jeopardize farmer's livelihoods.

Manual detection of plant diseases, particularly through the examination of leaf images, is a labor-intensive and time-consuming process. Early detection of diseases is crucial to mitigate their impact on crop yield and quality. Therefore, there is an urgent need to develop computational methods for automated disease detection and classification using leaf images, leveraging the advancements in technology.

Key Points to Address:

1. **Significance of Plant Disease Detection:** The health and economic prosperity of farmers rely heavily on the timely detection and management of plant diseases. Detecting diseases at their initial stages is essential for preventing widespread crop damage and ensuring sustainable agricultural practices.
2. **Automated Disease Detection:** Manual inspection of plants for disease symptoms is impractical due to its tedious nature. Developing automated techniques for disease detection using computational methods is imperative to streamline the process and enable early intervention.
3. **Symptom Manifestation in Leaves:** Symptoms of plant diseases often manifest visibly in various parts of the plant, including leaves. Analyzing leaf images for disease symptoms provides valuable insights into the health status of the plant and aids in early diagnosis.
4. **Challenges in Manual Detection:** The manual detection of plant diseases from leaf images poses several challenges, including subjectivity in interpretation, time consumption, and limited scalability. Addressing these challenges requires the adoption of automated computational approaches.
5. **Role of Computational Methods:** Leveraging computational methods, such as deep learning and image processing techniques, offers a promising avenue for automating the detection and classification of plant diseases. These methods can efficiently analyze large volumes of leaf images and accurately identify disease symptoms.
6. **Context of Indian Agriculture:** Given the significance of agriculture in India and its contribution to the economy, there is a particular emphasis on developing tailored solutions for detecting and managing plant diseases prevalent in Indian crops.

1.3 Objectives

The primary objective of this project is to develop an automated disease detection system for Indian agriculture, focusing on the analysis of leaf images. Specifically, the project aims to:

1. **Data Collection and Compilation:**
   a. Gather a diverse dataset of leaf images representing various crops commonly cultivated in Indian agriculture.
2. **Preprocessing and Augmentation:**
   a. Preprocess the collected leaf images to standardize their size, resolution, and color space.
   b. Augment the dataset to increase its diversity and robustness, incorporating techniques such as rotation, flipping, and scaling.
3. **Model Development and Training:**
   a. Explore and implement state-of-the-art deep learning architectures suitable for image classification tasks.
   b. Train the chosen models using the preprocessed and augmented dataset to learn the features indicative of various plant diseases.
4. **Model Evaluation and Validation:**
   a. Evaluate the performance of the trained models using appropriate evaluation metrics such as accuracy, precision, recall, and F1-score.
   b. Validate the models using separate test datasets to assess their generalization capability and robustness to unseen data.
5. **Fine-tuning and Optimization:**
   a. Fine-tune the trained models to optimize their performance for detecting specific plant diseases prevalent in Indian agriculture.
   b. Explore techniques such as transfer learning and hyperparameter tuning to enhance model accuracy and efficiency.
6. **Deployment and Integration:**
   a. Develop a user-friendly interface or application for farmers and agricultural stakeholders to access the automated disease detection system.

By accomplishing these objectives, the project aims to deliver a robust and effective automated disease detection system tailored to the specific needs and challenges of Indian agriculture.

1.4 Performance Metrices

Performance metrics are frequently employed to assess the effectiveness of plant disease classification, detection, and segmentation models. The confusion matrix is a key tool in evaluating classification algorithms, depicting the predicted class versus the actual class for each sample. It offers a comprehensive view of correct and incorrect classifications, aiding in the identification of areas requiring enhancement and providing insights into algorithm performance.

Confusion Matrix : A confusion matrix provides a structured summary of predictions, illustrating the accuracy and errors for each class. It aids in identifying which classes are being mistakenly classified as others by the model.

Accuracy: Accuracy measures the proportion of correctly classified instances out of the total number of instances. Mathematically, it is represented as the ratio of true positives and true negatives to the total number of instances.

$$Accurracy = \frac{True\ Positive\ Ratio + True\ Negative\ Ratio}{Total\ number\ of\ Samples}$$

Precision: Precision quantifies the proportion of correctly classified positive instances out of the total number of predicted positive instances. Mathematically, it is calculated as the ratio of true positives to the sum of true positives and false positives.

$$Precision = \frac{True\ Positive\ Ratio}{True\ Positive\ Ratio + False\ Positive\ Ratio}$$

Recall (Sensitivity): Recall assesses the proportion of correctly classified positive instances out of the total number of actual positive instances. Mathematically, it is calculated as the ratio of true positives to the sum of true positives and false negatives.

$$Recall = \frac{True\ Positive\ Ratio}{True\ Positive\ Ratio + False\ Negative\ Ratio}$$

F1 Score: The F1 score represents the harmonic mean of precision and recall, providing a balanced measure of a model's performance. Mathematically, it is calculated as the harmonic mean of precision and recall, offering a single metric to evaluate classification performance.

$$F1 - Score = \frac{2*(Precision*Recall)}{Precision + Recall}$$

2. Literature Survey

[1] S.V. Militante, B.D. Gerardo, N.V. Dionisio, Plant leaf detection and disease recognition using deep learning, in: 2019 IEEE Eurasia Conference on IOT, Communication and Engineering, ECICE), 2019, pp. 579–582.

This research has leveraged deep learning for plant disease diagnosis using computer vision. One study achieved a 100% accuracy rate by employing a convolutional neural network (CNN) on a dataset of 35,000 images of healthy and diseased plants. This outperformed a previous method which achieved 96.5% accuracy but required significant data for precise recognition.

[2] A. Venkataramanan, D.K.P. Honakeri, P. Agarwal, Plant disease detection and classification using deep neural networks, Int. J. Comput. Sci. Eng. 11 (9) (2019) 40–46.

Another study focused on documenting tomato plant leaf infections using image processing techniques such as clustering and image segmentation. Utilizing a CNN algorithm, hierarchical features were extracted from input images and compared with a training set to classify them. Additionally, fuzzy logic, hybrid algorithms, and artificial neural networks were considered. While effective, this method was time-consuming during the training process.

[3]G.K. Sandhu, R. Kaur, Plant disease detection techniques: a review, in: 2019 International Conference on Automation, Computational and Technology Management (ICACTM), London, United Kingdom, Apr. 2019, pp. 34–38, https:// doi.org/10.1109/ICACTM.2019.8776827.

This study offered a comprehensive overview of plant leaf disease identification techniques, utilizing methods like Support Vector Machine (SVM) alongside algorithms such as Otsu's method and K-means clustering. However, automation was necessary for outdoor and diverse environmental conditions, resulting in significant time consumption for disease detection.

[4] M. Nagaraju and P. Chawla, "Systematic review of deep learning techniques in plant disease detection", Int. J. Syst. Assurance Eng. Manage., vol. 11, no. 3, pp. 547-560, Jun. 2020.

Existing surveys have covered some strategies and highlighted significant issues, but there is a need to assess the potential application of various techniques in real-world settings.

[5] A. Abade, P. A. Ferreira and F. de Barros Vidal, "Plant diseases recognition on images using convolutional neural networks: A systematic review", Comput. Electron. Agricult., vol. 185, Jun. 2021.
[6] R. I. Hasan, S. M. Yusuf and L. Alzubaidi, "Review of the state of the art of deep learning for plant diseases: A broad analysis and discussion", Plants, vol. 9, no. 10, pp. 1302, Oct. 2020.

studies predominantly focused on CNN models in PDD and concluded that there is still room for improvement in methods, especially in identical visual environments.

[7] Z. Gao, L. R. Khot, R. A. Naidu and Q. Zhang, "Early detection of grapevine leafroll disease in a red-berried wine grape cultivar using hyperspectral imaging", Comput. Electron. Agricult., vol. 179, Dec. 2020.

A recent review highlighted the importance of existing methods in plant disease detection (PDD), focusing on segmentation, classification, localization, and disease techniques. However, there remains a gap in comprehensively examining image-centered PDD.

[8] S. K. Noon, M. Amjad, M. A. Qureshi and A. Mannan, "Use of deep learning techniques for identification of plant leaf stresses: A review", Sustain. Comput. Informat. Syst., vol. 28, Dec. 2020.

This study evaluated the performance of Convolutional Neural Network (CNN) methods in PDD, particularly for fruits, vegetables, and various plants.

[9] R. Manavalan, "Automatic identification of diseases in grains crops through computational approaches: A review", Comput. Electron. Agricult., vol. 178, Nov. 2020.

Other research focused on barley, maize, rice, soybeans, and wheat, comparing the benefits and drawbacks of different image processing (IP) techniques, segmentations, feature selections, and classifications.

[10] J. G. A. Barbedo, "A review on the main challenges in automatic plant disease identification based on visible range images", Biosyst. Eng., vol. 144, pp. 52-60, Apr. 2016.

A comprehensive review from the early 2000s investigated existing approaches in PDD, with implications for fully automated detection and identification using digital methods.

[11] Z. Iqbal, M. A. Khan, M. Sharif, J. H. Shah, M. H. ur Rehman and K. Javed, "An automated detection and classification of citrus plant diseases using image processing techniques: A review", Comput. Electron. Agricult., vol. 153, pp. 12-32, Oct. 2018.

Many studies have used the "k-means" segmentation strategy for identifying plant diseases, indicating its frequent usage in the field.

The research explored methodologies for plant disease detection using deep learning and computer vision. They achieved high accuracy rates, reaching 100% in one study by incorporating various leaf diseases. Techniques like convolutional neural networks (CNN) were used for classification, but challenges remain regarding data requirements, time consumption, and the need for automation in diverse environmental conditions.

3. About DataSet

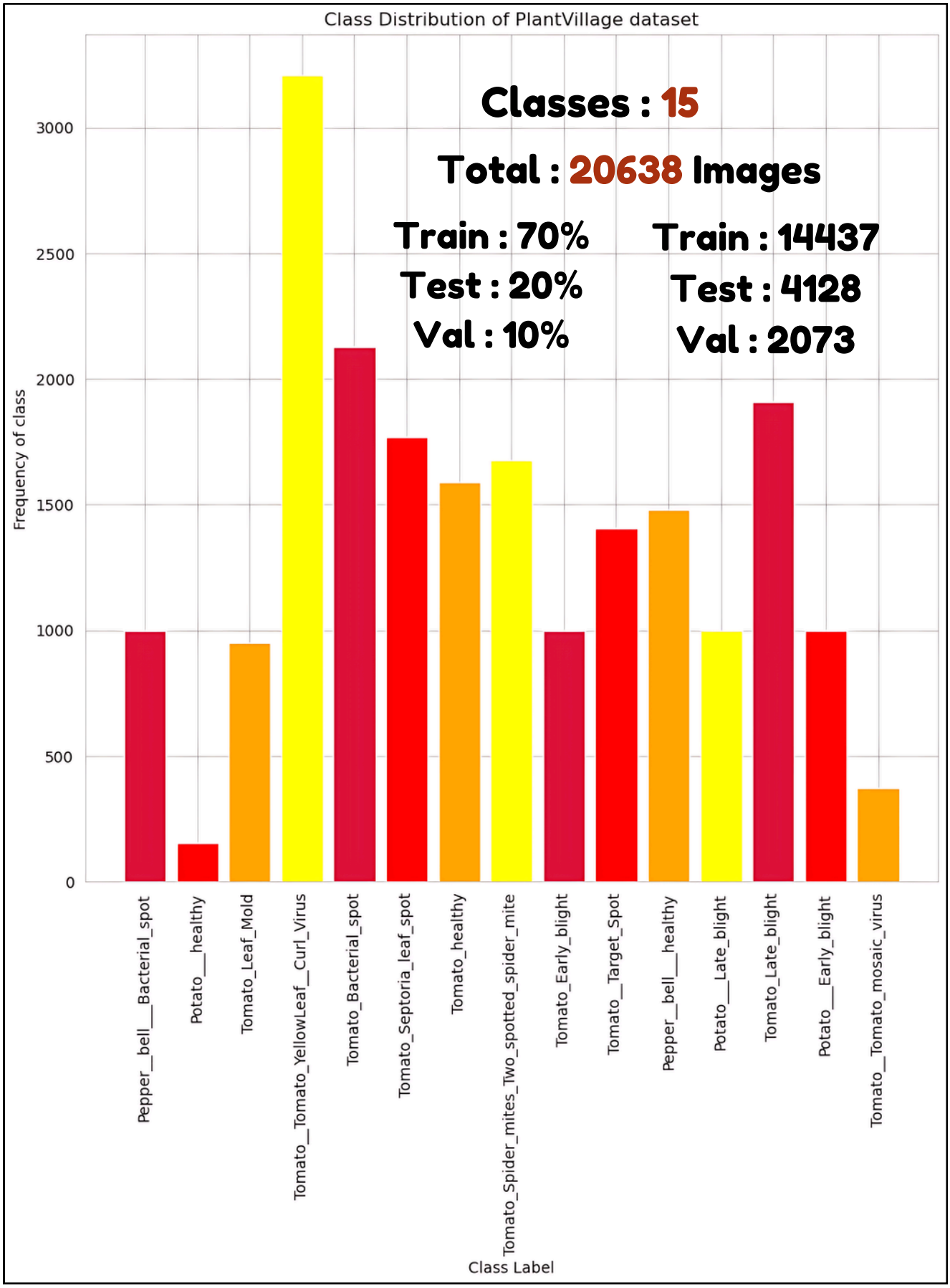
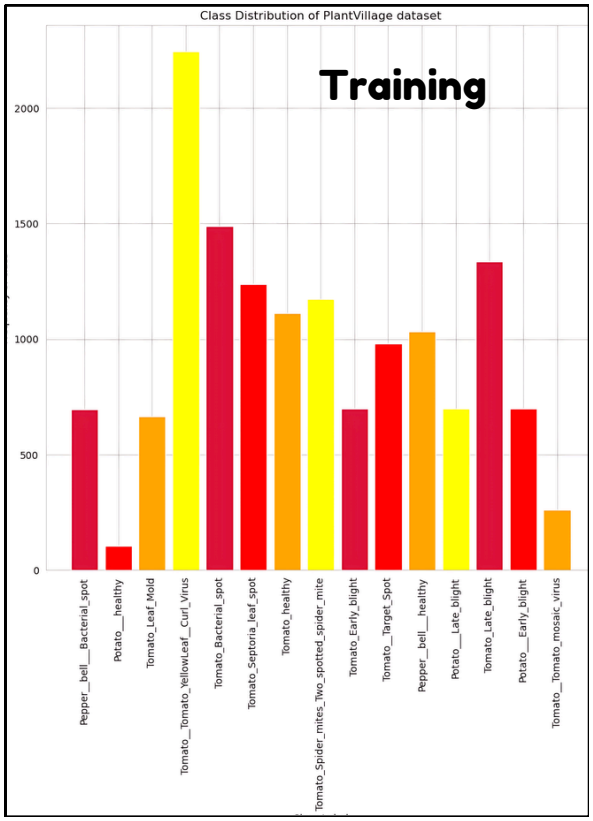
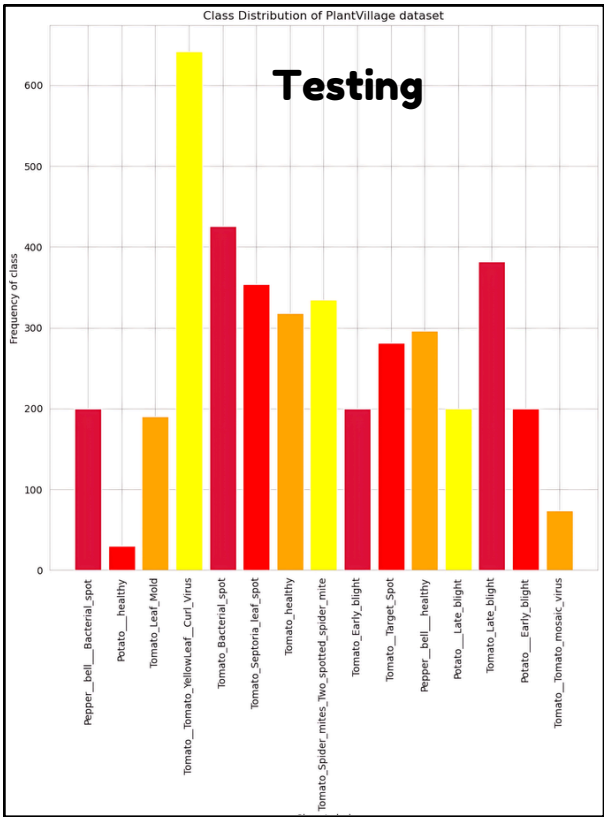


Figure 3 . Data-Set : Plant-Village





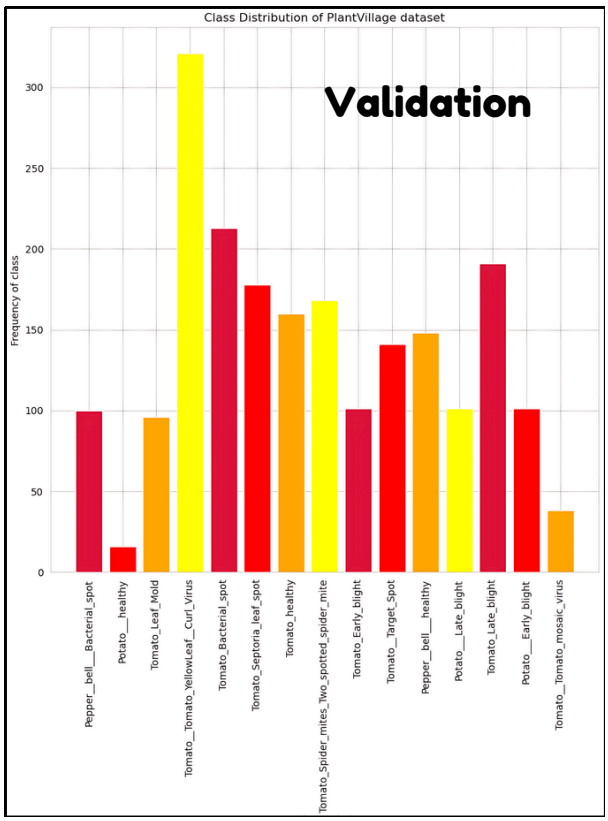


Figure 4,5,6 Data-Splitting

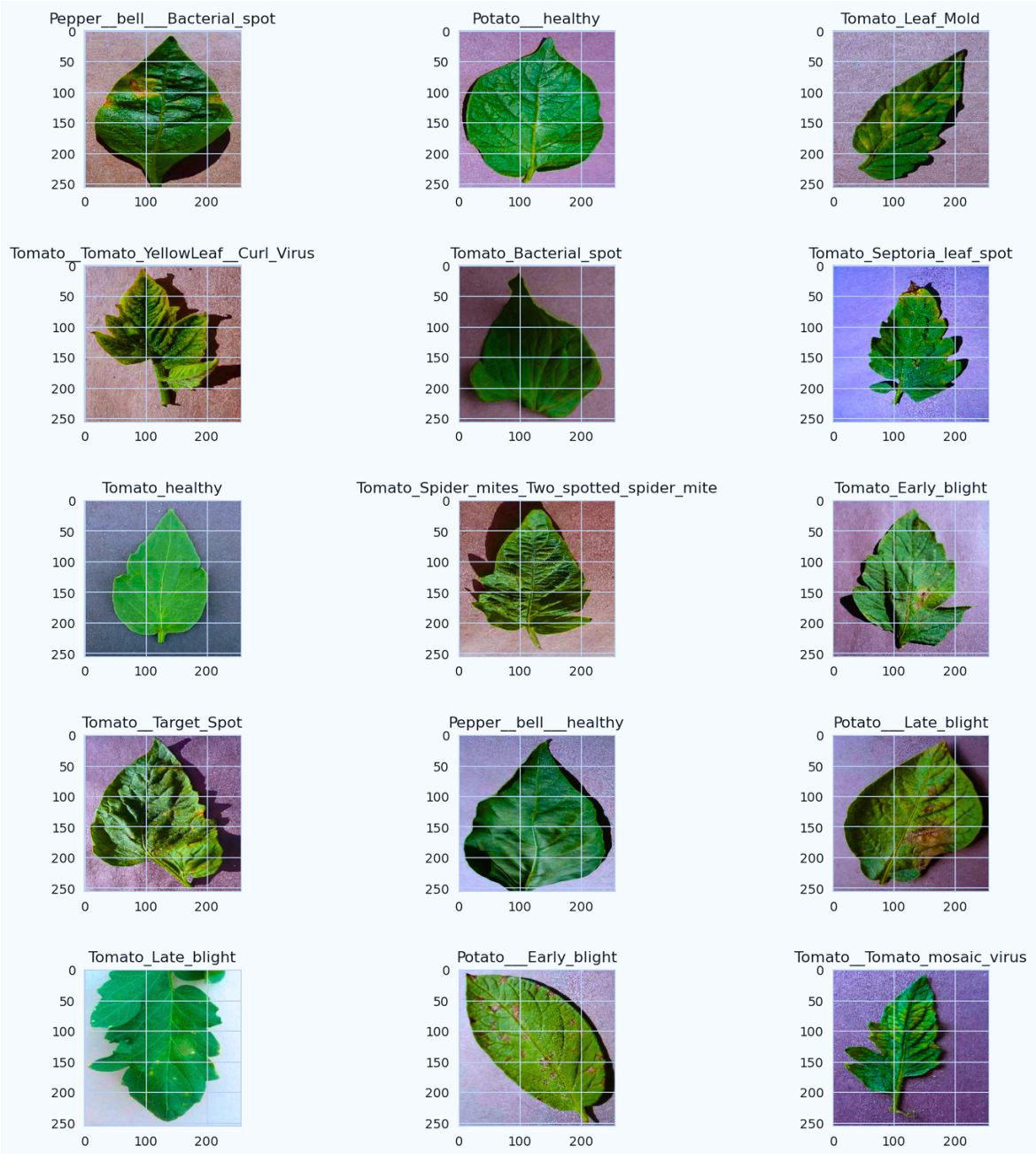


Figure 7 . Image Samples of Plant-Village DataSet

Classes : 15

'Pepper__bell___Bacterial_spot',
'Pepper__bell___healthy',
'Potato___Early_blight',
'Potato___Late_blight',
'Potato___healthy',
'Tomato_Bacterial_spot',
'Tomato_Early_blight',
'Tomato_Late_blight',
'Tomato_Leaf_Mold',
'Tomato_Septoria_leaf_spot',
'Tomato_Spider_mites_Two_spotted_spider_mite',
'Tomato__Target_Spot',
'Tomato__Tomato_YellowLeaf__Curl_Virus',
'Tomato__Tomato_mosaic_virus',
'Tomato_healthy'

4. PROPOSED METHOD

4.1 Support Vector Machine(SVM):

Support Vector Machine (SVM) is a powerful supervised machine learning algorithm widely used for classification tasks, including disease detection in plants. In the proposed system, SVM plays a crucial role in distinguishing between healthy and infected plant samples.

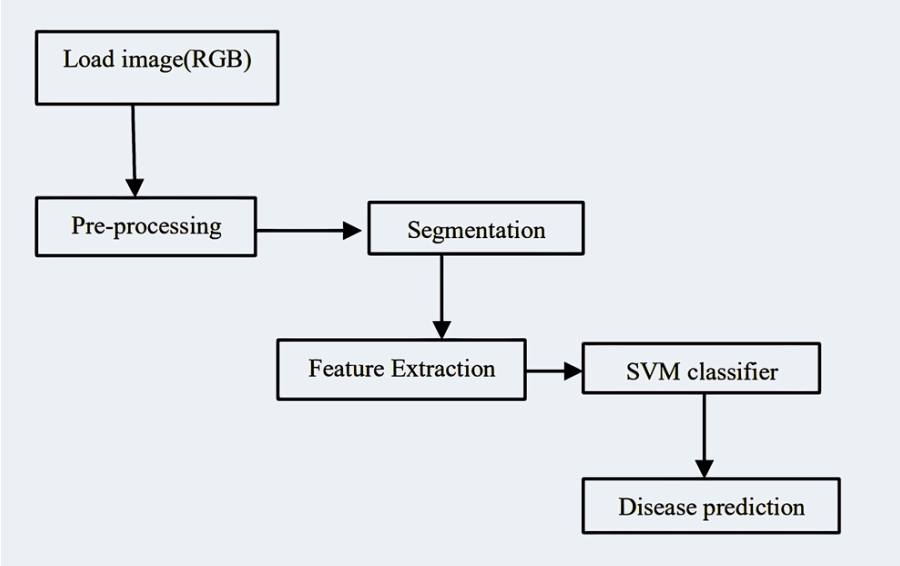


Figure 8 . System Block Diagram

Algorithm: Plant Disease Classification Using SVM

1. **Load leaf image:** Load the RGB image of the plant leaf.
2. **Contrast enhancement:** Adjust image contrast to improve accuracy.
3. **Pre-processing:** Perform pre-processing techniques to enhance image quality and remove noise.
4. **Segmentation:** Identify the affected area by segmenting the image into clusters:
   - Calculate the mean of each cluster.
   - Square the difference between two means.
   - Multiply different pixels into one cluster based on quantity.
5. **Feature extraction:** Extract features from the segmented area to identify the disease. Morphological approaches can enhance results.
6. **SVM classification:** Utilize Support Vector Machine (SVM) to classify the extracted features as healthy or infected.
7. **Disease prediction:** Predict the classified disease based on the SVM classification.

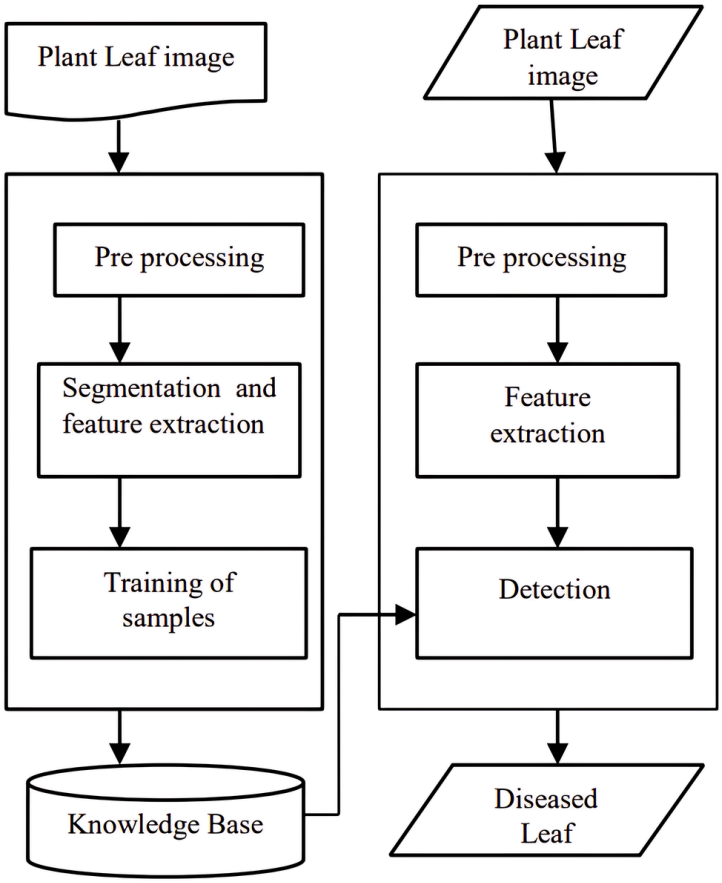


Figure 9 . System Architecture

System Overview:

The proposed disease detection system comprises two main phases: training and detection.

Training Phase:

1. **Image Acquisition:** Images of plant leaves are captured using a digital camera.
2. **Pre-processing:** The captured images undergo pre-processing to remove unwanted distortions and enhance brightness.
3. **Feature Extraction:** Feature extraction techniques are applied to the pre-processed images to extract relevant information such as color, shape, and texture.
4. **Image Database:** Pre-processed images with extracted features are stored in a database for training.

Detection Phase:

1. **Image Acquisition:** New images of plant leaves are captured using a digital camera.
2. **Pre-processing:** The newly captured images undergo the same pre-processing steps as in the training phase to ensure consistency.
3. **Segmentation:** Image segmentation is performed to separate the leaf image from the background, extracting the color of the leaf.
4. **Feature Extraction:** Features are extracted from the segmented leaf image, creating a tabular format for classification.
5. **Classification:** Using the extracted features, classification algorithms, such as SVM, classify the leaf as either healthy or infected based on the trained model.

Key Components:

1. **Image (BGR) Load:** Images of plant leaves are captured in BGR format using a digital camera.
2. **Pre-processing:** Removes undesired distortions and enhances brightness in the images.
3. **Segmentation:** Separates the leaf image from the background to focus on the affected area.
4. **Feature Extraction:** Identifies relevant features such as color, shape, and texture from the segmented leaf image.

This systematic approach ensures accurate and efficient detection of plant diseases, aiding in timely intervention to preserve crop health and yield.

4.2 Convolutional Neural Network(CNN):

Convolutional Neural Network (CNN) Overview:

- **Purpose**: CNNs are specialized deep learning models designed for image processing tasks.
- **Inspiration**: Inspired by the human visual system, they mimic how the brain processes and interprets visual data.
- **Key Components**:
  - **Convolutional Layers**: These layers apply filters to extract features like edges and textures from images.
  - **Pooling Layers**: They downsample the feature maps to reduce complexity and focus on essential information.
  - **Fully Connected Layers**: These layers make the final classifications based on the extracted features.
- **Deep Architecture**: CNNs have a deep, hierarchical structure that allows them to learn increasingly complex patterns.
- **Training**: They are trained through backpropagation, adjusting weights to minimize the difference between predicted and actual labels.
- **Applications**: CNNs excel in tasks like image classification, object detection, facial recognition, and medical image analysis.
- **Advantages**:
  - Effective in capturing spatial hierarchies in images.
  - Robust feature learning, reducing the need for manual feature engineering.
  - State-of-the-art performance in many image-related tasks.
- **Challenges**:
  - Requires large amounts of labeled data for training.
  - Computationally intensive, often requiring GPUs or TPUs.
  - Interpretability can be challenging due to complex architectures.
- **Future Directions**:
  - Continued research focuses on improving efficiency, interpretability, and generalization.
  - Integration with other AI techniques like attention mechanisms and reinforcement learning.

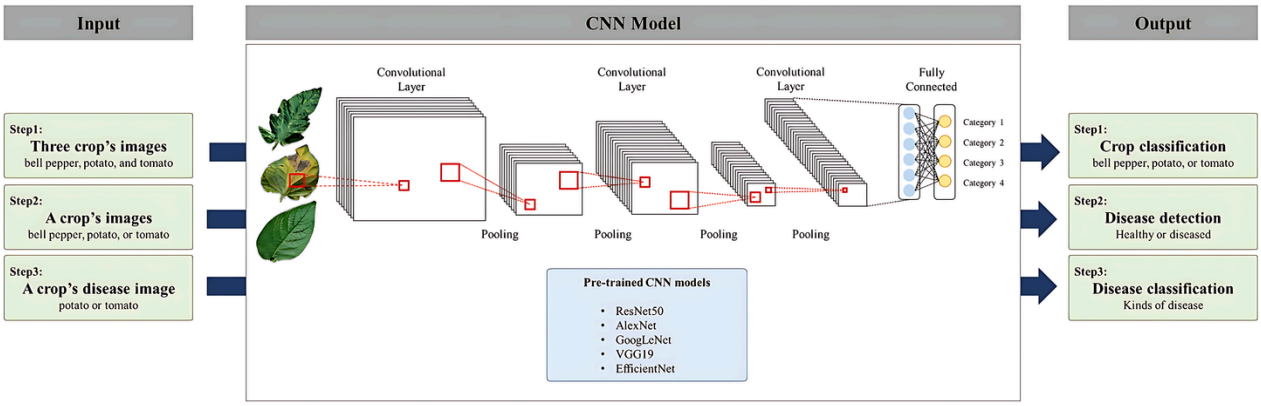


Figure 10 . CNN Architecture

Convolutional Neural Network (CNN) Architecture Overview

1. Input Layer

- **Purpose**: Receives the preprocessed input images.
- **Description**:
  - The input images are preprocessed to a fixed size of 64x64 pixels.
  - Each pixel value represents a color intensity (red, green, blue).

2. Convolutional Layers

- **Purpose**: Extracts features from the input images.
- **Conv2D with 32 Filters (Kernel Size: 5x5, Activation: ReLU)**
  - **Description**:
    - 32 filters (or kernels) are applied to the input image.
    - Each filter extracts different features (like edges, textures) from the image.
    - The ReLU (Rectified Linear Unit) activation function introduces non-linearity.
- **Conv2D with 64 Filters (Kernel Size: 5x5, Activation: ReLU)**
  - **Description**:
    - Additional convolutional layer with 64 filters for deeper feature extraction.
    - Helps the model learn more complex patterns in the images.

3. MaxPooling Layers

- **Purpose**: Reduces spatial dimensions, focuses on important features.
- **MaxPooling2D (Pool Size: 2x2)**
  - **Description**:
    - Divides the input image into smaller 2x2 regions.
    - Retains the maximum pixel value from each region, reducing image size.
    - Helps in capturing the most relevant information and reducing computation.

4. Dropout Layers

- **Purpose**: Prevents overfitting by randomly "dropping out" neurons during training.
- **Dropout with Rate 0.7**
  - **Description**:
    - Each neuron has a 70% probability of being dropped out during training.
    - Forces the network to learn more robust features and generalize better.

5. Flatten Layer

- **Purpose**: Converts 2D feature maps into a 1D vector.
- **Flatten**
  - **Description**:
    - Flattens the output from the convolutional layers into a single long vector.
    - Essential for connecting to fully connected (dense) layers.

6. Dense (Fully Connected) Layers

- **Purpose**: Makes final predictions based on extracted features.
- **Dense with 1000 Units (Activation: ReLU)**
  - **Description**:
    - Fully connected layer with 1000 units to learn complex patterns.
    - Each neuron is connected to every neuron in the previous layer.
    - Introduces higher-level abstractions in the learned features.
- **Dropout with Rate 0.7**
  - **Description**:
    - Another dropout layer applied to the dense layer to further prevent overfitting.
    - Ensures the model doesn't rely too heavily on any particular feature.

7. Output Layer

- **Purpose**: Produces final predictions in the form of class probabilities.
- **Dense with Softmax Activation (15 Units)**
  - **Description**:
    - Final layer with 15 units, corresponding to the 15 disease classes.
    - Softmax activation function converts raw scores into probabilities.
    - Each unit outputs the probability of the input image belonging to its corresponding class.
    - The class with the highest probability is the predicted class for the input image.

Summary

- **Input:** Preprocessed plant images of size 64x64 pixels.
- **Layers:** Convolutional layers for feature extraction, MaxPooling for downsampling, Dropout for regularization, Dense layers for final predictions.
- **Training:** Model trained using Adam optimizer and categorical crossentropy loss.
- **Evaluation:** Confusion matrix used to visualize performance, metrics such as accuracy, precision, recall, and F1-score calculated.

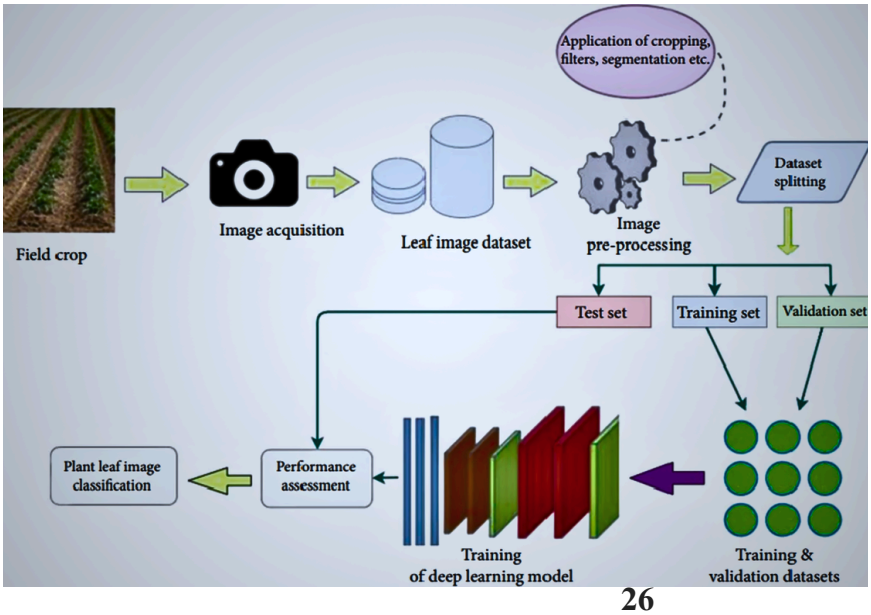


Figure 11. Computer vision-based techniques for plant disease detection and classification

4.3 Transfer Learning with Pre-trained Models:

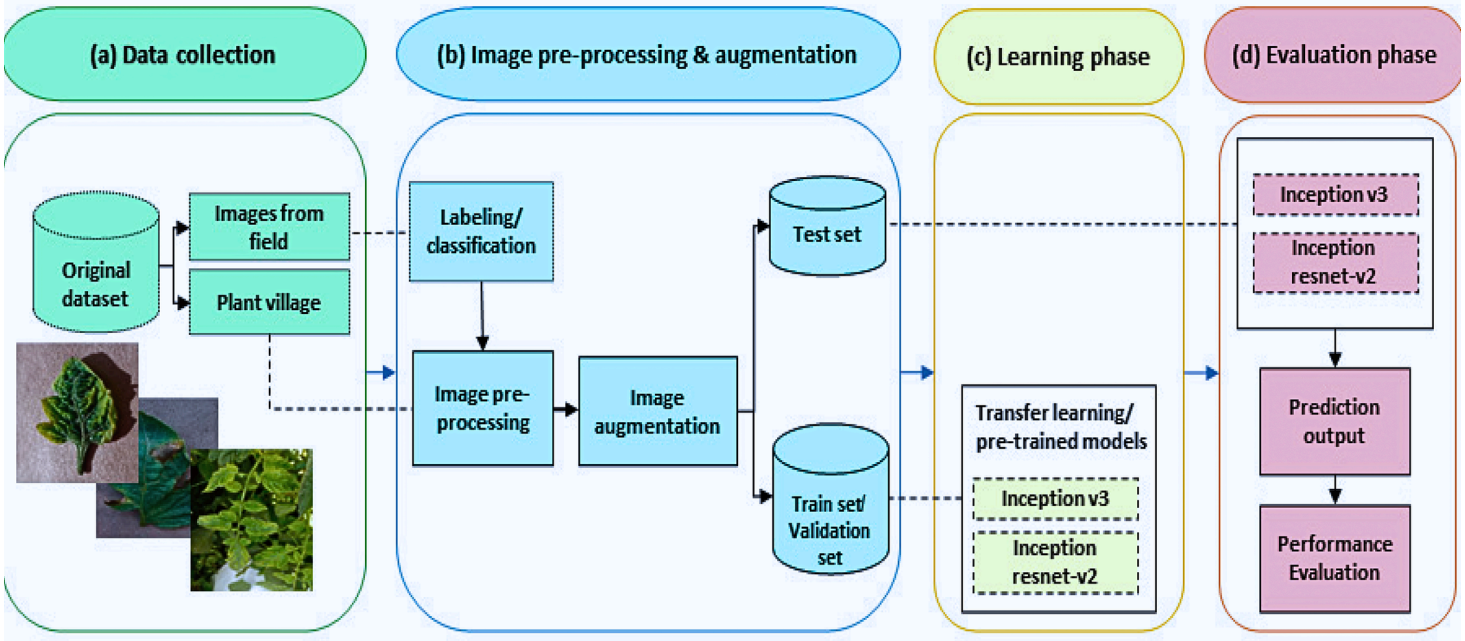


Figure 12 . Transfer Learning Architecture

4.3.1 Transfer Learning Workflow:

- **Choosing a Pre-trained Model**: The selection often depends on the task at hand. For instance, ResNet50 is suitable for complex image classification tasks, InceptionV3 excels in efficient resource utilization, while VGG-16/19 is ideal when detailed feature extraction is necessary.
- **Modifying Top Layers**: After selecting a pre-trained model, the top layers are either replaced or modified to suit the specific task. These top layers are responsible for making predictions, so they need to align with the new dataset's classes.
- **Freezing Base Layers**: The lower layers, which have already learned general features like edges and shapes, are usually "frozen" to preserve their learned representations. This ensures that the model doesn't forget the valuable knowledge it gained from the original dataset.
- **Fine-tuning (Optional)**: In some cases, particularly when the new dataset is large and similar to the original dataset, some of the frozen layers can be unfrozen and allowed to adapt to the new data. This process is called fine-tuning and can further improve performance.
- **Evaluation:** Evaluate the performance of the model on a validation or test dataset using metrics like accuracy, precision, recall, and F1-score.

Aspect	Traditional Learning (Training from Scratch)	Transfer Learning
Starting Point	Starts with no pre-existing knowledge	Starts with pre-trained model
Dataset Size	Requires large dataset for training	Can work well with smaller datasets
Training Time	Longer training time due to starting from scratch	Faster training as it builds on pre-trained model
Computational Resources	Demands more computational resources	Requires less computational power
Performance	May suffer from overfitting on smaller datasets	Often achieves better generalization
Customization	Entire model needs to be designed and trained	Last layers can be fine-tuned for specific task
Domain Knowledge	Requires deep understanding of model architecture	Less domain knowledge needed, easier to implement
Transferability	Features learned are not easily transferable	Features learned are transferable to similar tasks
Error Sensitivity	More prone to errors due to manual feature engineering	Less prone to errors as it leverages learned features
Real-world Applications	Used when specific data or tasks are unique	Widely used in various domains and tasks

Table 1 Traditional Learning vs Transfer Learning

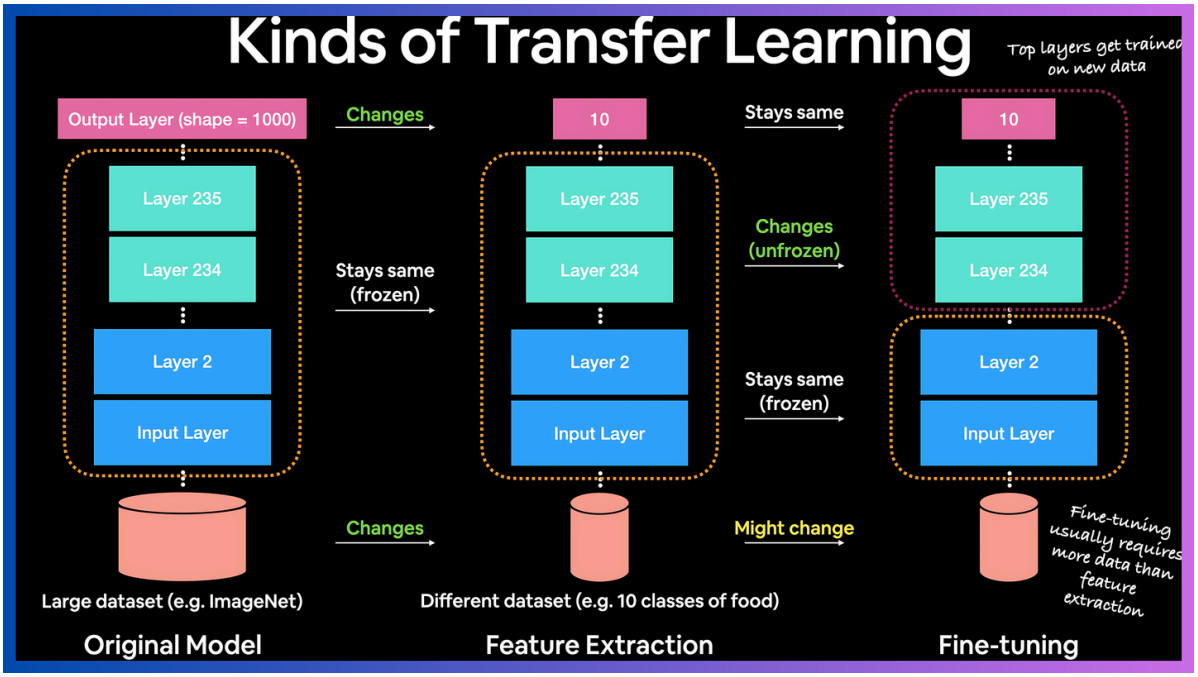


Figure 13 . Fine-Tuning VS Feature-Extraction

Applications of Transfer Learning

1. **Image Classification**:
   - Transfer learning is extensively used for image classification tasks.
   - Pre-trained models can be fine-tuned to identify objects, diseases, or anomalies in images.
   - Models trained on ImageNet can be adapted for specific classification tasks.
2. **Object Detection and Localization**:
   - Transfer learning benefits object detection tasks, where models classify and localize objects in images.
   - Pre-trained models can be fine-tuned to detect specific objects in images or videos.
   - Transfer learning aids in training models for accurate object localization.
3. **Natural Language Processing (NLP)**:
   - Transfer learning is prominent in NLP tasks, such as sentiment analysis and text classification.
   - Models like BERT and GPT are pre-trained and fine-tuned for various NLP tasks.
   - Transfer learning enables models to understand context and meaning in textual data.

Advantages of Transfer Learning

1. **Improved Training Speed**:
   a. Transfer learning significantly reduces training time by starting with pre-learned features.
   b. Models converge faster as they build on knowledge from pre-trained models.
   c. Training from scratch is avoided, saving time and resources.
2. **Better Generalization**:
   a. Pre-trained models have learned generic features from vast datasets, improving generalization.
   b. Models generalize well to new tasks or domains, even with limited labeled data.
   c. Transfer learning helps in achieving better performance on new tasks with small datasets.
3. **Effective Feature Extraction**:
   a. Transfer learning provides access to powerful feature extractors learned from massive datasets.
   b. Models like VGG, ResNet, and BERT have learned hierarchical features, beneficial for various tasks.
   c. Reusing pre-learned features improves model performance and accuracy.
4. **Reduced Overfitting:**
   a. Fine-tuning pre-trained models allows for better regularization.
   b. Models are less likely to overfit to small datasets as they adapt to the new task.

4.3.2 Inception-V3 :

InceptionV3 is a deep convolutional neural network architecture designed by **Google** Research for computer vision tasks. It was introduced in the paper "**Rethinking the Inception Architecture for Computer Vision**" by Christian Szegedy et al. in 2015. InceptionV3 is designed to achieve high accuracy on the ImageNet dataset while maintaining computational efficiency.

Key Concepts

- **Inception Modules**: InceptionV3 employs Inception modules, which are building blocks with parallel convolutional branches of different filter sizes. This allows the network to capture spatial hierarchies of features at multiple scales.
- **Factorization**: The model utilizes factorization into smaller convolutions (1x1, 3x3, 5x5) within the Inception modules. This reduces the number of parameters and computational load.
- **Global Average Pooling**: Similar to other architectures, InceptionV3 ends with a global average pooling layer. This computes the average of each feature map in the last convolutional layer.
- **Auxiliary Classifiers**: During training, InceptionV3 includes auxiliary classifiers at intermediate layers. These aid in regularization and combatting the vanishing gradient problem.

Data Preparation

- **Data Generators**: We used TensorFlow's **ImageDataGenerator** for data augmentation and loading images in batches.

Training

- **Pre-trained Weights**: We loaded the pre-trained weights of InceptionV3 from a local file **inception_v3_weights.h5**.
- **Freezing Layers**: All layers of the pre-trained InceptionV3 model were frozen to retain the learned features and prevent their modification during initial training.
- **Custom Layers**: Additional layers were added on top of the pre-trained model:
  - Flatten Layer
  - Dense Layer with 2048 units and ReLU activation
  - Dense Layer with 1024 units and ReLU activation
  - Dropout Layer with 20% dropout rate
  - Output Dense Layer with softmax activation for class probabilities.

Model Compilation and Training

- The model is compiled with:
  - Optimizer: RMSprop with a learning rate of 0.0001
  - Loss Function: Categorical Crossentropy
  - Metrics: Accuracy
- The model is trained for 10 epochs using the training and validation data.

Evaluation

- The trained model is evaluated on the test set using **evaluate_generator**.
- Evaluation metrics such as loss and accuracy are printed and visualized with plots.

Fine-Tuning:

- The last few layers of InceptionV3 are unfrozen for fine-tuning.
- The model is re-compiled and trained again with the fine-tuning dataset.

Further Evaluation:

- After fine-tuning, the model is evaluated again on the test set.
- Confusion matrix, classification report, and additional metrics are computed and visualized.

Advantages :

1. **Efficient Parameter Utilization**:
   - Utilizes "network-in-network" design.
   - Employs 1x1, 3x3, and 5x5 convolutions in parallel.
   - Reduces parameters, enhancing computational efficiency.
2. **Multi-Scale Feature Learning**:
   - Includes layers with varying receptive field sizes.
   - Learns features at different scales for better object recognition.
3. **State-of-the-Art Performance**:
   - Achieves top performance on datasets like ImageNet.
   - Superior accuracy for image classification, detection, and segmentation.
4. **Transfer Learning Capability**:
   - Pre-trained on ImageNet, ideal for transfer learning.
   - Allows fine-tuning on specific datasets, reducing training time.

These advantages collectively position InceptionV3 as a powerful and versatile deep learning architecture, capable of addressing the challenges of modern **computer vision applications**. Its efficient parameter usage, multi-scale feature learning, state-of-the-art performance, and transfer learning capabilities make it a preferred choice for a wide range of image-related tasks in both research and practical applications.

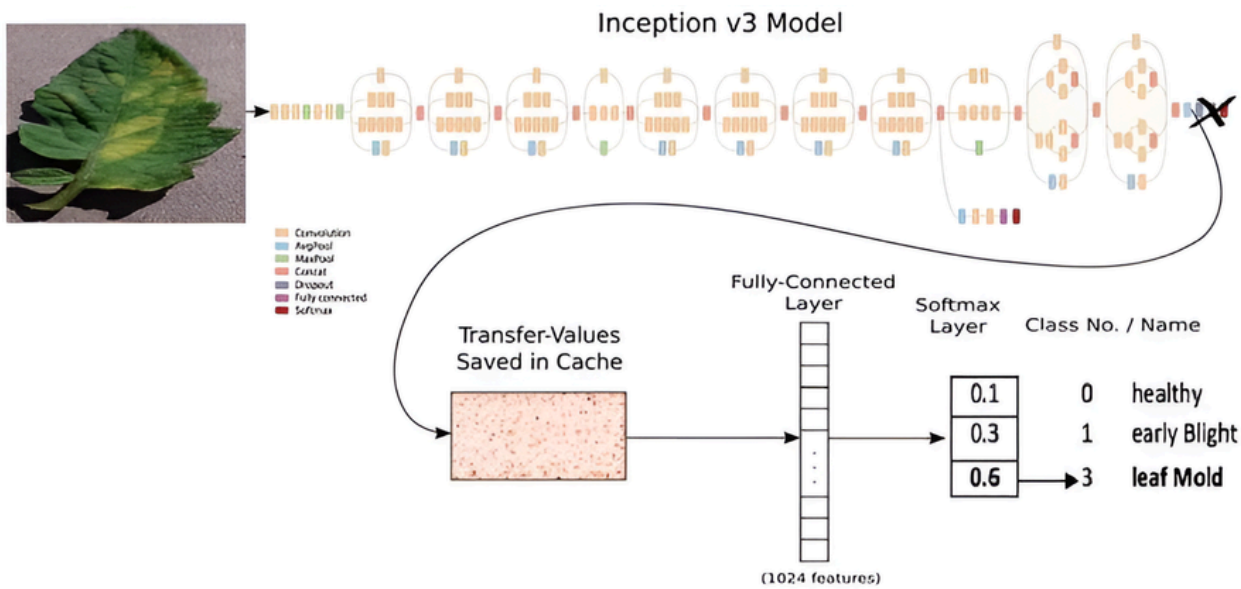


Figure 14 . Inception V3 Model

4.3.3 ResNet-50 :

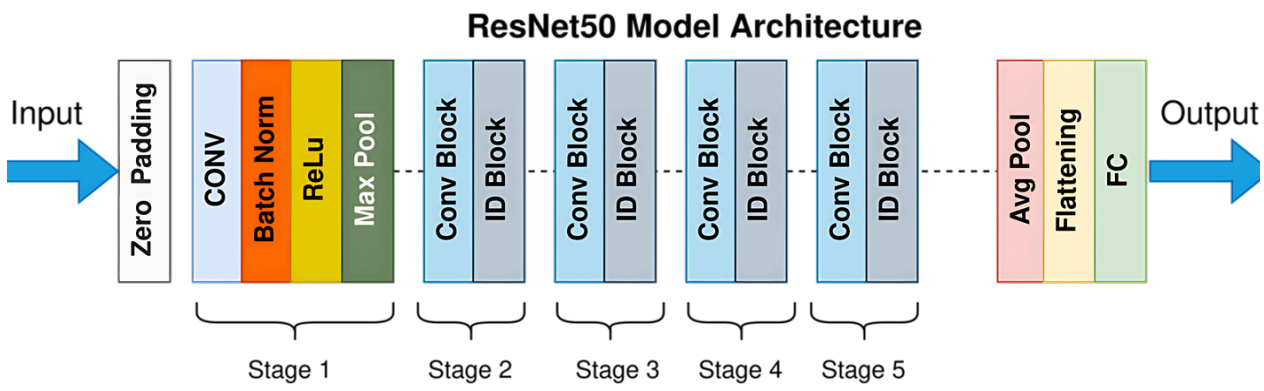


Figure 15 . ResNet-50 Model

ResNet-50 Architecture:

ResNet-50, short for Residual Network with **50 layers**, is a deep convolutional neural network architecture that was introduced by **Kaiming He et al**. from Microsoft Research in their paper "Deep Residual Learning for Image Recognition" in 2015. ResNet-50 is a variant of the original ResNet, designed to address the problem of vanishing gradients in very deep neural networks. It has become a widely used and highly effective architecture for various computer vision tasks, including image classification, object detection, and segmentation.

Key Concepts:

- **Residual Learning**: The core innovation of ResNet-50 is the introduction of residual learning blocks, or "shortcut connections." These connections enable the network to learn residual functions, which are the differences between the desired output and the actual output of a block of layers.
- **Deep Network Training**: The problem with training very deep neural networks is the vanishing gradient problem, where gradients diminish as they backpropagate through many layers, making training difficult. Residual connections mitigate this by allowing gradients to flow more directly through the network, enabling the training of very deep models.

Architecture Details:

- **Input**: Images are typically resized to a standard size, often 224×224 pixels with 3 color channels (RGB).
- **Building Blocks**: ResNet-50 is composed of several building blocks, including:
  - **Convolutional Layers**: Used for feature extraction.
  - **Batch Normalization**: Helps with faster convergence and regularization.
  - **Activation Functions**: Typically ReLU (Rectified Linear Unit).
  - **Max Pooling**: Downsamples feature maps.
  - **Residual Blocks**: The core building blocks with shortcut connections. ResNet-50 has 16 of these blocks, divided into 5 stages.
- **Stages**:
  - **Stage 1**: Initial Convolutional and Pooling layers.
  - **Stages 2-5**: Each stage contains multiple residual blocks with increasing numbers of filters.

- **Global Average Pooling**: Instead of using fully connected layers at the end, ResNet-50 ends with a global average pooling layer. This layer computes the average of each feature map in the last convolutional layer.
- **Output**: The final layer is a fully connected softmax layer that produces the output probabilities for each class in the classification task.

Layers:

- Input Layer: Images resized to (75, 75, 3).
- Pre-trained ResNet-50 Layers: Initially non-trainable to leverage pre-trained weights from ImageNet.
- Custom Layers: Added for classification.
  - Flatten Layer
  - Dense Layer with ReLU activation (1000 units)
  - Output Layer with Softmax activation (15 units for 15 classes)

Training:

- **Transfer Learning**:
  - Loaded pre-trained ResNet-50.
  - Marked initial layers as non-trainable.
  - Added custom layers.
  - Compiled the model with Adam optimizer and categorical cross-entropy loss.
- **Fine-Tuning**:
  - Loaded pre-trained ResNet-50.
  - Fine-tuned from a specific layer.
  - Added custom layers.
  - Compiled the model with Adam optimizer and categorical cross-entropy loss.
- **Training Details**:
  - Batch Size: 32
  - Epochs: 50 for transfer learning, 10 for fine-tuning.
  - Early Stopping: Used with a patience of 3 for both.

Advantages of ResNet-50:

- **Deeper Networks**: ResNet-50 was one of the first models to demonstrate the effectiveness of very deep networks, reaching up to 50 layers while maintaining good performance.
- **Better Gradient Flow**: The skip connections or shortcut connections ensure that gradients can flow directly through the network, making training easier.
- **State-of-the-Art Performance**: ResNet-50 and its variants have achieved state-of-the-art performance on various image classification benchmarks, including the ImageNet dataset.

4.3.4 VGG-16 :

The VGG-16 (Visual Geometry Group) model is a deep convolutional neural network architecture that was developed by the Visual Geometry Group at the University of Oxford. It is widely recognized for its simplicity and effectiveness, particularly in image classification tasks. In this case, we have utilized the VGG-16 model with pre-trained weights on the ImageNet dataset to perform plant disease classification.

Model Architecture

The VGG-16 model consists of 16 weight layers, including 13 convolutional layers and 3 fully connected layers. The architecture can be divided into two main parts:

1. **Base Model (VGG16)**:
   - The base VGG-16 model is instantiated with pre-trained ImageNet weights.
   - This pre-trained model serves as a feature extractor, capturing hierarchical features from input images.
   - The pre-trained model is frozen, meaning the weights are not updated during training.
2. **Top Layers** (Customized for Plant Disease Classification):
   - On top of the base model, we added custom fully connected layers for classification.
   - The architecture of the top layers is as follows:
     - Flatten Layer: Converts the 3D feature maps into a 1D feature vector.
     - Dense Layer (2048 units): Fully connected layer with ReLU activation, He normal initialization, and 35% dropout.
     - Dense Layer (2048 units): Another fully connected layer with ReLU activation, He normal initialization, and 35% dropout.
     - Dense Layer (15 units): Final fully connected layer with Softmax activation for 15 disease classes, initialized with Glorot normal weights.

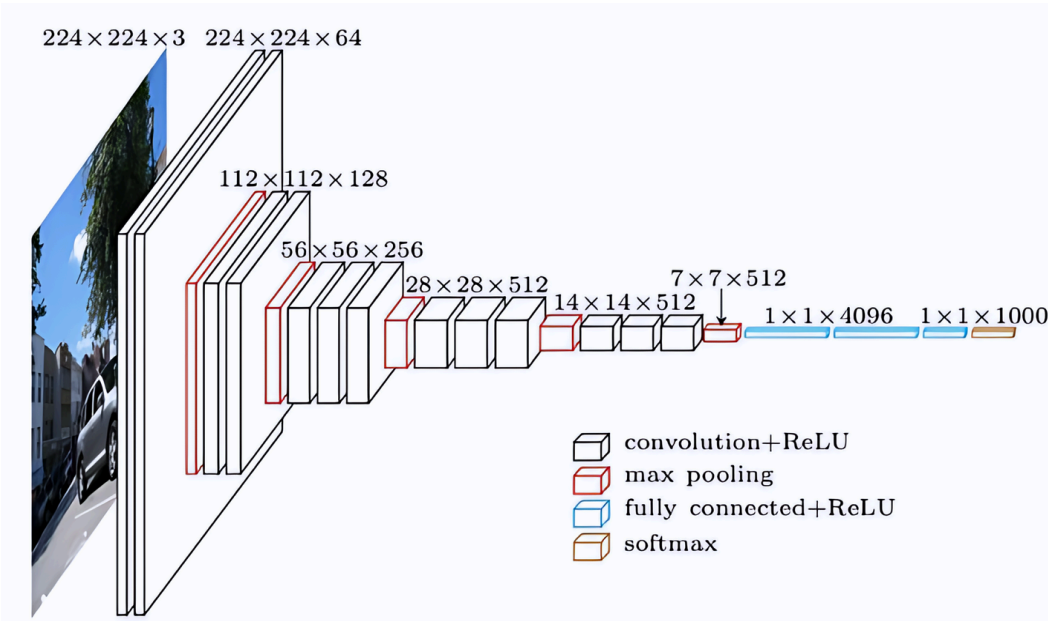


Figure 16 . VGG-16 Model

Training and Evaluation

- **Training Dataset**: The model is trained on a training dataset consisting of images of diseased plants. Data augmentation techniques such as shearing, zooming, and horizontal flipping are applied to increase the diversity of training samples.
- **Validation Dataset**: A separate validation dataset is used to monitor the model's performance during training and prevent overfitting.
- **Testing Dataset**: The model's final performance is evaluated on an unseen testing dataset, ensuring an unbiased assessment of its generalization ability.

Fine-Tuning

- **Freezing and Unfreezing Layers**:
  - Initially, the base VGG-16 layers are frozen, meaning their weights are not updated during training.
  - For fine-tuning, we selectively unfreeze the last two blocks (block4 and block5) of the base model.
  - The unfrozen layers allow the model to learn more specific features related to plant diseases from the training dataset.
- **Learning Rate**:
  - During fine-tuning, a lower learning rate is used to ensure that the model adapts slowly and avoids catastrophic forgetting of previously learned features.

Visualizations

- **Training and Validation Curves**:
  - Plots of training and validation accuracy/loss over epochs provide insights into the model's learning progress and potential overfitting.

Advantages

1. **Pre-Trained Weights**: By using pre-trained weights from ImageNet, the model benefits from learned features like edges, textures, and shapes. This allows the model to perform better on tasks even with limited training data.
2. **Transfer Learning**: VGG-16 supports transfer learning, enabling us to fine-tune the model on a smaller dataset specific to plant diseases. This is especially useful when working with limited labeled data as it leverages knowledge from a larger, diverse dataset.
3. **Depth and Capacity**: With 16 weight layers, including multiple convolutional layers, the model has a deeper architecture compared to simpler models. This depth allows it to learn more complex and abstract features, potentially improving its ability to discriminate between different types of diseases.
4. **State-of-the-Art Performance**: VGG-16 has shown strong performance in various image classification challenges and benchmarks. Its architecture has been a basis for many subsequent models, demonstrating its effectiveness.

4.3.5 VGG-19 :

VGG-19, a variant of the VGG (Visual Geometry Group) model, is utilized for classifying plant diseases using images. This model employs transfer learning with pre-trained weights from ImageNet. The dataset consists of images of plant leaves affected by various diseases. The workflow involves data preprocessing, model creation, training, and evaluation.

Model Architecture :

1. **Input Layer**:
   - The model takes an input image of size 224x224 pixels with 3 color channels (RGB).
2. **Convolutional Blocks**:
   - VGG-19 consists of 16 convolutional layers, followed by 3 fully connected layers.
   - Each convolutional block comprises 2 or 3 convolutional layers with a 3x3 kernel size and a stride of 1.
   - The convolutional layers are followed by ReLU activation functions.
3. **Pooling Layers**:
   - After each set of convolutional layers, there are max-pooling layers with a 2x2 window and a stride of 2.
   - These pooling layers reduce spatial dimensions while preserving important features.
4. **Fully Connected Layers**:
   - The final layers of VGG-19 include 3 fully connected layers.
   - The first two dense layers have 4096 units each with ReLU activation.
   - The last dense layer has 1000 units with softmax activation, suitable for classifying 1000 categories in ImageNet.

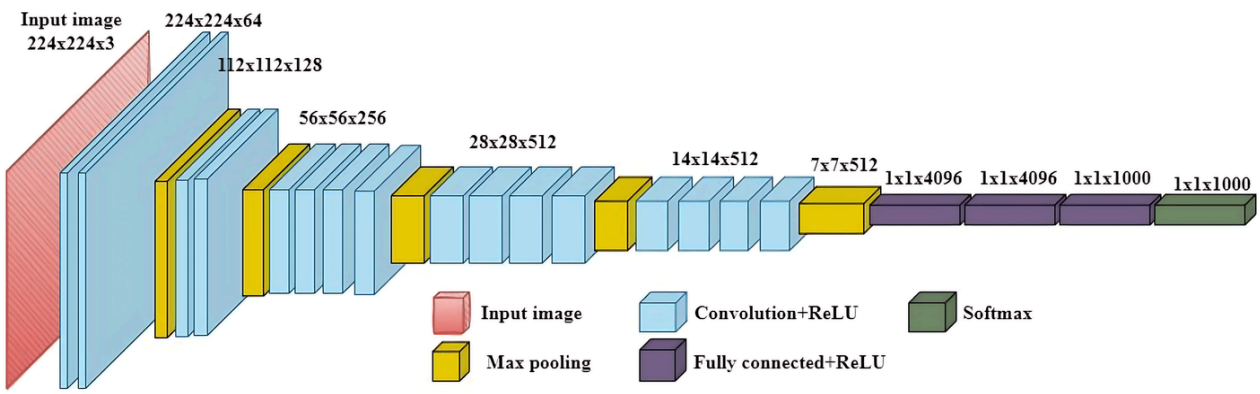


Figure 17 . VGG-19 Model

Workflow

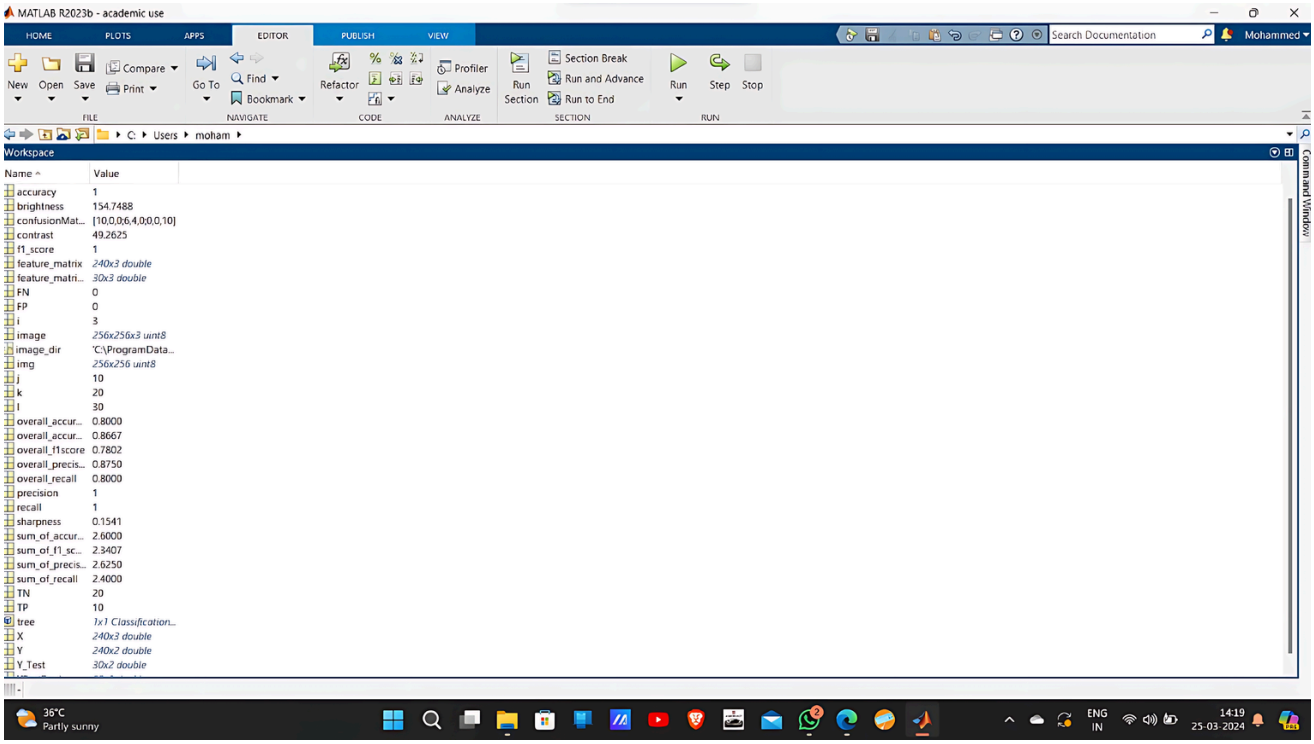
1. **Data Preprocessing**
   - The images are preprocessed using VGG-19 specific preprocessing functions.
   - Generators are created for training, validation, and testing datasets.
   - Training data undergoes augmentation with shear, zoom, and horizontal flip.
2. **Model Building**
   - VGG-19 is instantiated with pre-trained weights from ImageNet.
   - The base model is frozen to retain its learned features.
   - A new model is created by adding fully connected layers on top of the base model.
   - The model architecture includes dense layers with ReLU activation and dropout for regularization.
3. **Model Training**
   - The model is compiled with an Adam optimizer and categorical cross-entropy loss.
   - Training is performed on the training dataset for 20 epochs.
   - Validation data is used to monitor the model's performance.
4. **Model Evaluation**
   - Accuracy and loss curves are plotted to visualize model performance.
   - The model is evaluated on the test dataset to assess its generalization.
   - A confusion matrix is generated to understand classification results.
5. **Fine-tuning**
   - Last two blocks of the base VGG-19 model are unfrozen for fine-tuning.
   - The model is recompiled with a lower learning rate for fine-tuning.
   - Additional training is performed to fine-tune the model on the dataset.
6. **Final Evaluation**
   - The model is evaluated again on the test dataset post fine-tuning.
   - Metrics such as accuracy, precision, recall, and F1-score are calculated.

Advantages

1. **Deep Architecture**: VGG-19 is a deep convolutional neural network with 19 layers, allowing it to learn complex patterns and features from images effectively.
2. **Transfer Learning**: The use of pre-trained weights from ImageNet enables rapid learning and adaptation to new datasets. This reduces the need for extensive data and training time.
3. **Highly Effective Feature Extraction**: VGG-19's architecture, with repeated convolutional layers, enables it to capture intricate details and textures from images, enhancing its classification accuracy.
4. **Generalization**: The model's ability to generalize well on unseen data is crucial for real-world applications. VGG-19, with its robust feature learning, demonstrates good generalization capabilities.

5. Results

5.1 Support Vector Machine



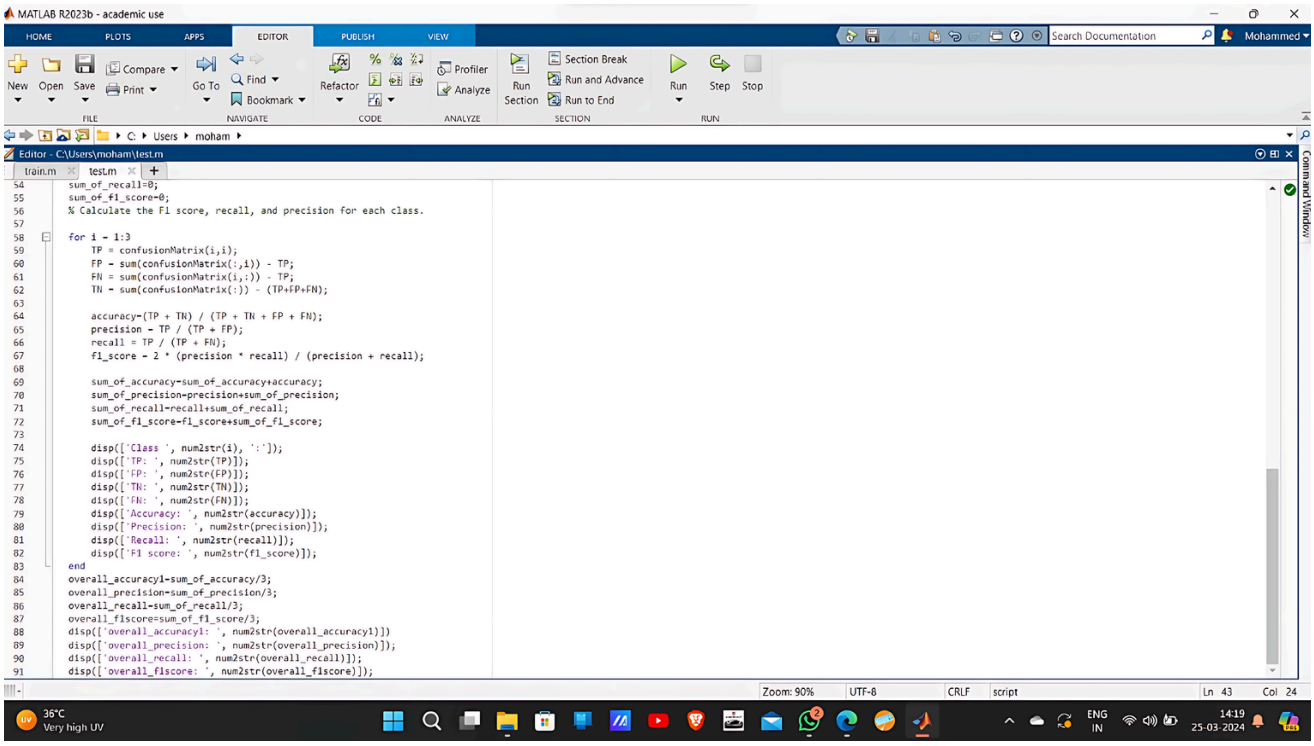
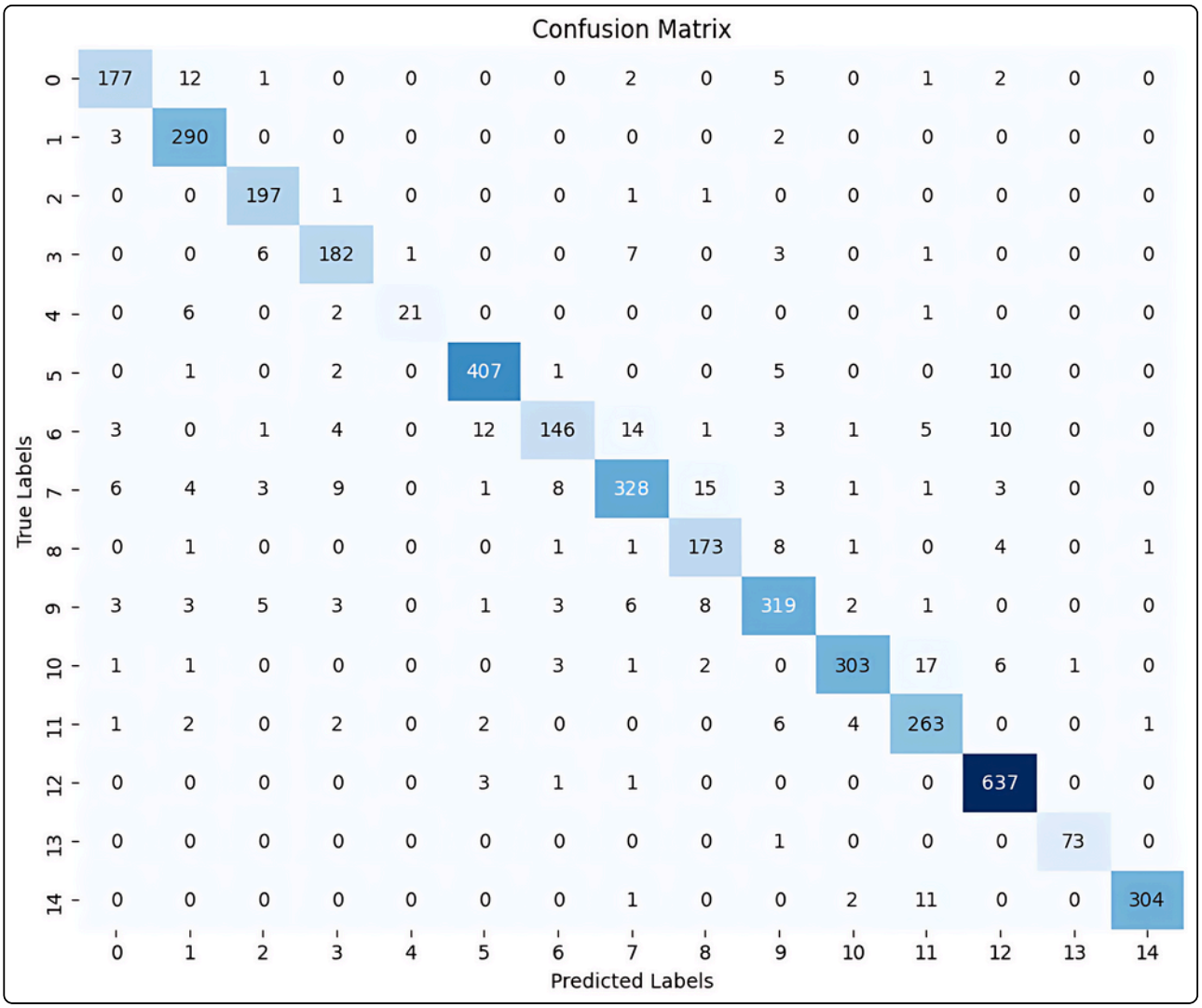
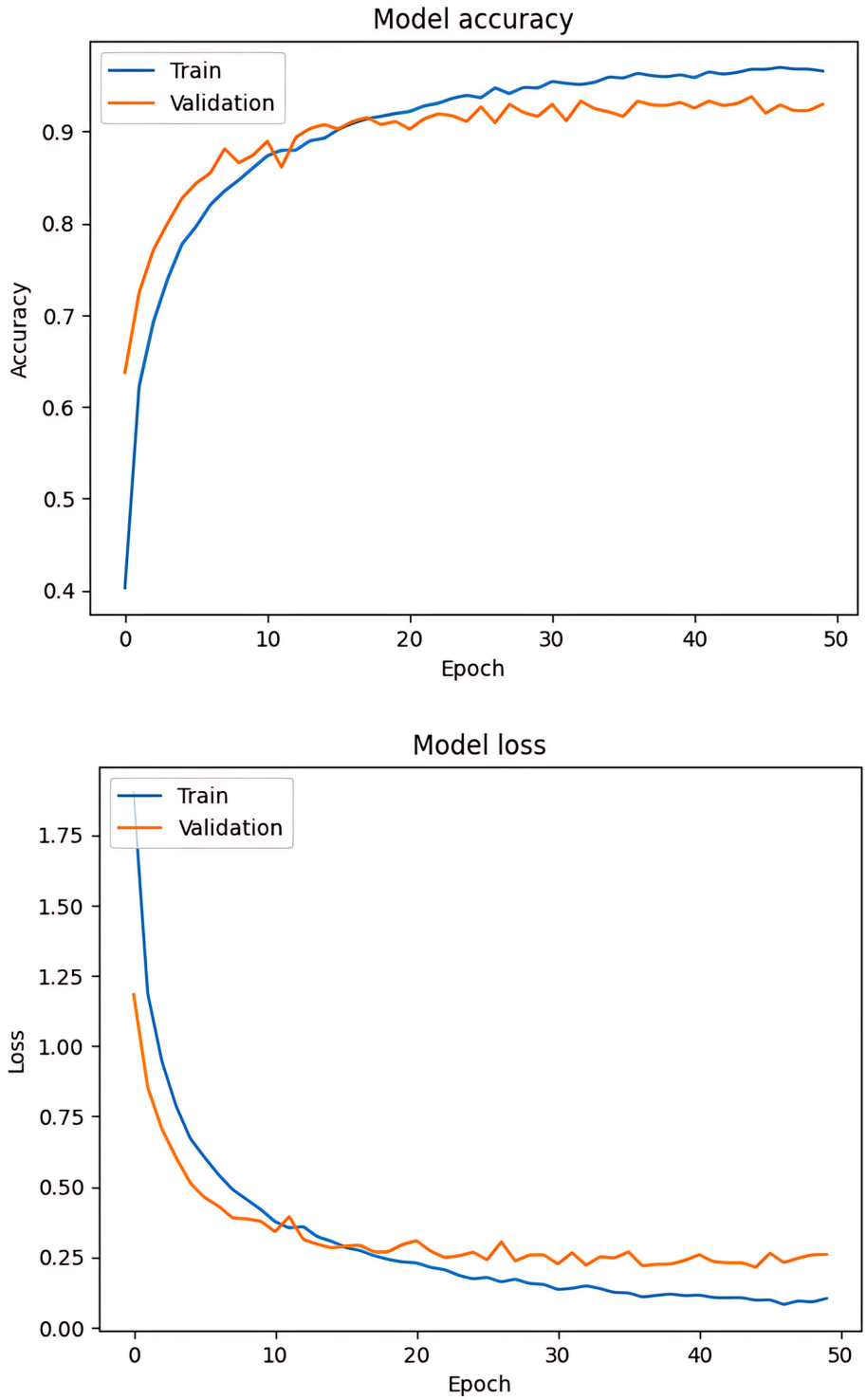


Figure 18,19 SVM Results

5.2 CONVOLUTIONAL NEURAL NETWORK



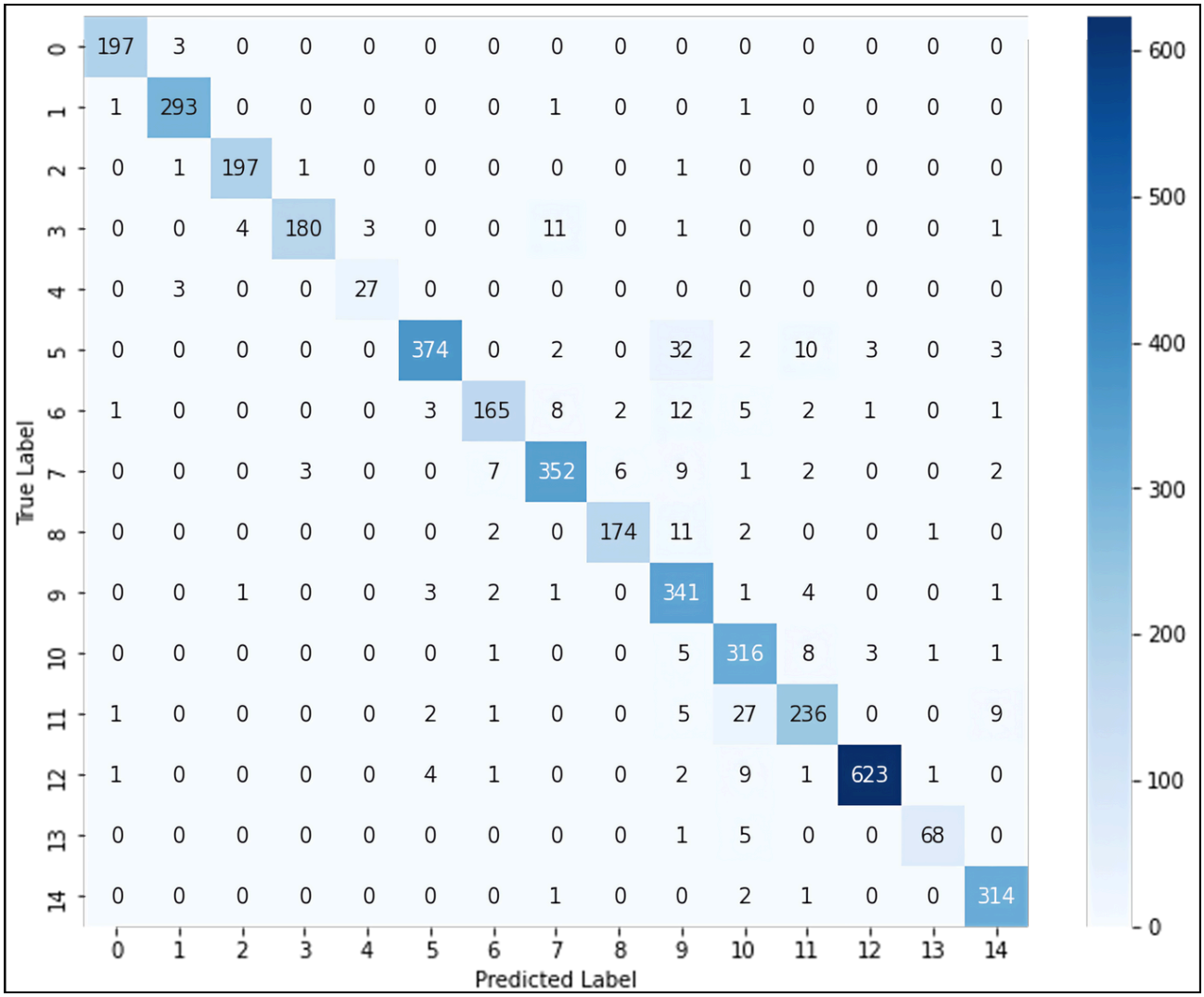

```
Epoch 46/50
181/181 [==============================] - ETA: 0s - loss: 0.0793 - accuracy: 0.9731
Epoch 46: val_loss did not improve from 0.21208
181/181 [==============================] - 78s 432ms/step - loss: 0.0793 - accuracy: 0.9731 - val_loss: 0.2194 - val_accuracy: 0.
9301
Epoch 47/50
181/181 [==============================] - ETA: 0s - loss: 0.0818 - accuracy: 0.9735
Epoch 47: val_loss did not improve from 0.21208
181/181 [==============================] - 79s 436ms/step - loss: 0.0818 - accuracy: 0.9735 - val_loss: 0.2348 - val_accuracy: 0.
9231
Epoch 48/50
181/181 [==============================] - ETA: 0s - loss: 0.0921 - accuracy: 0.9688
Epoch 48: val_loss did not improve from 0.21208
181/181 [==============================] - 78s 433ms/step - loss: 0.0921 - accuracy: 0.9688 - val_loss: 0.2249 - val_accuracy: 0.
9342
Epoch 49/50
181/181 [==============================] - ETA: 0s - loss: 0.0677 - accuracy: 0.9773
Epoch 49: val_loss did not improve from 0.21208
181/181 [==============================] - 78s 433ms/step - loss: 0.0677 - accuracy: 0.9773 - val_loss: 0.2518 - val_accuracy: 0.
9294
Epoch 50/50
181/181 [==============================] - ETA: 0s - loss: 0.0826 - accuracy: 0.9713
Epoch 50: val_loss did not improve from 0.21208
181/181 [==============================] - 82s 453ms/step - loss: 0.0826 - accuracy: 0.9713 - val_loss: 0.2184 - val_accuracy: 0.
9301
Traing finished.
Loading the best model...
Best Model loaded!
```



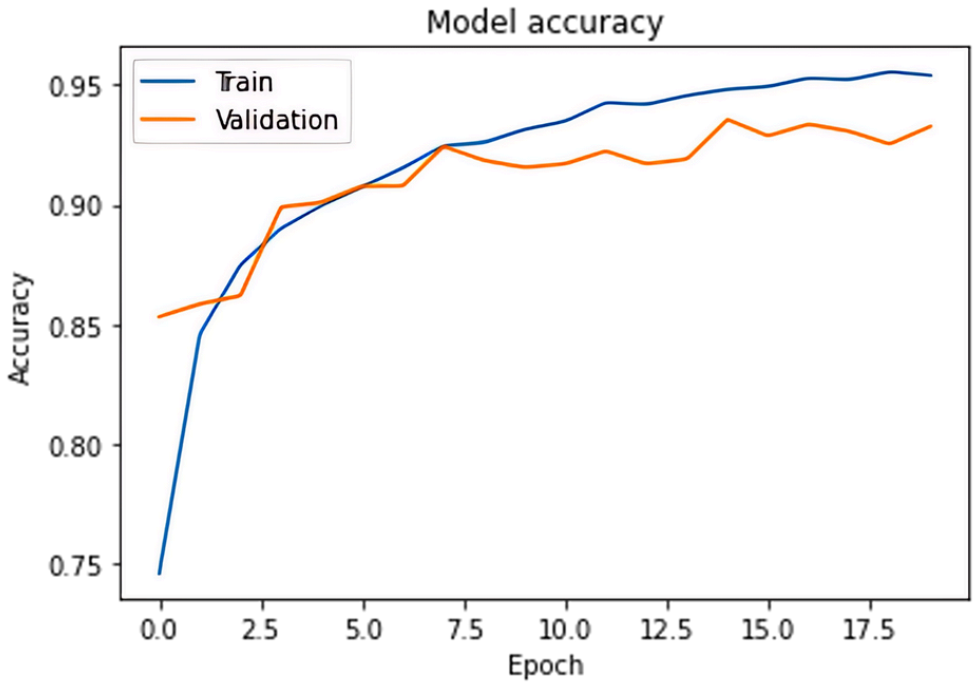

```
Accuracy: 0.9256118245699055
Precision: 0.9248811391147373
Recall: 0.9062559167291212
F1-score: 0.9132342323116737
```

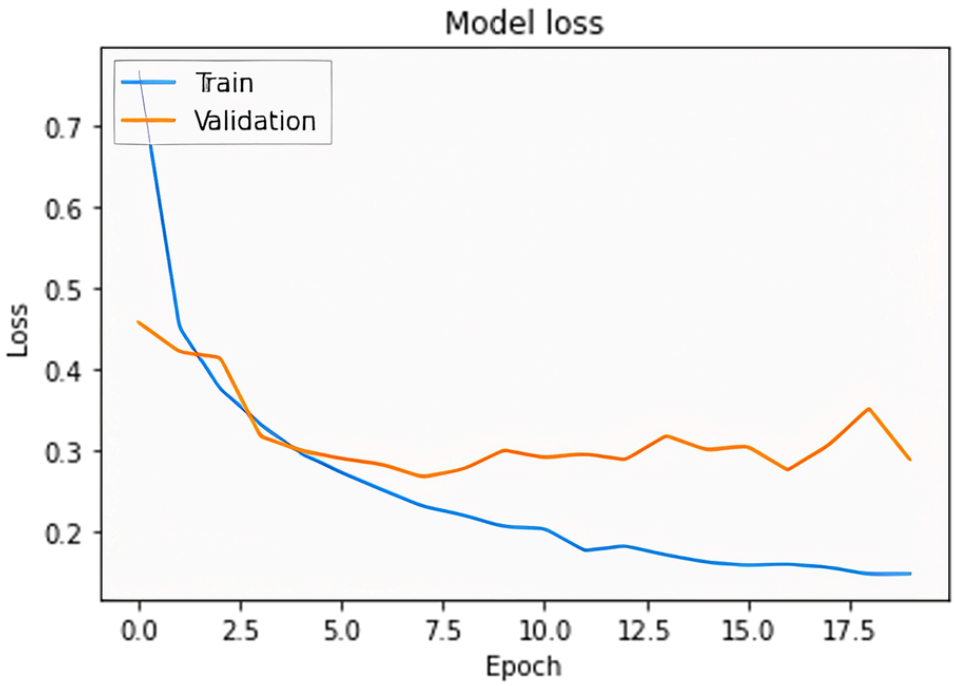
Figure 20-24 . CNN Results

5.3 Inception-V3




```
Epoch 1/10
451/451 [==============================] - 129s 273ms/step - loss: 0.4643 - accuracy: 0.8952 -
val_loss: 0.4302 - val_accuracy: 0.9067
Epoch 2/10
451/451 [==============================] - 120s 267ms/step - loss: 0.2494 - accuracy: 0.9237 -
val_loss: 0.3493 - val_accuracy: 0.9268
Epoch 3/10
451/451 [==============================] - 118s 262ms/step - loss: 0.2110 - accuracy: 0.9350 -
val_loss: 0.3237 - val_accuracy: 0.9282
Epoch 4/10
451/451 [==============================] - 122s 271ms/step - loss: 0.1988 - accuracy: 0.9427 -
val_loss: 0.3356 - val_accuracy: 0.9292
Epoch 5/10
451/451 [==============================] - 120s 266ms/step - loss: 0.1729 - accuracy: 0.9486 -
val_loss: 0.3626 - val_accuracy: 0.9272
Epoch 6/10
451/451 [==============================] - 123s 272ms/step - loss: 0.1645 - accuracy: 0.9497 -
val_loss: 0.3071 - val_accuracy: 0.9409
Epoch 7/10
451/451 [==============================] - 121s 268ms/step - loss: 0.1541 - accuracy: 0.9544 -
val_loss: 0.3639 - val_accuracy: 0.9375
Epoch 8/10
451/451 [==============================] - 120s 266ms/step - loss: 0.1554 - accuracy: 0.9537 -
val_loss: 0.3851 - val_accuracy: 0.9346
Epoch 9/10
451/451 [==============================] - 122s 271ms/step - loss: 0.1558 - accuracy: 0.9548 -
val_loss: 0.3304 - val_accuracy: 0.9409
Epoch 10/10
451/451 [==============================] - 118s 262ms/step - loss: 0.1362 - accuracy: 0.9599 -
val_loss: 0.3835 - val_accuracy: 0.9375
```

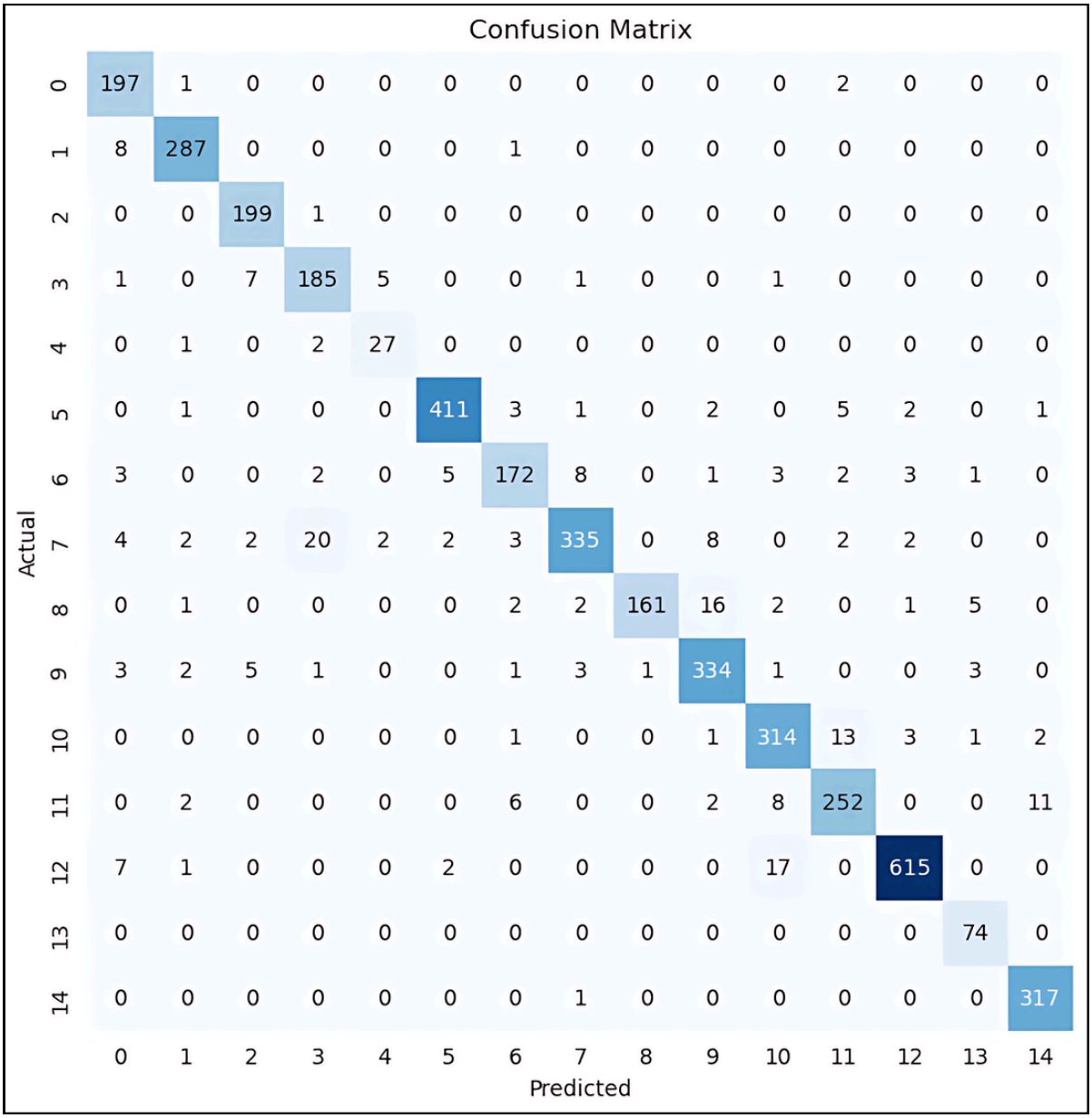




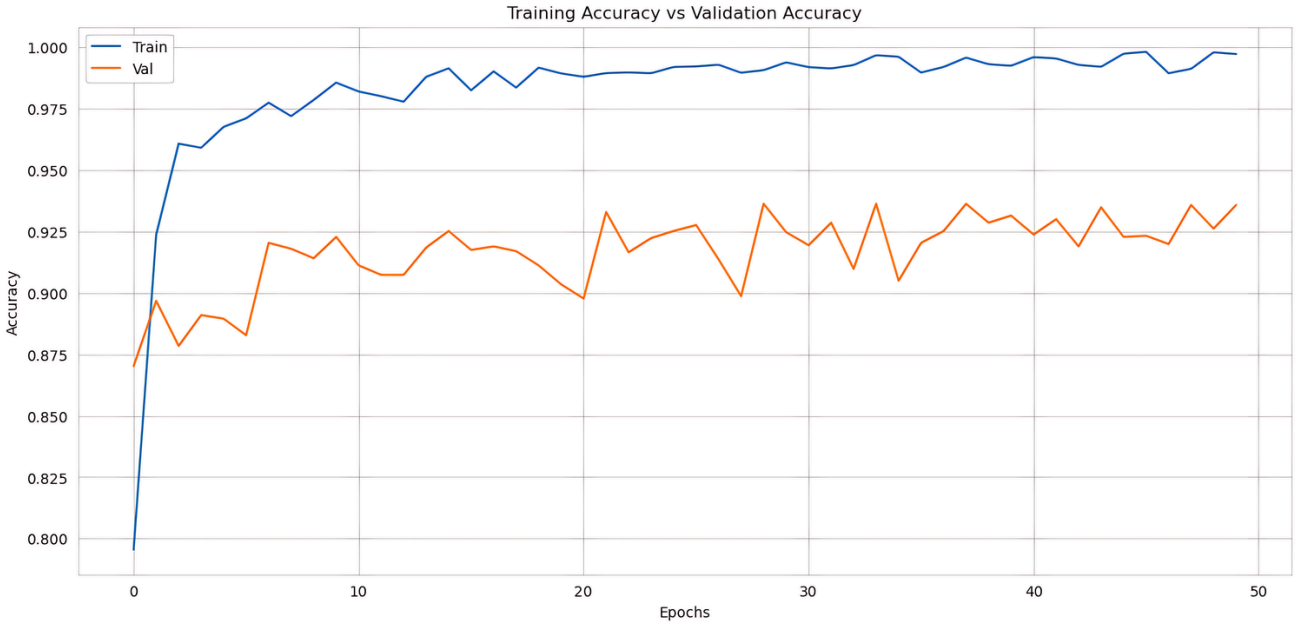

```
Accuracy: 0.9343507751937985
Precision: 0.9374131368908795
Recall: 0.9343507751937985
F1-score: 0.9345562835766121
```

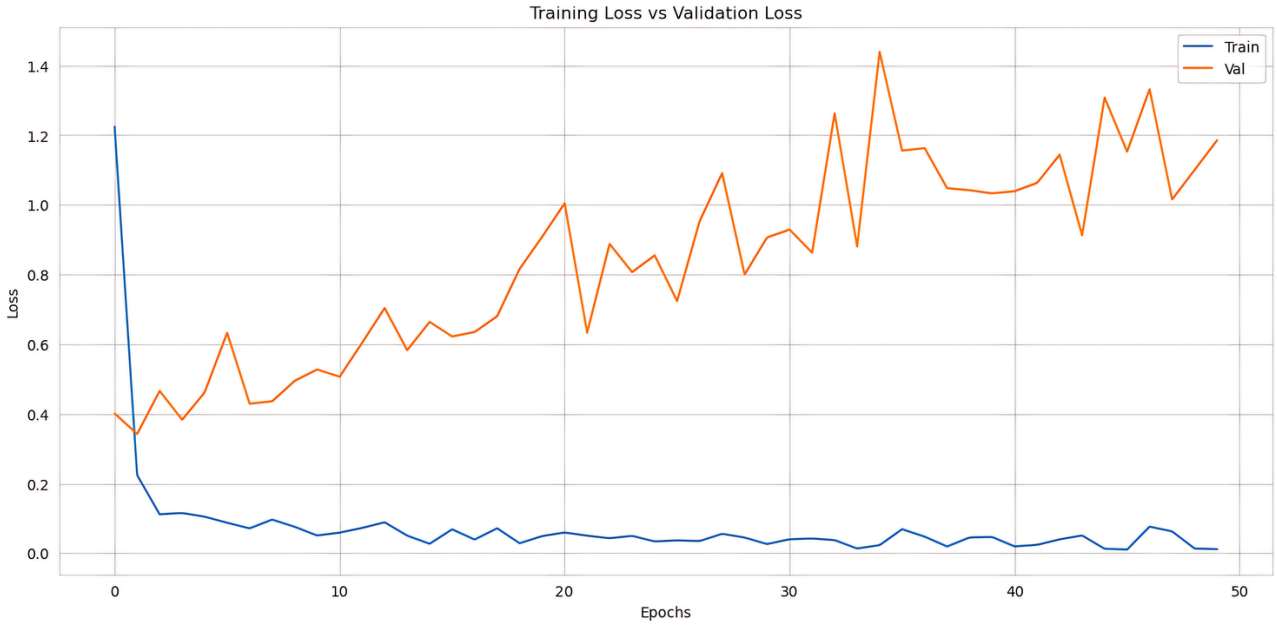
Figure 25-29 Inception-V3 Results

5.4 ResNet-50




```
Epoch 1/10
452/452 [==============================] - 50s 66ms/step - loss: 0.8035 - accuracy: 0.8476 - va
l_loss: 0.4733 - val_accuracy: 0.8794
Epoch 2/10
452/452 [==============================] - 29s 64ms/step - loss: 0.1403 - accuracy: 0.9546 - va
l_loss: 0.2705 - val_accuracy: 0.9291
Epoch 3/10
452/452 [==============================] - 28s 63ms/step - loss: 0.2480 - accuracy: 0.9560 - va
l_loss: 0.5926 - val_accuracy: 0.9252
Epoch 4/10
452/452 [==============================] - 27s 60ms/step - loss: 0.0848 - accuracy: 0.9761 - va
l_loss: 0.1714 - val_accuracy: 0.9600
Epoch 5/10
452/452 [==============================] - 28s 63ms/step - loss: 0.0893 - accuracy: 0.9781 - va
l_loss: 0.2404 - val_accuracy: 0.9411
Epoch 6/10
452/452 [==============================] - 28s 62ms/step - loss: 0.0697 - accuracy: 0.9812 - va
l_loss: 0.2313 - val_accuracy: 0.9503
Epoch 7/10
452/452 [==============================] - 28s 63ms/step - loss: 0.0536 - accuracy: 0.9875 - va
l_loss: 0.2088 - val_accuracy: 0.9542
```





Accuracy: 0.939922481

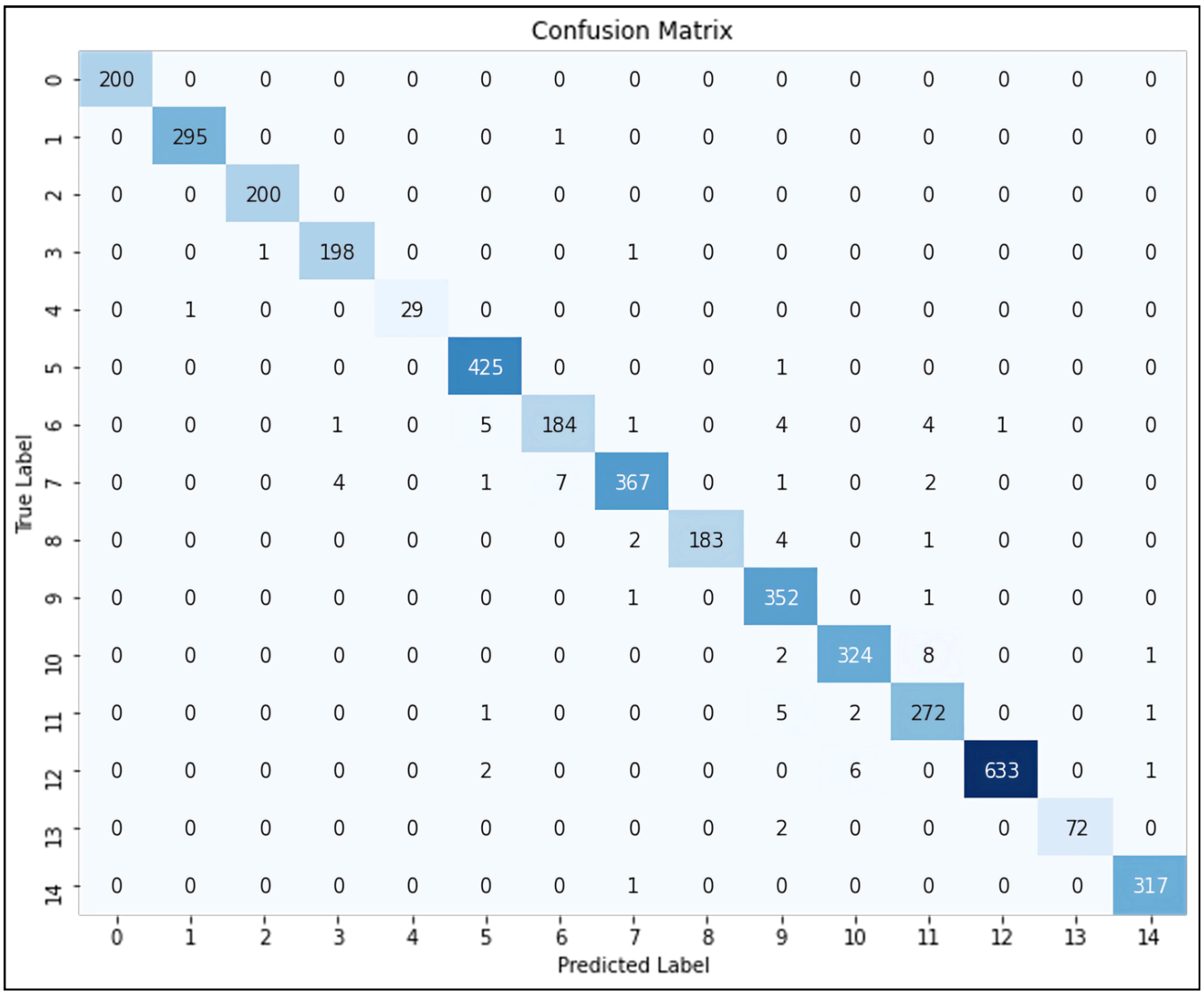
Precision: 0.941364064

Recall: 0.939922481

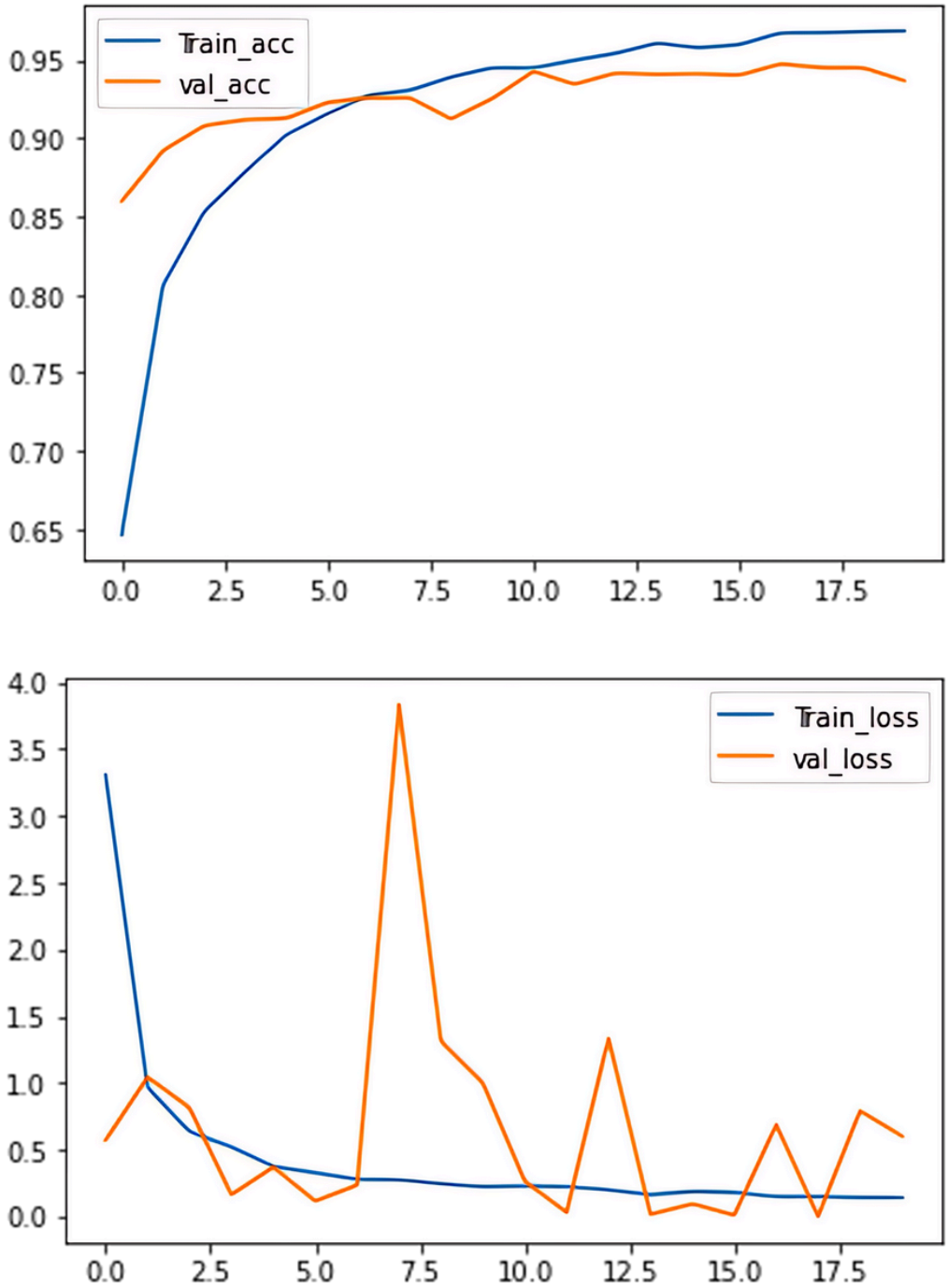
F1-score: 0.939786310

Figure 30-34
ResNet-50 Results

5.5 VGG-16




```
Epoch 1/5
225/226 [============================>.] - ETA: 0s - loss: 0.0066 - accuracy: 0.9983Epoch 1/5
Epoch 1/5
226/226 [==============================] - 126s 558ms/step - loss: 0.0065 - accuracy: 0.9983 -
val_loss: 0.1126 - val_accuracy: 0.9850
Epoch 2/5
226/226 [==============================] - 114s 506ms/step - loss: 0.0027 - accuracy: 0.9990 -
val_loss: 2.1934e-07 - val_accuracy: 0.9850
Epoch 3/5
226/226 [==============================] - 115s 507ms/step - loss: 0.0051 - accuracy: 0.9988 -
val_loss: 2.5548e-04 - val_accuracy: 0.9846
Epoch 4/5
226/226 [==============================] - 111s 492ms/step - loss: 0.0026 - accuracy: 0.9992 -
val_loss: 0.0023 - val_accuracy: 0.9836
Epoch 5/5
226/226 [==============================] - 119s 526ms/step - loss: 0.0020 - accuracy: 0.9994 -
val_loss: 0.0035 - val_accuracy: 0.9850
```



Accuracy: 0.9813468992248062

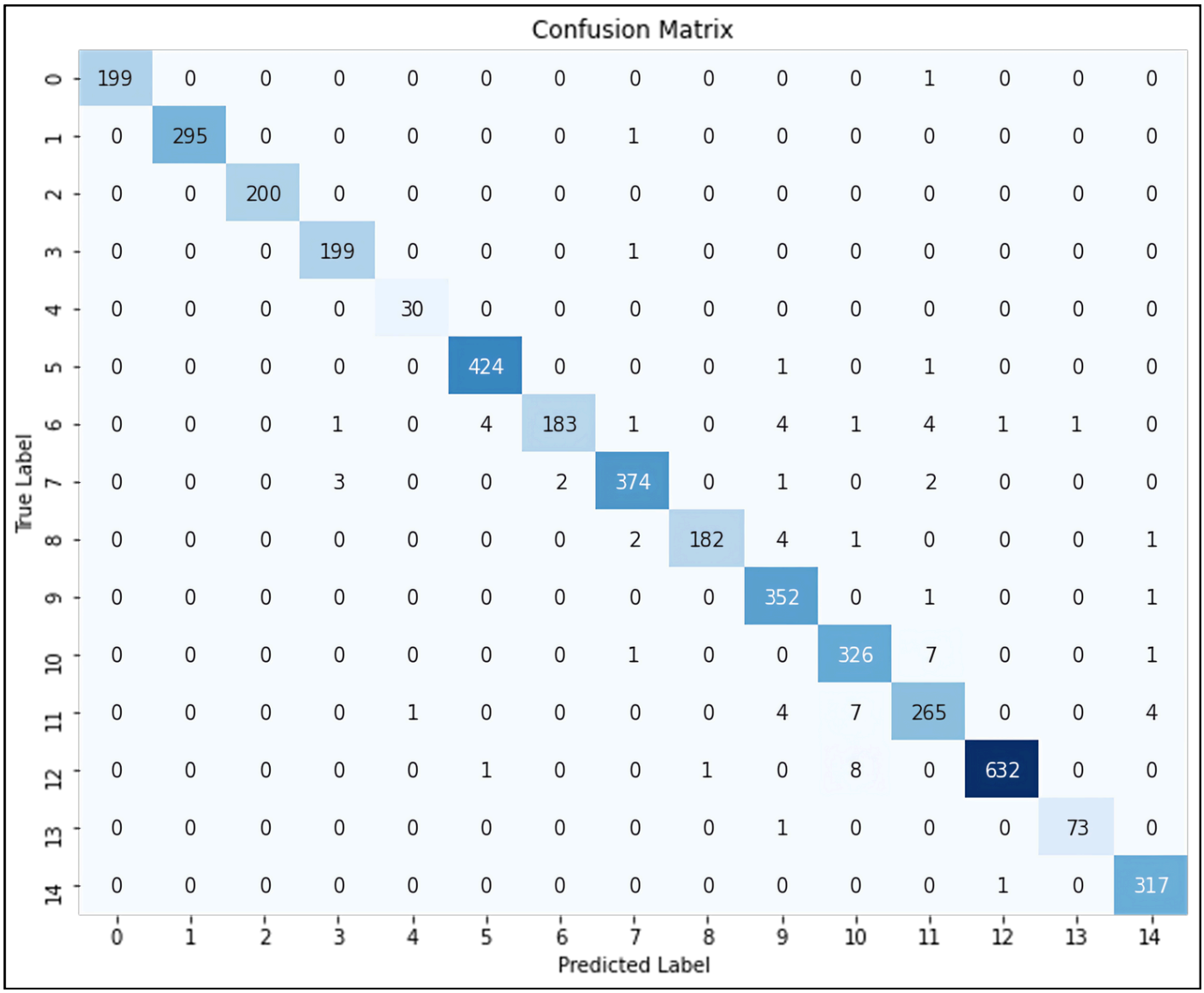
Precision: 0.9815794601012678

Recall: 0.9813468992248062

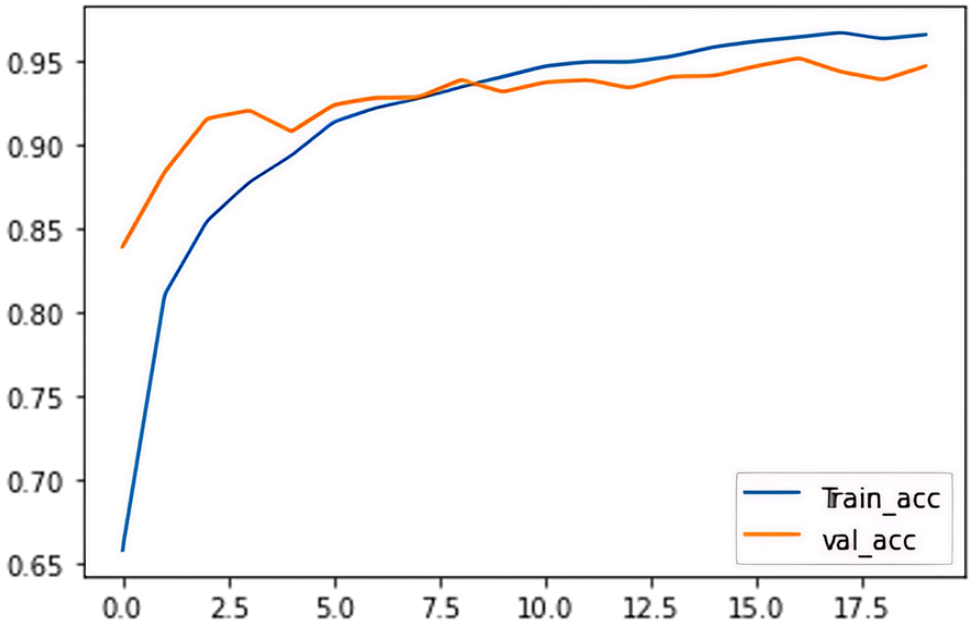
F1-score: 0.9813343195679487

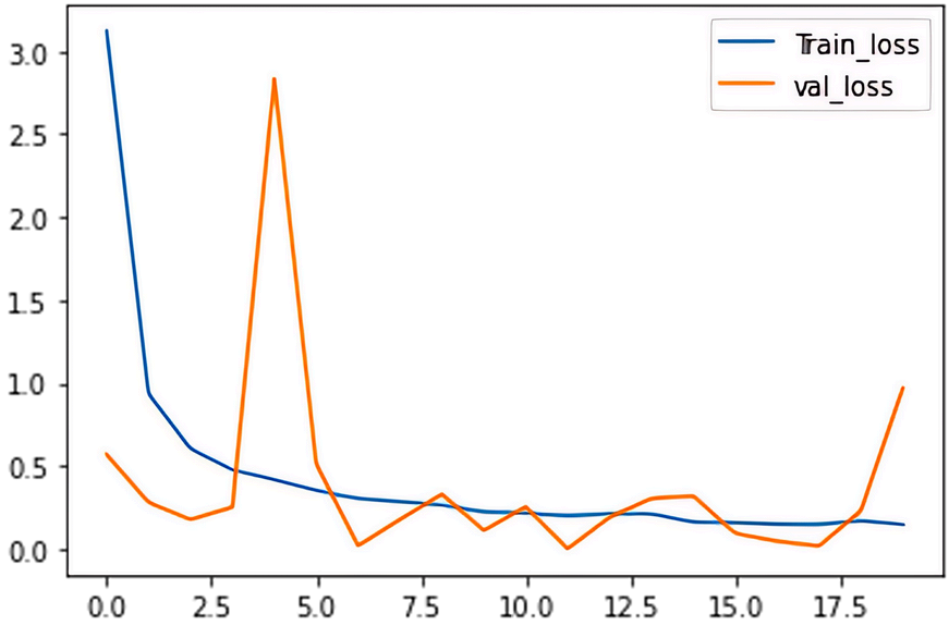
Figure 35-39
VGG-16 Results

5.6 VGG-19




```
Epoch 1/5
226/226 [==============================] - 148s 655ms/step - loss: 0.0124 - accuracy: 0.9962 -
val_loss: 0.2115 - val_accuracy: 0.9768
Epoch 2/5
226/226 [==============================] - 142s 627ms/step - loss: 0.0075 - accuracy: 0.9981 -
val_loss: 0.0123 - val_accuracy: 0.9783
Epoch 3/5
226/226 [==============================] - 140s 620ms/step - loss: 0.0052 - accuracy: 0.9985 -
val_loss: 0.0966 - val_accuracy: 0.9773
Epoch 4/5
Epoch 3/5
226/226 [==============================] - 138s 612ms/step - loss: 0.0035 - accuracy: 0.9990 -
val_loss: 8.3421e-05 - val_accuracy: 0.9797
Epoch 5/5
226/226 [==============================] - 141s 624ms/step - loss: 0.0047 - accuracy: 0.9985 -
val_loss: 0.0024 - val_accuracy: 0.9822
```






```
Accuracy: 0.9813468992248062
Precision: 0.9815713598760859
Recall: 0.9813468992248062
F1-score: 0.9813155167218475
```

Figure 40-44 VGG-19 Results

5.7 Comparision of Various Models

MODEL	ACCURACY	PRECISION	RECALL	F1-SCORE
VGG-16*	98.13*	98.15*	98.13*	98.13*
VGG-19*	98.13	98.15	98.13	98.13
ResNet-50	93.99	94.13	93.99	93.97
Inception-V3	93.43	93.74	93.43	93.45
CNN	92.56	92.48	90.62	91.32
SVM	80.00	87.50	80.00	78.02

Our Best Models

VGG-16 and VGG-19

5. Conclusion

Plant growth and health play a significant role in the economy in courtesies like India, where a huge sector of the population is dependent on agriculture. Due to many reasons like droughts, floods. Improper rain and other, a large part of crops got wasted. Since we can't control the natural factors we have to take proper care of our agriculture to minimize the losses and hence the plant health issue can't be ignored.

Based on the comparison of different models used in this project for plant disease detection, the following conclusions can be drawn:

1. **VGG-16 and VGG-19**: These deep convolutional neural networks (CNNs) performed exceptionally well, achieving the **highest accuracy** of **98.13%**. They also demonstrated high precision, recall, and F1-score, all around **98.13%**. This indicates that VGG-16 and VGG-19 are robust and reliable models for accurately identifying plant diseases.
2. **ResNet-50**: This model achieved an accuracy of **93.99**%, with slightly lower precision, recall, and F1-score compared to VGG-16 and VGG-19. However, it still performed well with precision and recall around 94%, showing its effectiveness in disease classification.
3. **Inception-V3**: Similar to ResNet-50, Inception-V3 also showed good performance with an accuracy of **93.43**%. Its precision and recall were slightly lower than ResNet-50 but remained above 93%, indicating its capability in disease identification.
4. **CNN (Convolutional Neural Network)**: The custom CNN architecture achieved an accuracy of **92.56**%, slightly lower than the pretrained models like VGG-16 and VGG-19. However, it showed competitive precision and recall around 92%, making it a viable option for plant disease detection tasks.
5. **SVM (Support Vector Machine)**: Among the models, SVM exhibited the lowest accuracy of **80.00%**. While its precision was relatively high at 87.50%, the recall and F1-score were lower compared to the deep learning models. This suggests that SVM may not be the best choice for this particular task of plant disease classification, especially when compared to the deep learning architectures.

In summary, the pretrained deep learning models like VGG-16 and VGG-19 outperformed the custom CNN and SVM models in terms of accuracy, precision, recall, and F1-score. Therefore, for accurate and reliable plant disease detection, VGG-16 and VGG-19 are recommended choices. However, ResNet-50 and Inception-V3 also demonstrated good performance and can be considered for this task. The SVM model, while showing respectable precision, lagged behind in overall accuracy and recall compared to the deep learning models, indicating its limitations in this context.

6. Future Scope

The system developed in this project has demonstrated significant efficiency in the analysis of data tested on the platform. In future iterations, we plan to incorporate additional features such as spot detection and shape/edge analysis of plant images. Leaves play a vital role in photosynthesis, as they are where the process occurs, enabling plants to produce their food. Adequate exposure to sunlight on leaf surfaces is crucial for plant health, making leaf shape an important factor.

1. **Real Environment Consideration**: Future research should focus on datasets that reflect the real environment in which plants grow. The PlantVillage dataset, while useful, has a simple/plain background. Incorporating real-world environmental factors can improve the practicality and robustness of disease detection models.
2. **Hyperspectral/Multispectral Imaging**: Utilizing hyperspectral or multispectral imaging technology in conjunction with efficient DL architectures can enable the detection of plant diseases even before visible symptoms appear. These advanced imaging techniques offer a deeper understanding of plant health and can enhance early detection capabilities.
3. **Improved Disease Spot Visualization**: Introducing more efficient methods for visualizing disease spots on plants can lead to cost savings by reducing the unnecessary application of chemicals like fungicides, pesticides, and herbicides. Advanced visualization techniques can provide clearer insights into disease severity and distribution.
4. **Device Camera Integration**: The application will be designed to interface with device cameras, enabling users to capture images in real-time for immediate analysis. This feature will be particularly beneficial for farmers, providing early warnings and actionable insights.
5. **Drone Integration**: Integrating the system with drones will allow farmers to cover larger agricultural areas efficiently. Drones equipped with our system can capture images of fields and provide quick disease diagnosis and recommendations.
6. **Expanded Plant Coverage**: The system will be updated to encompass a broader range of plant species and vegetables. This expansion will enhance its utility across various crops, enabling farmers to detect diseases and take preventive measures early.
7. **Data Science Algorithms(Predictive Algorithms)**: Incorporating data science algorithms will enable the system to predict disease occurrences or possibilities. Early warnings based on predictive analysis will assist farmers in taking proactive steps to protect their crops.

References

1. **X.-D. Yu, M.-J. Yang, H.-Q. Zhang, D. Li, Y.-Q. Tang, and X. Yu, "Research and application of crop diseases detection method based on transfer learning," Trans. Chin. Soc. Agricult. Eng., vol. 51, no. 10, pp. 252–258, May 2020.**
2. **J. Chen, J. Chen, D. Zhang, Y. Sun, and Y. A. Nanehkaran, "Using deep transfer learning for image-based plant disease identification," Comput. Electron. Agricult., vol. 173, Jun. 2020, Art. no. 105393**
3. **Saleem, M. H., Potgieter, J., & Arif, K. M. Plant disease detection and classification by deep learning. Plants, 8(11), 468, 2019**
4. **Arsenovic, M., Karanovic, M., Sladojevic, S., Anderla, A., & Stefanovic, D. (2019). Solving current limitations of deep learning based approaches for plant disease detection. Symmetry, 11(7), 939, 2019.**
5. **Ramesh, S., Hebbar, R., Niveditha, M., Pooja, R., Shashank, N., & Vinod, P. V. Plant disease detection using machine learning. In 2018 International conference on design innovations for 3Cs compute communicate control (ICDI3C) (pp. 41-45). IEEE, 2018**
6. **Agarwal, M., Singh, A., Arjaria, S., Sinha, A., Gupta, S. (2020). ToLeD: Tomato leaf disease detection using convolution neural network. Proc. Comput. Sci. 167 (2019), 293–301. doi: 10.1016/j.procs.2020.03.225**
7. **C.-S. Wang, J. Zhou, H.-R. Wu, G.-F. Teng, C.-J. Zhao, and J.-X. Li, "Identification of vegetable leaf diseases based on improved multi-scale ResNet," Trans. Chin. Soc. Agricult. Eng., vol. 36, no. 20, pp. 209–217, Doc. 2020.**
8. **M. A. Ebrahimi, M. H. Khoshtaghaza, S. Minaei and B. Jamshidi, "Vision-based pest detection based on SVM classification method", Comput. Electron. Agricult., vol. 137, pp. 52-58, May 2017.**
9. **J. G. A. Barbedo, "Factors influencing the use of deep learning for plant disease recognition", Biosyst. Eng., vol. 172, pp. 84-91, Aug. 2018.**
10. **M. Türkoglu and D. Hanbay, "Plant disease and pest detection using deep learning-based features", TURKISH J. Electr. Eng. Comput. Sci., vol. 27, no. 3, pp. 1636-1651, May 2019.**