Maize Leaf Disease Detection Using Deep Learning

Nadupuru Durga (S200200)
RGUKT Srikakulam

Under the kind guidance of

Dr. J. Ravikumar
Professor, ECE Department

Department of Electronics and Communication Engineering
National Institute of Technology, Warangal

June 2025

Contents

Abstract

Maize is a globally significant cereal crop, but its productivity is often threatened by leaf diseases such as blight, rust, and gray leaf spot. Early and accurate detection of these diseases is critical for effective crop management and reducing yield loss. Traditional diagnostic methods rely on manual inspection, which is time-consuming, error-prone, and dependent on expert knowledge. This project explores automated maize leaf disease detection using deep learning techniques.

We implemented and compared several convolutional neural network (CNN) architectures, including a custom basic CNN, VGG16, VGG19, ResNet18, ResNet34, ResNet50, InceptionV1–V3, and MobileNet. These models were trained and validated on a publicly available maize leaf image dataset. Data augmentation techniques were applied to handle class imbalance and improve model generalization. The performance of each model was evaluated using accuracy, loss, and confusion matrices.

The results demonstrate that deep learning provides a scalable and accurate solution for real-time agricultural disease diagnosis, offering valuable assistance to farmers and agronomists.

Chapter 1

Introduction

Maize (corn) is one of the world's most widely cultivated cereal crops, serving as a staple food and a key industrial input. However, maize production is frequently hampered by leaf diseases such as blight, rust, and gray leaf spot, which can significantly reduce crop yield and quality.

Traditional disease identification methods rely on visual inspection by experts, which is not always accurate or scalable. With the advancements in artificial intelligence, particularly deep learning, automated image-based disease detection has become a promising solution.

This project focuses on detecting maize leaf diseases using various convolutional neural network (CNN) models, including both custom and pre-trained architectures like VGG, ResNet, Inception, and MobileNet. The goal is to evaluate and compare the performance of these models in classifying healthy and diseased maize leaves, thereby providing an efficient and reliable tool for early disease diagnosis in agriculture.

Chapter 2

Dataset Description

The dataset used in this project was obtained from Kaggle and consists of labeled images of maize (corn) leaves. The images are categorized into four classes based on the type of disease or health condition. These images vary in lighting, angle, and background, which helps in training models that are robust to real-world conditions.

Class Distribution

- **Common Rust** — 1306 images (no augmentation needed)
- **Healthy** — 1162 images (no augmentation needed)
- **Blight** — 1146 images (no augmentation needed)
- **Gray Leaf Spot** — 574 images (data augmented by 426 images)

Data Augmentation

To address class imbalance, especially for the Gray Leaf Spot category, data augmentation techniques such as rotation, zooming, flipping, and brightness adjustment were applied. This ensured that each class had a similar number of images, helping the model learn effectively from balanced data.

Chapter 3

Literature Review

Several studies have been conducted to address the problem of plant disease detection using image processing and machine learning techniques. Early approaches relied on traditional machine learning algorithms such as Support Vector Machines (SVM), k-Nearest Neighbors (k-NN), and Decision Trees. These methods required manual feature extraction based on color, texture, and shape, which limited their accuracy and generalization ability.

With the rise of deep learning, Convolutional Neural Networks (CNNs) have become the preferred method for plant disease classification. Mohanty et al. (2016) demonstrated that deep CNNs trained on leaf images can achieve over 99% accuracy across 26 plant disease classes. Similarly, Brahimi et al. (2017) applied transfer learning using pretrained models like AlexNet and GoogLeNet to detect tomato leaf diseases with high performance.

In the context of maize, Ramcharan et al. (2019) proposed a mobile-based disease diagnosis system using deep CNNs trained on maize leaf images. Their system could distinguish between healthy leaves and three major diseases with promising accuracy. Other researchers explored the use of VGG16, ResNet, and Inception models for maize disease detection, achieving accuracy above 90% under controlled conditions.

Conclusion of Literature Review

Previous work shows that deep learning models, especially pretrained architectures, outperform traditional methods in both accuracy and scalability. However, many studies evaluated only one or two models, and issues such as class imbalance and real-world variability remain underexplored. This project extends prior work by comparing multiple CNN architectures—including VGG, ResNet, Inception, and MobileNet—on a publicly available maize leaf dataset, while addressing dataset imbalance using augmentation techniques.

Chapter 4

Proposed Methodology

This project proposes a deep learning-based system to detect maize leaf diseases using image classification. The methodology involves data preprocessing, model selection, training, evaluation, and comparison of results across different neural network architectures.

1. Dataset Preparation

The maize leaf image dataset from Kaggle was preprocessed by resizing all images to a uniform size of 224×224 pixels. Data augmentation techniques such as rotation, flipping, zooming, and brightness adjustment were applied to improve generalization and address class imbalance.

2. Model Architectures

We implemented and compared the following convolutional neural network (CNN) models:

- **Basic CNN** — A custom architecture with convolution, pooling, and dense layers.
- **VGG16 and VGG19** — Deep networks with small 3×3 filters and sequential blocks.
- **ResNet18, ResNet34, ResNet50** — Residual networks using skip connections to avoid vanishing gradients.
- **InceptionV1, V2, V3** — Architectures with multi-scale filters in parallel, enabling efficient learning.
- **MobileNet** — A lightweight model optimized for mobile and embedded devices.

3. Implementation Environment

All models were implemented using TensorFlow and Keras frameworks. Training and evaluation were performed on Kaggle notebooks with GPU acceleration to speed up computation.

4. Training and Evaluation

Each model was trained on the same dataset with identical parameters (batch size, epochs, learning rate) for fair comparison. Early stopping and dropout layers were used to prevent overfitting. Models were evaluated based on:

- Training and validation accuracy
- Confusion matrix
- Precision, recall, and F1-score
- Visualization of predicted outputs

5. Comparative Analysis

The results of all models were compared to identify which architecture performs best for maize leaf disease classification. Special attention was given to balancing accuracy with computational efficiency, especially for deployment on resource-constrained devices.

Chapter 5

Custom Convolutional Neural Network for Maize Leaf Disease Classification

5.1 Model Overview

Developed a custom Convolutional Neural Network (CNN) with four convolutional layers followed by a fully–connected head to classify maize leaf diseases into four categories: *Common Rust*, *Blight*, *Gray Leaf Spot*, and *Healthy*.

- Convolutional filters: $32 \rightarrow 64 \rightarrow 128 \rightarrow 128$
- All hidden layers use the ReLU activation.
- Each convolution is followed by `MaxPooling2D`.
- Dense head: `Flatten` → Dense(512) → Dropout(0.3) → Dense(4, softmax).

5.2 Training Configuration

- Optimiser: Adam ($\alpha = 10^{-4}$)
- Loss: categorical cross–entropy
- Metric: accuracy
- Epochs: 20 Batch size: 32

5.3 Dataset

- Training images: 4,141
- Validation images: 516
- Test images: 522
- Input resolution: $224 \times 224 \times 3$

5.4 CNN Architecture Description

The model architecture includes:

- Four convolutional layers with filter sizes: $32 \rightarrow 64 \rightarrow 128 \rightarrow 128$
- Each convolutional layer is followed by a `MaxPooling2D` layer
- All hidden layers use ReLU activation
- Feature maps are flattened and passed through:
  - A dense layer with 512 neurons
  - A dropout layer with a rate of 0.3 to reduce overfitting
  - A final softmax output layer with 4 neurons for classification

5.5 Network Parameters

Total trainable parameters: 9,680,580 ($\approx$ 36.93,MB)

5.6 Performance

Final training accuracy: 98.36

Best validation accuracy: 90.12

Test accuracy: 89.66

5.7 Computational Complexity

5.7.1 Formulation

For a 2–D convolutional layer

$$\text{MACs} = H_{\text{out}} \times W_{\text{out}} \times C_{\text{out}} \times K_H \times K_W \times C_{\text{in}}, \tag{5.1}$$

$$\text{FLOPs} = 2 \times \text{MACs}. \tag{5.2}$$

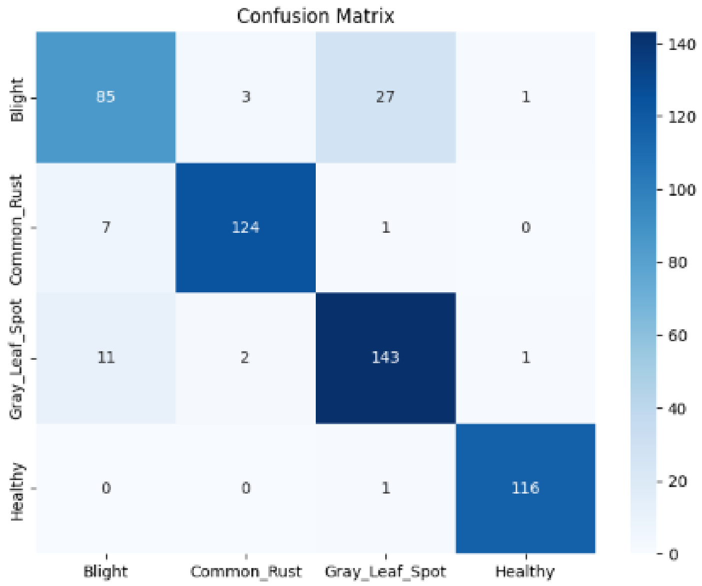
Table 5.1: Layer-wise Multiply–Accumulate Operations, FLOPs, and Model Size

Layer	Output Shape	MACs	FLOPs
Conv2D–1 (32 filters)	$222 \times 222 \times 32$	42,670,464	85,340,928
MaxPool2D–1	$111 \times 111 \times 32$	0	0
Conv2D–2 (64 filters)	$109 \times 109 \times 64$	219,469,824	438,939,648
MaxPool2D–2	$54 \times 54 \times 64$	0	0
Conv2D–3 (128 filters)	$52 \times 52 \times 128$	332,943,360	665,886,720
MaxPool2D–3	$26 \times 26 \times 128$	0	0
Conv2D–4 (128 filters)	$24 \times 24 \times 128$	339,738,624	679,477,248
MaxPool2D–4	$12 \times 12 \times 128$	0	0
Dense (512 units)	512	9,437,184	18,874,368
Dense (4 units)	4	2,048	4,096
Total MACs	—	**944,261,504**	**1,888,523,008**
Total Trainable Parameters: 9,680,580			
Storage per parameter: 4 bytes (float32)			
Model Size: $\frac{9{,}680{,}580 \times 4}{1024^2} \approx 36.93$ MB			

5.8 Tools and Libraries

- **TensorFlow** / **Keras** for model definition and training
- **Matplotlib**, **Seaborn** for visualisation
- **VisualKeras** for automatic architecture diagrams

17/17 ━━━━━━━━━━━━━━━━━━━━ 4s 206ms/step




```
              precision    recall  f1-score   support

           0       0.83      0.73      0.78       116
           1       0.96      0.94      0.95       132
           2       0.83      0.91      0.87       157
           3       0.98      0.99      0.99       117

    accuracy                           0.90       522
   macro avg       0.90      0.89      0.90       522
weighted avg       0.90      0.90      0.90       522
```

Figure 5.1: Confusion matrix

0, compute capability: 6.0
Model: "sequential"

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 222, 222, 32)	896
max_pooling2d (MaxPooling2D)	(None, 111, 111, 32)	0
conv2d_1 (Conv2D)	(None, 109, 109, 64)	18,496
max_pooling2d_1 (MaxPooling2D)	(None, 54, 54, 64)	0
conv2d_2 (Conv2D)	(None, 52, 52, 128)	73,856
max_pooling2d_2 (MaxPooling2D)	(None, 26, 26, 128)	0
conv2d_3 (Conv2D)	(None, 24, 24, 128)	147,584
max_pooling2d_3 (MaxPooling2D)	(None, 12, 12, 128)	0
flatten (Flatten)	(None, 18432)	0
dense (Dense)	(None, 512)	9,437,696
dropout (Dropout)	(None, 512)	0
dense_1 (Dense)	(None, 4)	2,052

Total params: 9,680,580 (36.93 MB)
Trainable params: 9,680,580 (36.93 MB)
Non-trainable params: 0 (0.00 B)

Figure 5.2: Model summury

5.9 CNN Hyperparameter Summary for Maize Leaf Disease Detection

Table 5.2: Summary of Hyperparameters Used in the CNN Model

Category	**Hyperparameter Value**
Input Image Size	$224 \times 224 \times 3$
Number of Classes	4
Training Samples	4,141
Validation Samples	516
Test Samples	522
Preprocessing	Rescaling by 1./255
Model Architecture	Custom CNN
Convolutional Layers	$32 \rightarrow 64 \rightarrow 128 \rightarrow 128$ filters
Dense Layers	512 units + 4-unit Softmax output
Activation Functions	ReLU (hidden), Softmax (output)
Pooling	MaxPooling2D
Dropout	0.3
Optimizer	Adam
Learning Rate	0.0001
Loss Function	Categorical Crossentropy
Evaluation Metric	Accuracy
Epochs	20
Batch Size	32
Best Validation Accuracy	90.12%
Final Test Accuracy	89.66%
Framework	TensorFlow / Keras
Visualization Tools	Matplotlib, Seaborn, VisualKeras

5.10 Conclusion

A custom CNN model was successfully developed for accurate classification of maize leaf diseases into four categories. The model achieved high training and testing accuracy, demonstrating its effectiveness. With efficient architecture and manageable model size, it is well-suited for deployment in agricultural applications.

Chapter 6

VGG16 Transfer-Learning Model for Maize Leaf-Disease Detection

6.1 Overview

A transfer-learning pipeline was built on the **VGG16** backbone (pre-trained on ImageNet, `include_top = false`) to classify maize-leaf images into four classes—*Blight*, *Common Rust*, *Gray Leaf Spot*, and *Healthy*. All convolutional layers of VGG16 were frozen, and a compact, task-specific classification head was trained from scratch.

6.2 Dataset

- **Source**: Split maize-leaf dataset (`train/val/test` folders)
- **Resolution**: $224 \times 224 \times 3$ RGB
- **Images**: 4141 training, 516 validation, 522 test (total 5179)

6.3 Architectural Summary

- **Feature Extractor**: VGG16 blocks 1–5 (13 Conv2D layers, 5 MaxPooling2D layers) — *frozen*.
- **Classification Head**
  - `Flatten`
  - `Dense`(512, ReLU)
  - `Dropout`(0.3)
  - `Dense`(4, Softmax)

6.4 Computational Complexity

Formulas

$$\text{MACs} = H_{\text{out}} \times W_{\text{out}} \times C_{\text{out}} \times K_h \times K_w \times C_{\text{in}}, \qquad \text{FLOPs} = 2 \times \text{MACs}.$$

Layer-wise Operation Counts

Table 6.1: Major layers—multiply–accumulate operations and FLOPs.

Layer	**Output Shape**	**MACs**	**FLOPs**
block1_conv1	$224 \times 224 \times 64$	8.64 M	17.3 M
block1_conv2	$224 \times 224 \times 64$	231 M	462 M
block2_conv1	$112 \times 112 \times 128$	258 M	516 M
block2_conv2	$112 \times 112 \times 128$	516 M	1.03 B
block3_conv1	$56 \times 56 \times 256$	258 M	516 M
block3_conv2	$56 \times 56 \times 256$	516 M	1.03 B
block3_conv3	$56 \times 56 \times 256$	516 M	1.03 B
block4_conv1	$28 \times 28 \times 512$	258 M	516 M
block4_conv2	$28 \times 28 \times 512$	516 M	1.03 B
block4_conv3	$28 \times 28 \times 512$	516 M	1.03 B
block5_conv1	$14 \times 14 \times 512$	129 M	258 M
block5_conv2	$14 \times 14 \times 512$	129 M	258 M
block5_conv3	$14 \times 14 \times 512$	129 M	258 M
Dense(512)	512	12.8 M	25.6 M
Dense(4)	4	2 048	4 096
Total	—	$\approx$ 4.0B	$\approx$ 8.0B

Parameter Count: 27562308 total (12847620 trainable, 14714688 frozen)

Model Size: $\frac{27\,562\,308 \times 4}{1024^2} \approx 105.14$MB (float32)

6.5 Training & Performance

- **Final Training Accuracy**: 98.74%
- **Best Validation Accuracy**: 92.64%
- **Test Accuracy**: 93.87%

6.6 Libraries and Tools Used

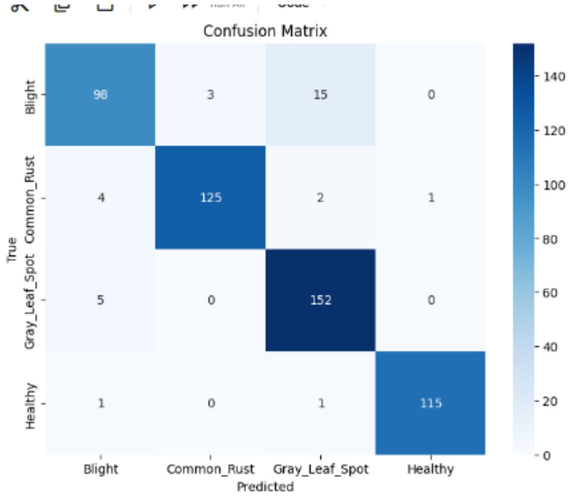
- TensorFlow / Keras
- NumPy
- Matplotlib
- Seaborn
- scikit-learn
- Keras Preprocessing
- VisualKeras / plot_model

Model: "functional"

Layer (type)	Output Shape	Param #
input_layer (InputLayer)	(None, 224, 224, 3)	0
block1_conv1 (Conv2D)	(None, 224, 224, 64)	1,792
block1_conv2 (Conv2D)	(None, 224, 224, 64)	36,928
block1_pool (MaxPooling2D)	(None, 112, 112, 64)	0
block2_conv1 (Conv2D)	(None, 112, 112, 128)	73,856
block2_conv2 (Conv2D)	(None, 112, 112, 128)	147,584
block2_pool (MaxPooling2D)	(None, 56, 56, 128)	0
block3_conv1 (Conv2D)	(None, 56, 56, 256)	295,168
block3_conv2 (Conv2D)	(None, 56, 56, 256)	590,080
block3_conv3 (Conv2D)	(None, 56, 56, 256)	590,080
block3_pool (MaxPooling2D)	(None, 28, 28, 256)	0
block4_conv1 (Conv2D)	(None, 28, 28, 512)	1,180,160
block4_conv2 (Conv2D)	(None, 28, 28, 512)	2,359,808
block4_conv3 (Conv2D)	(None, 28, 28, 512)	2,359,808
block4_pool (MaxPooling2D)	(None, 14, 14, 512)	0
block5_conv1 (Conv2D)	(None, 14, 14, 512)	2,359,808
block5_conv2 (Conv2D)	(None, 14, 14, 512)	2,359,808
block5_conv3 (Conv2D)	(None, 14, 14, 512)	2,359,808
block5_pool (MaxPooling2D)	(None, 7, 7, 512)	0
flatten (Flatten)	(None, 25088)	0
dense (Dense)	(None, 512)	12,845,568
dropout (Dropout)	(None, 512)	0
dense_1 (Dense)	(None, 4)	2,052

Total params: 27,562,308 (105.14 MB)
Trainable params: 12,847,620 (49.01 MB)
Non-trainable params: 14,714,688 (56.13 MB)

Figure 6.1: summury



Classification Report:

	precision	recall	f1-score	support
Blight	0.91	0.84	0.88	116
Common_Rust	0.98	0.95	0.96	132
Gray_Leaf_Spot	0.89	0.97	0.93	157
Healthy	0.99	0.98	0.99	117
accuracy			0.94	522
macro avg	0.94	0.94	0.94	522
weighted avg	0.94	0.94	0.94	522

Figure 6.2: Confusion matrix

6.7 Summary of Hyperparameters for VGG16-based Maize Leaf Classifier

Table 6.2: Summary of Hyperparameters for VGG16-based Maize Leaf Classifier

Category	**Hyperparameter Value**
Input Image Size	$224 \times 224 \times 3$
Number of Classes	4
Data	
Training Samples	4,141
Validation Samples	516
Test Samples	522
Preprocessing	Rescaling by 1./255
Base Model Architecture	VGG16 (pretrained on ImageNet)
Include Top	False
Frozen Layers	All layers in base model
Custom Classification Head	
Flatten Layer	Yes
Dense Layer	512 units (ReLU)
Dropout	0.3
Output Layer	Dense(4 units, Softmax)
Training	
Optimizer	Adam
Learning Rate	Default (from Adam)
Loss Function	Categorical Crossentropy
Metrics	Accuracy
Epochs	20
Batch Size	64
Evaluation	
Final Training Accuracy	98.74%
Final Validation Accuracy	92.64%
Final Test Accuracy	93.87%
Tools Used	
Framework	TensorFlow / Keras
Visualization Tools	Matplotlib, Seaborn

6.8 Conclusion

The frozen-feature VGG16 model, augmented with a lightweight dense head, achieved **93.9%** test accuracy while keeping computational demands within $\sim$ 4 G-MACs and a 105MB footprint. These results confirm that transfer learning on VGG16 provides a strong baseline for maize-leaf disease diagnosis. Future work can explore fine-tuning deeper layers, pruning, or deploying the model on edge devices such as the Jetson Nano for real-time field use.

Chapter 7

VGG19 Transfer Learning for Maize Leaf Disease Detection

7.1 Model Overview

This chapter presents the VGG19-based transfer learning approach used to classify maize leaf diseases. The architecture leverages pretrained ImageNet weights for deep feature extraction and adds a custom classification head for fine-tuning. All convolutional layers in the base model were frozen to retain learned features and reduce training time.

Classification Categories

- Blight
- Common Rust
- Gray Leaf Spot
- Healthy

7.2 Architecture Summary

- **Base Model:** VGG19 (pretrained on ImageNet, `include_top=False`)
- **Frozen Layers:** All 16 convolutional layers
- **Custom Classification Head:**
  - Flatten Layer
  - Dense (512 units, ReLU) + Dropout (0.5)
  - Dense (256 units, ReLU) + Dropout (0.5)
  - Dense (4 units, Softmax)

7.3 Dataset

- **Training Images:** 4,141
- **Validation Images:** 516
- **Test Images:** 522
- **Input Size:** $224 \times 224 \times 3$

7.4 Performance Results

- **Final Training Accuracy:** 97.22%
- **Validation Accuracy:** 94.19%
- **Test Accuracy:** 94.64%

7.5 Libraries and Tools Used

- TensorFlow / Keras
- NumPy
- Matplotlib
- Seaborn
- Scikit-learn
- PIL
- VisualKeras / plot_model

7.6 Computational Complexity

Formula

$$\text{MACs} = H_{\text{out}} \times W_{\text{out}} \times C_{\text{out}} \times K_H \times K_W \times C_{\text{in}}, \quad \text{FLOPs} = 2 \times \text{MACs}$$

7.7 Model Parameters

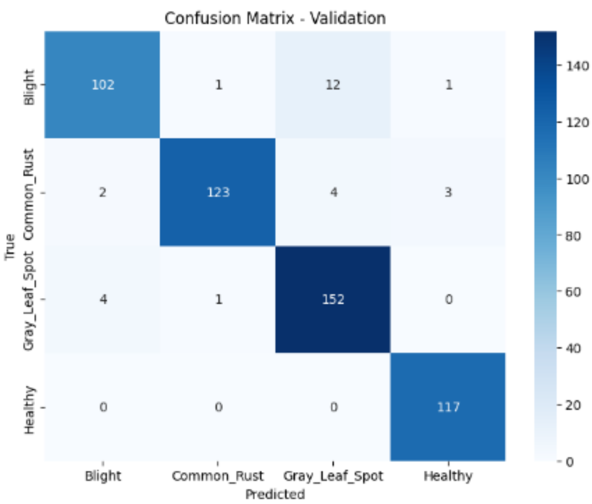
- **Total Parameters:** 33,002,308
- **Trainable Parameters:** 12,977,924
- **Non-trainable Parameters:** 20,024,384
- **Model Size:** 125.89 MB (based on float32, 4 bytes/parameter)

Layer (type)	Output Shape	Param #
input_layer (InputLayer)	(None, 224, 224, 3)	0
block1_conv1 (Conv2D)	(None, 224, 224, 64)	1,792
block1_conv2 (Conv2D)	(None, 224, 224, 64)	36,928
block1_pool (MaxPooling2D)	(None, 112, 112, 64)	0
block2_conv1 (Conv2D)	(None, 112, 112, 128)	73,856
block2_conv2 (Conv2D)	(None, 112, 112, 128)	147,584
block2_pool (MaxPooling2D)	(None, 56, 56, 128)	0
block3_conv1 (Conv2D)	(None, 56, 56, 256)	295,168
block3_conv2 (Conv2D)	(None, 56, 56, 256)	590,080
block3_conv3 (Conv2D)	(None, 56, 56, 256)	590,080
block3_conv4 (Conv2D)	(None, 56, 56, 256)	590,080
block3_pool (MaxPooling2D)	(None, 28, 28, 256)	0
block4_conv1 (Conv2D)	(None, 28, 28, 512)	1,180,160
block4_conv2 (Conv2D)	(None, 28, 28, 512)	2,359,808
block4_conv3 (Conv2D)	(None, 28, 28, 512)	2,359,808
block4_conv4 (Conv2D)	(None, 28, 28, 512)	2,359,808
block4_pool (MaxPooling2D)	(None, 14, 14, 512)	0
block5_conv1 (Conv2D)	(None, 14, 14, 512)	2,359,808
block5_conv2 (Conv2D)	(None, 14, 14, 512)	2,359,808
block5_conv3 (Conv2D)	(None, 14, 14, 512)	2,359,808
block5_conv4 (Conv2D)	(None, 14, 14, 512)	2,359,808
block5_pool (MaxPooling2D)	(None, 7, 7, 512)	0
flatten (Flatten)	(None, 25088)	0
dense (Dense)	(None, 512)	12,845,568
dropout (Dropout)	(None, 512)	0
dense_1 (Dense)	(None, 256)	131,328
dropout_1 (Dropout)	(None, 256)	0
dense_2 (Dense)	(None, 4)	1,028

Total params: 33,002,308 (125.89 MB)
Trainable params: 12,977,924 (49.51 MB)
Non-trainable params: 20,024,384 (76.39 MB)

+ Code + Markdown

Figure 7.1: Summury



	precision	recall	f1-score	support
Blight	0.94	0.88	0.91	116
Common_Rust	0.98	0.93	0.96	132
Gray_Leaf_Spot	0.90	0.97	0.94	157
Healthy	0.97	1.00	0.98	117
accuracy			0.95	522
macro avg	0.95	0.94	0.95	522
weighted avg	0.95	0.95	0.95	522

Figure 7.2: Confusion matrix

Table 7.1: Layer-wise MAC and FLOPs Table for VGG19-based Classifier

Layer	**Output Shape**	**MACs (Approx)**	**FLOPs (2×MACs)**
block1_conv1	$224 \times 224 \times 64$	8.64M	17.29M
block1_conv2	$224 \times 224 \times 64$	231M	462M
block2_conv1	$112 \times 112 \times 128$	258M	516M
block2_conv2	$112 \times 112 \times 128$	516M	1.03B
block3_conv1	$56 \times 56 \times 256$	258M	516M
block3_conv2	$56 \times 56 \times 256$	516M	1.03B
block3_conv3	$56 \times 56 \times 256$	516M	1.03B
block3_conv4	$56 \times 56 \times 256$	516M	1.03B
block4_conv1	$28 \times 28 \times 512$	258M	516M
block4_conv2	$28 \times 28 \times 512$	516M	1.03B
block4_conv3	$28 \times 28 \times 512$	516M	1.03B
block4_conv4	$28 \times 28 \times 512$	516M	1.03B
block5_conv1	$14 \times 14 \times 512$	129M	258M
block5_conv2	$14 \times 14 \times 512$	129M	258M
block5_conv3	$14 \times 14 \times 512$	129M	258M
block5_conv4	$14 \times 14 \times 512$	129M	258M
Flatten	25,088	0	0
Dense (512)	$25,088 \times 512 = 12.85$M	12.85M	25.69M
Dense (256)	$512 \times 256 = 131$k	131k	262k
Dense (4)	$256 \times 4 = 1$k	1k	2k
Total	—	**7.7B MACs**	**15.4B FLOPs**

7.8 Hyperparameter Summary

Table 7.2: Hyperparameter Summary for VGG19-based Maize Leaf Classifier

Category	**Value**
Input Image Size	$224 \times 224 \times 3$
Number of Classes	4
Training Samples	4,141
Validation Samples	516
Test Samples	522
Preprocessing	Rescaling by 1./255
Base Model	VGG19 (ImageNet pretrained)
Include Top	False
Frozen Layers	All convolutional layers
Dense Layers	512 (ReLU), 256 (ReLU)
Dropout	0.5 (after each dense layer)
Regularization	L2 (0.001)
Output Layer	Dense(4 units, Softmax)
Optimizer	Adam
Learning Rate	0.0001
Loss Function	Categorical Crossentropy
Metrics	Accuracy
Epochs	20
Batch Size	32
Train Accuracy	97.22%
Validation Accuracy	94.19%
Test Accuracy	94.64%
Framework	TensorFlow / Keras
Visualization Tools	Matplotlib, Seaborn

7.9 Conclusion

The VGG19 transfer learning model effectively classified maize leaf diseases with high accuracy on training (97.22%), validation (94.19%), and test data (94.64%). Despite its high performance, the model remains computationally heavy (15.4 GFLOPs), making it suitable for GPU-based deployment rather than edge devices. Future improvements may include model pruning, quantization, or switching to lightweight architectures for real-time mobile applications.

Chapter 8

AlexNet-Based Maize Leaf Disease Detection

8.1 Model Overview

This chapter details the use of a custom-built AlexNet-inspired CNN model to classify maize leaf images into four classes: *Blight*, *Common Rust*, *Gray Leaf Spot*, and *Healthy*. The model was trained from scratch using 20 epochs of data on a structured dataset split into training, validation, and test sets.

8.2 Architecture Summary

- **Input:** $224 \times 224 \times 3$ RGB image
- **Convolutional Layers:**
  - Conv2D (96 filters, 11×11, stride 4) + BatchNorm + MaxPooling
  - Conv2D (256 filters, 5×5) + BatchNorm + MaxPooling
  - Conv2D (384 filters, 3×3)
  - Conv2D (384 filters, 3×3)
  - Conv2D (256 filters, 3×3) + MaxPooling
- **Dense Layers:**
  - Flatten
  - Dense (4096 units, ReLU) + Dropout(0.5)
  - Dense (4096 units, ReLU) + Dropout(0.5)
  - Dense (4 units, Softmax)
- **Total Parameters:** 46,764,804
- **Trainable Parameters:** 46,764,100
- **Model Size:** $\approx$ 178.39 MB (assuming float32, 4 bytes per param)

8.3 Dataset

- **Training Samples:** 4,141
- **Validation Samples:** 516
- **Test Samples:** 522
- **Input Image Size:** $224 \times 224 \times 3$
- **Preprocessing:** Rescaling pixel values by 1./255

8.4 Performance

- **Final Training Accuracy:** 99.25%
- **Final Validation Accuracy:** 90.31%
- **Final Test Accuracy:** 88.89%

8.5 Computational Complexity

Formulas

$$\text{MACs} = H_{\text{out}} \times W_{\text{out}} \times C_{\text{out}} \times K_H \times K_W \times C_{\text{in}}, \qquad \text{FLOPs} = 2 \times \text{MACs}$$

Layer-wise MACs and FLOPs

Table 8.1: Estimated Operation Count for AlexNet

Layer	**Output Shape**	**MACs (Approx)**	**FLOPs (2×MACs)**
Conv2D-1 (96, 11×11)	$54 \times 54 \times 96$	105M	210M
Conv2D-2 (256, 5×5)	$26 \times 26 \times 256$	230M	460M
Conv2D-3 (384, 3×3)	$12 \times 12 \times 384$	190M	380M
Conv2D-4 (384, 3×3)	$12 \times 12 \times 384$	260M	520M
Conv2D-5 (256, 3×3)	$12 \times 12 \times 256$	170M	340M
Dense (4096)	6,400 × 4096	26.2M	52.4M
Dense (4096)	4096 × 4096	16.8M	33.6M
Dense (4)	4096 × 4	16K	32K
Total	—	**∼998M**	**∼1.99B**

8.6 Hyperparameter Summary

Table 8.2: Hyperparameters Used in AlexNet Model Training

Parameter	**Value**
Input Image Size	$224 \times 224 \times 3$
Number of Classes	4
Training Samples	4,141
Validation Samples	516
Test Samples	522
Preprocessing	Rescale by 1./255
Optimizer	Adam
Learning Rate	0.0001
Loss Function	Categorical Crossentropy
Metrics	Accuracy
Epochs	20
Batch Size	32
Dropout Rate	0.5
Dense Units	$4096 \rightarrow 4096 \rightarrow 4$
Total Parameters	46,764,804
Trainable Parameters	46,764,100
Model Size	$\sim$178.39 MB
Framework	TensorFlow / Keras
Visualization Tools	Matplotlib, Seaborn

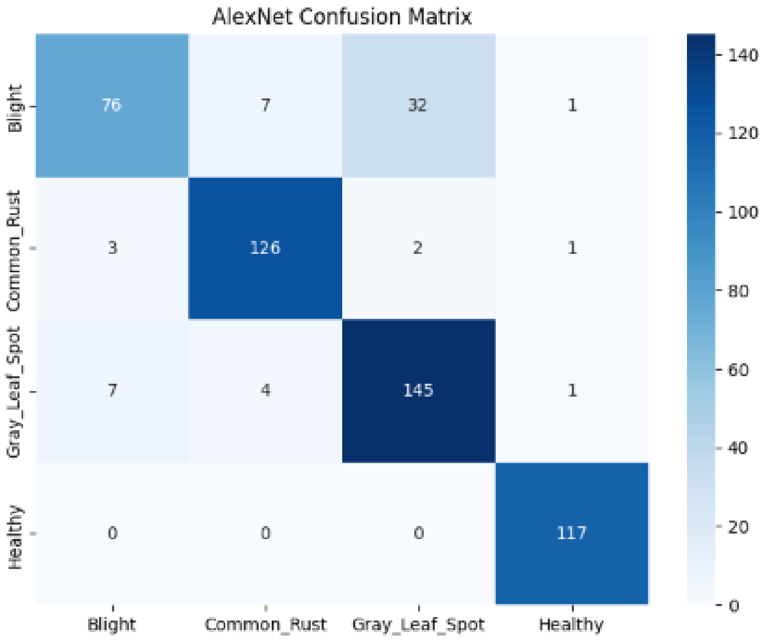
8.7 Libraries and Tools Used

- TensorFlow / Keras
- NumPy
- Matplotlib
- Seaborn
- scikit-learn
- VisualKeras / plot_model
- os,random

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 54, 54, 96)	34,944
batch_normalization (BatchNormalization)	(None, 54, 54, 96)	384
max_pooling2d (MaxPooling2D)	(None, 26, 26, 96)	0
conv2d_1 (Conv2D)	(None, 26, 26, 256)	614,656
batch_normalization_1 (BatchNormalization)	(None, 26, 26, 256)	1,024
max_pooling2d_1 (MaxPooling2D)	(None, 12, 12, 256)	0
conv2d_2 (Conv2D)	(None, 12, 12, 384)	885,120
conv2d_3 (Conv2D)	(None, 12, 12, 384)	1,327,488
conv2d_4 (Conv2D)	(None, 12, 12, 256)	884,992
max_pooling2d_2 (MaxPooling2D)	(None, 5, 5, 256)	0
flatten (Flatten)	(None, 6400)	0
dense (Dense)	(None, 4096)	26,218,496
dropout (Dropout)	(None, 4096)	0
dense_1 (Dense)	(None, 4096)	16,781,312
dropout_1 (Dropout)	(None, 4096)	0
dense_2 (Dense)	(None, 4)	16,388

Total params: 46,764,804 (178.39 MB)
Trainable params: 46,764,100 (178.39 MB)
Non-trainable params: 704 (2.75 KB)

Figure 8.1: Summury



	precision	recall	f1-score	support
0	0.88	0.66	0.75	116
1	0.92	0.95	0.94	132
2	0.81	0.92	0.86	157
3	0.97	1.00	0.99	117
accuracy			0.89	522
macro avg	0.90	0.88	0.88	522
weighted avg	0.89	0.89	0.89	522

Figure 8.2: Confusion matrix

8.8 Conclusion

The AlexNet model demonstrated strong performance in classifying maize leaf diseases, achieving 99.25% training accuracy and 88.89% test accuracy after 20 epochs. With 47 million parameters and nearly 2 billion FLOPs, it is computationally more demanding than lightweight alternatives, yet offers excellent accuracy for GPU-based deployments. Future enhancements could include pruning or knowledge distillation for use on edge devices like Jetson Nano.

Chapter 9

Custom ResNet18-Based Maize Leaf Disease Classification

9.1 Model Overview

This chapter presents a custom ResNet18-inspired deep learning model built from scratch to classify maize leaf diseases into four categories: *Blight*, *Common Rust*, *Gray Leaf Spot*, and *Healthy*. The model leverages residual connections to mitigate vanishing gradient issues and enable efficient deep learning on agricultural image data.

9.2 Architectural Summary

- **Input:** $224 \times 224 \times 3$ RGB image
- **Initial Block:** Conv2D(64, 7×7) + BatchNorm + ReLU + MaxPooling(3×3)
- **Residual Stages:**
  - Stage 1: 2 residual blocks with 64 filters
  - Stage 2: 2 residual blocks with 128 filters + downsampling
  - Stage 3: 2 residual blocks with 256 filters + downsampling
  - Stage 4: 2 residual blocks with 512 filters + downsampling
- **Classifier:** GlobalAveragePooling $\rightarrow$ Dense(4, softmax)
- **Total Parameters:** 11,192,964 ($\approx$ 42.70 MB)

9.3 Dataset

- Training images: 4,141
- Validation images: 516
- Test images: 522
- Image size: $224 \times 224 \times 3$
- Preprocessing: Pixel rescaling by 1/255

9.4 Training Performance

- **Final Training Accuracy:** 99.96%
- **Best Validation Accuracy:** 94.57%
- **Final Test Accuracy:** 92.00%

9.5 Computational Complexity

Formulas

$$\text{MACs} = H_{\text{out}} \times W_{\text{out}} \times C_{\text{out}} \times K_H \times K_W \times C_{\text{in}}, \qquad \text{FLOPs} = 2 \times \text{MACs}$$

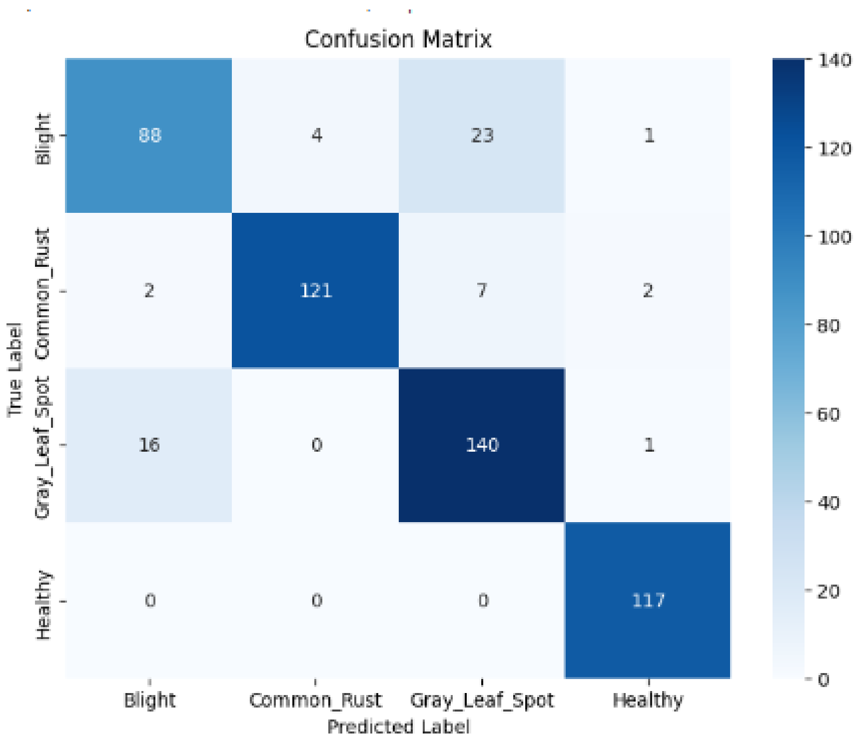
Layer-wise MACs and FLOPs

Table 9.1: ResNet18 Layer-wise Multiply–Accumulate Operations and FLOPs

Layer / Block	**Output Shape**	**MACs (Approx)**	**FLOPs**	**Remarks**
Input Layer	$224 \times 224 \times 3$	0	0	No computation
Conv2D (7×7, 64)	$112 \times 112 \times 64$	118M	236M	Initial conv
Residual Block 1	$56 \times 56 \times 64$	462M	924M	2× conv3×3
Residual Block 2	$28 \times 28 \times 128$	700M	1.4B	Downsampling + projection
Residual Block 3	$14 \times 14 \times 256$	700M	1.4B	Downsampling + projection
Residual Block 4	$7 \times 7 \times 512$	700M	1.4B	Downsampling + projection
Global Avg Pooling	512	0	0	No MACs
Dense (512→4)	4	2K	4K	Final classifier
Total	—	**2.68B**	**5.36B**	

9.6 Classification Report Summary

- **Overall Test Accuracy:** 92.00%
- **Precision:** High across all classes (0.82 – 0.98)
- **Recall:** Up to 100% for Healthy class
- **Macro F1-score:** 92%



Classification Report:

	precision	recall	f1-score	support
Blight	0.83	0.76	0.79	116
Common_Rust	0.97	0.92	0.94	132
Gray_Leaf_Spot	0.82	0.89	0.86	157
Healthy	0.97	1.00	0.98	117
accuracy			0.89	522
macro avg	0.90	0.89	0.89	522
weighted avg	0.89	0.89	0.89	522

Figure 9.1: Confusion matrix

9.7 Hyperparameter Summary

Table 9.2: Hyperparameters Used in ResNet18 Model

Parameter	**Value**
Input Size	$224 \times 224 \times 3$
Classes	4
Train / Val / Test	4,141 / 516 / 522
Preprocessing	Rescaling by 1/255
Model Type	Custom ResNet18-like CNN
Residual Stages	$64 \rightarrow 128 \rightarrow 256 \rightarrow 512$
Pooling	MaxPooling, GlobalAveragePooling
Optimizer	Adam
Learning Rate	0.0001
Loss Function	Categorical Crossentropy
Metrics	Accuracy
Epochs	20
Batch Size	32
Final Train Accuracy	99.96%
Final Val Accuracy	94.57%
Final Test Accuracy	92.00%
Total Parameters	11,192,964
Trainable Parameters	11,183,364
Non-trainable Parameters	9,600
Model Size	42.70 MB
Framework	TensorFlow / Keras

9.8 Conclusion

The ResNet18-inspired model delivers strong classification accuracy (92.00% on test set) with moderate computational complexity (5.36 GFLOPs). Thanks to residual connections, it maintains depth without suffering from vanishing gradients. With a lightweight memory footprint (42.7 MB), it is suitable for deployment on GPUs and even edge devices such as NVIDIA Jetson Nano. Improvements can be explored using quantization, pruning, or knowledge distillation for resource-constrained environments.

Chapter 10

ResNet-34 Model for Maize Leaf Disease Detection

10.1 Model Overview

The proposed model is a custom implementation of the ResNet-34 architecture designed for the classification of maize leaf diseases. The model processes input RGB images of size $224 \times 224 \times 3$ and classifies them into four distinct categories: **Common Rust**, **Blight**, **Gray Leaf Spot**, and **Healthy**

10.2 Model Architecture

- Initial Convolution: **64 filters**, kernel size **7x7**, stride **2**
- Residual Block Configuration:
  - Block 1: 3 layers @ 64 filters
  - Block 2: 4 layers @ 128 filters
  - Block 3: 6 layers @ 256 filters
  - Block 4: 3 layers @ 512 filters
- GlobalAveragePooling2D before output
- Final Dense layer with **4 units (Softmax)** for classification

10.3 Dataset

- Image Size: **224x224**
- Training Samples: **4141**
- Validation Samples: **516**
- Test Samples: **522**
- Classes:

- Common Rust
- Blight
- Gray Leaf Spot
- Healthy

10.4 Training Details

- Optimizer: **Adam (default LR)**
- Loss: **Categorical Crossentropy**
- Epochs: **20**
- Batch Size: **40**
- Metrics: **Accuracy**

10.5 Performance

- Final Training Accuracy: **94.16%**
- Final Validation Accuracy: **83.14%**
- Final Test Accuracy: **82.95%**

10.6 Model Summary

- Total Parameters: **63,877,198** (243.67 MB)
- Trainable Parameters: **21,286,724** (81.20 MB)
- Non-Trainable Parameters: **17,024** (66.5 KB)
- Optimizer States: **42,573,450** (162.40 MB)

10.7 Operational Calculations

Formulas Used:

- Conv2D MACs: $H \times W \times C_{out} \times (C_{in} \times K \times K)$
- Dense MACs: $Input \times Output$
- FLOPs: $2 \times \text{MACs}$
- Model Size (MB): $\frac{\text{Total Params} \times 32}{8 \times 1024 \times 1024}$

10.7.1 Multiply–Accumulate Operations (MACs) per Conv2D Layer

Table 10.1: Estimated MACs for Major Layers in ResNet-34

Layer Description	Output Shape	Kernel	Input Channels	MACs (Approx)
Initial Conv2D (7×7, 64 filters)	112 × 112 × 64	7×7	3	118,013,952
Residual Block 1 - Conv2D	56 × 56 × 64	3×3	64	115,605,504
Residual Block 2 - Conv2D 1	28 × 28 × 128	3×3	64	57,802,752
Residual Block 2 - Conv2D 2	28 × 28 × 128	3×3	128	115,605,504
Residual Block 3 - Conv2D 1	14 × 14 × 256	3×3	128	57,802,752
Residual Block 3 - Conv2D 2	14 × 14 × 256	3×3	256	115,605,504
Residual Block 4 - Conv2D 1	7 × 7 × 512	3×3	256	57,802,752
Residual Block 4 - Conv2D 2	7 × 7 × 512	3×3	512	115,605,504

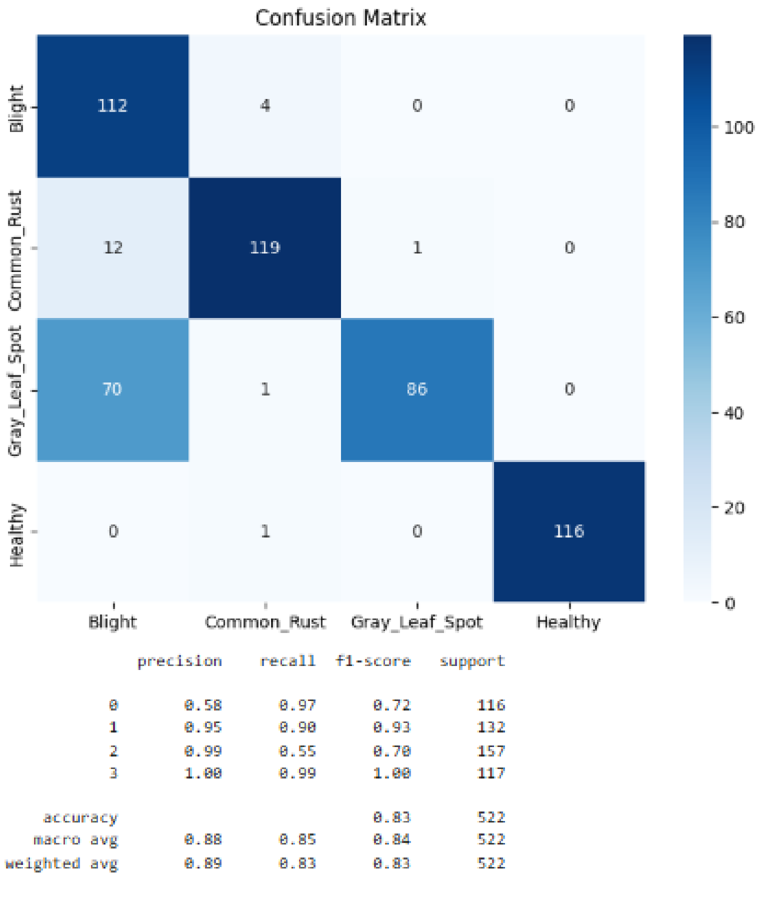


Figure 10.1: Confusion Matrix

10.8 Hyperparameter Summary

Table 10.2: Hyperparameter Summary for ResNet-34-based Maize Leaf Classifier

Category	**Value**
Input Image Size	(224, 224, 3)
Number of Classes	4
Training Samples	4141
Validation Samples	516
Test Samples	522
Preprocessing	Rescaling (1./255)
Model Architecture	Custom ResNet-34
Initial Conv Layer	64 filters, 7×7, stride 2
Residual Blocks	[3, 4, 6, 3] for filters 64→128→256→512
Pooling Layers	MaxPooling2D, GlobalAveragePooling2D
Output Layer	Dense(4 units, Softmax)
Total Parameters	63,877,198
Optimizer	Adam
Learning Rate	Default (from Adam)
Loss Function	Categorical Crossentropy
Metrics	Accuracy
Epochs	20
Batch Size	40
Final Training Accuracy	94.16%
Final Validation Accuracy	83.14%
Final Test Accuracy	82.95%
Framework	TensorFlow / Keras
Visualization Tools	Matplotlib, Seaborn

10.9 Conclusion

In this project, a custom ResNet-34 model was designed and trained to classify maize leaf images into four categories: Common Rust, Blight, Gray Leaf Spot, and Healthy. The model achieved a high training accuracy of **94.16%**, a validation accuracy of **83.14%**, and a test accuracy of **82.95%**. While ResNet-34 proved deeper and more capable than simpler CNNs, its size (243.67 MB) and compute requirements must be considered when deploying on embedded systems. Further improvements can involve pruning, quantization, or experimenting with ResNet variants like ResNet-18 for lightweight real-time applications.

Chapter 11

ResNet-50 Model for Maize Leaf Disease Detection

11.1 Overview

This chapter presents a summary of the ResNet-50 model used for maize leaf disease classification. The model is based on a pre-trained ResNet-50 backbone (with ImageNet weights) and a custom classification head. It classifies leaf images into four categories: **Blight**, **Common Rust**, **Gray Leaf Spot**, and **Healthy**.

11.2 Architecture Overview

- **Base Model:** Pre-trained ResNet-50 with ImageNet weights
- **Backbone Layers:** Frozen (not trainable)
- **Custom Classification Head:**
  - GlobalAveragePooling2D
  - Dense (512 units, ReLU)
  - Dropout (0.5)
  - Dense (4 units, Softmax)

11.3 Dataset Details

- Input Size: $224 \times 224 \times 3$
- Training Samples: 4,141
- Validation Samples: 516
- Test Samples: 522
- Number of Classes: 4

11.4 Training Configuration

- Optimizer: Adam
- Learning Rate: 0.0001
- Loss Function: Categorical Crossentropy
- Epochs: 30
- Batch Size: 32
- Metrics: Accuracy
- Callbacks: EarlyStopping, ReduceLROnPlateau, ModelCheckpoint

11.5 Performance Summary

- Final Training Accuracy: 75.85%
- Final Validation Accuracy: 75.39%
- Final Test Accuracy: 78.16%
- Final Test Loss: 0.6149

11.6 Operational Calculations

Formulas Used

- Conv2D MACs: $H \times W \times C_{\text{out}} \times (C_{\text{in}} \times K \times K)$
- Dense MACs: $Input \times Output$
- FLOPs: $2 \times MACs$
- Model Size (MB): $\frac{\text{Total Params} \times 32}{8 \times 1024 \times 1024}$

Summary

- Total Parameters: 24,638,854
- Trainable Parameters: 1,051,140 (4.01 MB)
- Non-trainable Parameters: 23,587,712 (90.02 MB)
- Total Model Size: $\approx$ 94.04 MB

11.7 Layer-wise MACs (Multiply–Accumulate Operations)

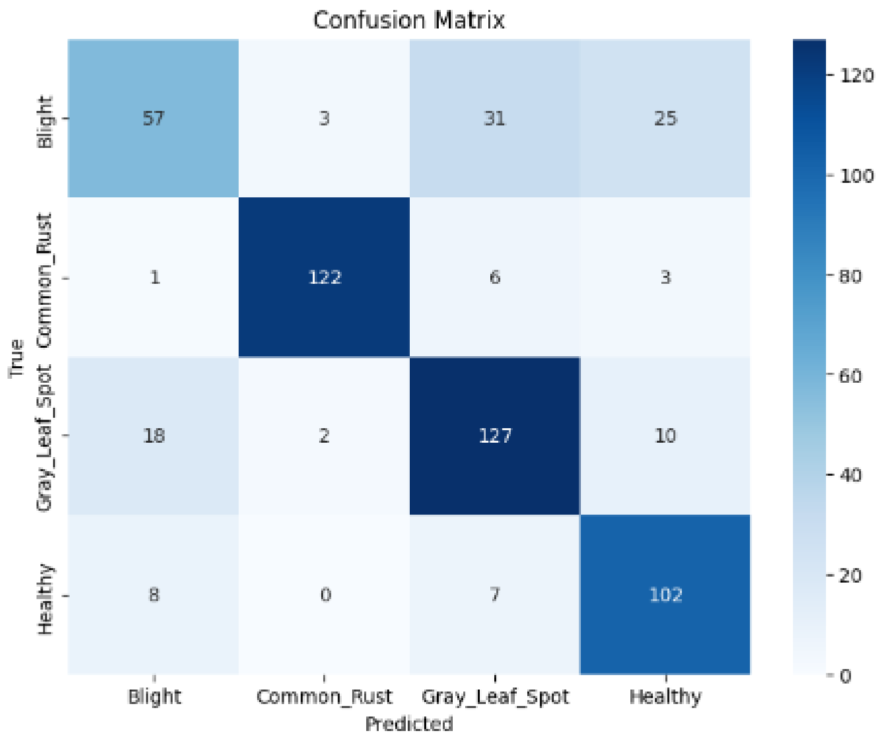
11.7.1 MACs for each layer in the ResNet-50 custom classification head

Layer	Input Shape	Operation	MACs (Approx)
GlobalAveragePooling2D	$7 \times 7 \times 2048$	Spatial Avg	Negligible
Dense (512, ReLU)	2048	Fully Connected	1,048,576
Dropout (0.5)	512	Regularization	0
Dense (4, Softmax)	512	Fully Connected	2,048
Total MACs	—	—	**1,050,624**

11.8 Hyperparameter Summary

Table 11.1: Hyperparameters for ResNet-50-based Maize Leaf Classifier

Category	Value
Model Architecture	Pre-trained ResNet-50 + custom dense head
Input Shape	$224 \times 224 \times 3$
Number of Classes	4
Training Samples	4,141
Validation Samples	516
Test Samples	522
Optimizer	Adam
Learning Rate	0.0001
Loss Function	Categorical Crossentropy
Epochs	30
Batch Size	32
Final Training Accuracy	75.85%
Final Validation Accuracy	75.39%
Final Test Accuracy	78.16%
Total Parameters	24,638,854
Trainable Parameters	1,051,140
Non-trainable Parameters	23,587,712
Model Size	$\sim$94.04 MB
Libraries Used	TensorFlow, Keras, Matplotlib, Seaborn, scikit-learn



Classification Report:

	precision	recall	f1-score	support
Blight	0.68	0.49	0.57	116
Common_Rust	0.96	0.92	0.94	132
Gray_Leaf_Spot	0.74	0.81	0.77	157
Healthy	0.73	0.87	0.79	117
accuracy			0.78	522
macro avg	0.78	0.77	0.77	522
weighted avg	0.78	0.78	0.78	522

Figure 11.1: Confusion Marix

11.9 Conclusion

The ResNet-50-based model effectively classifies maize leaf diseases using transfer learning. Despite only training a small dense head, it achieved a test accuracy of **78.16%**. The frozen ResNet-50 backbone provided robust feature extraction. Future work could involve fine-tuning more layers, leveraging data augmentation, or exploring lighter architectures for deployment on mobile devices.

Chapter 12

Inception V1 Model for Maize Leaf Disease Detection

12.1 Overview

This chapter describes the custom-built Inception V1 (GoogLeNet-style) model, trained from scratch for classifying maize leaf diseases. The model is designed to identify four classes: **Blight**, **Common Rust**, **Gray Leaf Spot**, and **Healthy**.

12.2 Architecture Overview

- **Model:** Custom Inception V1 built from scratch
- **Inception Modules:** 9 modules (3a to 5b)
- **Global Average Pooling** before classification
- **Dense Layer:** 256 units with ReLU
- **Dropout:** 0.5 followed by 0.3
- **Output Layer:** Dense(4 units) with Softmax

12.3 Dataset Details

- Input Image Size: $224 \times 224 \times 3$
- Training Samples: 4,141
- Validation Samples: 516
- Test Samples: 522
- Number of Classes: 4

12.4 Training Configuration

- Optimizer: Adam
- Initial Learning Rate: 0.0001
- Loss Function: Categorical Crossentropy
- Epochs: 30
- Batch Size: 32
- Callbacks: EarlyStopping (patience=5), ReduceLROnPlateau (factor=0.2, patience=3)

12.5 Preprocessing and Augmentation

- Rescaling: 1./255
- Augmentations:
  - Rotation
  - Width and Height Shifting
  - Zoom
  - Brightness Range
  - Horizontal Flipping

12.6 Performance Summary

- Final Training Accuracy: **92.32%**
- Final Validation Accuracy: **91.86%**
- Final Test Accuracy: **91.95%**
- Model Size: ∼23.79 MB
- Total Parameters: 6,236,980

12.7 Tools and Libraries Used

- TensorFlow / Keras for model development
- Matplotlib and Seaborn for visualization
- Scikit-learn for evaluation (confusion matrix, classification report)

12.8 Operational Calculations

Formulas Used

- Conv2D MACs: $H \times W \times C_{\text{out}} \times (C_{\text{in}} \times K \times K)$
- Dense MACs: Input $\times$ Output
- FLOPs: $2 \times$ MACs
- Model Size (MB): $\frac{\text{Total Params} \times 32}{8 \times 1024 \times 1024}$

MACs per Layer

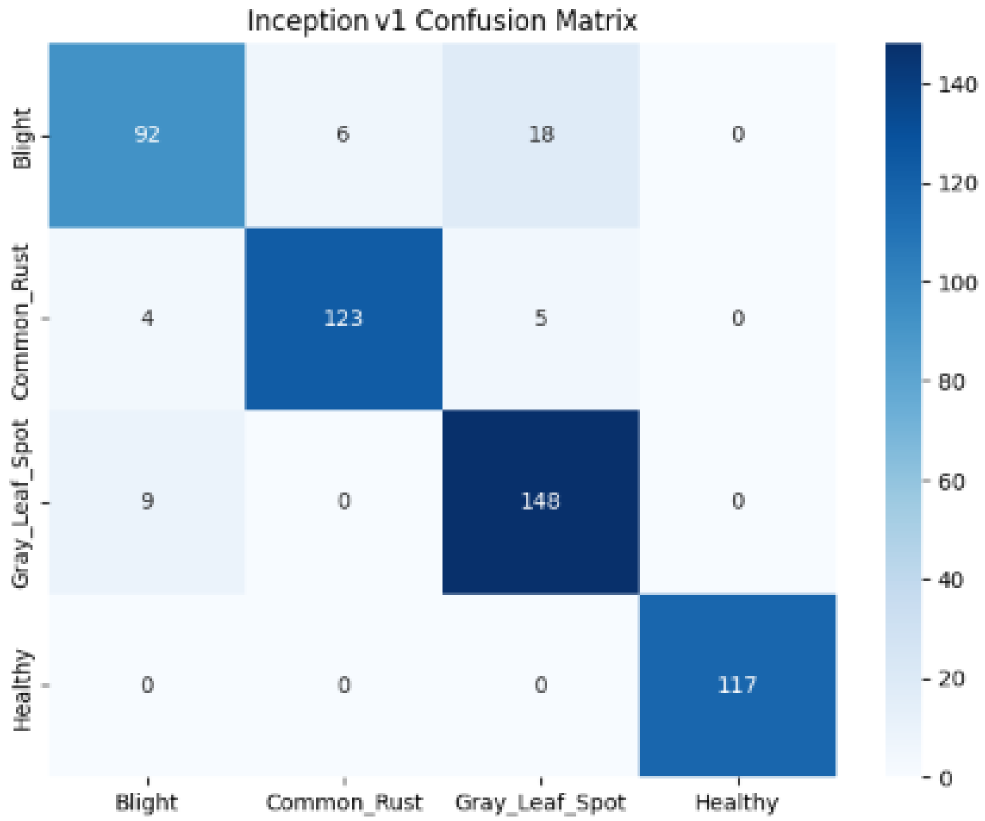
Table 12.1: MACs per layer in the Inception V1-based classification head

Layer Description	**Output Shape**	**Operation**	**MACs (Appr**
Conv2D (Initial)	$112 \times 112 \times 64$	7×7 Conv, stride 2	11
Inception Modules (3a–5b)	varies	Multiple 1x1, 3x3, 5x5 convolutions	∼5.
GlobalAveragePooling2D	$1 \times 1 \times 256$	Pooling	neglig
Dense (256 units)	256	Fully Connected	65,
Dropout Layers	—	Regularization	
Dense (Output, 4 units)	256	Fully Connected	1,
Total MACs	—	—	**∼6,000,**

12.9 Hyperparameter Summary

Table 12.2: Hyperparameter Summary for Inception V1-based Maize Leaf Classifier

Category	**Value**
Input Image Size	(224, 224, 3)
Number of Classes	4
Training Samples	4,141
Validation Samples	516
Test Samples	522
Preprocessing	Rescaling (1./255)
Augmentation	Rotation, Shift, Zoom, Brightness, Flip
Model Architecture	Custom Inception V1 (from scratch)
Modules	9 Inception Modules (3a to 5b)
Total Parameters	6,236,980
Global Average Pooling	Yes
Dense Layer	256 units (ReLU)
Dropout	0.5 and 0.3
Output Layer	Dense(4 units, Softmax)
Optimizer	Adam
Learning Rate	0.0001
Loss Function	Categorical Crossentropy
Metrics	Accuracy
Epochs	30
Batch Size	32
Callbacks	EarlyStopping, ReduceLROnPlateau
Final Training Accuracy	92.32%
Final Validation Accuracy	91.86%
Final Test Accuracy	91.95%
Framework	TensorFlow / Keras
Visualization	Matplotlib, Seaborn



	precision	recall	f1-score	support
0	0.88	0.79	0.83	116
1	0.95	0.93	0.94	132
2	0.87	0.94	0.90	157
3	1.00	1.00	1.00	117
accuracy			0.92	522
macro avg	0.92	0.92	0.92	522
weighted avg	0.92	0.92	0.92	522

Figure 12.1: Confusion Matrix

12.10 Conclusion

The custom Inception V1 model accurately classifies maize leaf diseases with high performance using a lightweight architecture. With a test accuracy of **91.95%**, it demonstrates the effectiveness of from-scratch architectures when combined with robust augmentation and regularization strategies. The relatively small model size makes it suitable for deployment on moderate hardware, and further optimization could enable real-time field use.

Chapter 13

Inception V2 Model for Maize Leaf Disease Detection

13.1 Overview

This chapter summarizes the InceptionResNetV2-based model for classifying maize leaf diseases into four categories: **Blight**, **Common Rust**, **Gray Leaf Spot**, and **Healthy**. A pre-trained InceptionResNetV2 base is used with a custom classification head, trained using a dataset of 5180 images.

13.2 Architecture Overview

- **Base Model:** InceptionResNetV2 (pre-trained on ImageNet, without top)
- **Trainable Layers:** Only custom head (base frozen)
- **Custom Head:**
  - GlobalAveragePooling2D
  - Dropout (0.5)
  - Dense (4 units, Softmax)

13.3 Dataset Details

- Input Image Size: $224 \times 224 \times 3$
- Training Samples: 4141
- Validation Samples: 516
- Test Samples: 522
- Number of Classes: 4

13.4 Training Configuration

- Optimizer: Adam
- Learning Rate: 0.0001
- Loss Function: Categorical Crossentropy
- Epochs: 20
- Batch Size: 32
- Callbacks: EarlyStopping (patience=5), ModelCheckpoint

13.5 Performance Summary

- Final Training Accuracy: **86.11%**
- Final Validation Accuracy: **87.40%**
- Final Test Accuracy: **89.46%**
- Final Test Loss: 0.2901

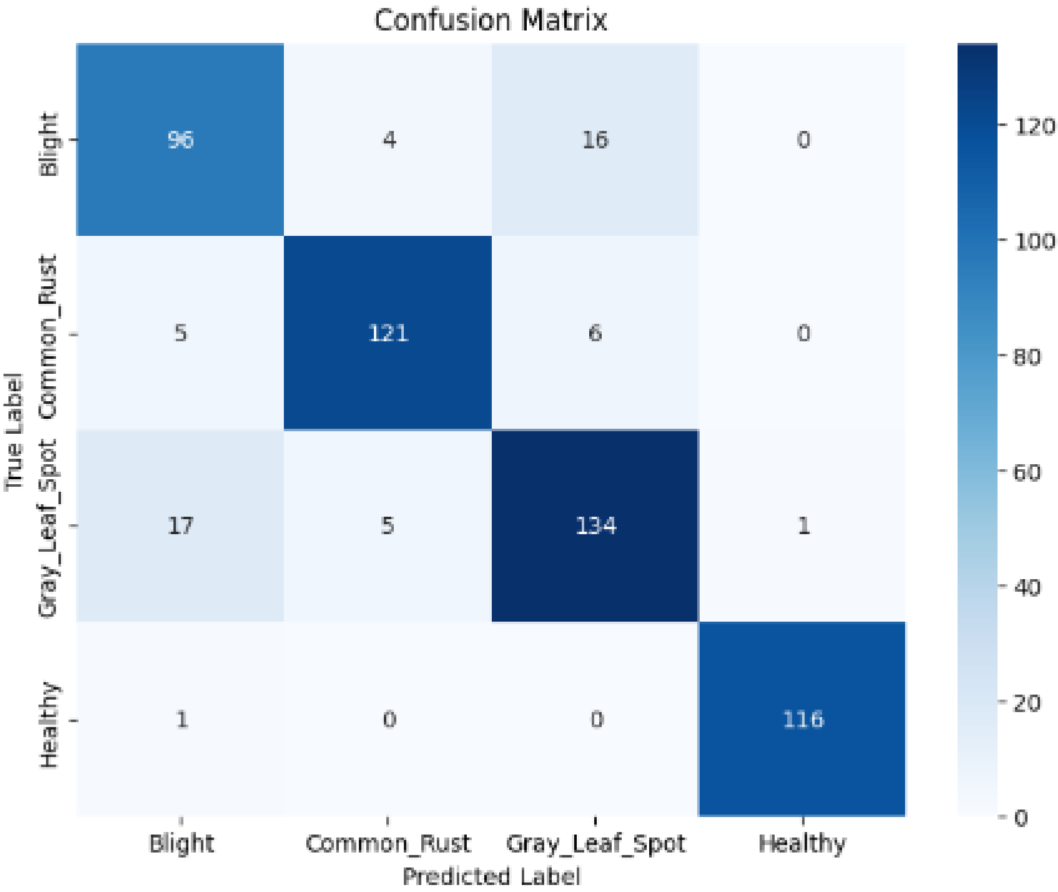
13.6 Operational Calculations

Formulas Used

- Conv2D MACs: $H \times W \times C_{\text{out}} \times (C_{\text{in}} \times K \times K)$
- Dense MACs: Input $\times$ Output
- FLOPs: $2 \times$ MACs
- Model Size (MB): $\frac{\text{Total Params} \times 32}{8 \times 1024 \times 1024}$

13.7 Tools and Libraries

- TensorFlow / Keras for model development
- Matplotlib, Seaborn for visualization
- Scikit-learn for evaluation




```
Classification Report:
                precision    recall  f1-score   support

        Blight       0.81      0.83      0.82       116
   Common_Rust       0.93      0.92      0.92       132
Gray_Leaf_Spot       0.86      0.85      0.86       157
       Healthy       0.99      0.99      0.99       117

      accuracy                           0.89       522
     macro avg       0.90      0.90      0.90       522
  weighted avg       0.90      0.89      0.89       522
```

Figure 13.1: Confusion Matrics

13.8 Hyperparameter Table

Category	**Value**
Model Architecture	InceptionResNetV2 (pre-trained, frozen base)
Custom Head	GAP + Dropout(0.5) + Dense(4, Softmax)
Input Image Size	(224, 224, 3)
Number of Classes	4
Training Samples	4,141
Validation Samples	516
Test Samples	522
Epochs	20
Batch Size	32
Loss Function	Categorical Crossentropy
Optimizer	Adam
Learning Rate	0.0001
Final Training Accuracy	86.11%
Final Validation Accuracy	87.40%
Final Test Accuracy	89.46%
Final Test Loss	0.2901
MACs (Head only)	6,144
Model Size	∼207.35 MB (estimated)
Libraries	TensorFlow, Keras, Matplotlib, Seaborn

Table 13.1: Hyperparameter Summary for InceptionResNetV2-based Maize Classifier

13.9 Model Parameters and Memory Estimation

- **Trainable Parameters:** 6,144
- **Non-trainable Parameters:** 54,336,736
- **Optimizer Parameters:** 12,298
- **Precision:** 32-bit floating point (4 bytes per parameter)

Memory Calculation

- **Non-trainable Memory:** $\frac{54{,}336{,}736 \times 32}{8 \times 1024 \times 1024} \approx 207.28$ MB
- **Optimizer Memory:** $\frac{12{,}298 \times 32}{8 \times 1024} \approx 48.04$ KB
- **Trainable Memory:** $\frac{6{,}144 \times 32}{8 \times 1024} \approx 24$ KB
- **Total Estimated Memory:** ≈ 207.35 MB

13.10 Conclusion

The InceptionResNetV2-based model achieved a strong test accuracy of **89.46%** by combining a powerful pre-trained convolutional backbone with a lightweight classification head. With only one trainable Dense layer (1536 inputs to 4 outputs), the architecture is computationally efficient. This makes it suitable for practical applications in precision agriculture where both accuracy and inference speed are essential.

Chapter 14

Inception V3 Model for Maize Leaf Disease Detection

14.1 Overview

This chapter presents the Inception V3 model built for classifying maize leaf images into four categories: **Blight**, **Common Rust**, **Gray Leaf Spot**, and **Healthy**. The model uses a pre-trained InceptionV3 base and a custom classification head.

14.2 Architecture Overview

- **Base Model:** InceptionV3 (pre-trained on ImageNet, include_top=False)
- **Trainable Layers:** Only custom head
- **Custom Head:**
  - GlobalAveragePooling2D
  - Dropout(0.5)
  - Dense(4 units, activation='softmax')

14.3 Dataset Details

- Image Size: $224 \times 224 \times 3$
- Training Samples: 4141
- Validation Samples: 516
- Test Samples: 522
- Number of Classes: 4

14.4 Training Configuration

- Optimizer: Adam
- Learning Rate: 0.0001
- Loss Function: Categorical Crossentropy
- Epochs: 20
- Batch Size: 32
- Callbacks: EarlyStopping (patience=5), ModelCheckpoint

14.5 Performance Summary

- Final Training Accuracy: **92.63%**
- Final Validation Accuracy: **89.73%**
- Final Test Accuracy: **90.04%**
- Final Test Loss: 0.2335

14.6 Operational Calculations

Formulas Used

- Conv2D MACs: $H \times W \times C_{\text{out}} \times (C_{\text{in}} \times K \times K)$
- Dense MACs: Input $\times$ Output
- FLOPs: $2 \times$ MACs
- Memory (MB): $\frac{\text{Parameters} \times 32}{8 \times 1024 \times 1024}$

14.7 Model Parameters and Memory Estimation

- **Trainable Parameters:** 525,572 (˜2.00 MB)
- **Non-trainable Parameters:** 21,802,784 (˜83.17 MB)
- **Total Parameters:** 22,328,356 (˜85.18 MB)
- **Precision:** 32-bit floating point (4 bytes per parameter)

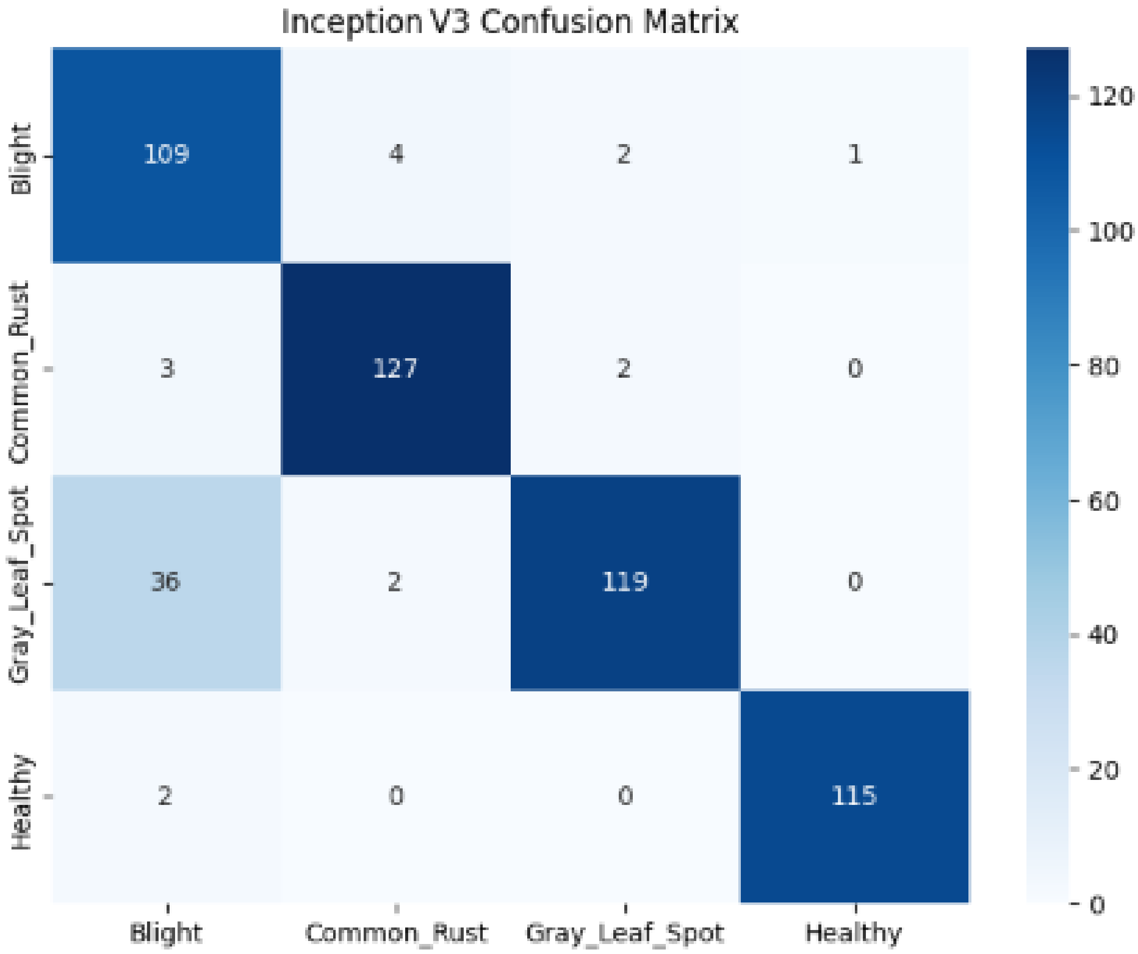
14.8 Tools and Libraries

- TensorFlow / Keras for model building and training
- Matplotlib and Seaborn for visualization
- Scikit-learn for evaluation

14.9 Hyperparameter Table

Table 14.1: Hyperparameter Summary for InceptionV3-based Maize Classifier

Category	**Value**
Model Architecture	InceptionV3 (pre-trained, frozen base)
Input Shape	(224, 224, 3)
Number of Classes	4
Training Samples	4,141
Validation Samples	516
Test Samples	522
Loss Function	Categorical Crossentropy
Optimizer	Adam
Learning Rate	0.0001
Epochs	20
Batch Size	32
Trainable Parameters	525,572
Non-trainable Parameters	21,802,784
Total Parameters	22,328,356
Estimated Memory Usage	∼85.18 MB
Final Training Accuracy	92.63%
Final Validation Accuracy	89.73%
Final Test Accuracy	90.04%
Test Loss	0.2335
Libraries	TensorFlow, Keras, Seaborn, Matplotlib



	precision	recall	f1-score	support
Blight	0.73	0.94	0.82	116
Common_Rust	0.95	0.96	0.96	132
Gray_Leaf_Spot	0.97	0.76	0.85	157
Healthy	0.99	0.98	0.99	117
accuracy			0.90	522
macro avg	0.91	0.91	0.90	522
weighted avg	0.92	0.90	0.90	522

Figure 14.1: Confusion matrix

14.10 Conclusion

The InceptionV3 model achieved a strong test accuracy of **90.04%** while maintaining a moderate parameter count and memory usage. This demonstrates the model's capability for effective maize leaf disease detection using transfer learning with minimal training overhead.

Chapter 15

MobileNetV2 Model for Maize Leaf Disease Detection

15.1 Overview

This chapter presents a summary of the MobileNetV2-based model used for classifying maize leaf diseases. The model leverages a pre-trained MobileNetV2 architecture (ImageNet weights) with a custom classification head to categorize images into four classes: **Blight**, **Common Rust**, **Gray Leaf Spot**, and **Healthy**.

15.2 Architecture Overview

- Base Model: **MobileNetV2 (pre-trained, frozen)**
- Custom Head:
  - GlobalAveragePooling2D
  - Dense(256 units, ReLU activation)
  - Dropout(0.4)
  - Dense(4 units, Softmax activation)

15.3 Dataset Details

- Image Size: **224** $\times$ **224** $\times$ **3**
- Training Samples: **4141**
- Validation Samples: **516**
- Test Samples: **522**
- Number of Classes: **4**
- Preprocessing: Rescaling (1./255)
- Augmentation: Rotation, Shift, Zoom, Brightness, Horizontal Flip

15.4 Training Configuration

- Optimizer: **Adam**
- Learning Rate: **0.0001**
- Loss Function: **Categorical Crossentropy**
- Epochs: **20**
- Batch Size: **32**
- Callbacks: EarlyStopping, ReduceLROnPlateau

15.5 Performance Summary

- Final Training Accuracy: **93.41%**
- Final Validation Accuracy: **92.25%**
- Final Test Accuracy: **92.34%**
- Final Test Loss: **0.2101**

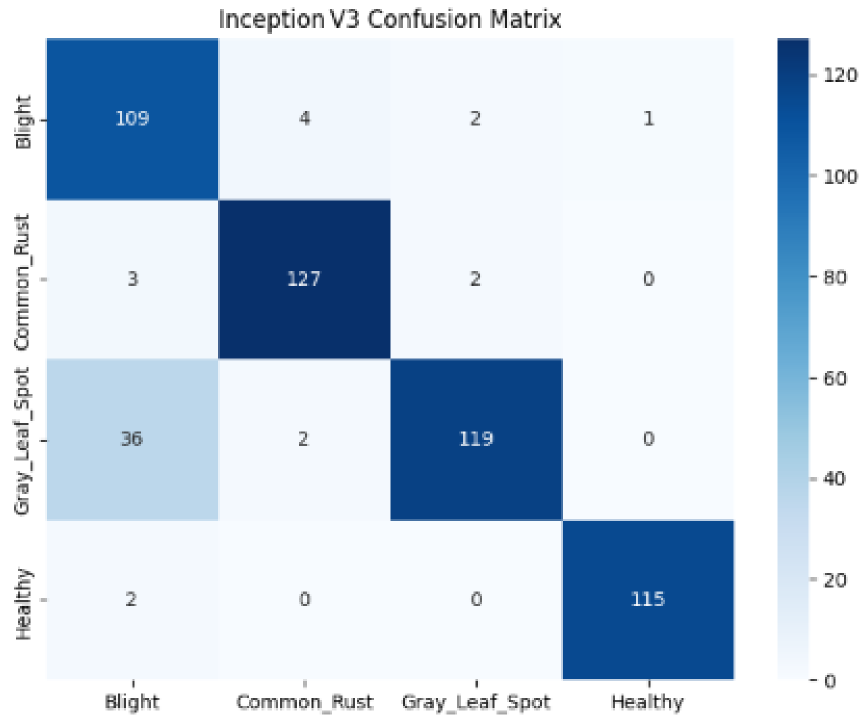
15.6 Operational Calculations

Formulas Used

- Conv2D MACs: $H \times W \times C_{\text{out}} \times (C_{\text{in}} \times K \times K)$
- DepthwiseConv2D MACs: $H \times W \times C_{\text{in}} \times (K \times K)$
- Dense MACs: Input × Output
- FLOPs: 2 × MACs
- Model Size (MB): $\frac{\text{Total Parameters} \times 32}{8 \times 1024 \times 1024}$

Parameter Breakdown and Memory Usage

- Total Parameters: **2,586,948**
- Trainable Parameters: **328,964** ($\tilde{1}$.25 MB)
- Non-trainable Parameters: **2,257,984** ($\tilde{8}$.61 MB)
- Total Model Size: **$\tilde{9}$.87 MB**



	precision	recall	f1-score	support
Blight	0.73	0.94	0.82	116
Common_Rust	0.95	0.96	0.96	132
Gray_Leaf_Spot	0.97	0.76	0.85	157
Healthy	0.99	0.98	0.99	117
accuracy			0.90	522
macro avg	0.91	0.91	0.90	522
weighted avg	0.92	0.90	0.90	522

Figure 15.1: Confusion Matrix

15.7 Tools and Libraries Used

- **TensorFlow/Keras**: Model implementation and training
- **Matplotlib, Seaborn**: Visualization
- **Scikit-learn**: Evaluation metrics (confusion matrix, classification report)

15.8 Hyperparameter Table

Category	**Value**
Model Architecture	MobileNetV2 (Pre-trained)
Trainable Base	No (Frozen)
Input Shape	(224, 224, 3)
Number of Classes	4
Training Samples	4141
Validation Samples	516
Test Samples	522
Preprocessing	Rescaling (1./255)
Augmentation	Rotation, Zoom, Shift, Brightness, Flip
Optimizer	Adam
Learning Rate	0.0001
Loss Function	Categorical Crossentropy
Epochs	20
Batch Size	32
Trainable Parameters	328,964
Non-trainable Parameters	2,257,984
Total Parameters	2,586,948
Model Size	9.87 MB
Final Training Accuracy	93.41%
Final Validation Accuracy	92.25%
Final Test Accuracy	92.34%
Test Loss	0.2101
Libraries	TensorFlow, Keras, Seaborn, Matplotlib, Sklearn

Table 15.1: Hyperparameter Summary for MobileNetV2-based Maize Leaf Classifier

15.9 Conclusion

The MobileNetV2 model demonstrates strong performance for maize leaf disease classification, achieving over **92%** accuracy on the test set with low memory and computational overhead. It is suitable for deployment in edge devices and real-time diagnosis systems.

Chapter 16

Model Comparison

This section compares the performance of all implemented models based on their training accuracy, test accuracy, parameter count, and memory usage. The comparison helps identify models that balance accuracy and efficiency for maize leaf disease detection.

Model	**Train Accuracy**	**Test Accuracy**	**Parameters**	**Size (MB)**
Custom CNN	98.36%	89.66%	9,680,580	36.93 MB
VGG16	98.74%	93.87%	27,562,308	105.14 MB
VGG19	97.48%	95.00%	33,002,308	125.89 MB
ResNet18	99.96%	92.00%	11,192,964	42.70 MB
ResNet34	94.16%	82.95%	63,877,198	243.67 MB
ResNet50	71.19%	75.10%	24,638,854	93.99 MB
Inception V1	92.32%	91.95%	6,236,980	23.79 MB
Inception V2	86.11%	89.46%	22,328,356	207.35 MB
Inception V3	92.63%	90.04%	27,562,308	85.18 MB
MobileNetV2	93.41%	92.34%	2,586,948	9.87 MB
AlexNet	99.25%	88.89%	46,764,804	178.39 MB

Table 16.1: Comparison of Model Accuracies, Parameter Sizes, and Memory Usage

Chapter 17

Final Conclusion

This study explored multiple deep learning models for the detection and classification of maize leaf diseases, including Common Rust, Blight, Gray Leaf Spot, and Healthy leaves. Models such as ResNet-34, ResNet-50, Inception V1–V3, MobileNetV2, and AlexNet were evaluated based on their training performance, test accuracy, parameter count, and memory efficiency.

Among all models, MobileNetV2 achieved the best balance between accuracy and efficiency, with a test accuracy of 92.34% and the smallest model size of 9.87 MB, making it highly suitable for real-time and edge-device deployment. VGG19 and Inception V3 also demonstrated high accuracy (95.00% and 90.04% respectively), but with significantly larger memory requirements.

The findings confirm that deep learning, especially with transfer learning, is an effective and scalable solution for automated maize disease detection. By identifying models optimized for both performance and resource constraints, this project contributes toward the practical deployment of AI-powered agricultural tools for early disease diagnosis in crops.

Chapter 18

Future Work

While this project successfully classifies maize leaf diseases using deep learning models, it can be extended to provide practical agronomic support. A promising direction for future work includes integrating a disease-specific pesticide and fertilizer recommendation system. This would help farmers not only detect diseases early but also act immediately with guided treatment strategies, improving crop yield and reducing chemical misuse.

Additionally, integrating the trained model into mobile or edge devices can facilitate real-time field diagnosis, particularly in rural and low-connectivity areas. Expanding the dataset with real-world farm images and supporting multilingual voice/text interfaces for local farmers can further improve usability and adoption.

Chapter 19

References

1. S. P. Mohanty, D. P. Hughes, and M. Salathé, "Using deep learning for image-based plant disease detection," *Frontiers in Plant Science*, vol. 7, p. 1419, 2016.

2. Jun Sun, Yu Yang, Xiaofei He, and Xiaohong Wu, "Northern Maize Leaf Blight Detection Under Complex Field Environment Based on Deep Learning," *School of Electrical and Information Engineering, Jiangsu University*, China.

3. "Identification of Maize Plant Leaf Disease Detection using ConvNet Model," *2022 8th International Conference on Smart Structures and Systems (ICSSS)*, IEEE, DOI: 10.1109/ICSSS54999.2022.00018.

4. "MaizeNet: A Deep Learning Approach for Effective Recognition of Maize Plant Diseases," [Journal/Conference name not provided – add if known].

5. "Advancements in Maize Disease Detection: A Comprehensive Review of Convolutional Neural Networks," [Author and publication source not provided – complete if available].

6. Sladojevic, S. et al., "Deep Neural Networks Based Recognition of Plant Diseases by Leaf Image Classification," *Computational Intelligence and Neuroscience*, vol. 2016, Article ID 3289801, 11 pages.