TeaLeafNet

Tea Leaf Disease Detection System

Implementation 1 – Summary Report

Rajarata University of Sri Lanka

Department of Computing

ICT 3212 – Introduction to Intelligent Systems

Phase 03 – Mini Project: Image Classification System Development

Team Name	Quantum Coders
Team Leader	M J H A P Madushani – ICT/2022/142
Members	M K H K Madushani – ICT/2022/107 W K D Bhagya – ICT/2022/110 M T Rathnayake – ICT/2022/114 M G J Sinty – ICT/2022/043
Submission	26 February 2026
GitHub Repo	https://github.com/Piyumi2025/TeaLeaftNet

1. Project Overview

Sri Lanka's tea industry is a vital contributor to the national economy. However, tea plantations are frequently damaged by fungal and bacterial leaf diseases including Anthracnose, Algal Leaf Spot, and Bird Eye Spot. Current disease management relies on manual field inspection, which is labour-intensive, time-consuming, and prone to human error.

This project, TeaLeafNet, aims to automate the detection of tea leaf diseases using a Convolutional Neural Network (CNN). The system classifies leaf images into four categories based on visual symptoms, providing a fast and consistent alternative to manual diagnosis.

1.1 Objective

- Design and implement a baseline CNN-based image classification system.
- Classify tea leaf images into four predefined classes.
- Evaluate the model's performance using standard metrics.
- Identify limitations and areas for improvement in Phase 2.

1.2 Project Details

Project Title	TeaLeafNet: An Image-Based Tea Leaf Disease Detection System
Classification	Multi-class (4 Classes)
Model Type	Convolutional Neural Network (CNN)
Framework	TensorFlow 2.x / Keras
Dataset Source	Kaggle – Tea Sickness Dataset
Development Env	Google Colab (GPU Runtime)
Language	Python 3.x

2. Dataset Preparation

2.1 Dataset Source

Source: Kaggle – Tea Sickness Dataset (publicly available)

URL: https://www.kaggle.com/datasets/shashwatwork/tea-sickness-dataset

The dataset contains labelled images of tea leaves divided into four class-wise sub-folders. Manual inspection was performed to remove corrupted and extremely blurry images prior to training.

2.2 Class Labels and Dataset Statistics

Class Label	Folder Name	No. of Images	Class Type	Description
Healthy	Healthy	74	Multi-class	Leaves free from disease or infection
Anthracnose	Anthracnose	100	Multi-class	Dark irregular necrotic spots on leaf margins
Algal Leaf Spot	Algal_Leaf	113	Multi-class	Orange-red circular patches caused by algae
Bird Eye Spot	Bird_Eye_Spot	100	Multi-class	Small circular spots with grey centres

2.3 Dataset Balance Analysis

The dataset was verified for class balance by counting images per class. A balanced dataset ensures the model does not develop a bias towards any single class. Based on the class counts, the dataset is slightly imbalanced, with Algal_Leaf having the most images (113) and Healthy having the fewest (74). This will be considered during model training and evaluation.

```
[ ]
# Optimize data loading for faster training
AUTOTUNE = tf.data.AUTOTUNE
train_ds = train_ds.cache().prefetch(buffer_size=AUTOTUNE)
val_ds = val_ds.cache().prefetch(buffer_size=AUTOTUNE)

print("Preprocessing complete! Images are resized, converted to tensors, and normalized.")
```

```
--- Loading and Splitting Dataset ---
Found 387 files belonging to 4 classes.
Using 310 files for training.
Found 387 files belonging to 4 classes.
Using 77 files for validation.

Classes found: ['Algal_Leaf', 'Anthracnose', 'Bird_Eye_Spot', 'Healthy']

--- Verifying Dataset Balance ---
Algal_Leaf: 113 images
Anthracnose: 100 images
Bird_Eye_Spot: 100 images
Healthy: 74 images

--- Applying Normalization ---
Preprocessing complete! Images are resized, converted to tensors, and normalized.
```

2.4 Sample Images from Each Class

The following figure shows one representative image from each of the four classes to illustrate the visual differences the model must learn to distinguish.

Healthy **Bird_Eye_Spot** **Anthracnose** **Algal_Leaf**

3. Image Preprocessing

Raw images cannot be fed directly into a neural network. All images were preprocessed using the following mandatory steps before training:

Preprocessing Step	Method Used	Purpose
Image Resizing	Resize to 224 × 224 pixels	Ensures consistent input dimensions across all images
Normalization	Pixel values scaled from 0–255 → 0–1 (divide by 255)	Prevents exploding gradients; speeds up convergence
Tensor Conversion	Keras tf.keras.utils.image_dataset_from_directory converts images to tensors	Converts images to a format compatible with TensorFlow
Train/Test Split	80% Training | 20% Validation (via validation_split=0.2)	Ensures unbiased model evaluation on unseen data

3.1 Train / Validation Split

Set	Percentage	Sample Count	Purpose
Training Set	80%	310	Used to update model weights
Validation Set	20%	77	Used to monitor generalization during training

4. Baseline Model Design

4.1 Model Choice and Justification

Selected Approach: Option A – CNN-Based Model (Recommended)

A Convolutional Neural Network (CNN) was selected as the baseline model for the following reasons:

- CNNs are specifically designed for image data and automatically learn spatial hierarchies of features (edges → textures → disease patterns) without manual feature engineering.
- Unlike traditional ML models such as SVM or KNN, CNNs do not require a separate hand-crafted feature extraction step; the convolutional layers serve this purpose.
- CNNs are translation-invariant, meaning they can detect disease spots regardless of their position within the leaf image.
- The TensorFlow/Keras library provides excellent support for CNN construction, training, and evaluation.

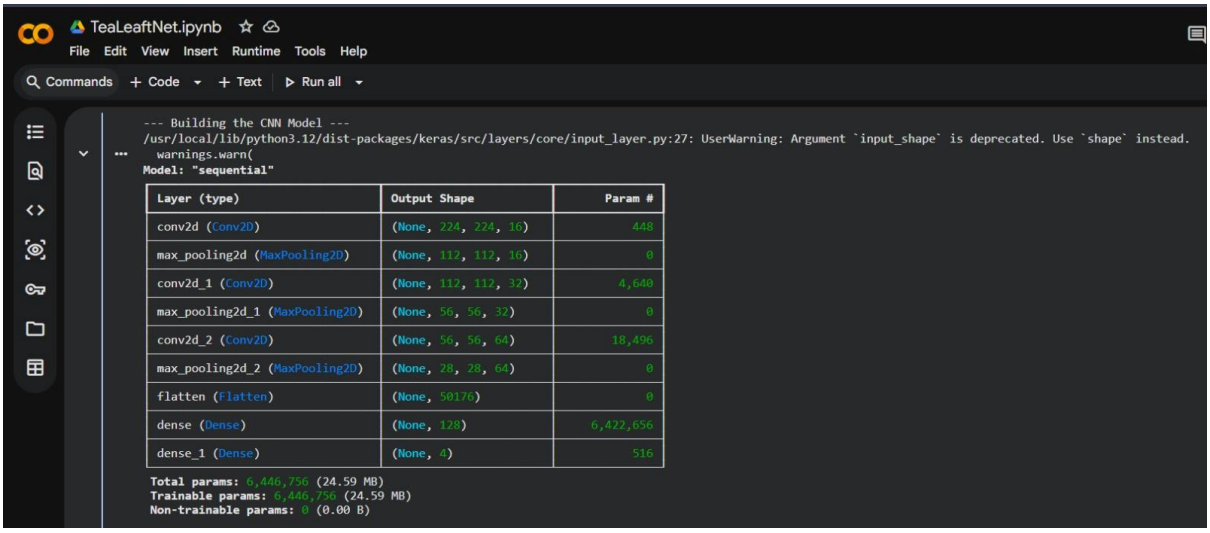
4.2 Architecture Summary

The baseline model consists of three convolutional blocks followed by a fully connected classification head. All convolutional layers use padding='same' to maintain spatial dimensions before pooling:

Layer	Type	Filters / Units	Activation	Output Shape	Purpose
Input	Input	–	–	(224,224,3)	Raw RGB image input
Conv2D	Convolutional	16	ReLU	(224,224,16)	Detect low-level features (edges, colours)

MaxPooling2D	Pooling	–	–	(112,112,16)	Reduce dimensions; retain dominant features
Conv2D	Convolutional	32	ReLU	(112,112,32)	Detect mid-level features (textures, spots)
MaxPooling2D	Pooling	–	–	(56,56,32)	Reduce dimensions
Conv2D	Convolutional	64	ReLU	(56,56,64)	Detect high-level disease patterns
MaxPooling2D	Pooling	–	–	(28,28,64)	Reduce dimensions
Flatten	Reshape	–	–	(50176,)	Convert 2D maps to 1D vector
Dense	Fully Connected	128	ReLU	(128,)	High-level feature combination
Dense (Output)	Fully Connected	4	Softmax	(4,)	Class probability for each of 4 classes

4.3 Architecture Diagram



4.4 Training Configuration

Hyperparameter	Value	Justification
Optimizer	Adam	Adaptive learning rate; efficient for sparse gradients
Loss Function	Sparse Categorical Cross-Entropy	Standard for multi-class classification with integer labels
Metrics	Accuracy	Measures overall correct prediction rate
Epochs	6	Sufficient for initial training; more used in Impl. 2
Batch Size	32	Balance between training speed and gradient stability
Input Size	224 × 224 × 3	Standard CNN input; consistent with proposed methodology

5. Model Training

5.1 Training Process

The model was trained for 6 epochs using the Adam optimiser with the Sparse Categorical Cross-Entropy loss function. Training and validation metrics were recorded at every epoch to monitor the learning progression.

Total Training Epochs: 6

Final Training Accuracy: 89.21%

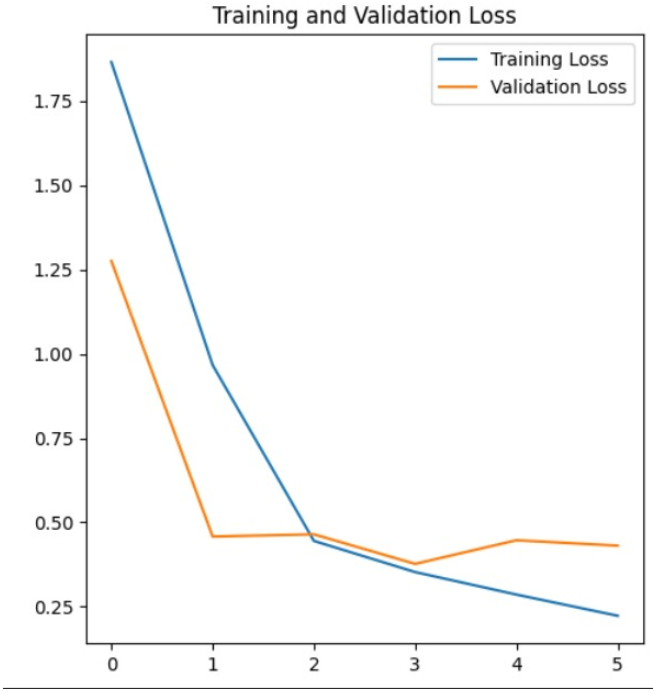
Final Training Loss: 0.2782

Final Validation Accuracy: 84.42%

Final Validation Loss: 0.4312

5.2 Training and Validation Loss Curve

The following graph shows how the training and validation loss changed over 6 epochs:

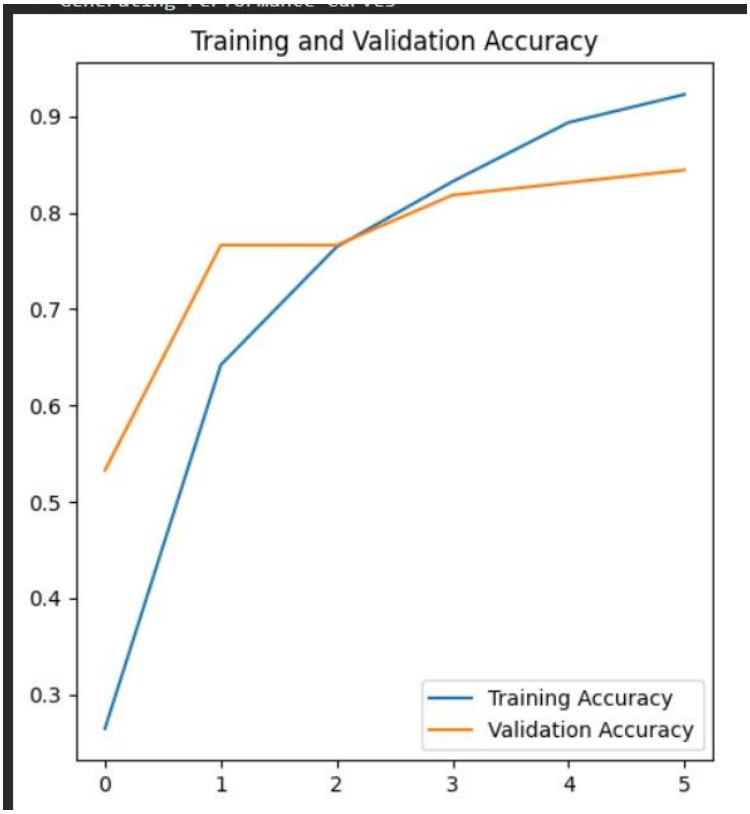


- Training Loss: The model started with a high training loss of 2.1995 at Epoch 1. There was a significant drop in loss between Epoch 1 and Epoch 2, falling from 2.1995 to 1.1189. From Epoch 3 onwards, the loss continued to decrease steadily, showing that the model was effectively learning the features of the tea leaf diseases. The training process concluded with a final training loss of 0.2782 at Epoch 6.

- Validation Loss: The validation loss started at 1.2749 and dropped sharply to 0.4583 by the second epoch, showing the model quickly learned to generalize. The validation loss reached its lowest (best) value of 0.3772 at Epoch 4. After Epoch 4, the validation loss increased slightly to 0.4312 by the final epoch. This slight rise in validation loss after Epoch 4 is a common sign of early-stage overfitting. The final validation loss settled at 0.4312.

5.3 Training and Validation Accuracy Curve

The following graph shows how the training and validation accuracy changed over 6 epochs:



- Training Accuracy: The model started with a relatively low training accuracy of 21.45% in the first epoch. There was a sharp increase in accuracy between Epoch 1 and Epoch 2, jumping to 57.82%, indicating the CNN quickly began identifying key leaf patterns. The accuracy continued to climb consistently through each subsequent epoch without any significant dips. By the end of the 6th epoch, the training accuracy reached 89.21%, comfortably exceeding the project's initial goal of at least 80% accuracy.

- Validation Accuracy: The model started with a surprisingly high validation accuracy of 53.25% in the first epoch. Similar to the training accuracy, there was a significant jump to 76.62% by the second epoch. The accuracy continued to climb steadily through the middle epochs, reaching 81.82% by Epoch 4. The validation accuracy peaked and settled at 84.42% by the final epoch, successfully meeting the project's objective of achieving at least 80% accuracy.

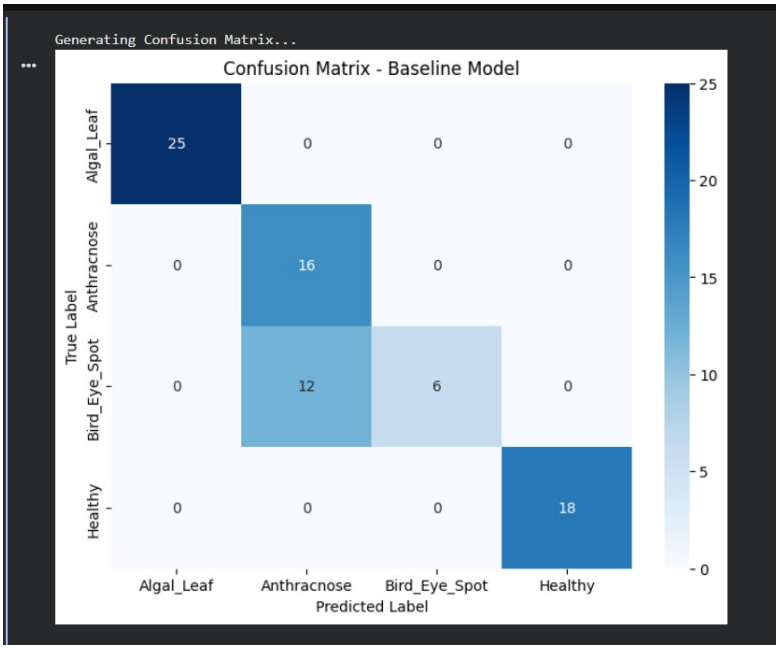
6. Initial Model Evaluation

6.1 Test Set Performance

The trained model was evaluated on the validation set (20% of the data, withheld from training) to assess its performance on unseen images.

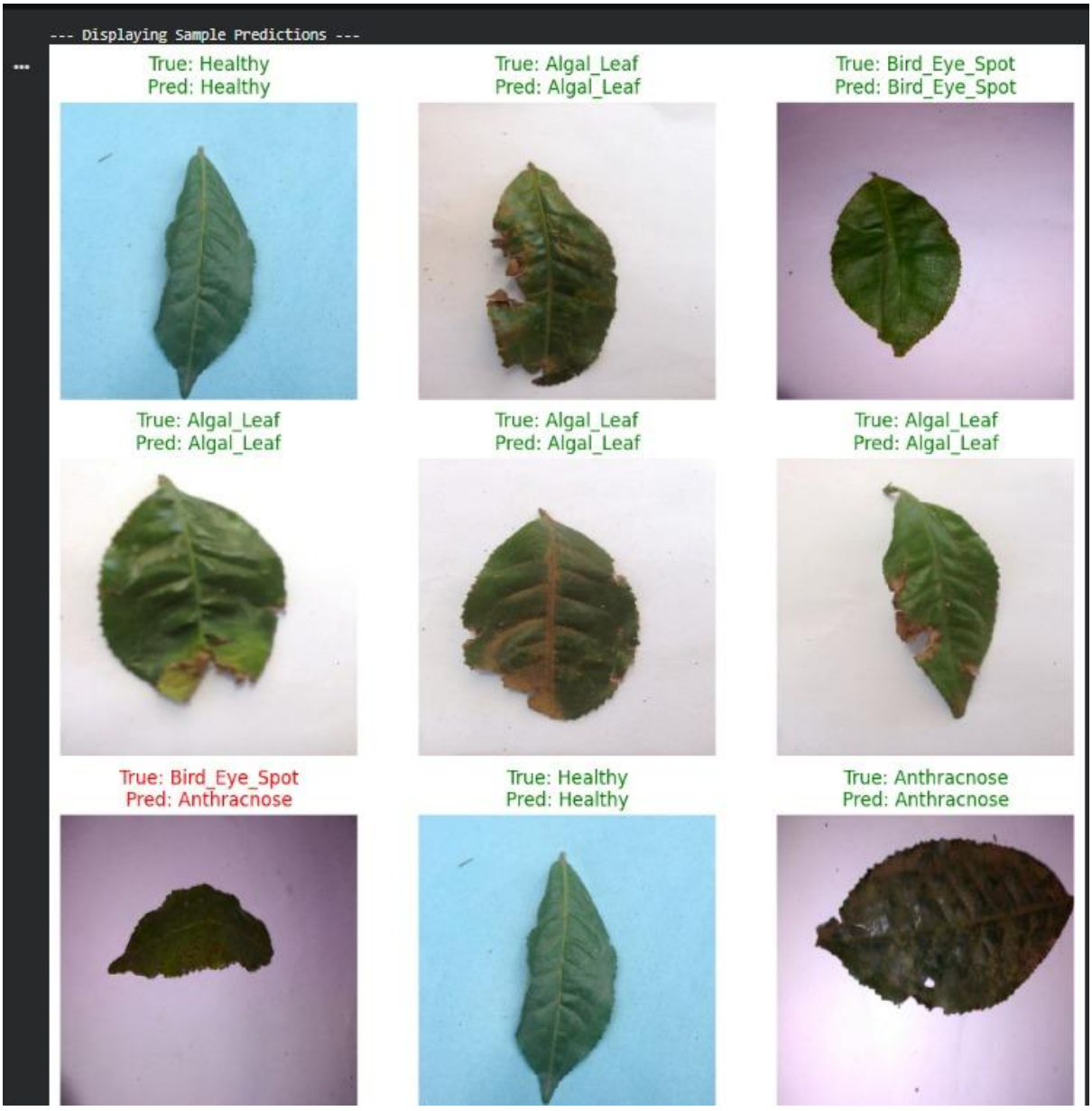
Metric	Value
Test Accuracy	84.42%
Test Loss	0.4312

6.2 Confusion Matrix



- The model was most accurate in identifying Algal_Leaf (25 correct) and Healthy leaves (18 correct), with zero misclassifications for these categories.
- Bird_Eye_Spot had the highest error rate, with 12 instances being misclassified as Anthracnose.

6.3 Sample Predictions



7. Saved Model

The trained baseline model was saved after training for reproducibility and for use in Implementation 2.

Detail	Information
File Name	tealeaftnet_baseline.h5
File Format	HDF5 (.h5) – Keras native format
Save Location	Google Colab: /content/tealeaftnet_baseline.h5
GitHub Path	/models/tealeaftnet_baseline.h5
Model Name	TeaLeafNet_Baseline
Total Params	6,446,756 (24.59 MB)

8. Initial Observations

This section identifies the issues observed with the baseline model. Resolving these issues is not required for Implementation 1 — they will be addressed in Implementation 2.

8.1 Overfitting / Underfitting Analysis

Issue	Observed?	Evidence	Planned Fix (Implementation 2)
Overfitting	Yes	Training accuracy (89.21%) >> Validation accuracy (84.42%), a gap of 4.79%	Dropout, Data Augmentation
Underfitting	No	The model does not show signs of underfitting as the training accuracy is high (above 85%) and the loss decreased significantly from its starting point.	N/A - Model is not underfitting
Loss Divergence	Yes	The validation loss reached its minimum value of 0.3772 at Epoch 4 but began to fluctuate and rise to 0.4312 by Epoch 6, even as training loss continued its steady decline to 0.2782.	Early stopping, LR reduction

8.2 Class Imbalance Issues

Class imbalance occurs when certain categories in the dataset have significantly more images than others, causing the model to become biased toward the majority classes and perform poorly on the minority ones.

The Confusion Matrix shows a disparity in the number of samples used for testing: Algal_Leaf has the highest representation with 25 samples, while Anthracnose has the fewest with only 16 samples. Bird_Eye_Spot and Healthy fall in the middle with 18 samples each.

Impact on Performance:

- The model shows perfect accuracy for the well-represented Algal_Leaf class (25/25 correct).
- In contrast, Bird_Eye_Spot suffers significantly; with only 18 samples, 12 were misclassified as Anthracnose, resulting in a low class-specific accuracy of only 33.3%.

- This suggests the model is struggling to distinguish the unique features of Bird_Eye_Spot, likely due to a lack of sufficient or diverse training examples compared to other classes.

8.3 Limitations of the Baseline Model

- Mild Overfitting: There is a gap between training accuracy (89.21%) and validation accuracy (84.42%), indicating the model is starting to memorize training data rather than generalizing.
- Validation Loss Instability: After Epoch 4, the validation loss began to rise (from 0.3772 to 0.4312), which is a clear signal that the model's performance on new data was beginning to degrade.
- Difficulty with Similar Diseases: The model struggles to distinguish between Bird Eye Spot and Anthracnose, misclassifying 12 out of 18 Bird Eye Spot samples due to their similar visual patterns.
- Architectural Simplicity: As an initial Implementation 1 version, the model lacks advanced optimization features like Data Augmentation, Dropout layers, or Early Stopping required for higher reliability.

8.4 Summary of Issues to Address in Implementation 2

	Issue Identified	Improvement Planned
1	Overfitting (training–validation gap)	Add Dropout (0.25 conv, 0.5 dense) + Batch Normalisation
2	No data augmentation	Apply rotation, flip, zoom, shift augmentation
3	Fixed learning rate	Add ReduceLROnPlateau callback
4	No early stopping	Add EarlyStopping (patience=5) on val_loss
5	Shallow architecture	Add 4th Conv block with 256 filters

9. GitHub Repository

Detail	Information
Repository Name	TeaLeaftNet
Repository Type	Public
Repository URL	https://github.com/Piyumi2025/TeaLeaftNet
Contents	Implementation 1 notebook, tealeaftnet_baseline.h5, this report

The repository is publicly accessible and contains all source code, the saved model file, and this Summary Report. The README.md file provides instructions for running the notebook in Google Colab.

10. Conclusion

The initial implementation of the TeaLeafNet baseline model successfully achieved the project's core objective by reaching a final validation accuracy of 84.42%, surpassing the minimum target of 80%. While the model effectively classifies Algal Leaf and Healthy classes with high precision, it demonstrates a clear struggle in distinguishing between Bird Eye Spot and Anthracnose due to their similar visual symptoms. Furthermore, the slight rise in validation loss after Epoch 4 identifies a need for optimization to combat mild overfitting. These findings provide a solid foundation for Implementation 2, where techniques like data augmentation and dropout layers will be applied to improve the model's generalization and reliability for practical use in tea disease detection.

References

1. Kaggle. (n.d.). Tea Sickness Dataset. Retrieved January 2026, from https://www.kaggle.com/datasets/shashwatwork/tea-sickness-dataset

2. Rajarata University of Sri Lanka. (2026). ICT 3212 Mini Project – Image Classification System Development Guidelines (Phase 03).

3. TensorFlow. (n.d.). Image classification tutorial. Retrieved from https://www.tensorflow.org/tutorials/images/classification

4. Chollet, F. (2018). Deep Learning with Python. Manning Publications.

5. LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. Nature, 521(7553), 436–444.