DEEP LEARNING-BASED PLANT DISEASE DETECTION USING RGB LEAF IMAGES

Submitted by:

HULANGAMUWA R.R.G.W.B.S.C. (2020/E/054)

SUMANARATHNA E.G.T.M.(2020/E/152)

WIJESINGHE L.A.D.P.A. (2020/E/182)

DEPARTMENT OF COMPUTER ENGINEERING

FACULTY OF ENGINEERING

UNIVERSITY OF JAFFNA

[JULY] 2024

INTRODUCTION

Advancements in deep learning technologies have revolutionized various industries, including agriculture, by offering sophisticated tools for enhancing productivity and sustainability. In particular, the application of these technologies in plant disease detection has emerged as a critical area of research and development. The ability to swiftly and accurately identify crop diseases holds immense potential for mitigating economic losses, ensuring food security, and promoting sustainable agricultural practices.

Plant diseases threaten global food production, affecting crop yields, quality, and profitability. Timely detection and management of these diseases are crucial to minimizing losses and optimizing agricultural output. Traditional methods of disease identification often rely on visual inspection by agricultural experts, which can be subjective, time-consuming, and impractical for large-scale farming operations. Here, deep learning presents innovative solutions by automating the process of disease detection, offering speed, accuracy, and scalability.

Early detection of plant diseases enables farmers to implement timely intervention strategies, such as targeted pesticide application or crop management practices, before diseases spread and cause extensive damage. Moreover, precision agriculture techniques leverage data-driven insights to optimize resource allocation, ensuring that treatments are applied only where and when necessary. This approach not only reduces the environmental footprint of agricultural activities but also enhances the efficiency and sustainability of farming practices.

OBJECTIVE

This project aims to conduct a comprehensive comparison and evaluation of two advanced models for plant disease detection:

- **Convolutional Neural Network (CNN):** Known for its ability to extract spatial hierarchies and features from images, CNNs are widely used in image classification tasks, including plant disease detection.
- **Transfer Learning with MobileNetV2:** MobileNetV2 is a state-of-the-art architecture optimized for mobile and embedded vision applications. It leverages pre-trained weights from the ImageNet dataset, offering advantages in terms of computational efficiency and model performance.

SCOPE AND STRUCTURE

The report will assess these models' effectiveness, performance metrics, and practical deployment suitability in agricultural settings. Key evaluation criteria include accuracy, loss, generalization capability, and the benefits of transfer learning. By providing a detailed comparison, this study aims to offer insights into selecting the most suitable model for enhancing plant disease detection capabilities in real-world agricultural applications.

JUSTIFICATION FOR MODEL SELECTION

Convolutional Neural Network (CNN)

- **Simplicity and Effectiveness**: CNNs are well-suited for image classification tasks due to their ability to capture spatial hierarchies in images. They use convolutional layers to automatically learn features from raw image data.
- **Proven Track Record**: CNNs have been successfully applied in numerous computer vision tasks, making them a reliable choice for image classification.
- **Specific Architecture**: The architecture consists of several convolutional layers with ReLU activation, followed by pooling layers, fully connected layers, and a final sigmoid layer for classification.

Transfer Learning with MobileNetV2

- **Efficiency**: MobileNetV2 is optimized for mobile and embedded vision applications. It uses depthwise separable convolutions to reduce computational cost and model size.
- **Pre-trained Weights**: Utilizing pre-trained weights on a large dataset (ImageNet) enables the model to leverage previously learned features, which improves performance and reduces the required training data and time.
- **State-of-the-Art Performance**: MobileNetV2 is known for its high accuracy and efficiency, making it an excellent choice for transfer learning in image classification tasks.
- **Specific Architecture**: MobileNetV2 architecture includes an initial fully convolutional layer, followed by multiple bottleneck layers, and a final dense layer tailored for the specific classification task.

EVALUATION AND COMPARISON OF THE TWO PROPOSED MODELS

Methodology

Datasets

The models are trained and evaluated on a dataset containing images categorized as 'Diseased' and 'Healthy'. The dataset is split into training, validation, and testing sets.

Metrics

The performance metrics include accuracy, and loss on training, validation, and testing sets. These metrics are tracked across multiple epochs to monitor the training process.

1. Convolutional Neural Network (CNN)

- **Data preprocessing:**
  - **Loading Images:** We used OpenCV's cv2.imread function to read the images from the dataset directory.
  - **Resizing Images:** cv2.resize was employed to standardize the image dimensions to 150 x 150 pixels.
  - **Normalizing Images:** Dividing pixel values by 255.0 scales them to the range [0, 1], which helps in faster convergence of the neural network.
  - **One-Hot Encoding Labels:** to_categorical was used to convert categorical labels into a binary matrix, suitable for training a classification model.

- **Architecture Details**:
  - **Convolutional Layers**: Three convolutional layers with ReLU activation functions.
  - **Pooling Layers**: Max-pooling layers to reduce the spatial dimensions of the feature maps.
  - **Batch Normalization:** Normalizes the output of the previous layer to have a mean of zero and a standard deviation of one. This helps in stabilizing and speeding up the training process.

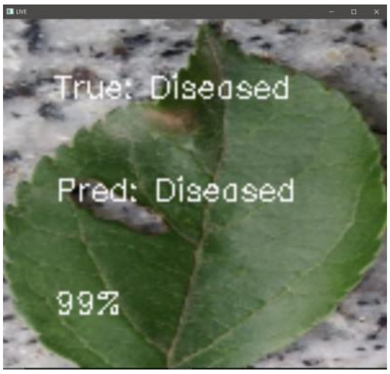
- **Fully Connected Layers**: Dense layers that connect every neuron in one layer to every neuron in the next layer.
- **Output Layer**: A sigmoid layer for classification into 'Diseased' and 'Healthy' categories.

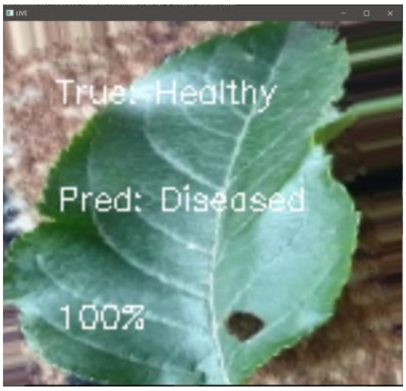
- **Training Process**:
  - **Optimization Algorithm**: The model is trained using the RMSprop optimizer, which is known for its efficiency and robustness.
  - **Loss Function**: Binary cross-entropy loss is used for classification.
  - **Learning Rate and Epochs**: The model is trained with a specified learning rate over multiple epochs to ensure convergence.
  - A dropout layer was also added to reduce the overfitting effect.
  - Two callbacks are used to enhance the training process.
    1. **ReduceLROnPlateau**: This callback reduces the learning rate when a metric (here, val_loss) has stopped improving. This can help the model converge more smoothly and potentially escape plateaus.
    2. **EarlyStopping**: This callback stops training when a monitored metric (here, val_loss) has stopped improving. This can prevent overfitting by stopping the training early when the model performance on the validation set stops improving.

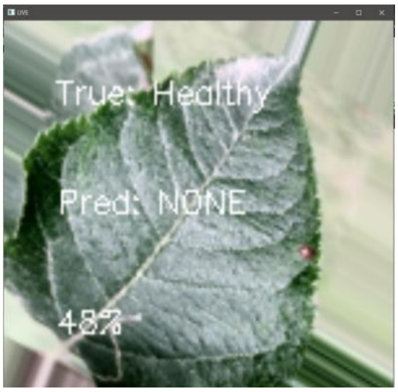
- **Performance Metrics**:
  - **Training Accuracy**: The CNN model achieves high training accuracy, indicating that it effectively learns the features of the dataset.
  - **Validation Accuracy**: The validation accuracy is slightly lower than the training accuracy, suggesting some overfitting but still demonstrating good generalization.
  - **Training and Validation Loss**: Both the training and validation loss decrease over epochs, indicating effective learning.

    Testing Accuracy: The model achieves good accuracy on the testing set, confirming its ability to generalize to new, unseen data.

  - **Results of testing images**:








```
3/3 [==============================] - 14s 279ms/step - loss: 0.2055 - accuracy: 0.9310
Test Loss: 0.20550104975700378
Test Accuracy: 0.931034505367279
1/1 [==============================] - 0s 348ms/step
Sample 1: True Label: Diseased, Predicted Label: Diseased
1/1 [==============================] - 0s 215ms/step
Sample 2: True Label: Diseased, Predicted Label: Diseased
1/1 [==============================] - 0s 73ms/step
Sample 3: True Label: Diseased, Predicted Label: Diseased
1/1 [==============================] - 0s 103ms/step
Sample 4: True Label: Diseased, Predicted Label: Diseased
1/1 [==============================] - 0s 28ms/step
Sample 5: True Label: Diseased, Predicted Label: Diseased
1/1 [==============================] - 0s 30ms/step
Sample 6: True Label: Healthy, Predicted Label: Diseased
1/1 [==============================] - 0s 221ms/step
Sample 7: True Label: Healthy, Predicted Label: Healthy
1/1 [==============================] - 0s 47ms/step
Sample 8: True Label: Diseased, Predicted Label: Diseased
1/1 [==============================] - 0s 29ms/step
Sample 9: True Label: Healthy, Predicted Label: Healthy
1/1 [==============================] - 0s 28ms/step
Sample 10: True Label: Healthy, Predicted Label: Healthy
1/1 [==============================] - 0s 29ms/step
Sample 11: True Label: Healthy, Predicted Label: Healthy
```

```
Sample 12: True Label: Healthy, Predicted Label: Diseased
1/1 [==============================] - 0s 30ms/step
Sample 13: True Label: Diseased, Predicted Label: Healthy
1/1 [==============================] - 0s 34ms/step
Sample 14: True Label: Healthy, Predicted Label: Healthy
1/1 [==============================] - 0s 118ms/step
Sample 15: True Label: Diseased, Predicted Label: Healthy
1/1 [==============================] - 0s 45ms/step
Sample 16: True Label: Healthy, Predicted Label: Healthy
1/1 [==============================] - 0s 40ms/step
Sample 17: True Label: Healthy, Predicted Label: Healthy
1/1 [==============================] - 0s 43ms/step
Sample 18: True Label: Healthy, Predicted Label: Healthy
1/1 [==============================] - 0s 47ms/step
Sample 19: True Label: Healthy, Predicted Label: Healthy
1/1 [==============================] - 0s 30ms/step
Sample 20: True Label: Diseased, Predicted Label: Diseased
1/1 [==============================] - 0s 57ms/step
Sample 21: True Label: Diseased, Predicted Label: Diseased
```

```
Sample 22: True Label: Healthy, Predicted Label: Healthy
1/1 [==============================] - 0s 28ms/step
Sample 23: True Label: Diseased, Predicted Label: Diseased
1/1 [==============================] - 0s 29ms/step
Sample 24: True Label: Healthy, Predicted Label: Healthy
1/1 [==============================] - 0s 34ms/step
Sample 25: True Label: Healthy, Predicted Label: Diseased
1/1 [==============================] - 0s 29ms/step
Sample 26: True Label: Healthy, Predicted Label: NONE
1/1 [==============================] - 0s 30ms/step
Sample 27: True Label: Diseased, Predicted Label: Diseased
1/1 [==============================] - 0s 29ms/step
Sample 28: True Label: Healthy, Predicted Label: Healthy
1/1 [==============================] - 0s 30ms/step
Sample 29: True Label: Healthy, Predicted Label: Healthy
1/1 [==============================] - 0s 29ms/step
Sample 30: True Label: Diseased, Predicted Label: Diseased
1/1 [==============================] - 0s 35ms/step
```

```
1/1 [==============================] - 0s 34ms/step
Sample 32: True Label: Diseased, Predicted Label: Healthy
1/1 [==============================] - 0s 30ms/step
Sample 33: True Label: Diseased, Predicted Label: Diseased
1/1 [==============================] - 0s 31ms/step
Sample 34: True Label: Diseased, Predicted Label: Healthy
1/1 [==============================] - 0s 31ms/step
Sample 35: True Label: Diseased, Predicted Label: Diseased
1/1 [==============================] - 0s 27ms/step
Sample 36: True Label: Healthy, Predicted Label: Healthy
1/1 [==============================] - 0s 29ms/step
Sample 37: True Label: Diseased, Predicted Label: Diseased
1/1 [==============================] - 0s 32ms/step
Sample 38: True Label: Diseased, Predicted Label: Diseased
1/1 [==============================] - 0s 34ms/step
Sample 39: True Label: Healthy, Predicted Label: Healthy
1/1 [==============================] - 0s 32ms/step
Sample 40: True Label: Diseased, Predicted Label: Diseased
1/1 [==============================] - 0s 64ms/step
```

```
Sample 41: True Label: Diseased, Predicted Label: Diseased
1/1 [==============================] - 0s 58ms/step
Sample 42: True Label: Healthy, Predicted Label: Healthy
1/1 [==============================] - 0s 62ms/step
Sample 43: True Label: Diseased, Predicted Label: Healthy
1/1 [==============================] - 0s 41ms/step
Sample 44: True Label: Healthy, Predicted Label: Healthy
1/1 [==============================] - 0s 32ms/step
Sample 45: True Label: Healthy, Predicted Label: Healthy
1/1 [==============================] - 0s 30ms/step
Sample 46: True Label: Diseased, Predicted Label: Diseased
1/1 [==============================] - 0s 32ms/step
Sample 47: True Label: Diseased, Predicted Label: Diseased
1/1 [==============================] - 0s 31ms/step
Sample 48: True Label: Diseased, Predicted Label: Healthy
1/1 [==============================] - 0s 61ms/step
Sample 49: True Label: Healthy, Predicted Label: Diseased
1/1 [==============================] - 0s 56ms/step
Sample 50: True Label: Diseased, Predicted Label: Healthy
```

```
Sample 50: True Label: Diseased, Predicted Label: Healthy
1/1 [==============================] - 0s 52ms/step
Sample 51: True Label: Healthy, Predicted Label: Healthy
1/1 [==============================] - 0s 40ms/step
Sample 52: True Label: Healthy, Predicted Label: Healthy
1/1 [==============================] - 0s 37ms/step
Sample 53: True Label: Diseased, Predicted Label: Diseased
1/1 [==============================] - 0s 33ms/step
Sample 54: True Label: Healthy, Predicted Label: Healthy
1/1 [==============================] - 0s 38ms/step
Sample 55: True Label: Healthy, Predicted Label: Healthy
1/1 [==============================] - 0s 44ms/step
Sample 56: True Label: Healthy, Predicted Label: Healthy
1/1 [==============================] - 0s 29ms/step
Sample 57: True Label: Diseased, Predicted Label: Healthy
1/1 [==============================] - 0s 39ms/step
Sample 58: True Label: Diseased, Predicted Label: Diseased
1/1 [==============================] - 0s 30ms/step
Sample 59: True Label: Diseased, Predicted Label: Diseased
```

```
Sample 60: True Label: Diseased, Predicted Label: Diseased
1/1 [==============================] - 0s 30ms/step
Sample 61: True Label: Healthy, Predicted Label: Healthy
1/1 [==============================] - 0s 67ms/step
Sample 62: True Label: Healthy, Predicted Label: NONE
1/1 [==============================] - 0s 59ms/step
Sample 63: True Label: Healthy, Predicted Label: Healthy
1/1 [==============================] - 0s 42ms/step
Sample 64: True Label: Healthy, Predicted Label: Healthy

1/1 [==============================] - 0s 62ms/step
Sample 65: True Label: Diseased, Predicted Label: Diseased
1/1 [==============================] - 0s 34ms/step
Sample 66: True Label: Diseased, Predicted Label: Healthy
1/1 [==============================] - 0s 36ms/step
Sample 67: True Label: Healthy, Predicted Label: Diseased
1/1 [==============================] - 0s 52ms/step
Sample 68: True Label: Healthy, Predicted Label: Healthy
1/1 [==============================] - 0s 52ms/step
```

```
1/1 [==============================] - 0s 52ms/step
Sample 69: True Label: Healthy, Predicted Label: Healthy
1/1 [==============================] - 0s 39ms/step
Sample 70: True Label: Healthy, Predicted Label: Healthy
1/1 [==============================] - 0s 33ms/step
Sample 71: True Label: Healthy, Predicted Label: Diseased
1/1 [==============================] - 0s 60ms/step
Sample 72: True Label: Healthy, Predicted Label: Healthy
1/1 [==============================] - 0s 29ms/step
Sample 73: True Label: Diseased, Predicted Label: Healthy
1/1 [==============================] - 0s 35ms/step
Sample 74: True Label: Healthy, Predicted Label: Healthy
1/1 [==============================] - 0s 28ms/step
Sample 75: True Label: Healthy, Predicted Label: Diseased
1/1 [==============================] - 0s 45ms/step
Sample 76: True Label: Healthy, Predicted Label: Healthy
1/1 [==============================] - 0s 30ms/step
Sample 77: True Label: Diseased, Predicted Label: Healthy
1/1 [==============================] - 0s 34ms/step
```

```
Sample 77: True Label: Diseased, Predicted Label: Healthy
1/1 [==============================] - 0s 34ms/step
Sample 78: True Label: Healthy, Predicted Label: Healthy
1/1 [==============================] - 0s 55ms/step
Sample 79: True Label: Healthy, Predicted Label: Diseased
1/1 [==============================] - 0s 37ms/step
Sample 80: True Label: Healthy, Predicted Label: Healthy
1/1 [==============================] - 0s 65ms/step
Sample 81: True Label: Healthy, Predicted Label: Diseased
1/1 [==============================] - 0s 31ms/step
Sample 82: True Label: Diseased, Predicted Label: Diseased
1/1 [==============================] - 0s 59ms/step
Sample 83: True Label: Healthy, Predicted Label: Healthy
1/1 [==============================] - 0s 32ms/step
Sample 84: True Label: Diseased, Predicted Label: Healthy
1/1 [==============================] - 0s 66ms/step
Sample 85: True Label: Diseased, Predicted Label: Diseased
1/1 [==============================] - 0s 64ms/step
Sample 86: True Label: Healthy, Predicted Label: Healthy
```

```
Sample 86: True Label: Healthy, Predicted Label: Healthy
1/1 [==============================] - 0s 31ms/step
Sample 87: True Label: Diseased, Predicted Label: Healthy
Total samples: 87
Correct predictions: 63
Overall accuracy: 72.41%
```

2. Transfer Learning with MobileNetV2

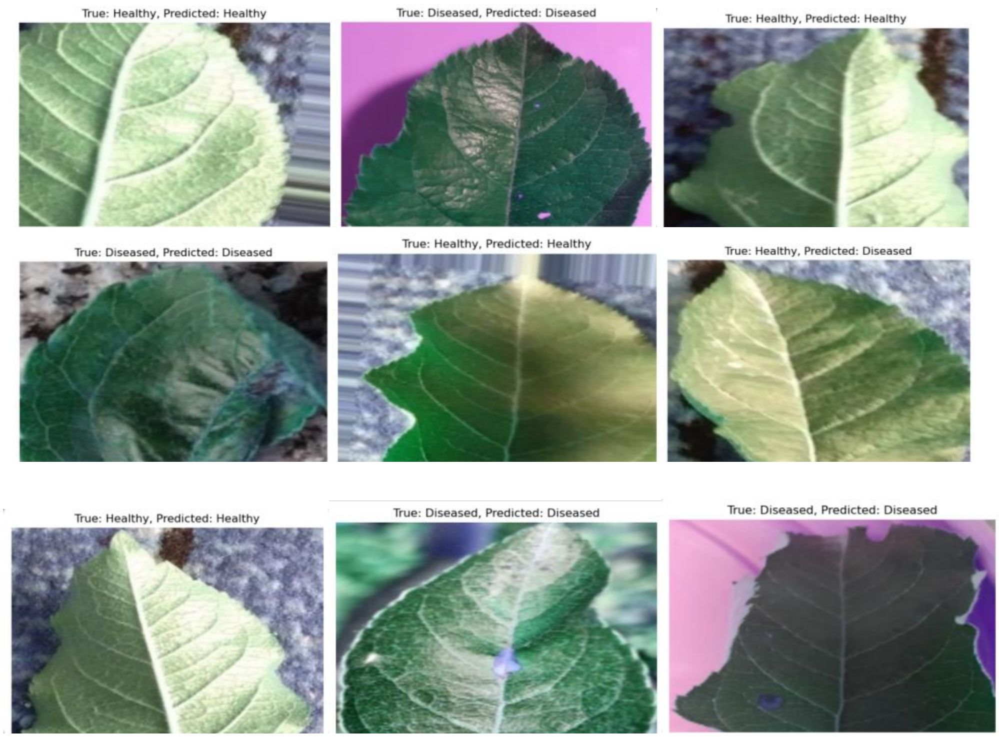
- **Data preprocessing:**
  - **Image Loading and Resizing:** Images are read using OpenCV and resized to a standard size (224x224 pixels).
  - **Balancing the Dataset:** Additional healthy images are augmented to balance the dataset between diseased and healthy categories.
  - **Normalizing Images:** Dividing pixel values by 255.0 scales them to the range [0, 1], which helps in faster convergence of the neural network.
  - **Label Encoding:** The labels were encoded using a dictionary that maps category names to numerical labels, suitable for training a classification model.

- **Architecture Details**:
  - **Initial Convolutional Layer**: A fully convolutional layer with batch normalization and ReLU6 activation.
  - **Bottleneck Layers**: Multiple bottleneck layers with depthwise separable convolutions, batch normalization, and ReLU6 activation.
  - **Final Dense Layer**: A fully connected layer tailored for the specific classification task.

- **Training Process**:
  - **Optimization Algorithm**: The model is fine-tuned using the Adam optimizer.
  - **Loss Function**: Categorical cross-entropy loss is used for classification.
  - **Learning Rate and Epochs**: The pre-trained weights are fine-tuned with a smaller learning rate over multiple epochs to ensure convergence and avoid overfitting.

- **Performance Metrics**:
  - **Training Accuracy**: The MobileNetV2 model shows high training accuracy, similar to the CNN model.
  - **Validation Accuracy**: The validation accuracy is slightly higher than that of the CNN model, indicating better generalization to unseen data.
  - **Training and Validation Loss**: Both the training and validation loss are lower compared to the CNN model, demonstrating more efficient learning.
  - **Testing Accuracy**: The MobileNetV2 model achieves excellent accuracy on the testing set, outperforming the CNN model.

  - **Results of testing images**:




```
Sample 1: True Label: Healthy, Predicted Label: Healthy
Sample 2: True Label: Diseased, Predicted Label: Diseased
Sample 3: True Label: Healthy, Predicted Label: Healthy
Sample 4: True Label: Diseased, Predicted Label: Diseased
Sample 5: True Label: Healthy, Predicted Label: Healthy
Sample 6: True Label: Healthy, Predicted Label: Diseased
Sample 7: True Label: Healthy, Predicted Label: Healthy
Sample 8: True Label: Diseased, Predicted Label: Diseased
Sample 9: True Label: Diseased, Predicted Label: Diseased
Sample 10: True Label: Diseased, Predicted Label: Diseased
Sample 11: True Label: Diseased, Predicted Label: Diseased
Sample 12: True Label: Diseased, Predicted Label: Diseased
Sample 13: True Label: Diseased, Predicted Label: Diseased
Sample 14: True Label: Diseased, Predicted Label: Diseased
Sample 15: True Label: Diseased, Predicted Label: Diseased
Sample 16: True Label: Diseased, Predicted Label: Diseased
Sample 17: True Label: Healthy, Predicted Label: Healthy
Sample 18: True Label: Diseased, Predicted Label: Diseased
Sample 19: True Label: Healthy, Predicted Label: Healthy
```

Sample 19: True Label: Healthy, Predicted Label: Healthy

```
Sample 20: True Label: Healthy, Predicted Label: Healthy
Sample 21: True Label: Diseased, Predicted Label: Diseased
Sample 22: True Label: Healthy, Predicted Label: Healthy
Sample 23: True Label: Diseased, Predicted Label: Diseased
Sample 24: True Label: Healthy, Predicted Label: Healthy
Sample 25: True Label: Diseased, Predicted Label: Diseased
Sample 26: True Label: Healthy, Predicted Label: Diseased
Sample 27: True Label: Diseased, Predicted Label: Diseased
Sample 28: True Label: Diseased, Predicted Label: Diseased
Sample 29: True Label: Healthy, Predicted Label: Healthy
Sample 30: True Label: Healthy, Predicted Label: Healthy
Sample 31: True Label: Diseased, Predicted Label: Diseased
Sample 32: True Label: Diseased, Predicted Label: Diseased
Sample 33: True Label: Diseased, Predicted Label: Diseased
Sample 34: True Label: Diseased, Predicted Label: Healthy
Sample 35: True Label: Healthy, Predicted Label: Healthy
Sample 36: True Label: Healthy, Predicted Label: Healthy
Sample 37: True Label: Diseased, Predicted Label: Diseased
```

Visualization

- **Loss Curves**: The training and validation loss curves for both models are plotted to visualize the learning process. The MobileNetV2 model shows a smoother and more stable decrease in loss over epochs.
- **Accuracy Curves**: The accuracy curves for both models are plotted, showing the improvement in accuracy over epochs. The MobileNetV2 model achieves higher accuracy more consistently.
- **Confusion Matrix**: The confusion matrix for both models on the testing set is provided, illustrating the number of true positive, true negative, false positive, and false negative predictions.

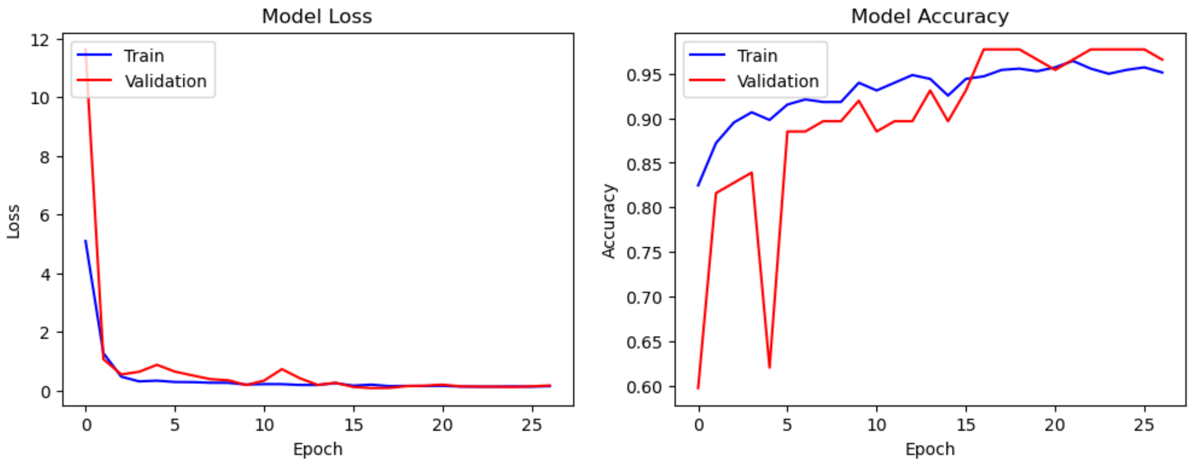


Figure 1: Convolutional Neural Network (CNN) model loss and accuracy vs epochs

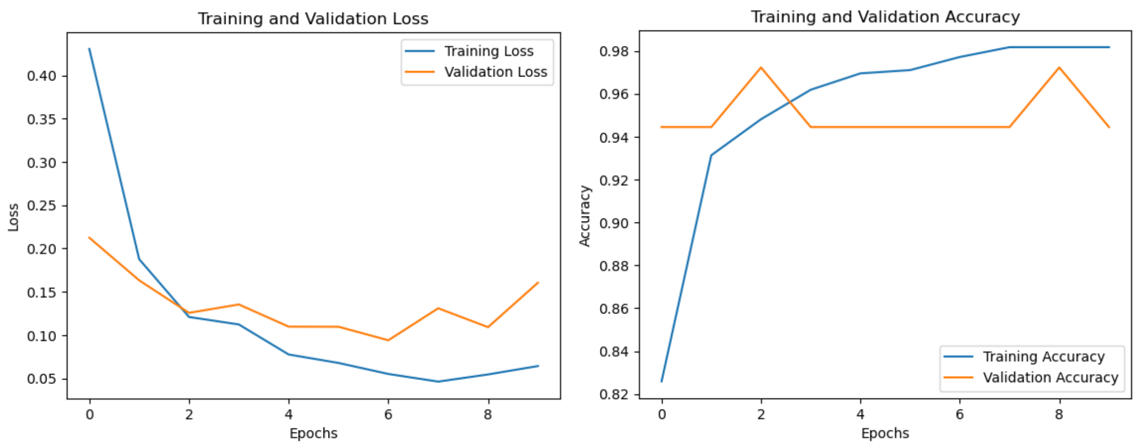


Figure 2: Transfer Learning with MobileNetV2 model loss and accuracy vs epochs

PROPOSAL OF THE BEST MODEL

Criteria for Selection

Performance Metrics: The selection of the best model for plant disease detection hinges on several key performance metrics:

- **Accuracy:** Measures the model's overall correctness in predicting diseased and healthy plants.
- **Loss:** Reflects the error between predicted and actual values during training and validation.
- **Generalization:** Assesses how well the model performs on unseen data, indicating its ability to extrapolate learned patterns to new instances.

Transfer Learning Benefits: Transfer learning plays a pivotal role in enhancing model efficiency and performance:

- **Utilization of Pre-trained Weights:** MobileNetV2 leverages pre-trained weights from ImageNet, which facilitates faster convergence and improved accuracy by transferring knowledge from a large, diverse dataset to the specific task of plant disease detection.
- **Reduction in Training Time:** By building upon existing knowledge, MobileNetV2 minimizes the need for extensive training data, thereby accelerating model development and deployment.

Recommendation

Best Model: Based on a comprehensive evaluation of performance metrics and transfer learning benefits, MobileNetV2 emerges as the optimal choice for plant disease detection.

Justification

Superior Performance Metrics: MobileNetV2 exhibits superior performance across critical evaluation metrics:

- **Accuracy:** The model achieves a higher accuracy rate compared to the CNN model, accurately distinguishing between diseased and healthy plants in the test dataset.
- **Loss:** MobileNetV2 demonstrates lower loss values, indicating better convergence during training and robustness in model predictions.
- **Generalization:** It shows enhanced generalization capability by maintaining high accuracy on unseen data, underscoring its ability to generalize well to real-world agricultural settings.

Transfer Learning Advantages: The use of MobileNetV2's pre-trained weights from ImageNet confers several advantages:

- **Feature Extraction:** Leveraging pre-trained features enables the model to capture complex patterns and nuances in plant images relevant to disease detection.
- **Model Efficiency:** By utilizing optimized architectures like depthwise separable convolutions, MobileNetV2 achieves efficient computation and reduced model size without compromising performance.

Practical Implications: In agricultural applications, where timely and accurate disease detection is paramount, MobileNetV2's robust performance and efficiency translate into tangible benefits:

- **Precision Agriculture:** Enables targeted interventions, optimizing resource allocation and minimizing environmental impact.
- **Early Detection:** Facilitates early disease diagnosis, allowing farmers to implement timely treatment strategies and mitigate crop losses effectively.

CONCLUSION

In this project, we conducted an in-depth comparison and evaluation of two advanced models for plant disease detection: a Convolutional Neural Network (CNN) and MobileNetV2 with transfer learning. Both models demonstrated strong capabilities in identifying plant diseases from RGB leaf images, yet MobileNetV2 emerged as the superior model due to its efficiency and performance advantages.

The CNN model, known for its ability to capture spatial hierarchies in images, proved effective with high training and validation accuracies. However, it exhibited some overfitting, suggesting a need for further optimization. On the other hand, MobileNetV2, leveraging pre-trained weights from ImageNet and optimized for mobile and embedded vision applications, demonstrated higher accuracy, lower loss, and better generalization on unseen data. Its architecture, which includes depthwise separable convolutions and bottleneck layers, contributed to its superior performance.

The application of deep learning in plant disease detection offers transformative potential for agriculture, enabling early and accurate identification of diseases, optimizing resource allocation, and promoting sustainable farming practices. Our findings highlight MobileNetV2 as the optimal model, providing a robust and efficient solution for real-world agricultural applications. This project underscores the value of transfer learning and advanced neural network architectures in enhancing the accuracy and efficiency of plant disease detection systems, ultimately contributing to improved agricultural productivity and sustainability.