Abstract

Crop diseases pose a significant threat to global food security by reducing yield quality and quantity. Early and accurate detection of plant leaf diseases is therefore critical for sustainable agriculture. In this study, we investigate deep learning approaches for classifying leaf diseases across three important crops: cotton, rice, and sugarcane. Two convolutional neural network (CNN) architectures are evaluated: MobileNetV2, a lightweight model optimized for speed and resource efficiency, and EfficientNetB3, a more advanced model designed for high accuracy through compound scaling.

Both models were trained using a standardized pipeline to ensure fair comparison, including identical preprocessing steps, augmentation techniques, and training phases (transfer learning followed by fine-tuning). The datasets included multiple disease classes for each crop, along with healthy leaf samples. Model performance was assessed using metrics such as accuracy, precision, recall, and F1-score.

Experimental results show that while MobileNetV2 performs competitively with faster inference, EfficientNetB3 consistently achieves higher accuracy and better generalization across all three crops. Visualization techniques, including heatmaps and class activation overlays, were also used to interpret the models and highlight the regions most influential in disease prediction.

This work demonstrates that deep learning can be effectively applied to multi-crop disease detection, and highlights the trade-off between lightweight models suitable for real-time applications and deeper models that achieve higher predictive performance. Future work includes extending the approach to bounding box detection for localization and deploying the models in mobile-based agricultural decision support systems.

Datasets

Rice Leaf Dataset

This dataset was curated specifically for the detection and classification of six distinct types of rice leaf diseases: Hispa, Brown Spot, Leaf Blast, Leaf Scald, Narrow Brown Spot, and Neck Blast. Each category represents a unique pathology that significantly impacts rice crop health and yield. The dataset provides annotated images that enable the training and evaluation of machine learning models, supporting automated detection systems for early diagnosis. Researchers and practitioners in agricultural technology and plant pathology can leverage this dataset to build robust solutions for disease management, ultimately improving food security and sustainable farming practices.

Sugarcane Leaf Dataset

Image datasets play a crucial role across diverse fields, including computer vision, machine learning, and agricultural research. In this context, the sugarcane leaf disease dataset serves as a valuable benchmark for identifying, classifying, and studying various sugarcane

leaf pathologies. The dataset consists of 6,748 high-resolution images of sugarcane leaves stored in JPEG format with dimensions of 768 × 1024 pixels and a resolution of 72 dpi.

These images are categorized into 12 distinct classes, including 10 disease categories, a healthy leaves category, and a dried leaves category. The dataset was compiled through extensive field surveys, capturing different stages of infection and multiple viewpoints, ensuring a comprehensive representation of visual disease characteristics. Its diversity and clarity make it highly suitable for training deep learning models, enabling the development of accurate disease detection algorithms, early warning systems, and effective crop management practices.

Cotton Dataset

The cotton dataset was obtained from the SAR-CLD-2024: A Comprehensive Dataset for Cotton Leaf Disease Detection repository. It includes both original and augmented images, combined into a single dataset to enhance diversity and model generalization.

Dataset Composition:
A total of 9,137 images across 7 classes, distributed as follows:

- Bacterial_Blight: 1,250
- Curl_Virus: 1,431
- Healthy: 1,257
- Herbicide: 1,280
- Leaf_Hopper: 1,225
- Leaf_Reddening: 1,578
- Leaf_Variegation: 1,116

Models

MobileNetV2

MobileNetV2 is a lightweight convolutional neural network (CNN) architecture designed for mobile and embedded vision applications. It uses depthwise separable convolutions to reduce computational cost while maintaining strong accuracy. The architecture introduces inverted residual blocks with linear bottlenecks, which allow for efficient feature reuse and improved gradient flow. Because of its smaller size and faster inference speed, MobileNetV2 is well-suited for real-time agricultural disease detection tasks where computational resources may be limited.

EfficientNetB3

EfficientNet is a family of CNN architectures that scale model depth, width, and resolution in a principled way using a compound scaling method. EfficientNetB3, in particular, offers a good balance between accuracy and computational efficiency. It builds on the Mobile Inverted Bottleneck (MBConv) blocks, similar to MobileNetV2, but introduces squeeze-and-excitation optimization to improve channel-wise feature recalibration. EfficientNetB3 typically achieves higher accuracy than MobileNetV2 on image classification tasks, making it a strong candidate for crop disease detection across diverse datasets.

Training Pipeline Implementation

To optimize model performance while preventing overfitting, we adopted a two-phase training strategy with the following callbacks:

Callbacks Used

- EarlyStopping: Monitored validation loss and stopped training if no improvement was observed after 5 epochs, restoring the best weights.
- ModelCheckpoint: Saved the best-performing model based on validation loss.
- ReduceLROnPlateau: Reduced the learning rate by a factor of 0.3 when validation loss plateaued, with a minimum threshold of 1e-6.
- CSVLogger: Recorded the training history (loss, accuracy, and validation metrics) for further analysis.

Phase 1 – Classifier Head Training (Feature Extraction)

- Initialized the backbone (MobileNetV2 or EfficientNetB3) with ImageNet weights.
- Kept the backbone frozen, training only the newly added classification head.
- Optimizer: Adam, learning rate = 1e-3.
- Duration: 10 epochs.

Phase 2 – Fine-Tuning

- Unfroze the last 50 layers of the backbone for fine-tuning, while keeping BatchNormalization layers frozen to maintain stability.
- Optimizer: Adam, learning rate = 1e-5 (smaller to avoid catastrophic forgetting).
- Duration: 20 epochs.

This two-step approach ensured that the models first leveraged generic ImageNet features and then gradually adapted to crop-specific disease features without losing previously learned representations

Evaluation Metrics

To comprehensively assess the performance of the models across the three crop datasets (cotton, rice, and sugarcane), we employed a diverse set of evaluation metrics. These metrics not only quantify overall accuracy but also provide deeper insights into class-wise performance, robustness, and the reliability of predictions.

Accuracy

- The proportion of correctly classified images across all classes.
- Serves as the primary measure of overall model performance.

$$Accuracy=TP+TN / FP+FN+TP+TN$$

Where:

- TP = True Positives
- TN = True Negatives
- FP = False Positives
- FN= False Negatives

Precision, Recall, and F1-Score

- **Precision**: Indicates the fraction of correctly predicted positive samples among all predicted positives.

  $$Precision=TP/FP+TP$$

  Indicates how many of the predicted positives are actually correct.

- **Recall (Sensitivity)**: Measures the proportion of correctly identified samples out of all actual positives.

  $$Recall=TP/FN+TP$$

  Measures how many actual positives are correctly identified.

- **F1-Score**: Harmonic mean of precision and recall, providing a balanced view of classification performance.

F1-Score=2 x (Precision×Recall / Precision+Recall)

Balances **Precision** and **Recall** into a single metric.

- These metrics are reported **per class** and as a **macro/micro average** across all classes.

Confusion Matrix

- Visualizes classification results by showing correct predictions (diagonal) and misclassifications (off-diagonal).
- Helps identify classes with high confusion (e.g., visually similar diseases).

Training Curves

- Plotted **loss and accuracy trends** during training and validation across both phases:
    - **Phase 1 (Classifier Head Training)**: Typically shows rapid convergence.
    - **Phase 2 (Fine-Tuning)**: Demonstrates gradual improvement as deeper layers adapt to crop-specific features.
- Used to monitor overfitting, underfitting, and general training stability.

MAP

mAP = mean Average Precision
It's a metric mostly used in object detection tasks (like YOLO, Faster R-CNN, etc.), but sometimes people also talk about it in multi-class classification.

1. Average Precision (AP)

- For one class, AP is basically the area under the Precision–Recall curve (PR curve).
- Instead of looking at precision at just one point (like threshold = 0.5), we check precision at all recall levels and average it.

So AP = overall score of how well precision & recall trade-off is maintained for that class.

2. mean Average Precision (mAP)

If you have multiple classes (like your sugarcane dataset with several diseases), you compute AP for each class and then take the mean:

Example

Suppose your dataset has 3 classes:

- Bacterial Blight → AP = 0.82
- Leaf Curl → AP = 0.74
- Healthy → AP = 0.88

Then:

mAP=(0.82+0.74+0.88)/3=0.81

So your model's mAP = 81% → on average, your model maintains 81% precision-recall performance across all classes.

Key Difference from Precision/Recall

- Precision & Recall are point metrics (at a single threshold).
- AP looks at the *whole precision-recall curve*.
- mAP averages AP across all classes → a stronger, more complete metric.

Top-K Accuracy

Instead of looking only at the single top prediction (Top-1), we check whether the true class is within the top K predictions.

Example with K = 3 (Top-3 Accuracy):

True class = Healthy
Model predictions:

```
Bacterial Blight → 0.45
Curl Virus       → 0.30
Healthy          → 0.20
```

```
Leaf Hopper      → 0.05
```

- Top-1 prediction = Bacterial Blight → Wrong (so normal accuracy fails).
- But Healthy is in the top-3 predictions → Correct under Top-3 Accuracy.

    Top-1 Accuracy = 85%

    Top-3 Accuracy = 95%

    This means: even when the model's first guess is wrong, the true label is present in the top-3 predictions 95% of the time.

Why is this useful?

- In fine-grained tasks (like leaf diseases, bird species, handwriting digits), some classes look very similar.
- The model might not always put the correct class at rank-1, but often the correct answer is near the top.
- Doctors or farmers could still use the "Top-3 suggestions" as a decision support system.

ROC Curve & AUC (Area Under Curve)

What is ROC Curve?

- ROC = Receiver Operating Characteristic
- It's a graph that shows the trade-off between sensitivity (Recall/True Positive Rate) and specificity (1 − False Positive Rate) at different classification thresholds.

For binary classification (2 classes), it's straightforward:

- TPR (y-axis) = Recall = TP / (TP + FN)
- FPR (x-axis) = FP / (FP + TN)

We draw the curve by changing the threshold (default = 0.5) to see how predictions behave at 0.1, 0.2, …, 0.9.

AUC (Area Under Curve)

- AUC = Area under the ROC curve.
- It measures how well the model separates the classes.

Interpretation:

- AUC = 1.0 → perfect model
- AUC = 0.5 → random guessing
- AUC = 0.7–0.8 → fair
- AUC = 0.8–0.9 → good
- AUC ≥ 0.9 → excellent

ROC & AUC in Multi-Class Classification (like your 7 leaf diseases)

Since you have 7 classes, ROC/AUC is computed in one-vs-rest style:

- For each class, treat it as “positive” vs all other classes “negative”.
- Plot ROC curve for each class.
- Optionally, calculate macro-average and micro-average AUC to summarize.

Log Loss (Cross-Entropy Loss)

- Log Loss measures how uncertain the model is when making predictions.
- It penalizes wrong confident predictions more heavily than uncertain ones.

Formula (for multi-class classification):

$$\text{LogLoss} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C} y_{i,c} \cdot \log(\hat{p}_{i,c})$$

- N = number of samples
- C = number of classes
- $y_{i,c}$ = 1 if sample i belongs to class c, else 0
- $\hat{p}_{i,c}$ = predicted probability of class c for sample i

Why use it?

- Accuracy just checks right/wrong.
- Log Loss also considers confidence:
    - If the true class = “Healthy” but model predicts 99% Leaf_Hopper, the penalty is very high.
    - If it predicts 51% Healthy, 49% Leaf_Hopper, the penalty is smaller.

So, lower Log Loss = better calibrated probabilities.