Final Project

# Cassava Leaf Disease Image Classification

國立陽明交通大學

NATIONAL YANG MING CHIAO TUNG UNIVERSITY

Prepared by

110550124 林家甫

110550065 尤茂為

110550108 施柏江

110550052 楊沁瑜

# Kaggle Competition: Cassava Leaf Disease Classification

Group 11 : AAAI
GitHub Link : https://github.com/MwYuREZ/NYCU_CV_2025_Final_Project

## Introduction

### Problem Statement

Image classification has become a cornerstone of modern computer vision, enabling automated interpretation of visual data across domains. In this chapter, we focus on the Cassava Leaf Disease Classification problem from a Kaggle competition. The objective is to build a model that, given an input image of a cassava leaf, accurately assigns it to one of five categories: four distinct disease types (Cassava Mosaic Disease, Cassava Bacterial Blight, Cassava Green Mottle, Cassava Brown Streak Disease) or healthy. We leverage TensorFlow Hub's CropNet feature extractor [1], fine-tuned on our dataset, to directly predict disease labels.

### Importance of the Problem

Cassava is a staple crop for over 800 million people worldwide,  early and accurate detection of foliar diseases is critical for maintaining crop yields. Manual inspection is time-consuming , while traditional lab testing can be costly and slow. Automated image-based classification offers a scalable, low-cost alternative, enabling rapid field diagnostics.

### Motivation and Challenges

The dataset exhibits significant class imbalance: leaves with cassava mosaic disease (CMD) vastly outnumber certain disease categories, risking bias toward the majority class. Second, varying lighting conditions, leaf orientations, and background clutter introduce high intra-class variability. Finally, subtle visual differences among some diseases demand a model with both strong feature extraction and fine discrimination.

## Related Works

1. Convolutional Neural Networks (CNNs)
   CNNs, such as AlexNet [5], VGGNet [6], and ResNet [7], have been foundational in advancing image classification tasks. They utilize convolutional layers to extract hierarchical features from images, enabling effective pattern recognition.
   - Pros
     - High accuracy in image classification tasks
     - Ability to learn complex features through deep architectures.
   - Cons
     - Require substantial computational resources-constrained devices.

2. LightWeight CNNs
   Architecture like MobileNet [8] and SqueezeNet [9] are designed for efficiency, making them suitable for deployment on mobile and embedded devices.
    - Pros
        - Efficient inference suitable for real-time applications.
        - Smaller model sizes facilitate deployment on devices with limited resources.
    - Cons
        - Potential trade-offs in accuracy compared to larger CNNs.
        - Struggle with complex classification tasks due to reduced capacity.
3. Transformer-Based Models
   Vision Transformers (ViTs) [3] represent a shift from convolutional architectures by employing self-attention mechanisms to process image data. They have shown promise in capturing global context within images.
    - Pros
        - Effective at modeling long-range dependencies in image data.
        - Scalable architectures that can achieve high performance with sufficient data.
    - Cons
        - Require large datasets for effective training.
        - High computational demands can be a barrier to training and deployment.

What are the advantages of your method over all existing methods?

Unlike lightweight CNNs, it maintains strong feature richness and discriminative power without sacrificing inference efficiency. And compared to Vision Transformers, it demands far fewer computational resources and converges faster.

# Method / Approach

## Data Preprocessing

The original dataset was provided in TFRecord format by the competition host. We directly loaded these records using *tf.data.TFRecordDataset*. And each record contains: ‘image’, ‘image_name’, ‘target’. During training, the data pipeline included the following steps:

- Decoding: Images were decoded from raw JPEG bytes.
- Resizing: Images were reshaped to 224*224.
- Normalization: Pixel values were converted to float32 and normalized to the [0.0, 255.0] range, followed by optional scaling to [0.0, 1.0].

To further improve accuracy, data augmentation was applied during training. This included following operations:

- Random horizontal flip
- Random vertical flip
- Random brightness adjustment
- Random saturation adjustment
- Random 90-degree rotation (0°, 90°, 180°, or 270°)
- Random crop after padding/resizing

- Random hue adjustment
- Pixel value clipping to the [0.0, 1.0] range

This setup was optimized using tf.data pipeline features like prefetch, AUTOTUNE, and batch for efficient training.

## Model Architecture

### Baseline Model (CropNet via TensorFlow Hub)

1. Introduction to CropNet
   CropNet is a lightweight, transfer-learning-based classifier developed by Google on TensorFlow Hub for agricultural disease detection. It leverages a MobileNet V3 backbone pre-trained on ImageNet. Optimized with depthwise separable convolutions and squeeze-and-excitation blocks, CropNet strikes a strong balance between accuracy and efficiency. It accepts 224*224 RGB inputs and produces size softmax probabilities.
2. Model Definition
   - Contruct a tf_keras_Sequential model
     - Input Layer : Accepts RGB images resized to 224*224*3
     - Hub Layer : Appended as the sole hidden layer, effectively replacing any custom head.

### Training Details

- Framework: TensorFlow + TensorFlow Hub
- Optimizer: Adam
- Callbacks:
  EarlyStopping (patience = 4)
  ReduceLROnPlateau
- Epochs: Training typically stopped at ~16 due to callbacks
- Batch Size: 64
- Validation Split: 75/25 manually by slicing TFRecord list

### Inference Pipeline

1. Model Initialization
   - TensorFlow Hub-based classifier is wrapped in a simple Sequential model, taking 224*224*3 inputs.
   - Pre-trained weights (best_model.h5), which is generated by our training process, is loaded to ensure the network is ready for prediction.
2. Data Parsing & Preprocessing
   - TFRRecord files are read in parallel and each example is decoded, JPEG images are cast to float32 and reshaped to 512*512*3, labels are one-hot encoded.
   - Images are normalized, resized to 224*224, and labels are extended to include a default class.
3. Batching & Prefetching

    - The preprocessed dataset is batched and prefetched with AUTOTUNE for optimal throughput.
4. Prediction & Submission
    - The model predicts per-image class probabilities, argmax yields discrete labels.
    - Image_id are decoded from bytes, paired with predictions and written out as submission.csv.

## Late fusion

We maintain two independent inference pipelines, a ViT-B-16 [3] and an EfficientNet-B7 [2], each fine-tuned on cassava leaf disease images, and fuse them at the score level rather than by concatenating deep features.

- Backbone Fine-tuning
    - ViT-B-16 : all backbone parameters frozen, only the transformer head is replaced with a two-layer MLP classifier. (Linear → ReLU → Dropout(0.5) → Linear → 5 classes).
    - EfficientNet-B7 : Freeze all except the last two feature blocks, replace the classifier with Dropout(0.5) → Linear(→512) → ReLU → Dropout(0.4) → Linear(→5).
- Score-Level fusion [10]
    - At test time, each branch produces a 5-dim softmax probability vector. We set the weights to be equal, with each being half, to get the final prediction of each image.
- Advantages
    - Complementary inductive biases : CNN and transformer features each capture different aspects of leaf patterns.
    - Lightweight fusion : No extra trainable MLP beyond each branch's classifier, so inference stays fast.
    - Scalable fine-tuning : It's available to freeze most weights and only train small heads without any changes to the fusion logic.

# Experimental Results

## Learning Curve

As shown in Figure 1, the CropNet model demonstrates stable training and validation performance.The left plot shows the accuracy trends: training accuracy rises quickly from around 65% to 95%, while validation accuracy remains stable around 91%. The close alignment between the two curves indicates good generalization performance and no significant signs of overfitting. The right plot illustrates the loss curves, where both training and validation loss drop significantly during the initial epochs and eventually converge below

0.3. This suggests that the model training was stable and well-converged.

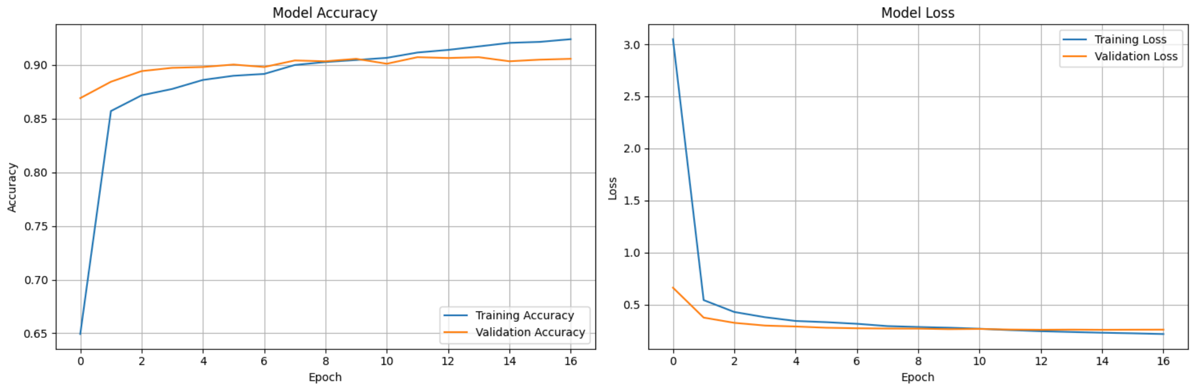
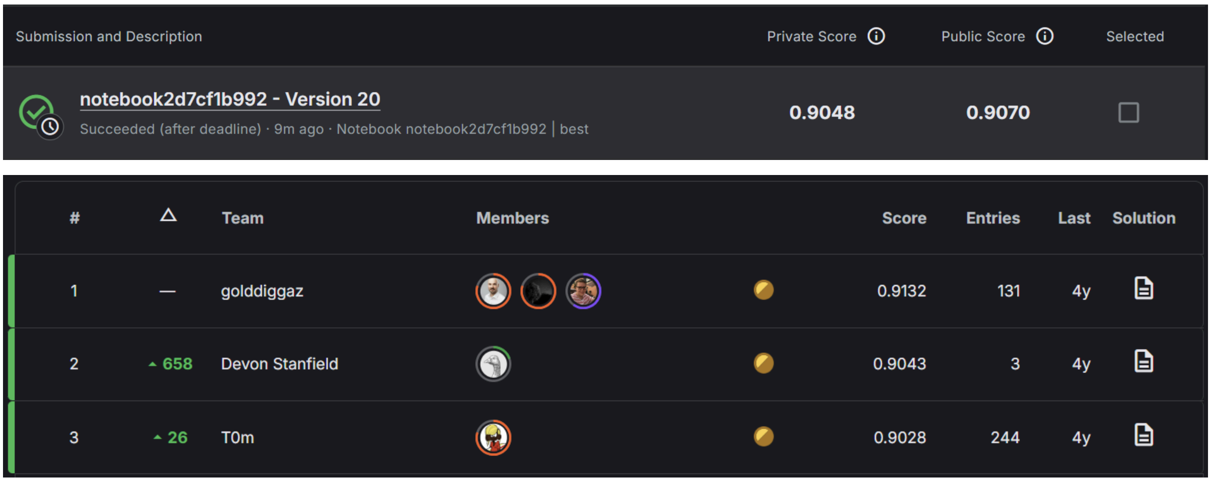


**Figure 1.** Learning curves of the CropNet model. The left plot shows training vs. validation accuracy; the right plot displays corresponding loss curves.

## Performance

Figure 2 and 3 show our final submission results on the Cassava Leaf Disease Classification competition. Our model achieved a private leaderboard score of 0.9048 and a public score of 0.9070. As illustrated in the leaderboard snapshot, this performance slightly outperformed the second-place team, which scored 0.9043.



| Submission and Description | Private Score ⓘ | Public Score ⓘ | Selected |
|---|---|---|---|
| notebook2d7cf1b992 - Version 20<br>Succeeded (after deadline) · 9m ago · Notebook notebook2d7cf1b992 \| best | 0.9048 | 0.9070 | □ |

| # | △ | Team | Members | | Score | Entries | Last | Solution |
|---|---|---|---|---|---|---|---|---|
| 1 | — | golddiggaz | | ● | 0.9132 | 131 | 4y | 🗎 |
| 2 | ▲ 658 | Devon Stanfield | | ● | 0.9043 | 3 | 4y | 🗎 |
| 3 | ▲ 26 | T0m | | ● | 0.9028 | 244 | 4y | 🗎 |

**Figure 2/3.** Final submission results and leaderboard ranking on the Cassava Leaf Disease Classification competition.

## Ablation Studies

We conducted an ablation study to investigate the impact of different training configurations on the performance of the CropNet model for cassava leaf disease classification. The baseline model achieved the highest validation accuracy of 0.9059 with a training time of approximately 9.8 minutes. Three variations were explored (see Figure 3):

- No Data Augmentation: Slight decrease in accuracy (0.9023) but also the shortest training time (8.9 minutes), indicating augmentation contributes to generalization.
- Batch Size = 16: Resulted in marginally lower accuracy (0.9019) and the longest training time (12.4 minutes), reflecting the inefficiency of smaller batches despite similar performance.

- No Weight Decay: Accuracy remained comparable (0.9034) but training time was also relatively high (11.9 minutes), suggesting regularization had minimal influence under current settings.

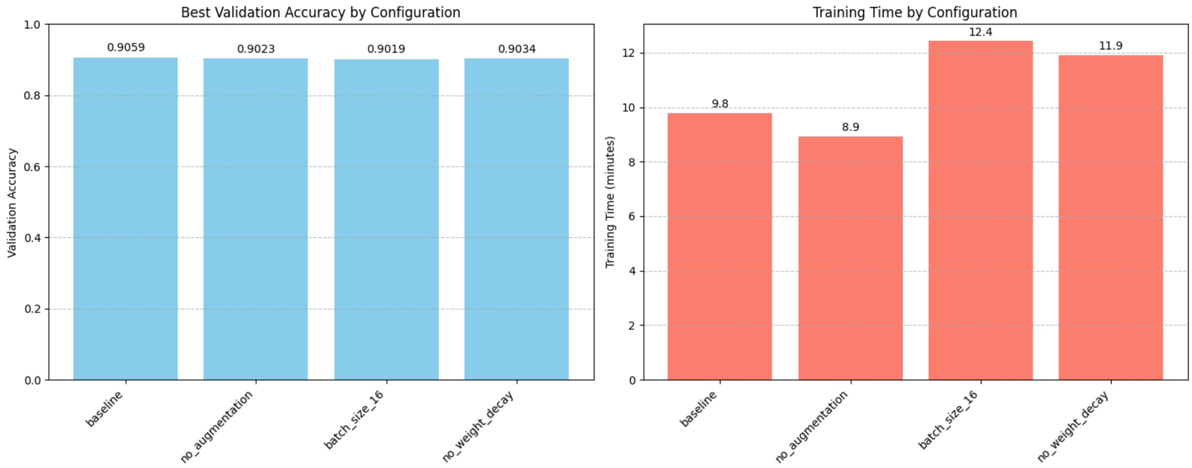


**Figure 4.** Comparison of validation accuracy and training time under different configurations.

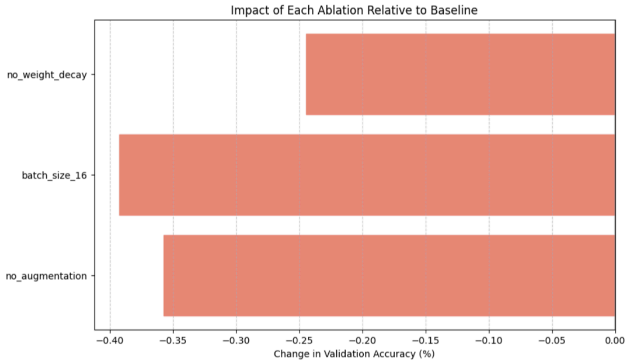


**Figure 5.** Change in validation accuracy relative to the baseline.

Learning curves reveal all configurations converged well, with the baseline showing stable training and validation performance. Overall, data augmentation and optimal batch size trade-offs were key factors affecting efficiency and model accuracy.

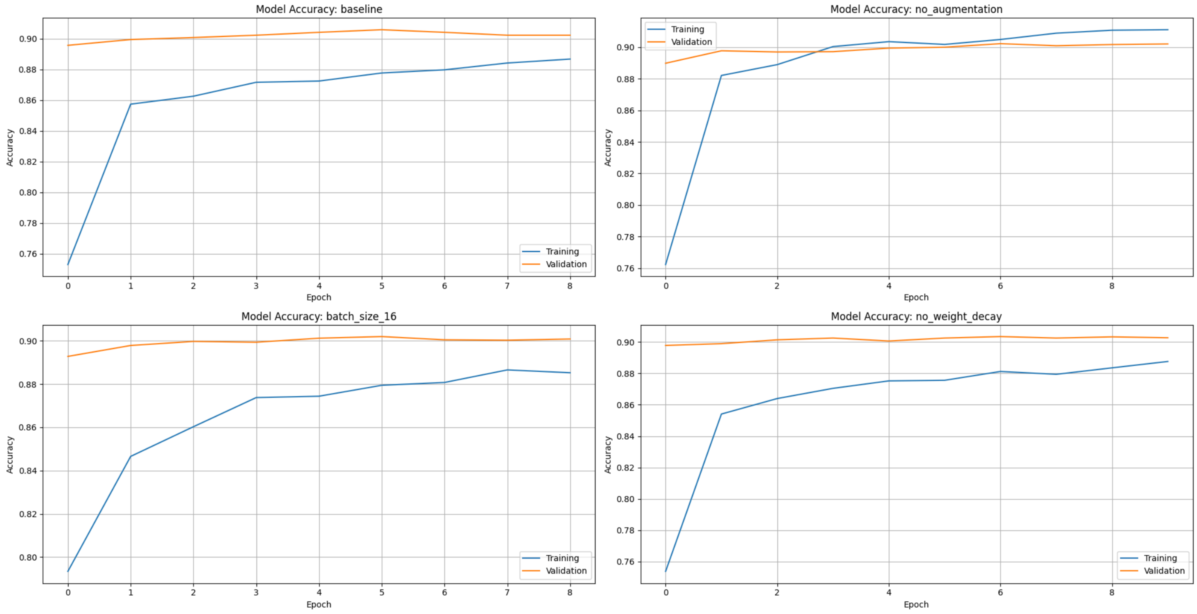


**Figure 5.** Training and validation accuracy curves for all ablation configurations.

# Conclusion

In this project, we successfully tackled the cassava leaf disease classification task by leveraging a lightweight yet powerful model—CropNet from TensorFlow Hub. Through effective data preprocessing, augmentation, and training strategies, our CropNet-based model achieved strong validation performance and a final score of 0.9048 (private) and 0.9070 (public).

We also evaluated a score-level late fusion of EfficientNet-B7 [2] and ViT-B-16 [3] by simply averaging their softmax outputs, but it failed to surpass CropNet in overall accuracy and stability. Accordingly, we retained CropNet for submission, as it offers the best balance of accuracy, robustness, and efficiency.

Overall, our findings reaffirm the strength of transfer learning with lightweight architectures for real-world agricultural image classification and highlight the importance of balancing model complexity with practical effectiveness.

# Reference

[1] Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, et al. Searching for mo bilenetv3. In ICCV, 2019.

[2] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In International conference on machine learning, pages 6105–6114. PMLR, 2019.

[3] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Syl vain Gelly, et al. An image is worth 16x16 words: Transform ers for image recognition at scale. In ICLR, 2021.

[4] Paradisa, R.H.; Bustamam, A.; Mangunwardoyo, W.; Victor, A.A.; Yudantha, A.R.; Anki, P. Deep Feature Vectors Concatenation for Eye Disease Detection Using Fundus Image. Electronics 2022, 11, 23

[5] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2017. ImageNet classification with deep convolutional neural networks. Commun. ACM 60, 6 (June 2017), 84–90.

[6] Simonyan, K.; Zisserman, A. Very Deep Convolutional Networks for Large-Scale Image Recognition. arXiv 2014, arXiv:1409.1556

[7] K. He, X. Zhang, S. Ren and J. Sun, "Deep Residual Learning for Image Recognition," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, USA, 2016, pp. 770-778, doi: 10.1109/CVPR.2016.90.

[8] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco An dreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861, 2017.

[9] Forrest N Iandola, Song Han, Matthew W Moskewicz, Khalid Ashraf, William J Dally, and Kurt Keutzer. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and 0.5 mb model size. arXiv preprint arXiv:1602.07360, 2016.

[10] S. Taheri and Ö. Toygar, ‘‘Animal classification using facial images with score-level fusion,’’ IET Comput. Vis., vol. 12, no. 5, pp. 679–685, Aug. 2018.

| | 110550124 | 110550065 | 110550108 | 110550052 |
|---|---|---|---|---|
| Literature survey | 10 | 30 | 30 | 30 |
| Approach design | 20 | 20 | 50 | 10 |
| Approach implementation (experiment) | 20 | 20 | 50 | 10 |
| Report writing | 10 | 40 | 10 | 40 |
| Slide making and oral presentation | 70 | 10 | 10 | 10 |