CSE455/555

Produce Classification and Variation Detection Using Deep Learning

Jacky He

Abstract

This project explores a vision-based produce classification system using deep learning techniques. Specifically, we develop a two-stage ResNet-based pipeline designed to classify produce type (i.e., Corn vs. Ginger) and their respective variations (e.g., Husked, Kernels, Minced-Sliced). The dataset comprises over 6,000 labeled images, curated from various sources and organized to represent real-world variation. The first model (Type Classifier) distinguishes between corn and ginger with a test accuracy of 76%, while the second model (Variation Classifier) classifies the physical state of the produce into one of six categories, achieving 48.6% test accuracy. Despite not including NLP components due to team limitations, the project delivers a functional proof of concept and demonstrates how computer vision can assist in automated agricultural inspection. A Streamlit-based app was developed for demonstration and testing.

Introduction

In agricultural settings, being able to identify produce types and their physical variations automatically is valuable for streamlining packaging, storage, and retail processes. Visual classification is a promising alternative to manual inspection, especially when implemented with deep learning. This project focuses on two specific crops (i.e. corn and ginger) which are commonly encountered in multiple physical forms that may vary due to processing, handling, or presentation.

We construct a two-stage classification system: one model determines the type of produce, and another infers its visual variation. Using a dataset of over 6,000 labeled images sourced from publicly available sets and refined through internal curation, the models were trained using ResNet-based convolutional neural networks. The Type Classifier reached a 76% test accuracy, while the Variation Classifier achieved 48.6% across six predefined classes.

Although our team lacked an NLP engineer to extend functionality beyond visual classification, we focused on developing a stable and modular image based pipeline. The final product is delivered via a Streamlit web interface, allowing for quick demonstration and user interaction with both classifiers.

Methodology

This project follows a two-stage classification pipeline, each built and trained separately using deep learning techniques. The primary objective is to classify images into two produce types (corn or ginger), and then into one of six visual variations relevant to each. The overall flow is described below in **Dataset Overview**.

Dataset Overview

A combined dataset of over 6,000 images was constructed, covering the following labels:

- Type Classes: **Corn**, **Ginger**
- Variation Classes:
    - **Corn**: Husked, Kernels, Un-Husked
    - **Ginger**: Broken-Peeled, Minced-Sliced, Whole-Hand

Images were sourced from a mixture of public datasets and manually curated samples, ensuring diversity in lighting, orientation, and visual quality. The dataset was then split into:

- **Training Set**: 80%
- **Validation Set**: 10%
- **Test Set**: 10%

Images were resized to 224×224 and normalized to fit model input constraints.

Preprocessing

All images underwent a standard preprocessing pipeline including:

- Resizing to a uniform resolution (224×224)
- Normalization of pixel values to range [0, 1]

Model Architecture

Two independent models were trained using the ResNet50 architecture, initialized with ImageNet weights and fine-tuned on our specific dataset:

- **Type Classifier**
    - **Binary classification:** Corn vs. Ginger
    - **Output Layer:** 2 neurons, Softmax activation
        - Though Sigmoid is the conventional choice for Binary Classification, Softmax was opted as a design choice.
    - **Optimizer:** Adam (lr = 0.0001)
    - **Loss Function:** Categorical Crossentropy
- **Variation Classifier**
    - **Multiclass classification:** 6 variation classes
    - **Output Layer:** 6 neurons, Softmax activation
    - **Optimizer:** Adam (lr = 0.0001)
    - **Loss Function:** Categorical Crossentropy

Both models included Dropout (0.5) in their dense layers to reduce overfitting.

Training Strategy

Each model was trained separately using a supervised learning approach with cross-entropy loss and the Adam optimizer. Both models used a batch size of 32 and ran for a fixed number of epochs, 10 for the Type classifier, and 15 for the Variation classifier.

To monitor performance and prevent overfitting, training metrics were collected per epoch and visualized post-training:

- **Training and Validation Accuracy** (tracked over epochs)

- **Training and Validation Loss** (tracked over epochs)
- **Final Test Accuracy** (evaluated on held-out test set)
- **Confusion Matrix** (plotted for test predictions)
- **Classification Report** (precision, recall, F1-score per class)

Although misclassified examples were visually inspected during evaluation, they were not formally recorded or included in this report.

Both models were saved in .h5 format for compatibility with deployment tools such as Streamlit.

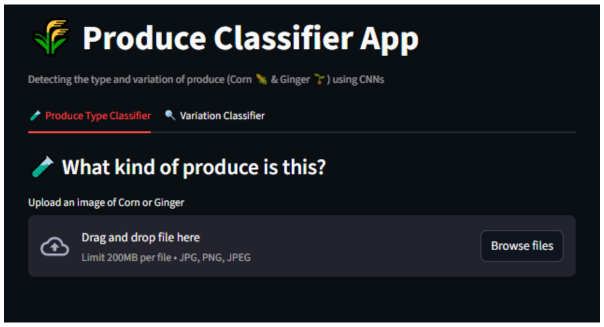
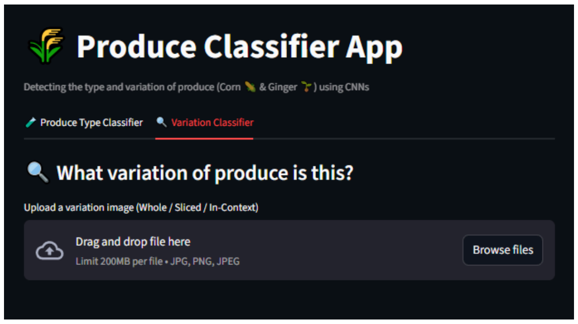
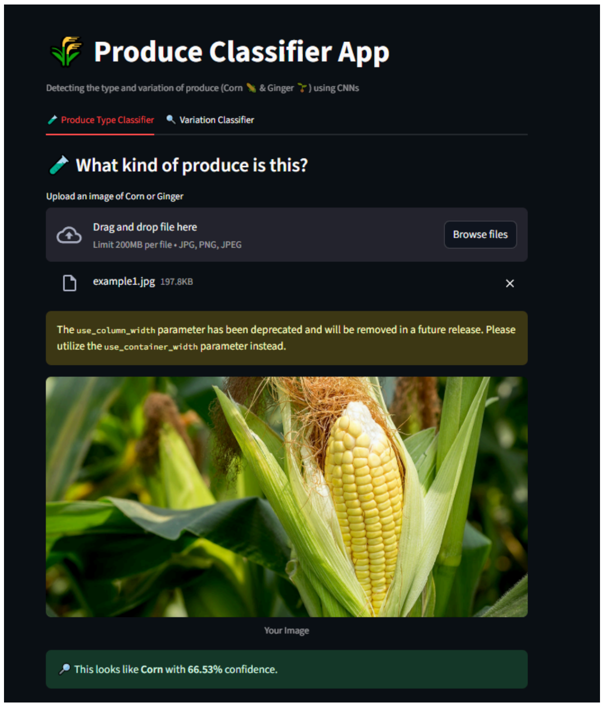
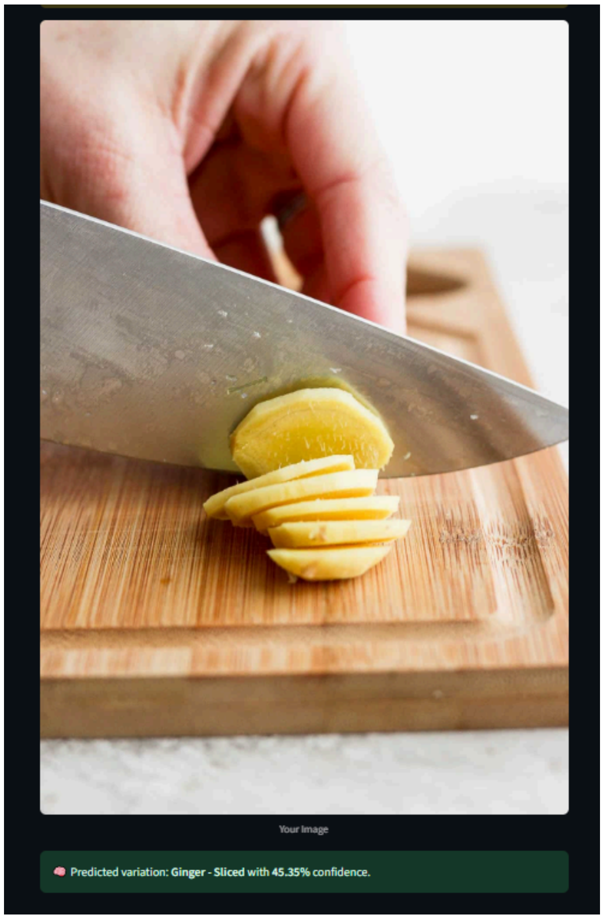
Implementation

The trained models were integrated into a user-facing application using Streamlit, a lightweight Python framework for interactive web apps. Two separate .h5 models—one for produce type classification and another for variation classification, were loaded dynamically based on user selection via tabs or buttons.

Key implementation highlights:

- **Frontend**: Streamlit with image upload interface and prediction outputs.
- **Backend**: TensorFlow/Keras for loading and inferring models.
- **Class Labels:** Manually mapped to ensure semantic clarity (e.g., Corn ->Minced, Ginger -> Whole-Hand, etc.).
- **Runtime:** Designed for local execution due to cloud limitations (e.g., missing TensorFlow packages on Streamlit Cloud).
- **Deployment Attempts:** Though initial deployment was attempted, technical constraints (e.g., tensorflow installation issues) led to a decision to keep it as a locally hosted web app.

The final interface allows users to upload an image and receive both a type (Corn or Ginger) and variation classification (e.g., Husked, Kernels, Broken-Peeled, etc.).

Produce Classifier App Demonstration	Produce Variation App Demonstration
 Produce Classifier App Detecting the type and variation of produce (Corn 🌽 & Ginger 🫚) using CNNs Produce Type Classifier · Variation Classifier What kind of produce is this? Upload an image of Corn or Ginger Drag and drop file here Limit 200MB per file • JPG, PNG, JPEG Browse files	 Produce Classifier App Detecting the type and variation of produce (Corn 🌽 & Ginger 🫚) using CNNs Produce Type Classifier · Variation Classifier What variation of produce is this? Upload a variation image (Whole / Sliced / In-Context) Drag and drop file here Limit 200MB per file • JPG, PNG, JPEG Browse files
 Produce Classifier App Detecting the type and variation of produce (Corn 🌽 & Ginger 🫚) using CNNs Produce Type Classifier · Variation Classifier What kind of produce is this? Upload an image of Corn or Ginger Drag and drop file here Limit 200MB per file • JPG, PNG, JPEG Browse files example1.jpg 197.8KB The `use_column_width` parameter has been deprecated and will be removed in a future release. Please utilize the `use_container_width` parameter instead. Your Image This looks like **Corn** with **66.53%** confidence.	 Your Image Predicted variation: **Ginger - Sliced** with **45.35%** confidence.

Results & Evaluation

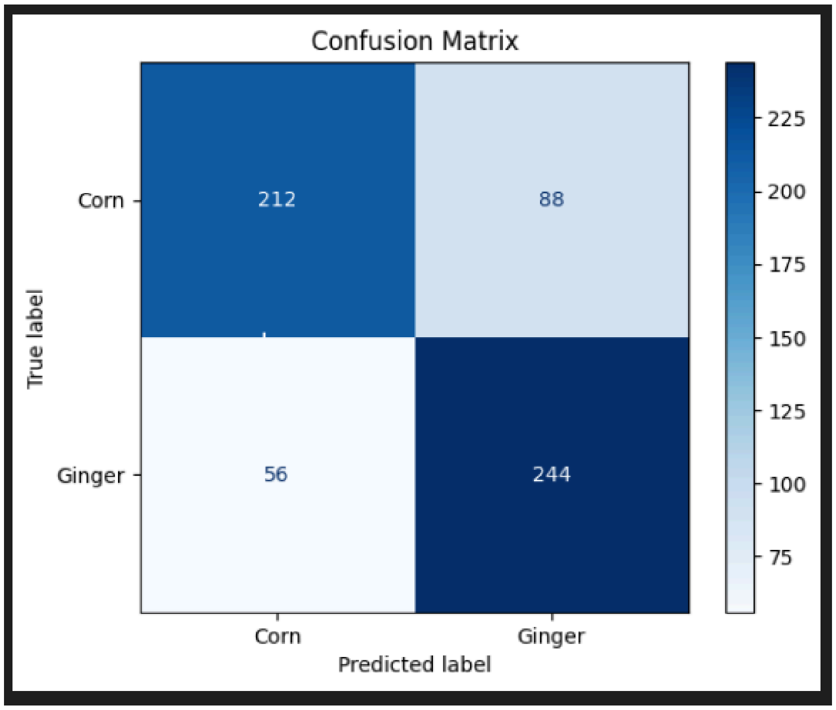
To assess the performance of the classification pipeline, both the Produce Type Classifier and the Variation Classifier were evaluated using standard classification metrics on a test set. Key results include accuracy scores, confusion matrices, classification reports, and training and validation learning curves.

Produce Type Classifier (Corn vs Ginger)

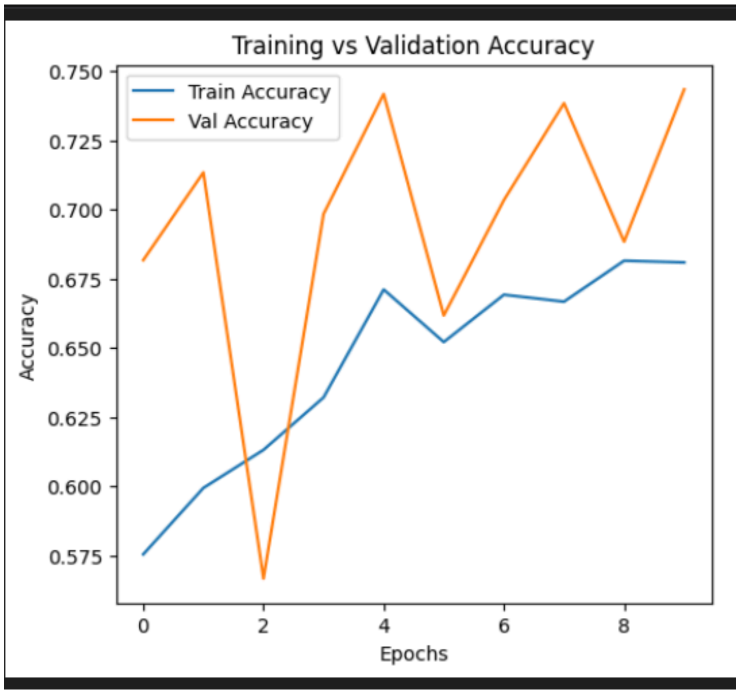
- Test Accuracy: 76.0%
- Epochs Trained: 10
- Metrics Table:

Metric	**Value**
Training Accuracy	~99%
Validation Accuracy	~75%
Test Accuracy	76.0%

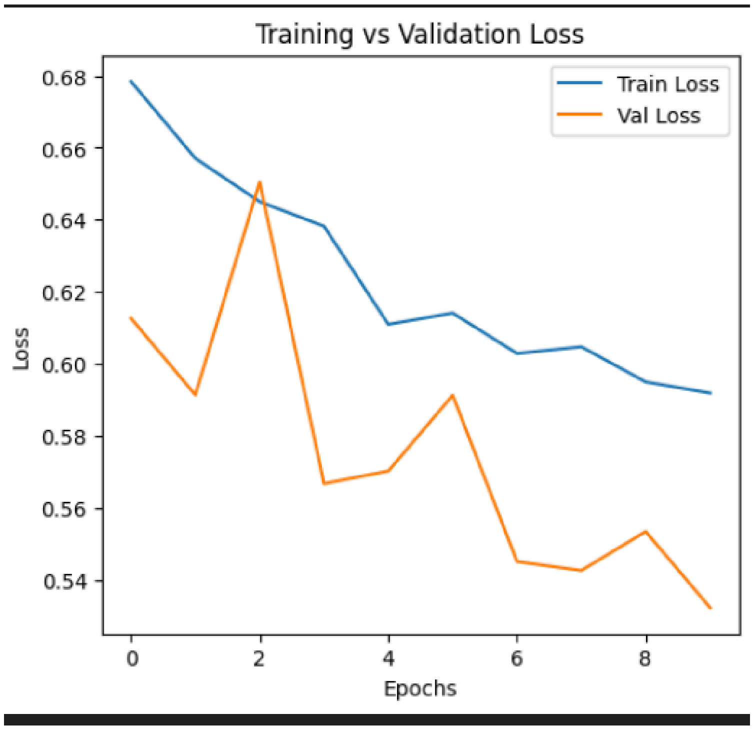
- Confusion Matrix



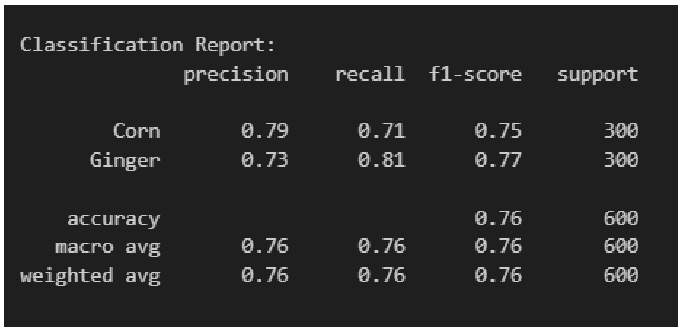
- Accuracy Curve



- Loss Curve



- Classification Report



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

        Corn       0.79      0.71      0.75       300
      Ginger       0.73      0.81      0.77       300

    accuracy                           0.76       600
   macro avg       0.76      0.76      0.76       600
weighted avg       0.76      0.76      0.76       600
```

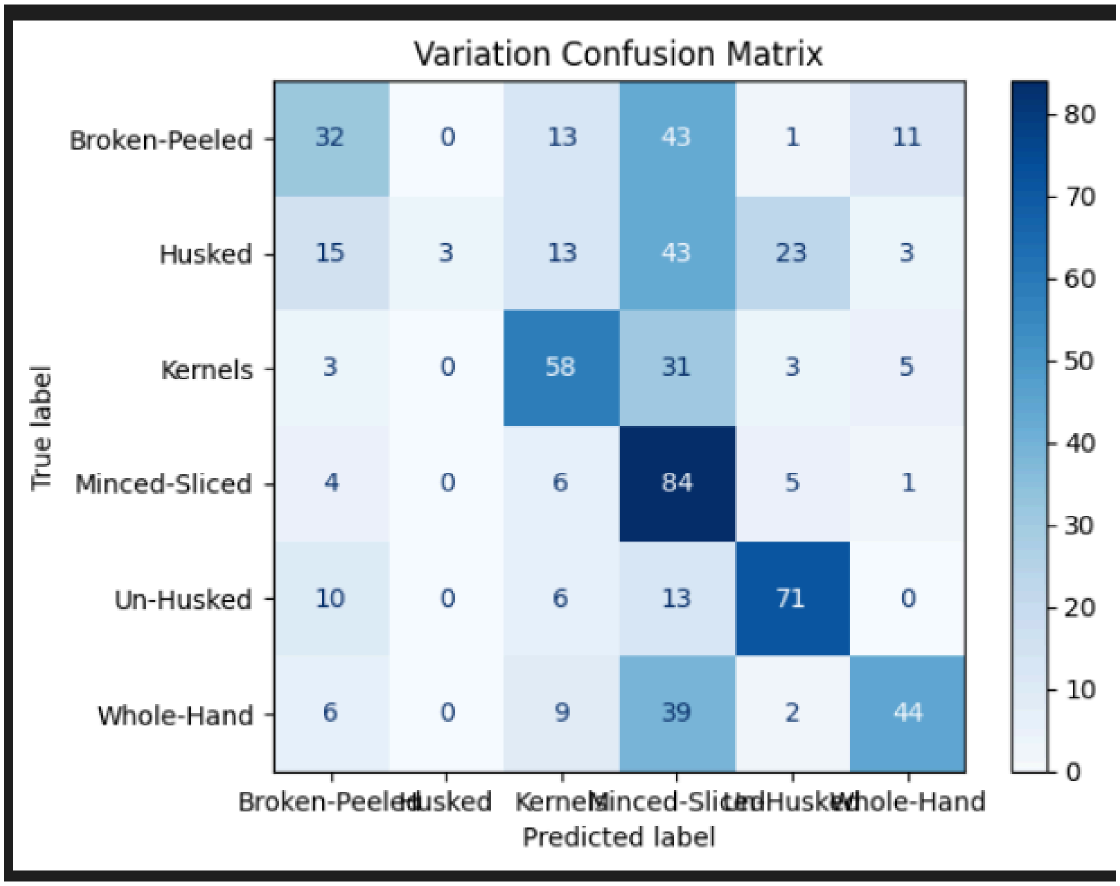
Observations: The model performed well given the binary nature of the task. The small performance drop on validation and test sets suggests some overfitting, but not too bad. Corn and Ginger were both correctly predicted most of the time.

Produce Variation Classifier (6 Classes)

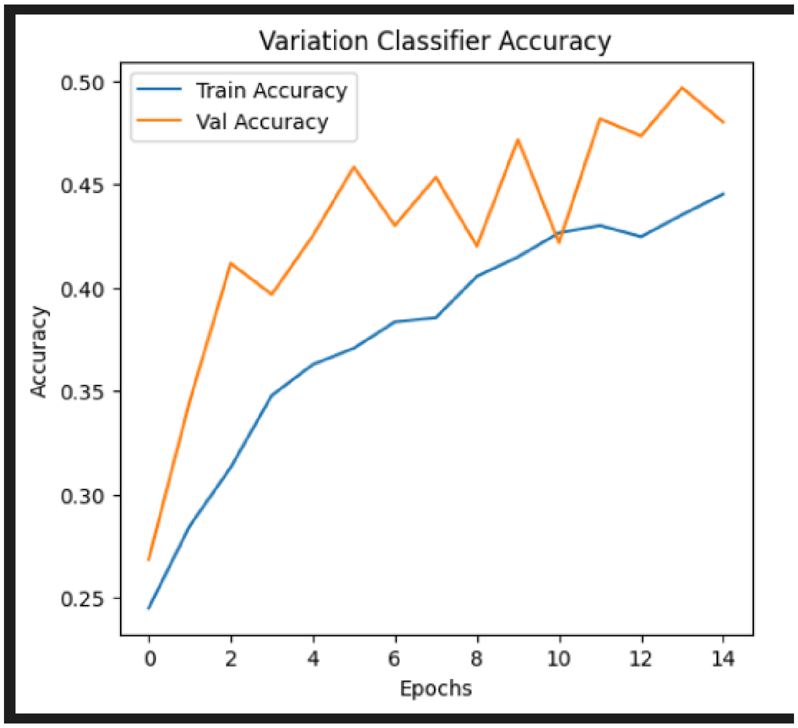
- Test Accuracy: 48.6%
- Epochs Trained: 15
- Metrics Table:

Metric	Value
Training Accuracy	~90%
Validation Accuracy	~48.5%
Test Accuracy	48.6%

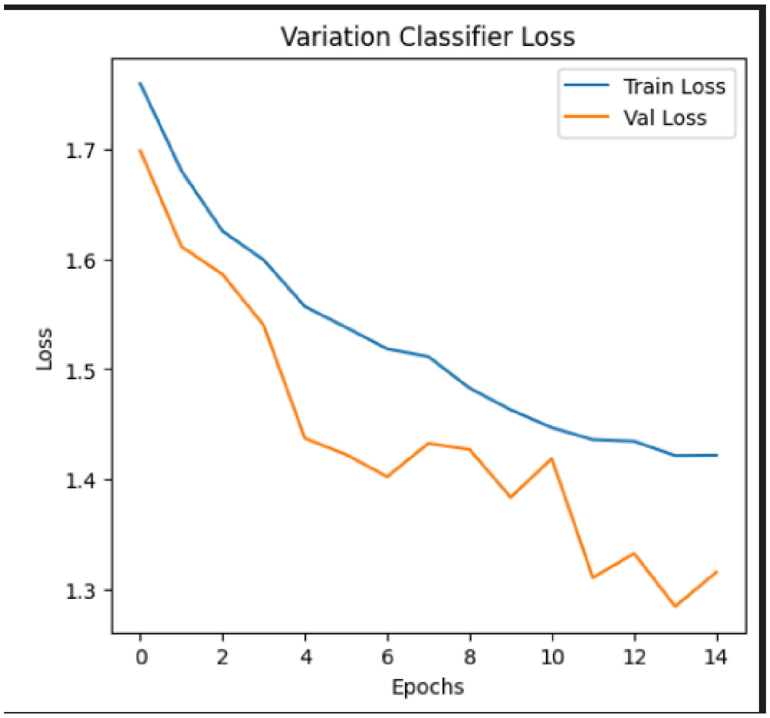
- Confusion Matrix



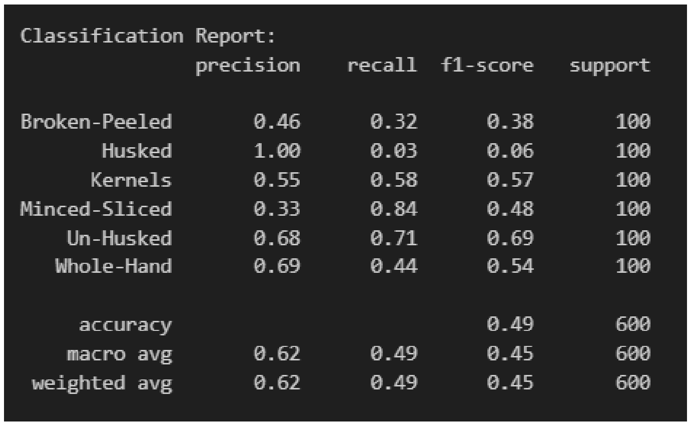
- Accuracy Curve



- Loss Curve



- Classification Report



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

Broken-Peeled       0.46      0.32      0.38       100
       Husked       1.00      0.03      0.06       100
      Kernels       0.55      0.58      0.57       100
Minced-Sliced       0.33      0.84      0.48       100
    Un-Husked       0.68      0.71      0.69       100
   Whole-Hand       0.69      0.44      0.54       100

     accuracy                           0.49       600
    macro avg       0.62      0.49      0.45       600
 weighted avg       0.62      0.49      0.45       600
```

Observations: The variation classifier showed lower generalization. While the model fit the training data well, it struggled to distinguish subtle visual features between classes such as "Broken-Peeled" and "Minced-Sliced."

Discussion

This project revealed both the potential and the limitations of deep learning for agricultural produce classification. The Produce Type Classifier achieved strong generalization, with a test accuracy of 76%, demonstrating that broader, more distinct categories like “Corn” and “Ginger” can be reliably distinguished using standard CNN architectures, such as ResNet.

However, the Variation Classifier, trained to differentiate between six fine-grained visual states, reached a lower accuracy of 48.6%. This was anticipated due to factors like high visual similarity and varying lighting conditions. For example, ginger variations such as "Minced-Sliced" and "Broken-Peeled" share texture and color characteristics that are difficult to isolate visually.

Despite the performance drop in the variation model, both models showed proper convergence behavior with stable training and validation loss curves. Furthermore, results suggest that additional training data, and attention-based architectures, just to name a few, may be beneficial for improving fine-grained variation classification.

While NLP and automatic label processing were excluded due to role limitations, the image pipeline remained the core focus. A functional Streamlit web interface was developed locally serving as a strong proof of concept for real-time application.

Conclusion

This project successfully implemented a deep learning-based pipeline for agricultural produce recognition, focusing exclusively on corn and ginger. With over 6,000 curated images, the two-stage model, comprising a type classifier and variation classifier, was trained, evaluated, and integrated into a lightweight user-facing web interface.

The type classifier performed well, achieving 76% accuracy on the test set. The variation classifier, while less accurate (48.6%), still managed to differentiate among six subtle visual classes. Together, the models provide a foundation for more advanced agricultural visual systems.

Future work may include:

- Expanding the dataset to include more produce types and visual states.
- Applying transfer learning with specific pretraining on a dataset that is relevant to the target task (e.g. a large dataset of Fruits/Vegetables, Plant datasets, etc.)

Despite its challenges, this project demonstrates the real-world feasibility of applying computer vision to agricultural classification tasks.

References

ResNet Paper - “Deep Residual Learning for Image Recognition” (Kaiming He, et al.): https://arxiv.org/abs/1512.03385

Tensorflow Documentation: https://www.tensorflow.org/api_docs/python/tf/all_symbols

Keras Documentation: https://keras.io/