Introduction to Deep Learning - Final Project

Classification of food images through transfer learning and fine tuning of Resnet50 Convolutional Neural Network

Problem Statement

In this project, we address a computer vision task in the domain of food recognition: classifying Kenyan dishes based on images. The dataset consists of 8,174 images across 13 food categories, offering a challenging yet meaningful problem for real-world applications such as automated dietary tracking, restaurant menu digitization, and food logging systems.

Dataset Description

The dataset, obtained from Kaggle, is named **KenyanFood13**. It includes:

- **8,174 images** across **13 classes** (e.g., chapati, mandazi, ugali, etc.).
- A public **training subset** and a **private test subset**, simulating a real-world evaluation setting.
- We manually split the training subset into **training and validation sets** to monitor performance during model development.

The dataset is available at this link:
https://www.kaggle.com/competitions/opencv-pytorch-project-2-classification-round-3/data

Approach

We employed **transfer learning**, leveraging a pretrained **ResNet50** architecture to jumpstart learning with robust feature representations. The pipeline includes:

- **Freezing lower layers** to retain generic features.
- **Replacing the classifier head** to adapt to our 13-class problem.
- **Fine-tuning** upper layers for domain-specific feature refinement.

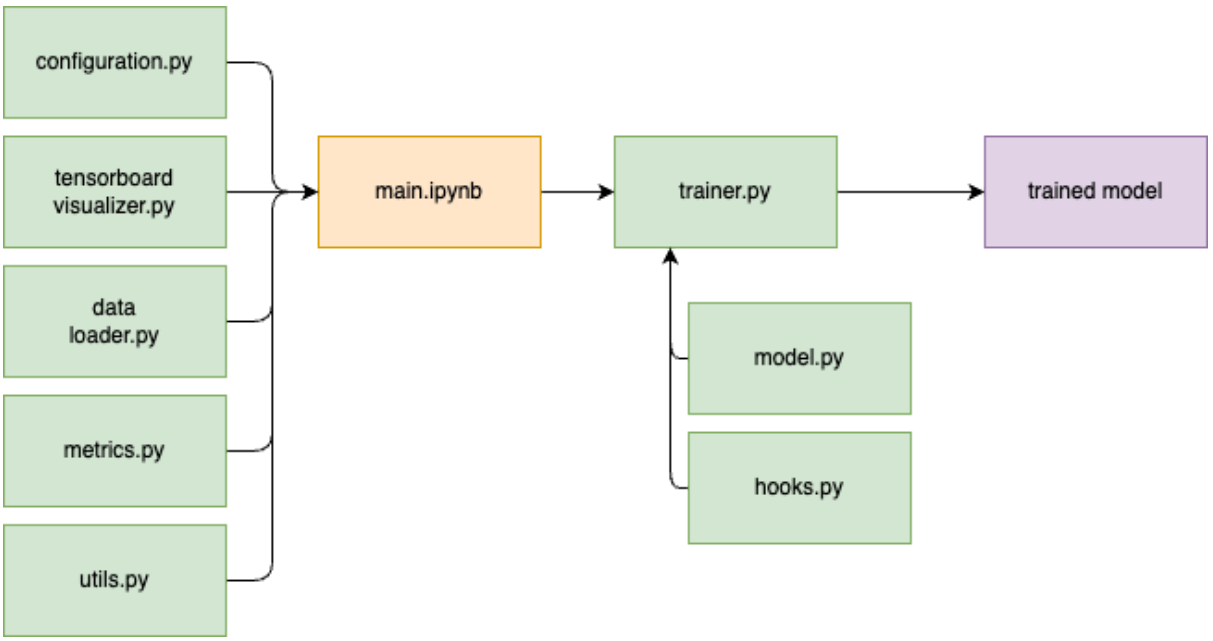
System Design

Our training pipeline was modularized into multiple scripts for maintainability and reuse. The diagram below outlines the structure:

- **`main.ipynb`**: Central notebook for orchestrating the training pipeline.

- **Scripts (under `trainer/`)**:

    - `data_loader.py`: Prepares training/validation datasets.
    - `model.py`: Loads and adapts ResNet50.
    - `trainer.py`: Encapsulates training loop, evaluation, and checkpointing.
    - `metrics.py`, `hooks.py`: Evaluation metrics and callback hooks.
    - `tensorboard_visualizer.py`: Logs to TensorBoard.
    - `configuration.py`, `utils.py`: Configuration and helper functions.

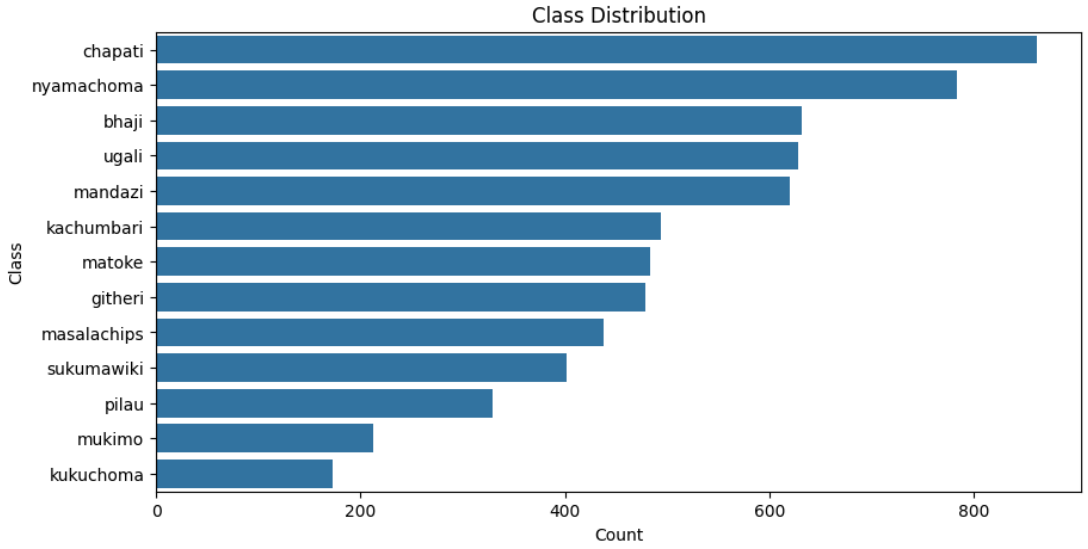

configuration.py
tensorboard visualizer.py
data loader.py
metrics.py
utils.py
main.ipynb
trainer.py
trained model
model.py
hooks.py

Exploratory Data Analysis (EDA)

To better understand the dataset and inform our modeling strategy, we conducted a detailed EDA in a separate notebook (EDA_KenyanFoodClassification.ipynb).

Class Distribution

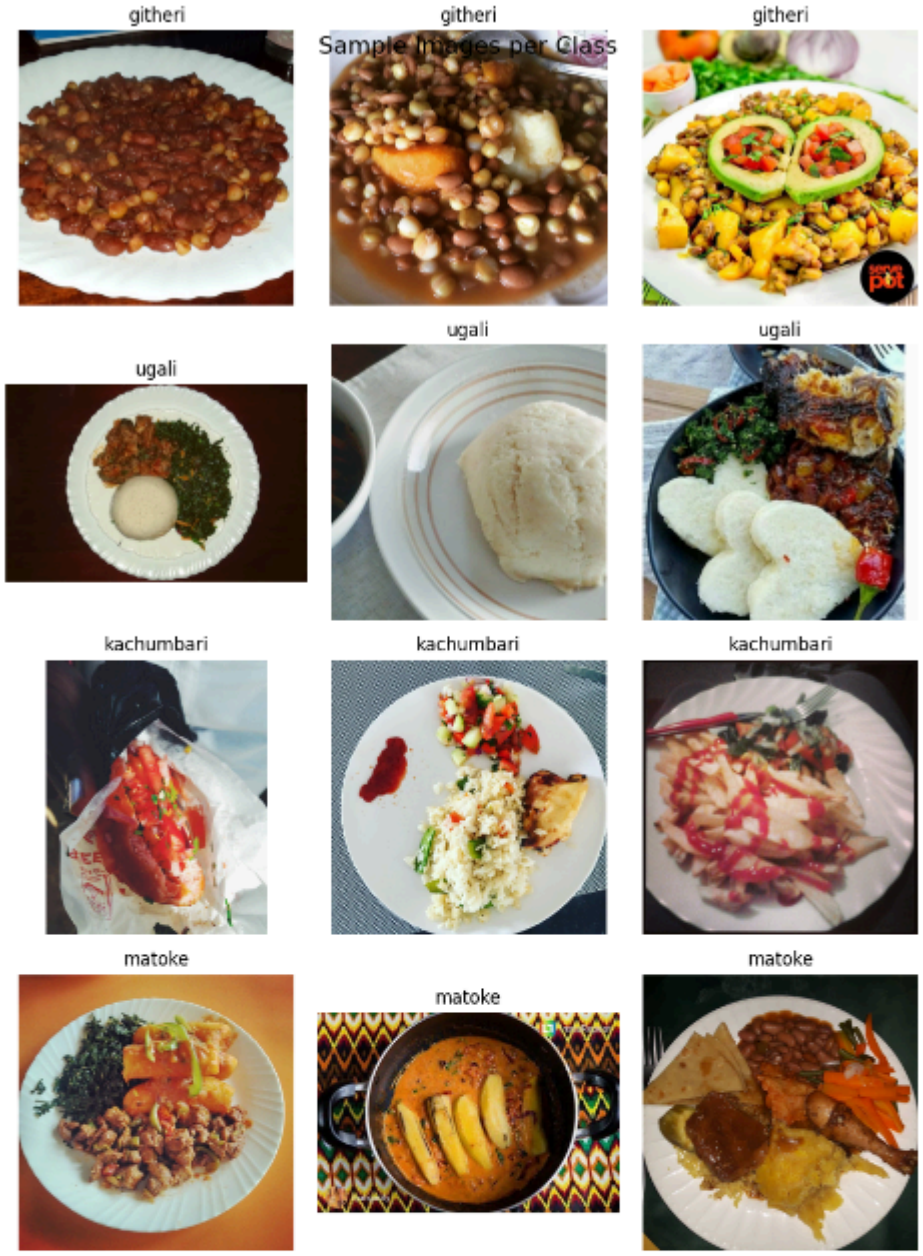
The dataset contains **13 food categories**, with some class imbalance. For example, classes like `chapati` and `nyama choma` have significantly more images than others like `mukimo` and `kukuchoma`.



Sample Images and Class Inspection

Sample images were visualized for each class to:

- Confirm label correctness.
- Observe visual variability.
- Identify noisy or ambiguous samples.



This inspection showed an appreciable intra-class consistency, which supports the viability of a classification model.

Data Augmentation

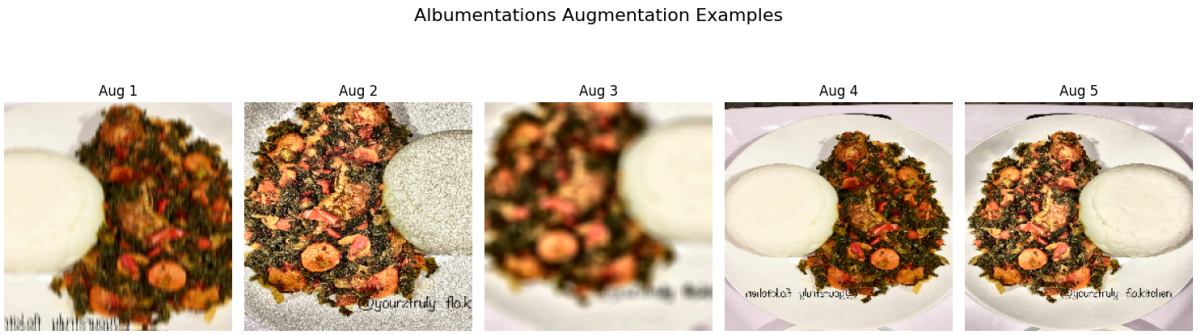
We applied and tested various data augmentation techniques using **Albumentations**, including:

- **Horizontal flipping and 90 degrees rotation**
- **Brightness/contrast adjustments**
- **Normalization** (ImageNet mean and std)

These augmentations were visualized and qualitatively verified to ensure they preserved semantic content while adding useful variance.

Augmentations helped improve generalization and reduce overfitting, especially for underrepresented classes.

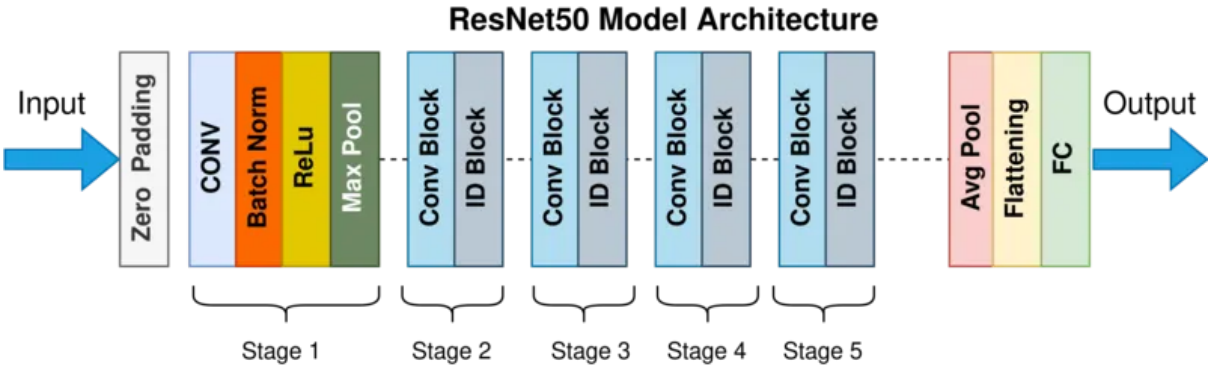
Before these procedures, a proper set of operations for resizing and cutting has been used.



Model Training

- **Architecture**: ResNet50 pretrained on ImageNet
- **Optimizer**: AdamW, with LR scheduler
- **Loss Function**: CrossEntropyLoss
- **Epochs**: 80
- **Augmentations**: as described above
- **Validation**: Monitored Top-1 Accuracy and Loss.

Training metrics were logged using TensorBoard to track learning progress.



https://medium.com/@nitishkundu1993/exploring-resnet50-an-in-depth-look-at-the-model-architecture-and-code-implementation-d8d8fa67e46f

Transfer learning approach: the pretrained Resnet50 model is updated with a new head consisting of a linear layer of perceptron having 2048 input connection and 13 (the number of classes) output connections.

Fine tuning approach: only the last 3 blocks of the Resnet50 were allowed to update their parameters during the training. If less blocks were left frozen, then the model would have tended to overfit.

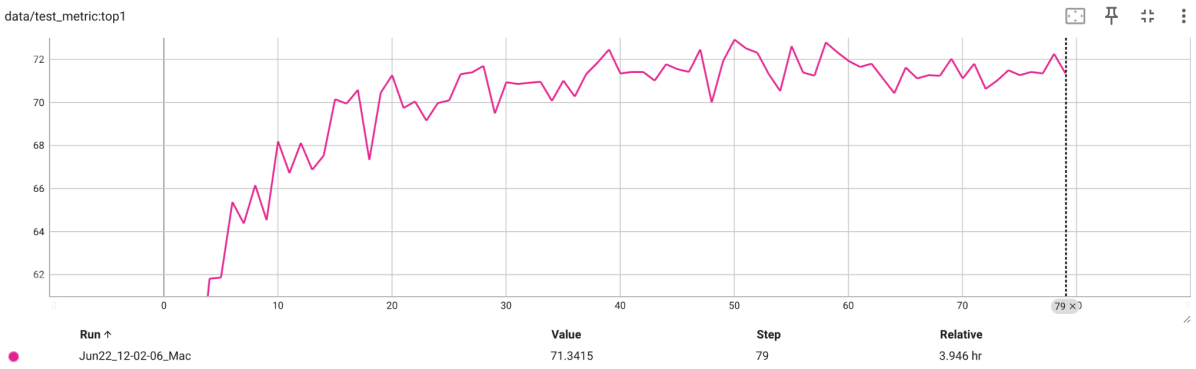
Results

The final trained model achieved:

- **Training Accuracy**: ~80%
- **Validation Accuracy**: ~70%
- **Test Performance** (on unseen data): 68%

Model checkpoints were saved, and the best-performing model was exported for inference.

Tensorboard output for validation accuracy:



Discussion & Conclusion

This project demonstrates the efficacy of transfer learning for food classification in low-data domains. Key insights include:

- ResNet50 generalizes well with limited domain-specific tuning.
- Image augmentations were critical to improving robustness.
- Modular training design facilitated fast iteration and debugging.