# Utilizing Machine Learning to Create an Effective Tool for Managing Food Waste

Rohan Gulati

Advisor: Dr. Kevin Crowthers

**A refrigerator system, which detects the state of food using visual cues, time data, and user-provided images, is a practical solution to food waste. When tested on both known and unknown products, the deep-learning focused system was able to accurately identify spoilage.**

## Need Statement

Urban households need a system which detects whether fresh and cooked foods in **refrigerators** are demonstrating signs of **spoilage** to reduce the amount of **food waste**.

## Objective

Identify food spoilage in common and unfamiliar items in household refrigerators using **visual cues**, insertion and removal times, and user-provided images, and to display the state of foods to **reduce waste**.

## Background

### Problem

Food spoilage leads to:

**Economic Losses**

**1 trillion** dollars worth of food wasted every year (Gravert & Mormann, 2025)

**Environmental Consequences**

**Greenhouse Gas** Production and Resource **Inefficiency** (Reddy et al., 2025)

**Food Safety Hazards**

**1.6 million** casualties due to food contamination (Chowdhury et al., 2025)

**Wasted Potential Impact**

**782 million** people worldwide struggling with **hunger**(Gravert & Mormann, 2025)

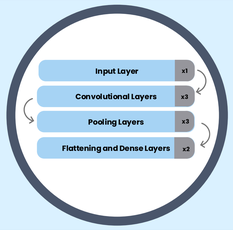
### Competitor Analysis

| Name | Pros | Cons |
|---|---|---|
| KitchenPal™ | • Barcode Scanning<br>• Wide Database<br>• Suggest Recipes<br>• Creates Shopping Lists | • Requires User Input for Every Item<br>• Limited Scans<br>• Cluttered User Interface |
| NoWaste™ | • Barcode Scanning<br>• Stores Quantity and Expiry Date<br>• AI Assistant for Shopping Assistance | • No Automatic Identification<br>• User Provided Expiration Dates<br>• Limited Database |
| This System | • Allows for Automatic Item and Spoilage Detection<br>• Trains Itself for Unrecognizable Items<br>• Clear User Interface | • Struggles with Cluttered Environments<br>• Limited Database<br>• Higher Error Frequency |

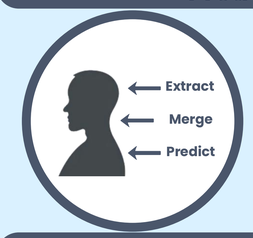
## Methodology

### Spoilage Detection Models

**Convolutional Neural Network**
Input Size: 64 * 64
Filters: 32, 64, 128
Stride: 1
Filter Size: 3 * 3
Max Pooling: 2 * 2
1 Dropout Layer: 40% dropout




### Produce Detection Model




**YOLOv8 Object Detection**
Input Size: 64 * 64
Backbone: 9 Convolutional layers, 8 C2f blocks, 1 SPPF
Neck: 2 Upsample, 3 Concat, 3 C2f blocks
Head: 1 Detection layer
Activation: SiLU

### Self Training Models

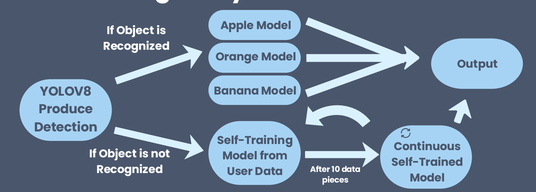
**Transfer Learning: ResNet-18**
pretrained on ImageNet
Input Type: 64×64 RGB images
Folders: Pending, Base
Retrain Trigger: Retrain when Pending reaches 10 new images
Epochs per Retraining: 3 epochs

### Systems Integration

Product Detection: YOLO model identifies the product
Spoilage Classification: Corresponding CNN evaluates spoilage
Unrecognized Items: If the product is not identified, the user inputs labels and images for training with transfer learning

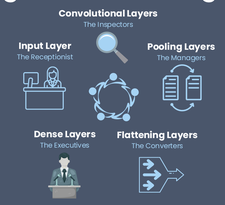
## Results

### Fig 1A. System Structure




### Fig 1B. Data Division


### Fig 1C. Model Design




### Fig 2A. Spoilage Epoch vs. Loss

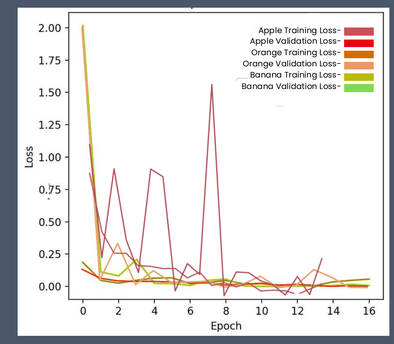
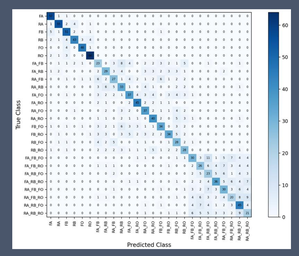



Figure 2A: A strong, constant decrease in both training loss and validation accuracy for all produce over 16 epochs. The similarity between the highest values of **0.0847** for training loss and **0.1594** for validation loss indicate effective learning and limited overfitting. However, there are multiple large spikes for apples, suggesting potential for further training.

### Fig 3A. System Confusion Matrix




### Fig 3B. System Bounding Box Visual

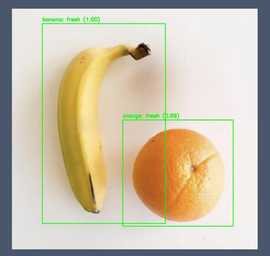



Figure 3A: The systems ability to identify multiple items in an image and whether they are spoiled. There is a major **increase** in error whenever **bananas** are present in an image. Additionally, error is more frequent with **more items** in the image.

Figure 3B: A visual of the **bounding boxes** created by the system to identify the objects. It identifies the item name, its spoilage state, and the confidence of the model's decision.

### Fig 2B. Spoilage ROC Curve

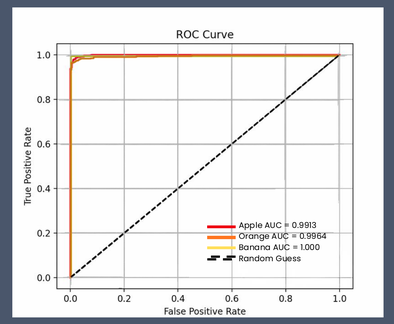



Figure 2B: The models' ability to distinguish between classes across all thresholds. The AUC (Area under the curve) values of **0.9913, 0.9964, and 1.000** show that the models can achieve high true positive rates while maintaining low false positive rates. This indicates a **strong overall performance** of the models. However, the AUC of 1.000 for the banana's model is concerning, as it may be a sign of overfitting.

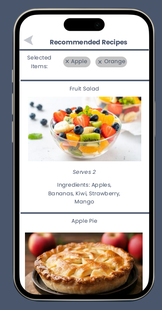
### Custom Food Recognition


Tracks item **names,** their **spoilage state**, and the approximate **time** to spoilage.

### Tracking the State of Produce


Allows the user to **select/add** an item and **attach** images for training.

### Smart Recipes from Spoiling Items




Displays recommended **recipes prioritizing** items approaching spoilage.

## Highlights

- The model shows a **strong ability** to identify spoilage in **pre-trained products** including apples, oranges , and bananas, with accuracies of 98.19%, 97.72%, and 98.49%, respectively.
- However, the YOLOv8 model struggles to classify the **location** of multiple products.
- For training, more data with a variety of crowded foods is required to eventually use **transfer learning** for an accurate model.

## Next Steps

Optimize the model for **real-time deployment** (motion blur) in environments which are **not ideal.**

To capture **more training data,** create a system which identifies images online, confirms their authenticity, and **creates datasets** for various labels.

A **collaborative** self-learning model, where user-labeled images of unfamiliar products are **shared across users** with the same labels.