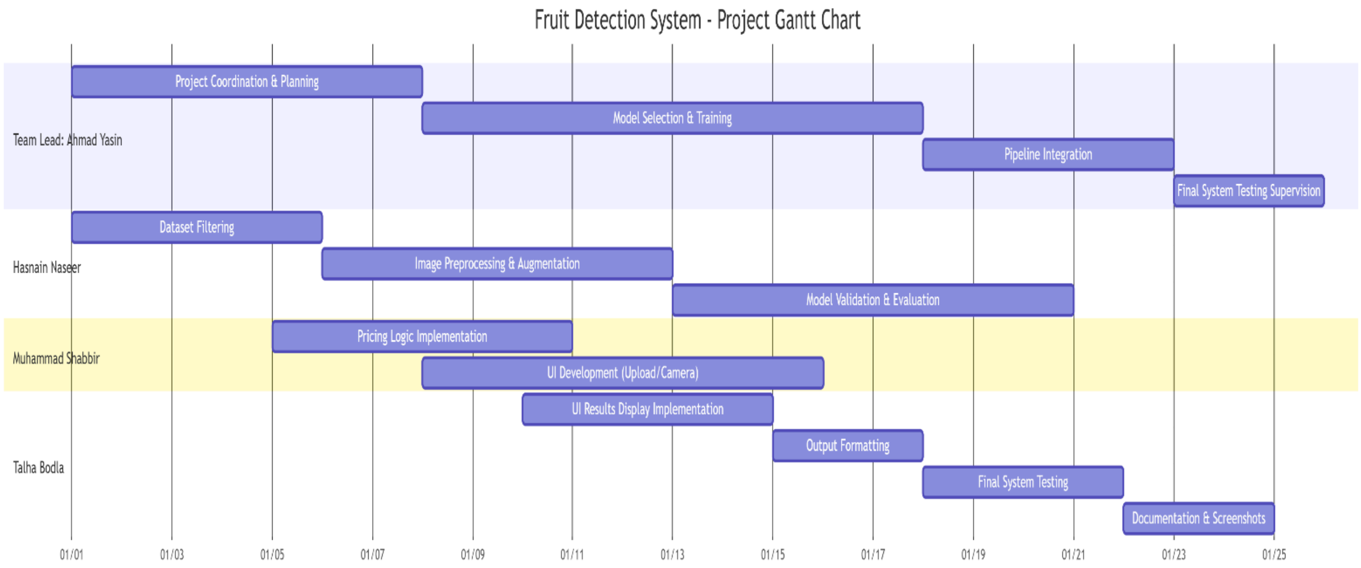
Project Proposal

Title: Fruit Classification System

Table of Contents

Gant Chart



Problem Statement

Manual fruit classification and billing requires human effort, can introduce errors, and is inefficient for fast-paced environments. There is a need for an automated solution that can:

- Detect multiple fruits in a single image
- Identify fruit types accurately
- Count the number of each fruit
- Assign predefined prices
- Calculate the total cost automatically

Team Members

Ahmad Yasin (Team Lead)

- Overall project coordination
- Model selection and training (YOLO)
- Integration of detection with pricing logic
- Final system testing

Hasnain Naseer

- Dataset filtering from Fruit-Images-Dataset
- Image preprocessing and augmentation
- Model evaluation (accuracy, precision)

Muhammad Shabbir

- Pricing logic implementation
- UI development (camera/image upload)
- Display of detected fruits, quantity, and total price

Talha Bodla

- Writing **project documentation sections** (Introduction, Objectives)
- Assisting in **report formatting** (tables, headings, references)
- Basic testing and screenshot collection for report

Introduction and Objectives

With the rapid advancement of artificial intelligence and computer vision, automated object detection systems are becoming increasingly useful in real-world applications such as smart retail, self-checkout systems, and agricultural monitoring. Traditional methods of identifying fruits, counting quantities, and calculating prices require manual effort, which can be time-consuming and prone to human error.

This project proposes an **AI-based Fruit Detection and Pricing System** that allows users to **capture images using a camera or upload images**, automatically detect **multiple fruits in a single image**, identify the **fruit type, quantity, and color**, and calculate the **total price using predefined**

rates. The system is designed to be simple, efficient, and user-friendly.

Key Objectives:

- To detect and classify fruits from images using computer vision
- To support **multiple fruits in one image**
- To estimate fruit quantity by counting detected objects
- To identify fruit color using image analysis
- To calculate total price using predefined fruit rates
- To develop a simple user interface for image upload or camera input

Dataset Description

1 Dataset Source

- **Name:** Fruit-Images-Dataset
- **Author:** Horea94
- **Link:** https://github.com/Horea94/Fruit-Images-Dataset

2 Dataset Overview

The Fruit-Images-Dataset contains high-quality images of various fruits captured under controlled conditions with consistent backgrounds. For this project, the dataset will be filtered to include selected fruit classes such as:

- Apple
- Banana
- Orange
- Mango

3 Data Preprocessing

- Image resizing (e.g., 416 × 416 pixels)
- Normalization
- Data augmentation (rotation, flipping, brightness adjustment)
- Dataset split:
    - 80% Training
    - 10% Validation
    - 10% Testing

Methodology and Techniques

- **Primary Technique: YOLOv8 (You Only Look Once)**:

  1. **Why?**: State-of-the-art for real-time object detection. It detects objects in a single pass, outputting bounding boxes and class probabilities, ideal for counting quantities.
  2. **Implementation**: Fine-tune a pre-trained YOLOv8 model on the dataset. Train for 50-100 epochs with data augmentation.
  3. **Advantages**: Fast inference (suitable for real-time apps), handles multiple objects per image, and provides confidence scores for filtering detections.

- **Supporting Techniques**:

  1. **CNN-Based Classification (Convolutional Neural Networks)**: Use ResNet50 or Efficient-Net as a backbone in YOLO for feature extraction and classification into the 4 classes.
  2. **Image Pre-processing**: OpenCV for resizing, normalization, and noise reduction.
  3. **Post-Processing**: Non-Maximum Suppression (NMS) to eliminate overlapping detections; threshold confidence >0.5 for accurate counting.
  4. **Quantity Estimation**: Count bounding boxes per class (e.g., 3 Apples detected).
  5. **Pricing Integration**: Use a Python dictionary for prices. Calculate total: `total = sum(quantity[class] * price[class] for class in detected_classes)`.

- **Training and Evaluation**:

  1. **Metrics**: mAP@0.5 (mean Average Precision) for detection accuracy; precision/recall for classification.
  2. **Tools**: PyTorch for YOLOv8, Weights & Biases for logging.
  3. **Expected Performance**: >85% mAP on test set, with low false positives.

Proposed System Overview

The proposed system is an AI-based fruit detection and pricing application that allows users to capture images using a camera or upload images from a device. The system automatically detects multiple fruits in a single image, identifies their type, quantity, and color, and calculates the total price using predefined rates. The overall system is designed to be efficient, user-friendly, and suitable for real-world applications such as smart retail and automated billing.

□ System Workflow

The system follows a structured workflow to ensure accurate detection and pricing:

1. The user captures or uploads an image using the system interface.
2. The input image is preprocessed (resizing and normalization).
3. The trained object detection model detects fruits in the image.
4. Each detected fruit is classified into its respective category.
5. The quantity of each fruit type is calculated by counting detections.
6. Fruit color is identified using image processing techniques.
7. Predefined prices are applied to each fruit category.
8. The total price is calculated and displayed to the user.

□ System Architecture

The system architecture consists of the following main components:

- **Image Acquisition Module:** Captures images through a camera or allows image upload.
- **Preprocessing Module:** Resizes and normalizes images for model input.
- **Fruit Detection Module:** Uses a YOLO-based deep learning model to detect and classify fruits.
- **Quantity & Color Analysis Module:** Counts detected fruits and determines their color.
- **Pricing Module:** Applies predefined prices and calculates the total cost.
- **User Interface Module:** Displays detected fruits, quantities, colors, and total price.

Tools and Technologies

- **Programming**: Python 3.x

- **Libraries**: PyTorch, Ultralytics YOLOv8, OpenCV, NumPy
- **Environment**:

  1. Google Colab or local machine
  2. Camera or image upload support

Performance Evaluation

1. Accuracy of fruit detection
2. Correct quantity estimation
3. Correct price calculation
4. System response time
5. Web app (Flask/Stream_lit) for image upload and results display (e.g., "Detected: 2 Apples (1.50) - Total: $5.50").
6. Report on accuracy, challenges (e.g., overlapping fruits), and improvements.

Applications

- Smart retail checkout systems
- Automated fruit shops
- Agricultural monitoring
- Educational AI projects

Conclusion

This project demonstrates the practical use of **computer vision and artificial intelligence** in automating fruit detection and pricing. By utilizing the **Fruit-Images-Dataset by Horea94**, the proposed system efficiently detects multiple fruits from a single image, estimates quantities, identifies colors, and calculates total prices automatically. The system highlights how AI can improve efficiency, accuracy, and user experience in real-world applications.

Reference

Link: https://github.com/Horea94/Fruit-Images-Dataset