FRUIT OBJECT DETECTION USING YOLOv8

By Anubriya Baskaran

B.Tech(CSE), M.Sc(Data Science in UK)

Final Project

Batch: DS-S-WD-T-B79

Hugging Face App Link

https://anubriya-fruit-detection-using-yolov8.hf.space/

Users can test images from anywhere.

Abstract

Object detection is one of the most widely applied tasks in computer vision, enabling machines to identify and localize multiple objects within an image.
This project focuses on detecting **three fruit classes — Apple, Banana, and Orange** — using the **YOLOv8 model** from Ultralytics.

A custom dataset was annotated, trained, validated, and evaluated using YOLOv8. The model achieved strong performance.
A **Streamlit web application** was developed to allow users to upload an image and view real-time object detection results.
Finally, the system was **deployed on Hugging Face Spaces**, making it accessible online.

1. Introduction

Fruit detection and classification play a significant role in modern computer vision applications, particularly in domains where automation, accuracy, and efficiency are critical. With the rapid advancement of deep learning and convolutional neural networks (CNNs), object detection models have achieved remarkable performance in identifying and localizing objects within images.

Automated fruit detection systems are widely useful in several real-world scenarios, including:

- Supermarkets and retail stores – enabling automated checkout systems, inventory management, and reduced human intervention
- Fruit sorting and grading systems – improving quality control by classifying fruits based on type and appearance
- Smart kitchens and home automation – assisting in food recognition and nutritional tracking
- Agricultural automation – supporting yield estimation, crop monitoring, and harvesting robots

Traditional image processing techniques struggle with variations in lighting, background clutter, occlusion, and different fruit sizes. Deep learning-based object detection models, such as YOLO (You Only Look Once), overcome these challenges by learning robust visual features directly from data.

The goal of this project is to design and implement an **end-to-end fruit object detection system** using a modern deep learning approach. A custom YOLOv8 model is trained to detect three fruit classes: **Apple, Banana, and Orange**. The project covers the complete machine learning pipeline, including dataset preparation, annotation, model training, validation, performance evaluation, and deployment.

To ensure usability beyond experimentation, a **Streamlit-based web application** is developed, allowing users to upload images and view real-time detection results. The application is deployed on **Hugging Face Spaces**, making it publicly accessible through a web interface. This project demonstrates not only the technical implementation of an object detection model but also its practical deployment in a real-world environment.

2. Objectives

 To build a object detection model for fruits using YOLOv8

 To prepare and organize a dataset with proper annotations

 To evaluate the trained model using standard metrics such as Precision, Recall, and mAP

 To visualize detection results on unseen test images

 To deploy the trained model using a web application

 To host the application online for public access

3. Dataset Description and Preparation

3.1 Dataset Overview

The dataset consists of images containing three fruit classes:

- **Apple**
- **Banana**
- **Orange**

Each image may contain **one or multiple fruits**, and bounding boxes are provided for every object.

3.2 Dataset Split

The dataset was divided as follows:

- **Training set:** 80% of images
- **Validation set:** 20% of images
- **Test set:** 100% separate unseen images

This split ensures proper learning, tuning, and unbiased evaluation of the model.

3.3 Annotation Format (YOLO Format)

Annotations are stored in **YOLO .txt format**, where each line represents one object:

class_id x_center y_center width height

- class_id → Numerical ID of the class
- x_center, y_center → Center of bounding box (normalized)
- width, height → Size of bounding box (normalized)

All values are normalized between **0 and 1** relative to image dimensions.

3.4 Annotation Conversion

Original annotations were available in XML format, which was converted to YOLO .txt format to match YOLOv8 requirements.

3.5 YAML Configuration File

A data.yaml file was created to define:

- Paths to training, validation, and test datasets
- Number of classes
- Class names

The YAML file acts as a **central configuration** that YOLO uses during training and evaluation.

4. Model Architecture – YOLOv8

YOLOv8 (You Only Look Once version 8) is a state-of-the-art object detection model developed by Ultralytics. It performs object detection in a single forward pass, making it extremely fast and accurate.

Key Features of YOLOv8:

- Anchor-free detection
- Improved loss functions
- Faster inference

- High accuracy with fewer parameters

For this project, a **YOLOv8 lightweight model** was trained to balance performance and speed.

5. Model Training

5.1 Training Process

The model was trained using:

- Custom fruit dataset
- YOLOv8 training pipeline
- 30 epochs
- GPU acceleration during training

YOLOv8 internally applies data augmentation such as flipping, scaling, and color variations to improve generalization.

5.2 Saving the Model

After training, the best performing model was saved as:

best_fruit_model.pt

This model file is later used for evaluation, prediction, and deployment.

6. Model Evaluation

The trained model was evaluated separately on:

- Validation set
- Test set

6.1 Evaluation Metrics

The following metrics were used:

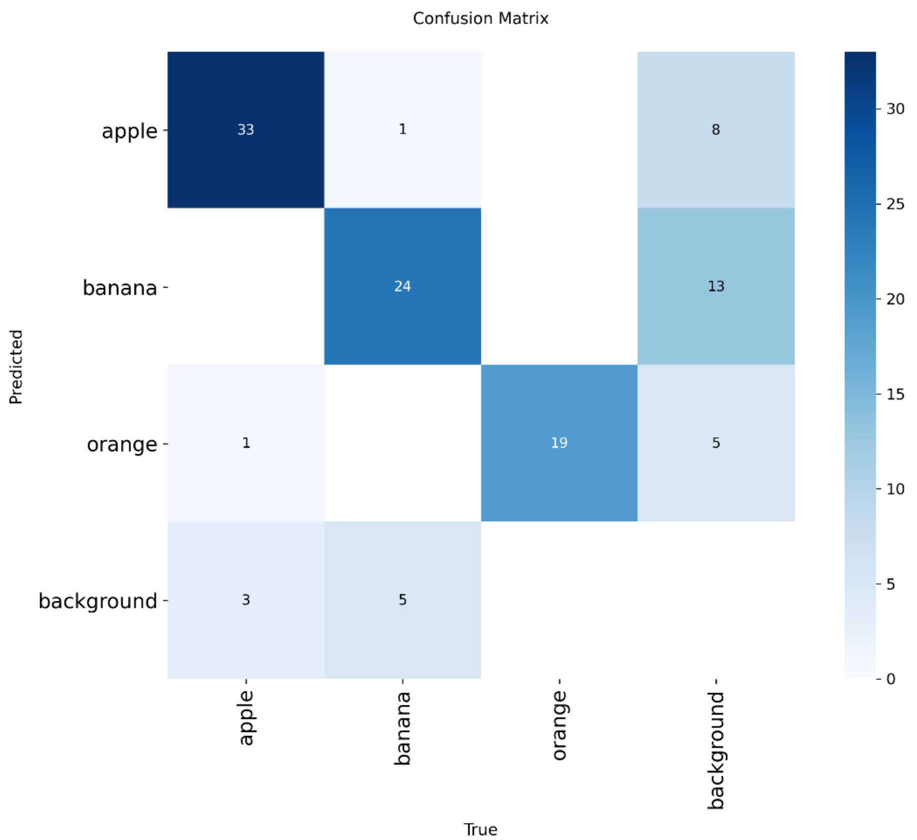
- **Precision (P)** – Correct detections among predicted detections
- **Recall (R)** – Correct detections among actual objects
- **mAP@50** – Mean Average Precision at IoU threshold 0.50
- **mAP@50-95** – Average mAP over IoU thresholds from 0.50 to 0.95

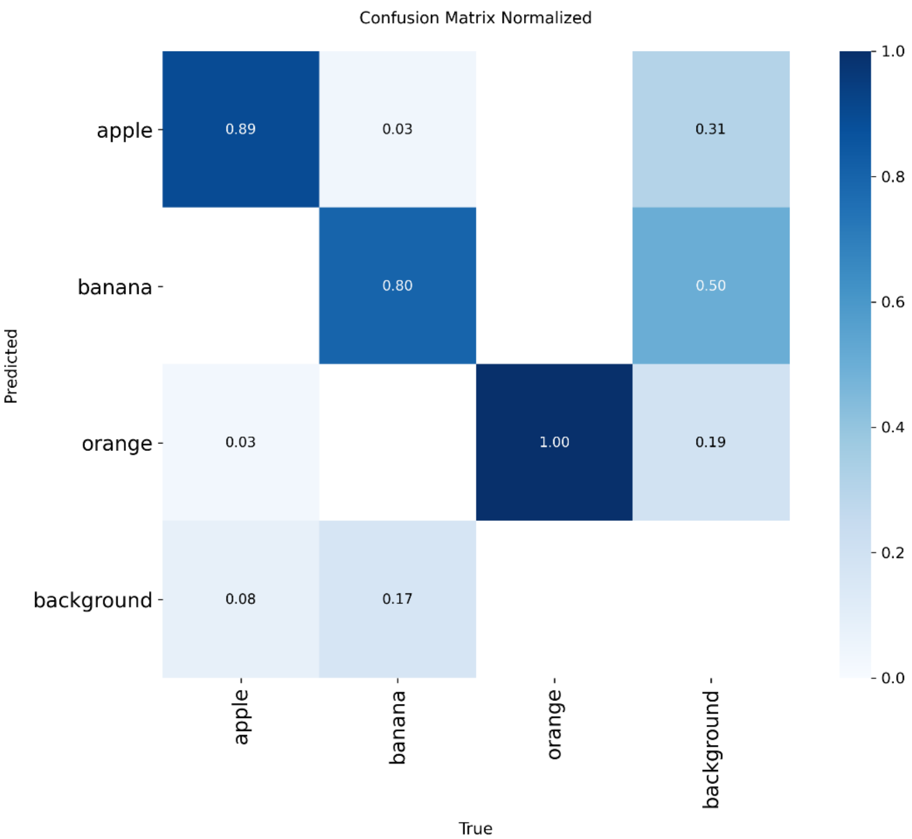
6.2 Confusion Matrix

- **Confusion Matrix** shows correct and incorrect class predictions
- **Normalized Confusion Matrix** shows class-wise performance in percentage

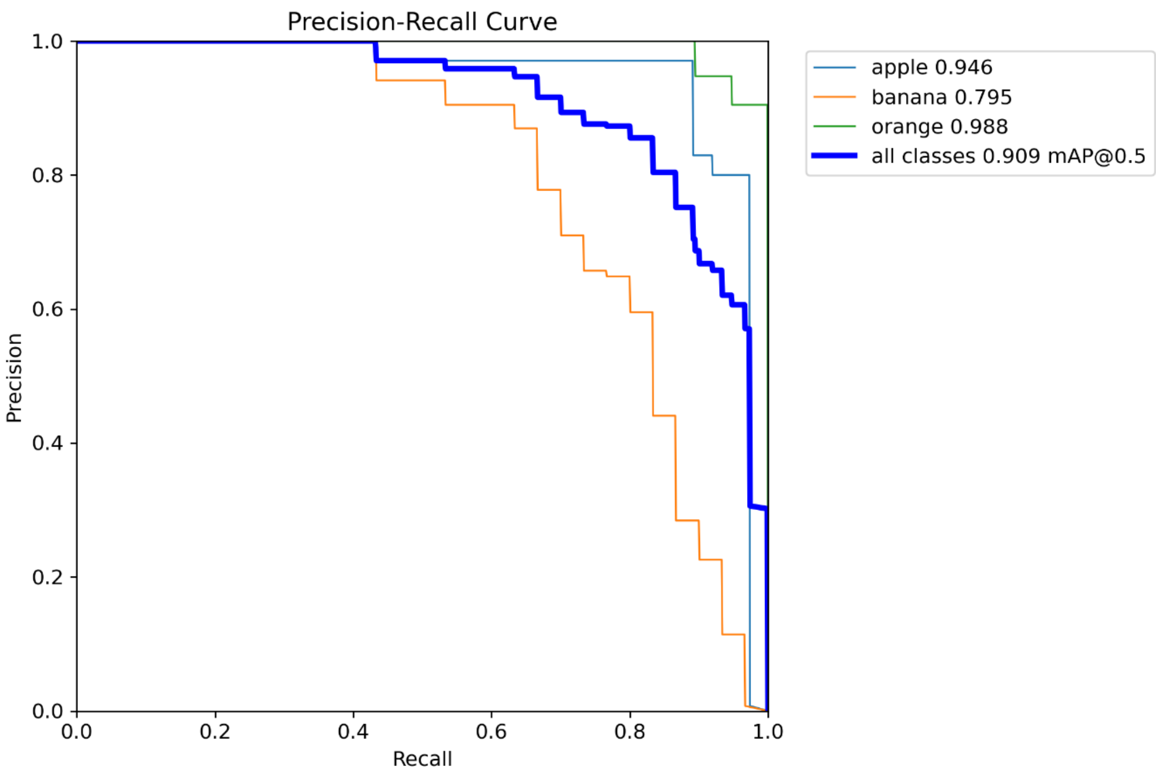
These visualizations help understand which classes are confused with others.

6.2.1 Validation Set



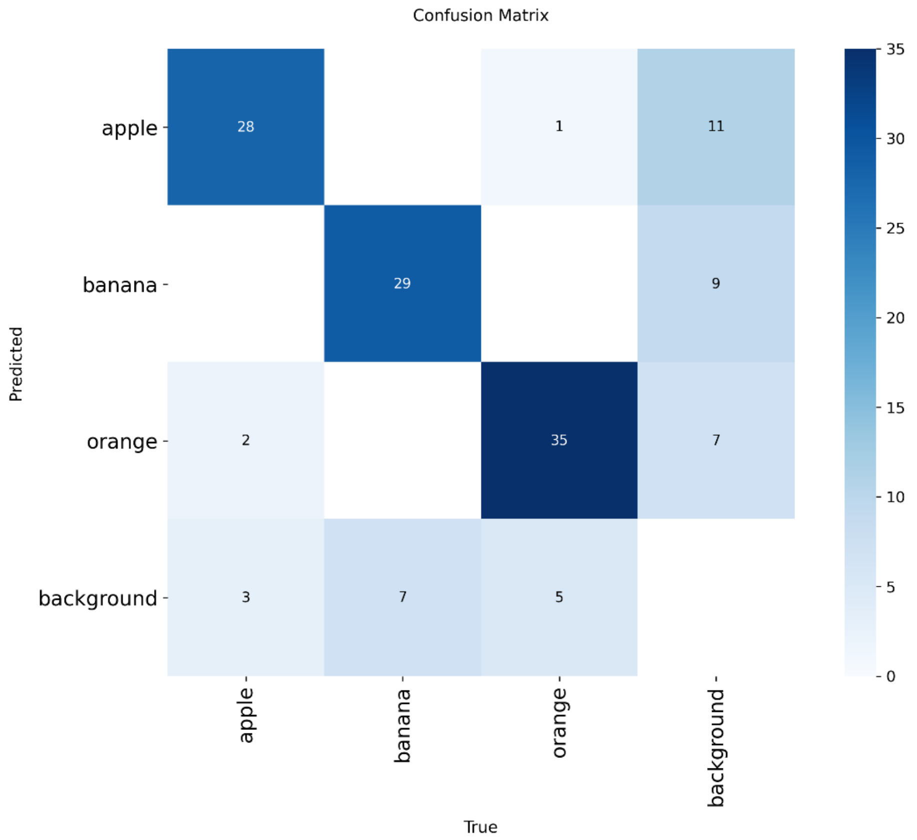


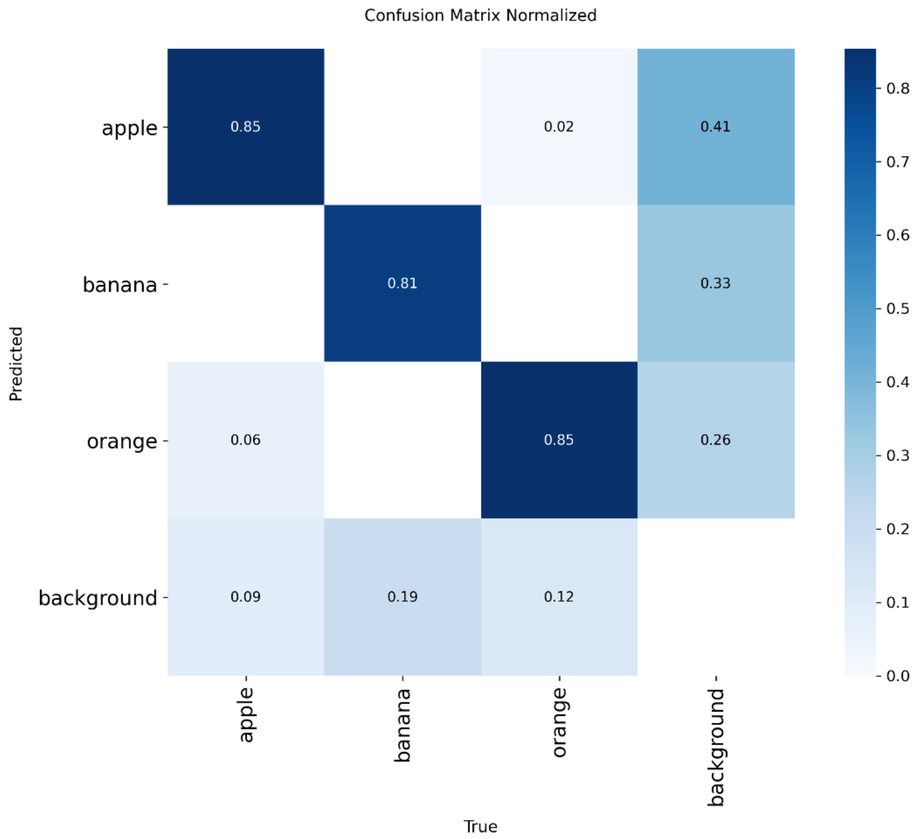
Confusion Matrix Normalized
apple
banana
orange
background
Predicted
0.89
0.03
0.31
0.80
0.50
0.03
1.00
0.19
0.08
0.17
apple
banana
orange
background
True
1.0
0.8
0.6
0.4
0.2
0.0

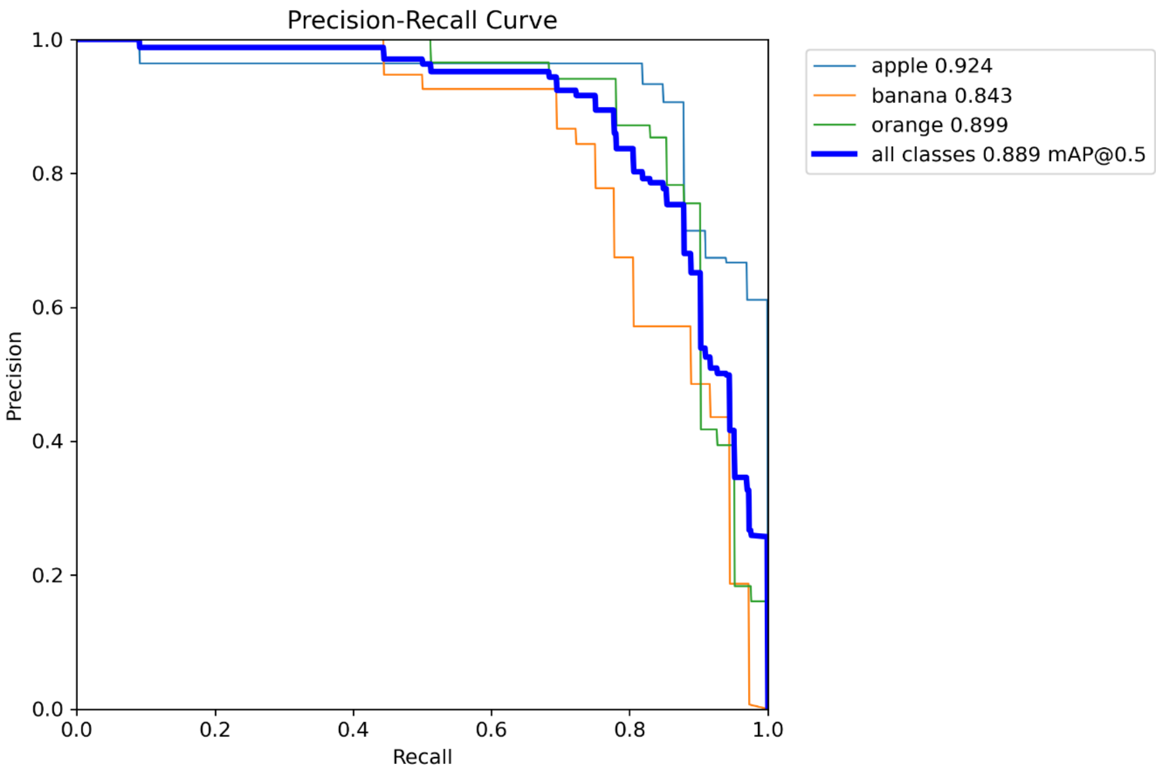


Precision-Recall Curve
1.0
0.8
0.6
0.4
0.2
0.0
Precision
0.0
0.2
0.4
0.6
0.8
1.0
Recall
apple 0.946
banana 0.795
orange 0.988
all classes 0.909 mAP@0.5

6.2.2 Test Set


Confusion Matrix
apple
banana
orange
background
Predicted
28
1
11
29
9
2
35
7
3
7
5
apple
banana
orange
background
True
35
30
25
20
15
10
5
0


Confusion Matrix Normalized
apple
banana
orange
background
Predicted
0.85
0.02
0.41
0.81
0.33
0.06
0.85
0.26
0.09
0.19
0.12
apple
banana
orange
background
True
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0



6.3 Performance Summary

- Apples and Oranges achieved **high precision and recall**
- Bananas showed slightly lower recall due to visual similarity in some images
- Overall model performance is **strong and reliable**

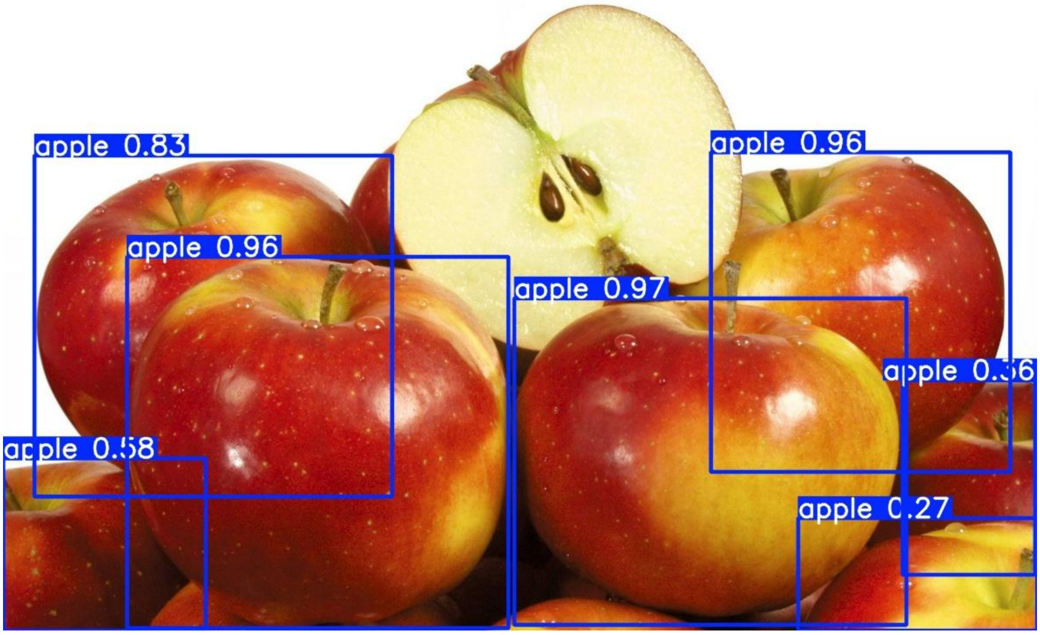
7. Prediction on Test Images

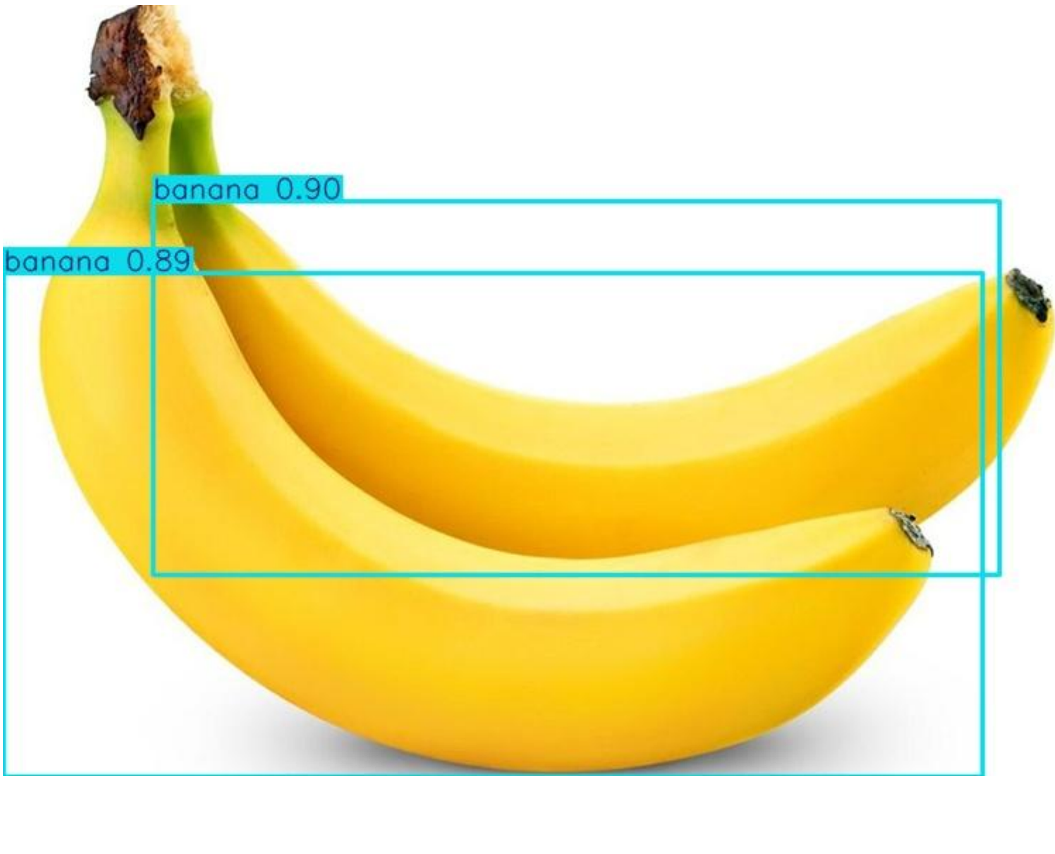
The trained model was used to predict fruits on **unseen test images**.

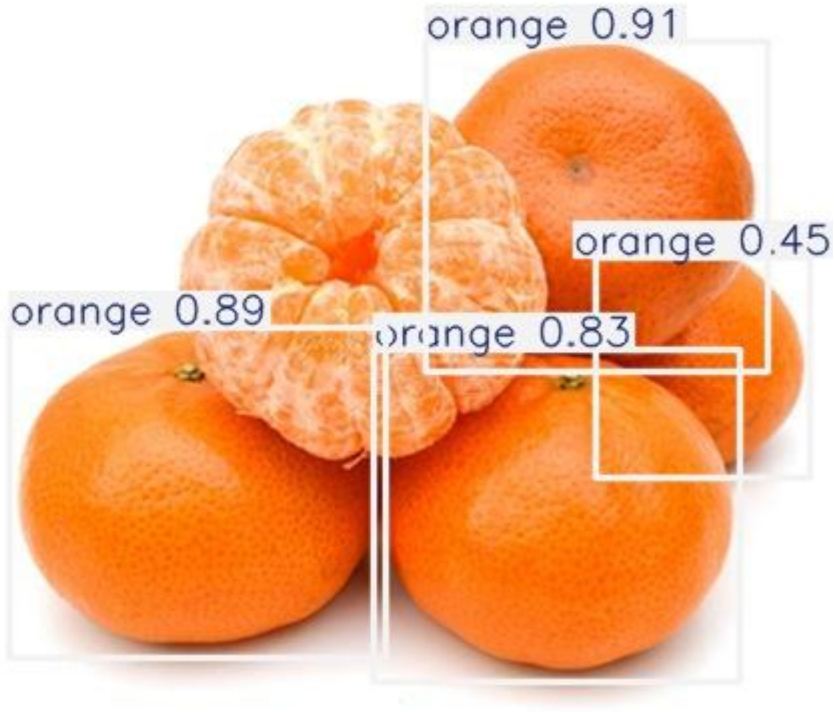
Prediction Output Includes:

- Bounding boxes drawn around detected fruits
- Class labels (Apple / Banana / Orange)
- Confidence scores

All predicted images were saved and visualized for qualitative analysis.


apple 0.83
apple 0.96
apple 0.96
apple 0.97


banana 0.90
banana 0.89



8. Web Application using Streamlit

A **Streamlit web application** was developed to make the model user-friendly.

Application Features:

- Upload an image
- Run fruit detection
- Display detected image with bounding boxes
- Supports multiple uploads (one at a time)

Streamlit enables rapid development of interactive ML applications with minimal code.

9. Deployment using Hugging Face Spaces

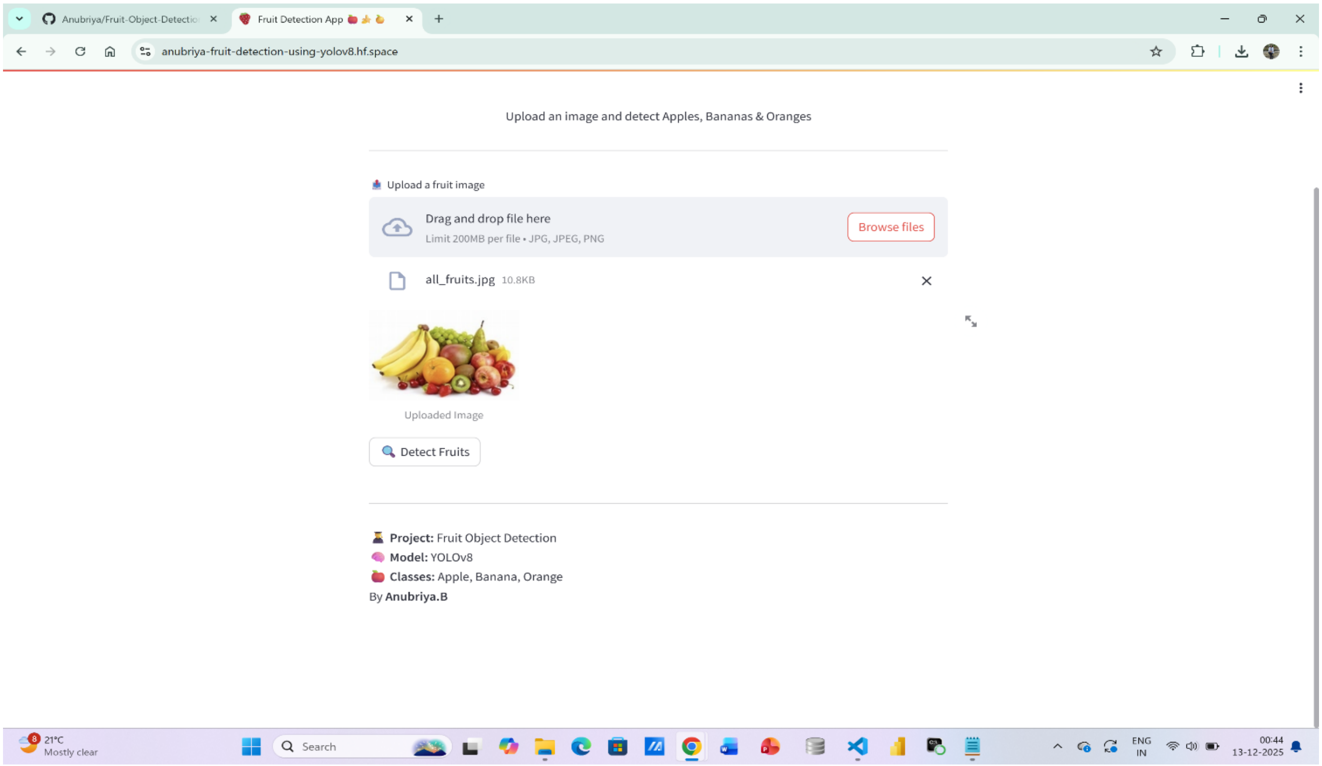
The Streamlit application was deployed using **Hugging Face Spaces**, making it publicly accessible online.

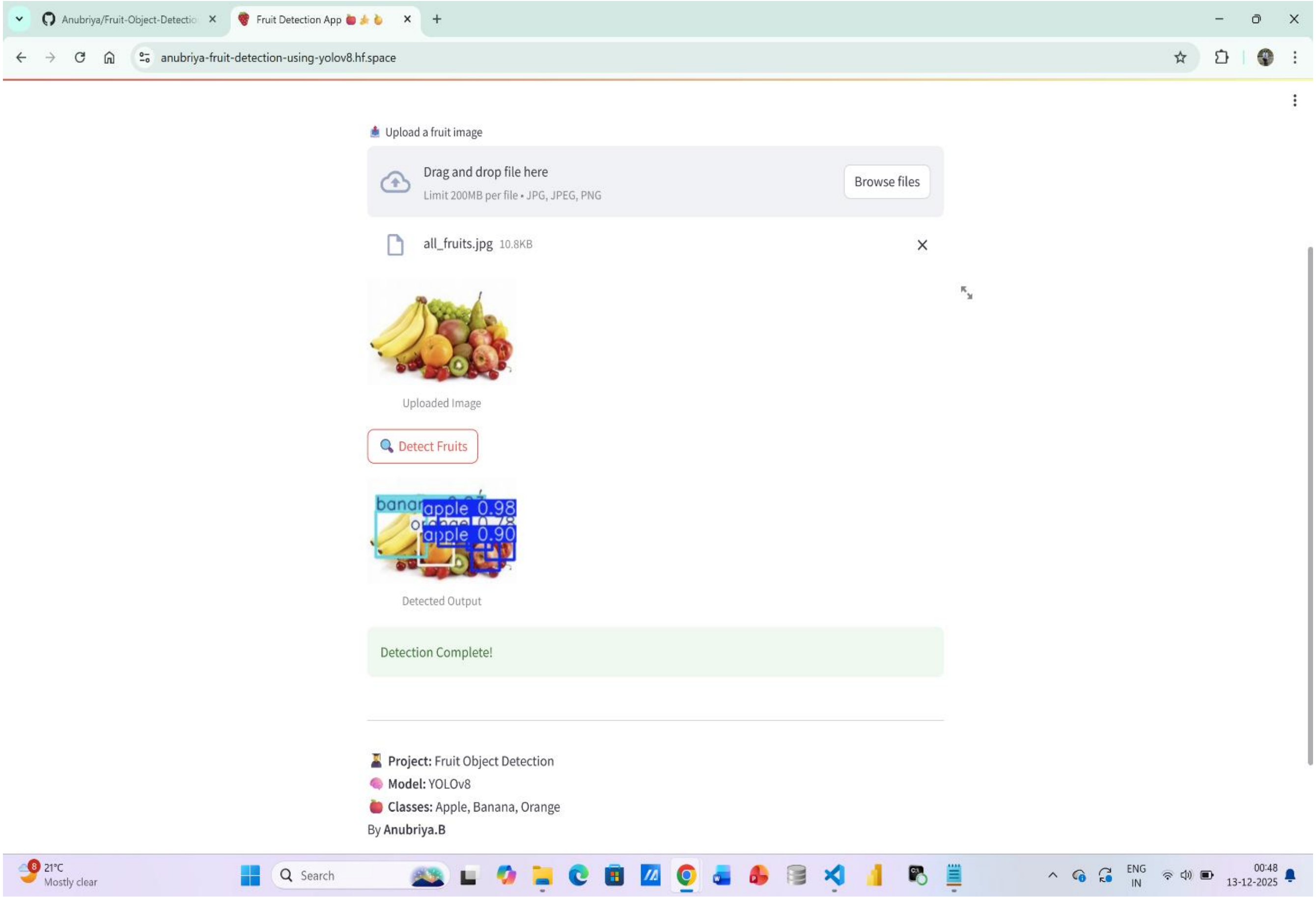
Benefits of Hugging Face Deployment:

- Free hosting
- Public accessibility
- Easy integration with GitHub
- Ideal for ML project showcasing

Live Application Link:

🔗 https://anubriya-fruit-detection-using-yolov8.hf.space/





10. GitHub Repository

All project files including:

- Dataset structure
- Training and evaluation scripts
- Streamlit app
- Evaluation results
- Project report

are maintained in a GitHub repository for version control and reproducibility.

GitHub Repository Link

https://github.com/Anubriya/Fruit-Object-Detection

11. Conclusion

This project successfully demonstrates a complete object detection pipeline using YOLOv8. From dataset preparation to model deployment, every stage follows industry-standard practices.

The system achieves high accuracy in detecting apples, bananas, and oranges and provides an interactive web interface for real-time usage. This project reflects strong practical knowledge in computer vision, deep learning, model evaluation, and deployment.

Future improvements may include:

- Adding more fruit classes
- Live camera detection
- Mobile or cloud-scale deployment

References

- Ultralytics YOLOv8 Documentation
- Streamlit Documentation
- Hugging Face Spaces Documentation
- Computer Vision and Deep Learning Resources