



PERFORMANCE BENCHMARKING OF YOLO ARCHITECTURES

A Comparative Study of YOLOv5, YOLOv8, and YOLOv11



TEAM MEMBERS

● Beyza GÜLER ● Ramazan YILDIZ ● Abdelrahman MOHAMED

ABSTRACT



Goal: Real-time object detection using YOLO models + Streamlit Web Interface



Tech: Pre-trained YOLO + Transfer Learning on Roboflow Data



Result: Working prototype achieving ~30 FPS

[VISUAL: Collage: Webcam Input → YOLO Box → Screen Output]

PROBLEM

High accuracy models are often too slow (low FPS) for real-time web use

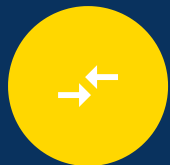
GOALS



Achieve Real-Time
Performance with High FPS



Create accessible Streamlit
Web Interface



Compare YOLO architectures
(v5 vs v8 vs v11)

Slide 3 of 15

[VISUAL: Icon showing a "Speedometer" vs.
"Target"]

METHODOLOGY

1 STEP 1

- Train YOLO models with Labeled Data

2 STEP 2

- Capture video using OpenCV

3 STEP 3

- Perform Object Detection

4 STEP 4

- Visualize & Deploy via Streamlit

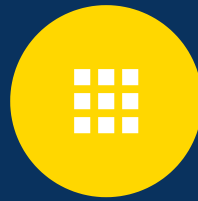
[VISUAL: Flowchart: Input → Preprocessing → YOLO → Streamlit UI]

THE DATASET



SOURCE

Roboflow "Fruit
Detection" Dataset



CLASSES

9 Distinct Fruit Classes



TOTAL IMAGES

2,974

[VISUAL: Grid of sample fruit images with bounding boxes]



DATASET SPLIT DETAILS



TRAINING SET

2,697 images

(Used to update weights)



VALIDATION SET

187 images

(Monitors performance during training)



TEST SET

90 images

(Reserved for final evaluation)



TOTAL

2,974

Images

[VISUAL: Pie Chart showing 2697/187/90 distribution]

YOLO MODEL ARCHITECTURE



TYPE

Single-Stage Detector

End-to-end detection in one pass



BACKBONE

Feature Extraction

Identifies key visual features



NECK

Feature Aggregation

Combines multi-scale features



HEAD

Bounding Box & Class Prediction

Predicts location and category

Slide 7 of 15

[VISUAL: Diagram of YOLO Neural Network Structure]



TRAINING CONFIGURATION



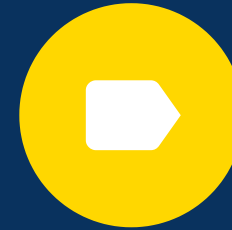
TOOLS

Python
PyTorch
YOLO
OpenCV



CONFIG

Custom data.yaml
Defining 9 classes



LABELS

Normalized YOLO format
Class ID • Center X/Y • W/H

[VISUAL: Example code snippet of a YOLO label line: `0 0.71 0.84 0.15 0.23`]



TRAINING RESULTS



LOSS CURVES

Demonstrated
consistently decreasing
loss



MAP

Accuracy increased
steadily over epochs



CONCLUSION

Models learned
successfully without
major overfitting

[INSERT: Training Curves Graph from Results]

Slide 9 of 15



CONFUSION MATRICES



ANALYSIS

Strong diagonal
concentration



MEANING

Models correctly
identified fruit classes
with minimal
confusion



CONCLUSION

High accuracy
achieved with strong
class separation



BENCHMARK RESULTS

The Core Findings



TARGET

Real-time detection achieved (~30 FPS)



ACCURACY WINNER

YOLOv11m (Highest detection accuracy)



SPEED WINNER

YOLOv5m (Fastest inference time)



BALANCED

YOLOv8m (Balanced performance)

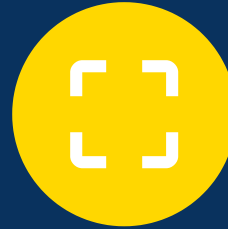
[VISUAL: Comparison Table. Slide 11 of 15 Model Name | Strength | Approx FPS]

WEB DEPLOYMENT



LIVE WEBCAM

Real-time video processing



BOUNDING BOXES

Object localization



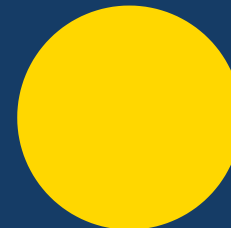
CONFIDENCE SCORES

Probability displayed



BROWSER-BASED

No local installation required for
users



STREAMLIT

Interactive web interface with live
detection

Slide 12 of 15

[VISUAL: Screenshot of Web App Live Detection Page]



CONCLUSION



STATUS: Working real-time prototype developed



PERFORMANCE: Stable ~30 FPS achieved



OUTCOME: Strong foundation for future improvements



FUTURE RECOMMENDATIONS



BIGGER DATASET

Train with more images for robustness



3D LOCALIZATION

Add depth estimation for 3D positioning



MOBILE APP

Deploy to Android/iOS platforms

[VISUAL: Icons for Database, 3D Cube, and Smartphone]

THANK YOU

TEAM MEMBERS

- Beyza GÜLER
- Ramazan YILDIZ
- Abdelrahman MOHAMED

 **Q&A SESSION** 

Questions & Discussion