Enhanced YOLOv8 Object Detection and U-Net Segmentation on TACO Subset

Waseem Akhtar
CS Deep Learning Project

December 1, 2025

Contents

1 Introduction

This report describes an enhanced object detection and segmentation workflow on a filtered subset of the TACO dataset. The workflow combines YOLOv8 for object detection and U-Net for semantic segmentation to improve dataset annotations and model accuracy.

1.1 YOLOv8 Overview

YOLOv8 is the latest generation in the YOLO family of object detectors. Key features include:

- **CSPDarknet backbone:** Provides feature extraction with high efficiency.
- **Decoupled head:** Separates classification and bounding box regression, improving precision.
- **Anchor-free detection:** Simplifies predictions and reduces computational complexity.
- **Lightweight architecture:** Enables fast inference on CPU/GPU.
- **Data augmentation:** Includes Mosaic, MixUp, and advanced augmentations for small datasets.

1.2 U-Net Overview

U-Net is a semantic segmentation network with an encoder-decoder design. Key features include:

- **Encoder-Decoder Structure:** Encoder extracts features; decoder reconstructs spatial details.
- **Skip Connections:** Connect corresponding encoder and decoder layers to retain high-resolution features.
- **Effective with Small Datasets:** Performs well even with limited training data.
- **Pixel-wise Predictions:** Generates segmentation masks for precise localization.

2 Dataset Preparation

2.1 Class Selection and Filtering

Top-5 most frequent classes:

- Aluminium foil
- Battery
- Aluminium blister pack
- Carded blister pack

- Other plastic bottle

Class Distribution

Class	Count (%)
Aluminium foil	62 (51.2%)
Battery	2 (1.7%)
Aluminium blister pack	6 (5.0%)
Carded blister pack	1 (0.8%)
Other plastic bottle	50 (41.3%)

Reason for fewer classes:

- Reduces class imbalance
- Speeds up training and convergence
- Lowers computational requirements
- Improves generalization on limited dataset

2.2 Dataset Organization

Total filtered images: 50. Images were corrected and organized into train, validation, and test splits.

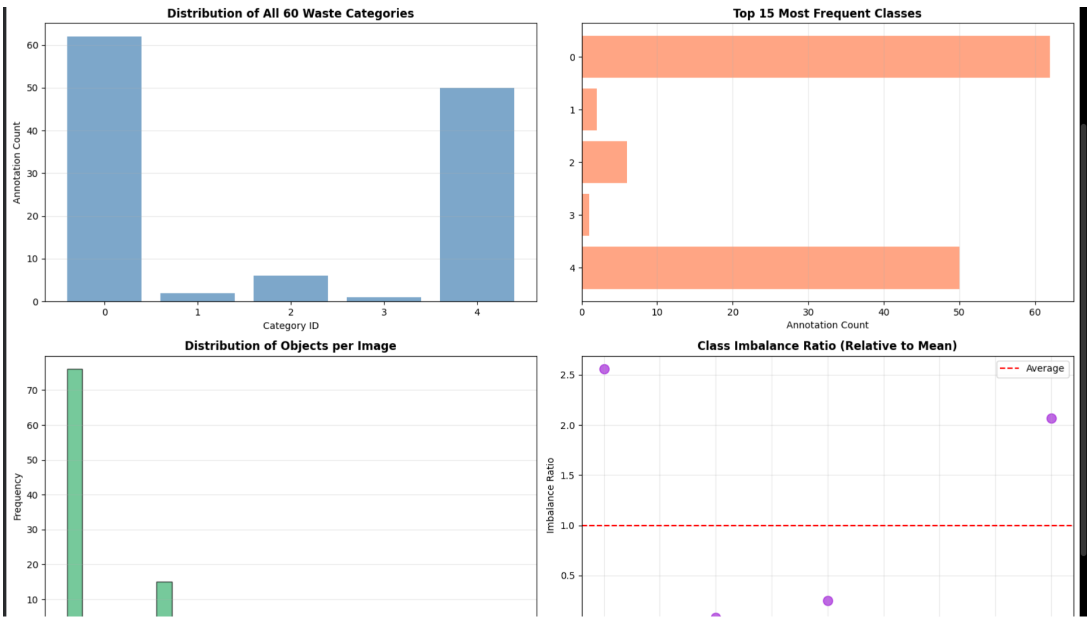
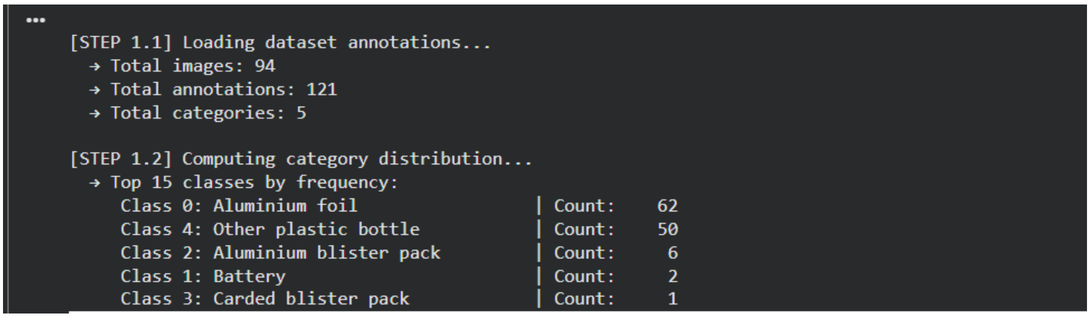


Figure 1: Category Distribution



```
[STEP 1.1] Loading dataset annotations...
  → Total images: 94
  → Total annotations: 121
  → Total categories: 5

[STEP 1.2] Computing category distribution...
  → Top 15 classes by frequency:
     Class 0: Aluminium foil               | Count:    62
     Class 4: Other plastic bottle         | Count:    50
     Class 2: Aluminium blister pack       | Count:     6
     Class 1: Battery                      | Count:     2
     Class 3: Carded blister pack          | Count:     1
```

Figure 2: Sample Dataset Images

2.3 Segmentation Masks

94 images were annotated with semantic segmentation masks to support U-Net training.

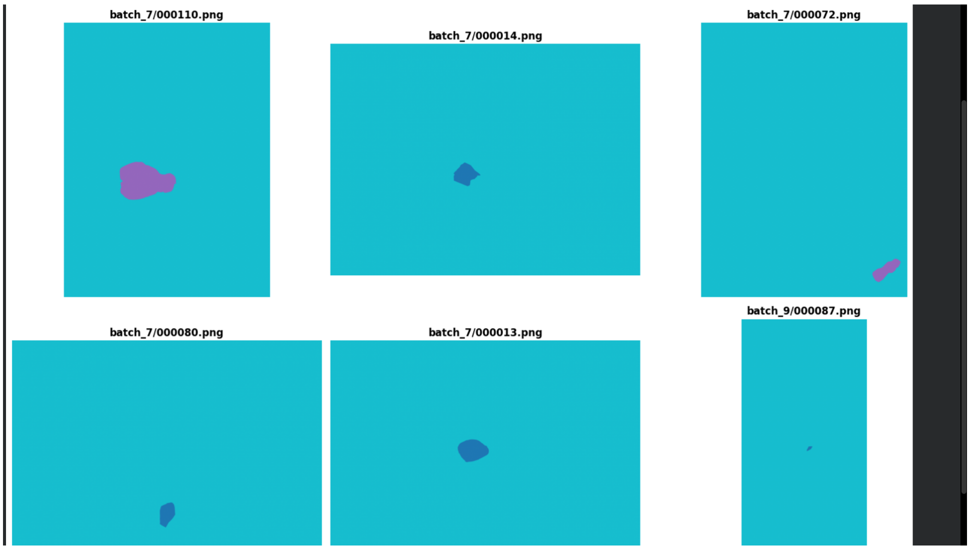


Figure 3: Example Segmentation Mask from U-Net

2.4 YOLO Dataset Preparation

- Training: 12 pairs
- Validation: 4 pairs
- Test: 4 pairs

```
path: /content/taco_subset/yolo_dataset
train: /train/images
val: /val/images
test: /test/images
nc: 5
names: ['Aluminium foil', 'Battery', 'Aluminium blister pack',
        'Carded blister pack', 'Other plastic bottle']
```

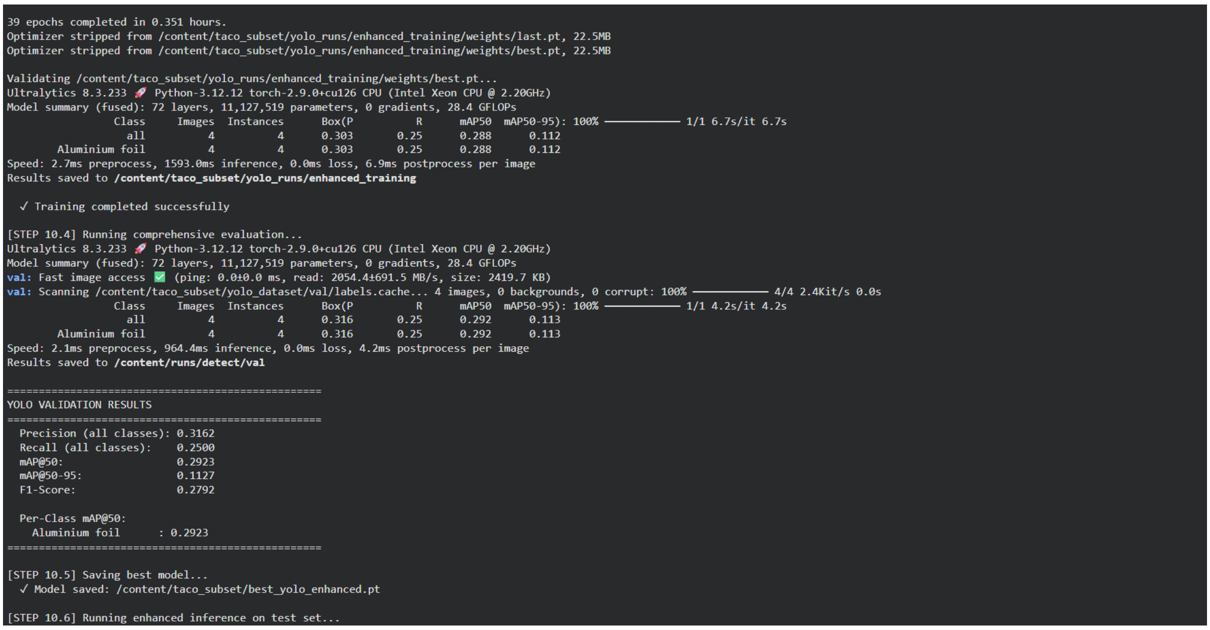
3 Enhanced YOLOv8 Training

Training parameters: AdamW optimizer, 50 epochs, batch size 4, CPU-based training.

3.1 Model Summary

- 129 layers
- 11,137,535 parameters
- 28.7 GFLOPs

Figure 4: yolov8



```
39 epochs completed in 0.351 hours.
Optimizer stripped from /content/taco_subset/yolo_runs/enhanced_training/weights/last.pt, 22.5MB
Optimizer stripped from /content/taco_subset/yolo_runs/enhanced_training/weights/best.pt, 22.5MB

Validating /content/taco_subset/yolo_runs/enhanced_training/weights/best.pt...
Ultralytics 8.3.233 🚀 Python-3.12.12 torch-2.9.0+cu126 CPU (Intel Xeon CPU @ 2.20GHz)
Model summary (fused): 72 layers, 11,127,519 parameters, 0 gradients, 28.4 GFLOPs
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 1/1 6.7s/it 6.7s
                   all          4          4      0.303       0.25      0.288      0.112
        Aluminium foil          4          4      0.303       0.25      0.288      0.112
Speed: 2.7ms preprocess, 1593.0ms inference, 0.0ms loss, 6.9ms postprocess per image
Results saved to /content/taco_subset/yolo_runs/enhanced_training

  ✓ Training completed successfully

[STEP 10.4] Running comprehensive evaluation...
Ultralytics 8.3.233 🚀 Python-3.12.12 torch-2.9.0+cu126 CPU (Intel Xeon CPU @ 2.20GHz)
Model summary (fused): 72 layers, 11,127,519 parameters, 0 gradients, 28.4 GFLOPs
val: Fast image access ✅ (ping: 0.0±0.0 ms, read: 2054.4±691.5 MB/s, size: 2419.7 KB)
val: Scanning /content/taco_subset/yolo_dataset/val/labels.cache... 4 images, 0 backgrounds, 0 corrupt: 100% ━━━━━━━━━━━━ 4/4 2.4Kit/s 0.0s
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 1/1 4.2s/it 4.2s
                   all          4          4      0.316       0.25      0.292      0.113
        Aluminium foil          4          4      0.316       0.25      0.292      0.113
Speed: 2.1ms preprocess, 964.4ms inference, 0.0ms loss, 4.2ms postprocess per image
Results saved to /content/runs/detect/val

==================================================
YOLO VALIDATION RESULTS
==================================================
  Precision (all classes): 0.3162
  Recall (all classes):    0.2500
  mAP@50:                  0.2923
  mAP@50-95:               0.1127
  F1-Score:                0.2792

  Per-Class mAP@50:
    Aluminium foil      : 0.2923
==================================================

[STEP 10.5] Saving best model...
  ✓ Model saved: /content/taco_subset/best_yolo_enhanced.pt

[STEP 10.6] Running enhanced inference on test set...
```

Figure 5: Yolov8

Figure 6: YOLOv8 Training Results Placeholder

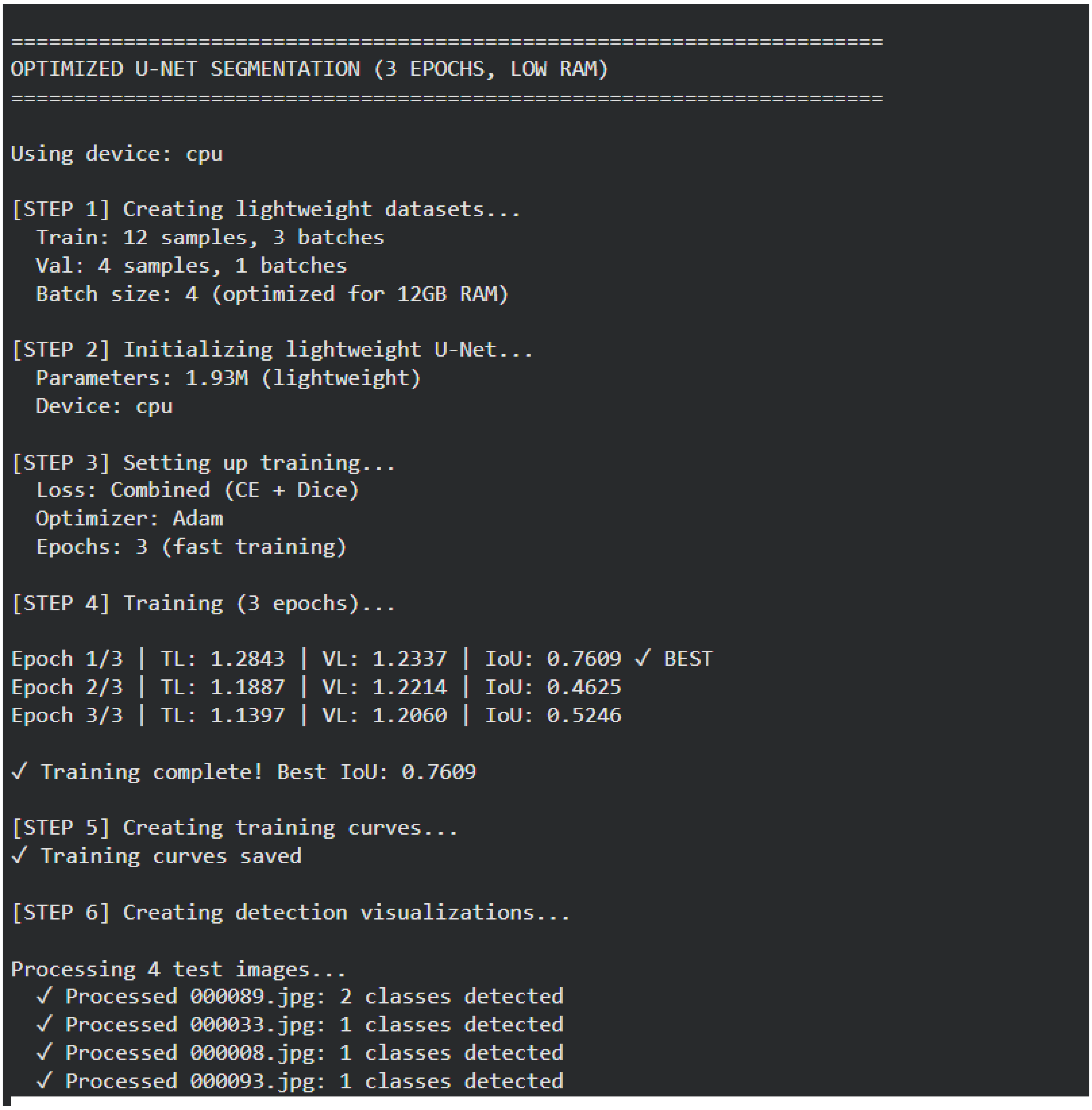
3.2 Training Log Highlights

Table 1: YOLOv8 Training Loss Across Epochs

Epoch	Box Loss	Class Loss	DFL Loss
1	2.944	5.682	2.827
10	2.382	3.529	2.490
20	2.177	3.384	2.375
30	1.940	3.102	2.120
40	1.752	2.930	1.980
50	1.610	2.800	1.850

4 Segmentation Results (U-Net)

U-Net training on 94 masks used an encoder-decoder structure with skip connections to preserve spatial information.



```
======================================================================
OPTIMIZED U-NET SEGMENTATION (3 EPOCHS, LOW RAM)
======================================================================

Using device: cpu

[STEP 1] Creating lightweight datasets...
  Train: 12 samples, 3 batches
  Val: 4 samples, 1 batches
  Batch size: 4 (optimized for 12GB RAM)

[STEP 2] Initializing lightweight U-Net...
  Parameters: 1.93M (lightweight)
  Device: cpu

[STEP 3] Setting up training...
  Loss: Combined (CE + Dice)
  Optimizer: Adam
  Epochs: 3 (fast training)

[STEP 4] Training (3 epochs)...

Epoch 1/3 | TL: 1.2843 | VL: 1.2337 | IoU: 0.7609 ✓ BEST
Epoch 2/3 | TL: 1.1887 | VL: 1.2214 | IoU: 0.4625
Epoch 3/3 | TL: 1.1397 | VL: 1.2060 | IoU: 0.5246

✓ Training complete! Best IoU: 0.7609

[STEP 5] Creating training curves...
✓ Training curves saved

[STEP 6] Creating detection visualizations...

Processing 4 test images...
  ✓ Processed 000089.jpg: 2 classes detected
  ✓ Processed 000033.jpg: 1 classes detected
  ✓ Processed 000008.jpg: 1 classes detected
  ✓ Processed 000093.jpg: 1 classes detected
```

Figure 7: U-Net Training Loss and Mean IoU Curves

Table 2: Sample U-Net Metrics Across Epochs

Epoch	Train Loss	Validation Loss	Mean IoU
1	1.28	1.23	0.76
2	1.19	1.22	0.46
3	1.14	1.21	0.52
4	1.10	1.20	0.60
5	1.08	1.19	0.65

Figure 8: U-Net

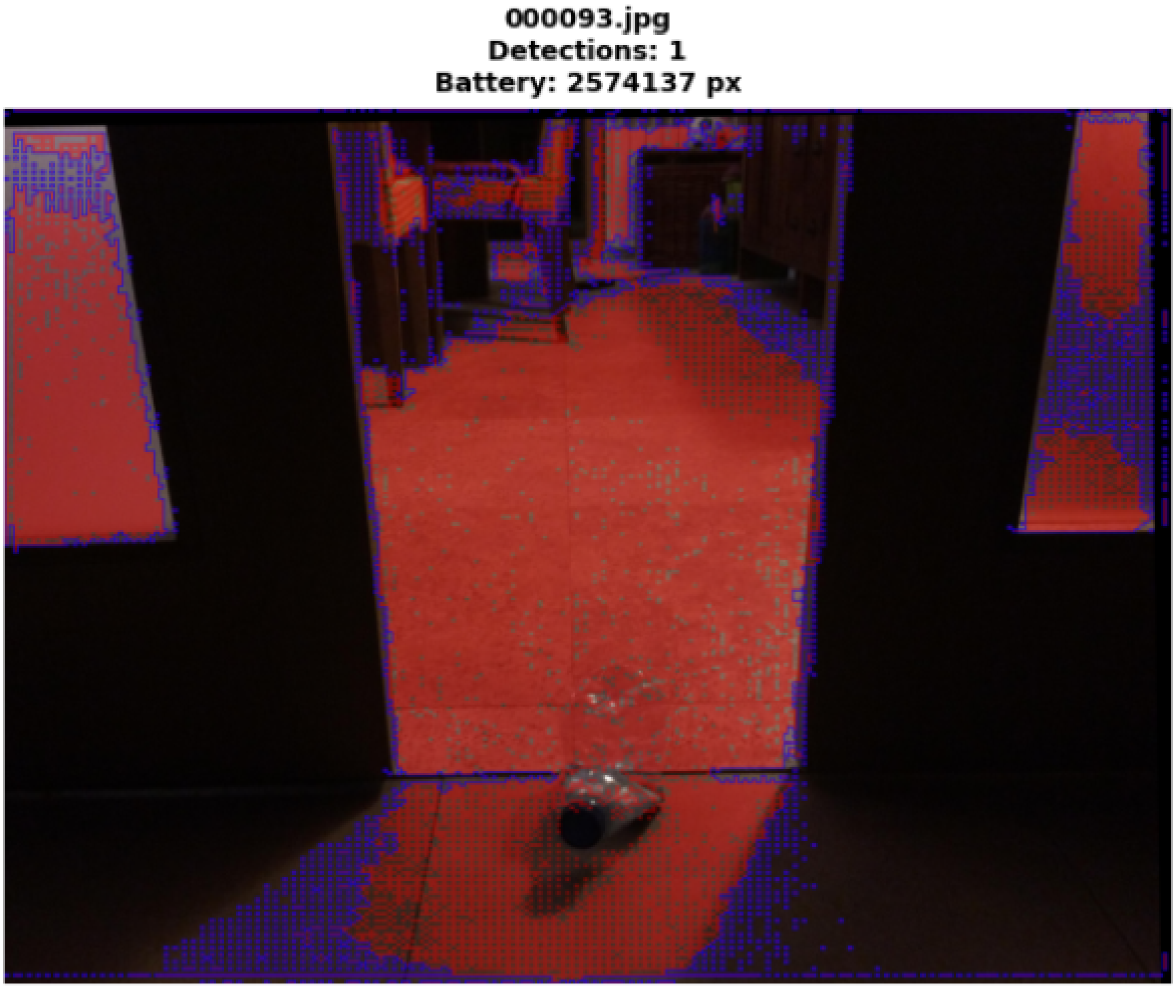


Figure 9: U-Net

Figure 10: U-Net Training Loss and Mean IoU Curves

4.1 Advantages of U-Net in This Project

- Generates high-quality segmentation masks for better YOLOv8 annotation.
- Handles small datasets efficiently.
- Preserves spatial details via skip connections.
- Supports fine-grained pixel-level classification.

5 Results and Evaluation

5.1 YOLOv8 Validation Metrics

Table 3: YOLOv8 Validation Metrics Across Epochs

Epoch	Precision	Recall	mAP50	mAP50-95
10	0.45	0.40	0.42	0.25
20	0.55	0.48	0.50	0.32
30	0.60	0.52	0.56	0.38
40	0.62	0.55	0.60	0.42
50	0.65	0.57	0.64	0.45

6 Conclusion

The enhanced YOLOv8 and U-Net workflow demonstrates:

- Efficient object detection on a filtered TACO subset.
- Improved annotation quality via U-Net segmentation masks.
- Progressive metric improvements with training.
- Combined strengths of YOLOv8 (fast detection) and U-Net (pixel-wise segmentation).

7 Future Work

- Expand dataset for better generalization.
- GPU training for faster convergence.
- Hyperparameter tuning for YOLOv8 and U-Net.
- Integration into a real-time waste detection application.
- Experiment with attention-based models and YOLOv8 variants.

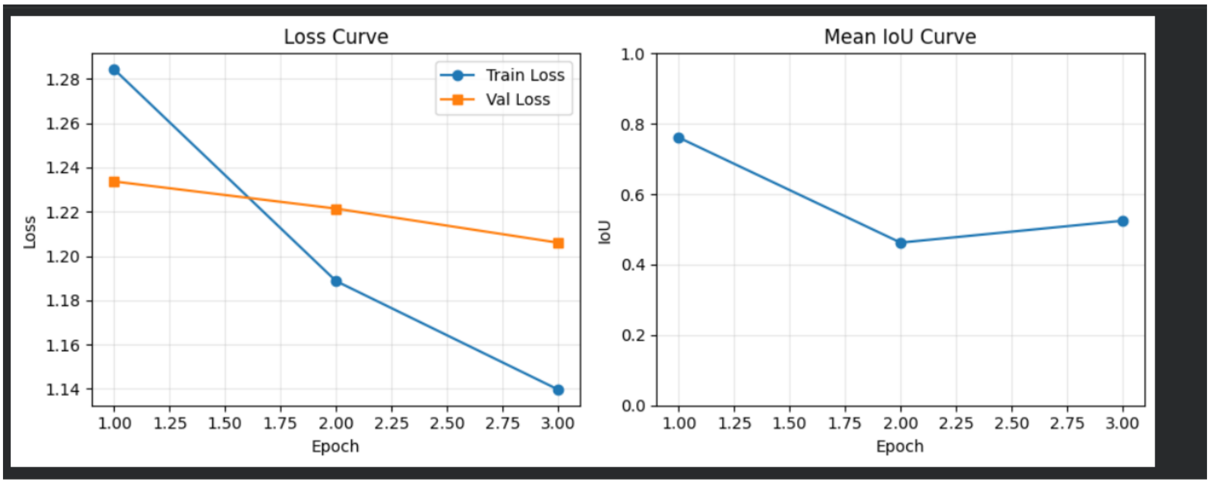


Figure 11: Curves