Prepared by group 6

GARBAGE DETETCTION using CCTV

EOC - MFC

15 April, 2025

THANUSHA P S - cb.sc.u4aie24154
VISHNU VARDAN - cb.sc.u4aie24159

INDEX

1. Introduction
2. Objective
3. Methodology
4. Implementations
5. Results and analysis
6. Discussion
7. Conclusion
8. Future work

INTRODUCTION

India is expected to generate 165 million tonnes of waste annually by 2030, much of it poorly managed. Manual waste monitoring is slow and inefficient, especially in crowded cities. At the same time, over 1.5 million CCTV cameras installed under smart city projects remain underutilized. This work proposes a real-time garbage detection system using YOLOv8n and SVD preprocessing, leveraging existing CCTV infrastructure for efficient and automated waste surveillance.

OBJECTIVE

- Automate garbage detection using existing CCTV systems.
- Improve object detection using image preprocessing (SVD).
- Train and evaluate a lightweight model (YOLOv8n).

MEHTODOLOGY

STEP 1:

- Collecting the Dataset from multiple sources such as roboflow, kaggle.
- The dataset contains 1683 images which can be annotated to various categories as Broken trash can, Close empty, Close full, Healthy trash can, Open empty, Open full, Trash flow.

STEP 2:

Preprocessing (SVD)

- Goal: Denoise input images without losing key features.
- Applied Singular Value Decomposition (SVD) in MATLAB.
  - Convert image to R, G, B matrices.
  - Perform SVD on each channel.
  - Truncate top 200 singular values.
  - Reconstruct cleaner image.
- Result: Sharper, more informative input for YOLO.

RESULT (SVD)

Fig: Image

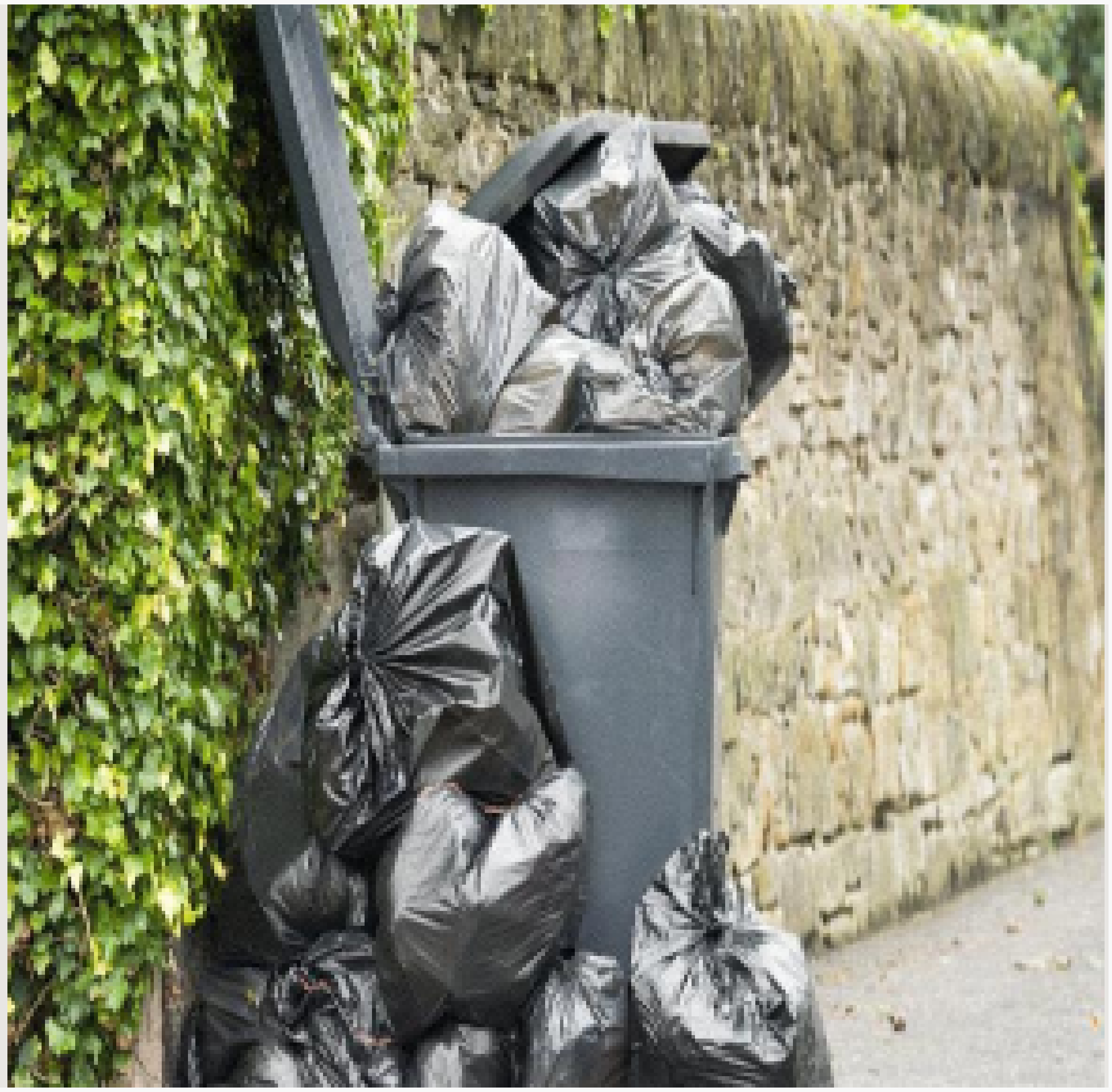


Fig: Image preprocessed

STEP 3:

- Annotate the images into 7 key classes are used for annotating Broken trash can, Close empty, Close full, Healthy trash can, Open empty, Open full, Trash flow

STEP 4:

Model: YOLOv8n (nano version by Ultralytics)

- Lightweight and optimized for real-time detection.

Input Image Size: 640 × 640 pixels

- Ensures uniformity and compatibility with the model.

Dataset Split:

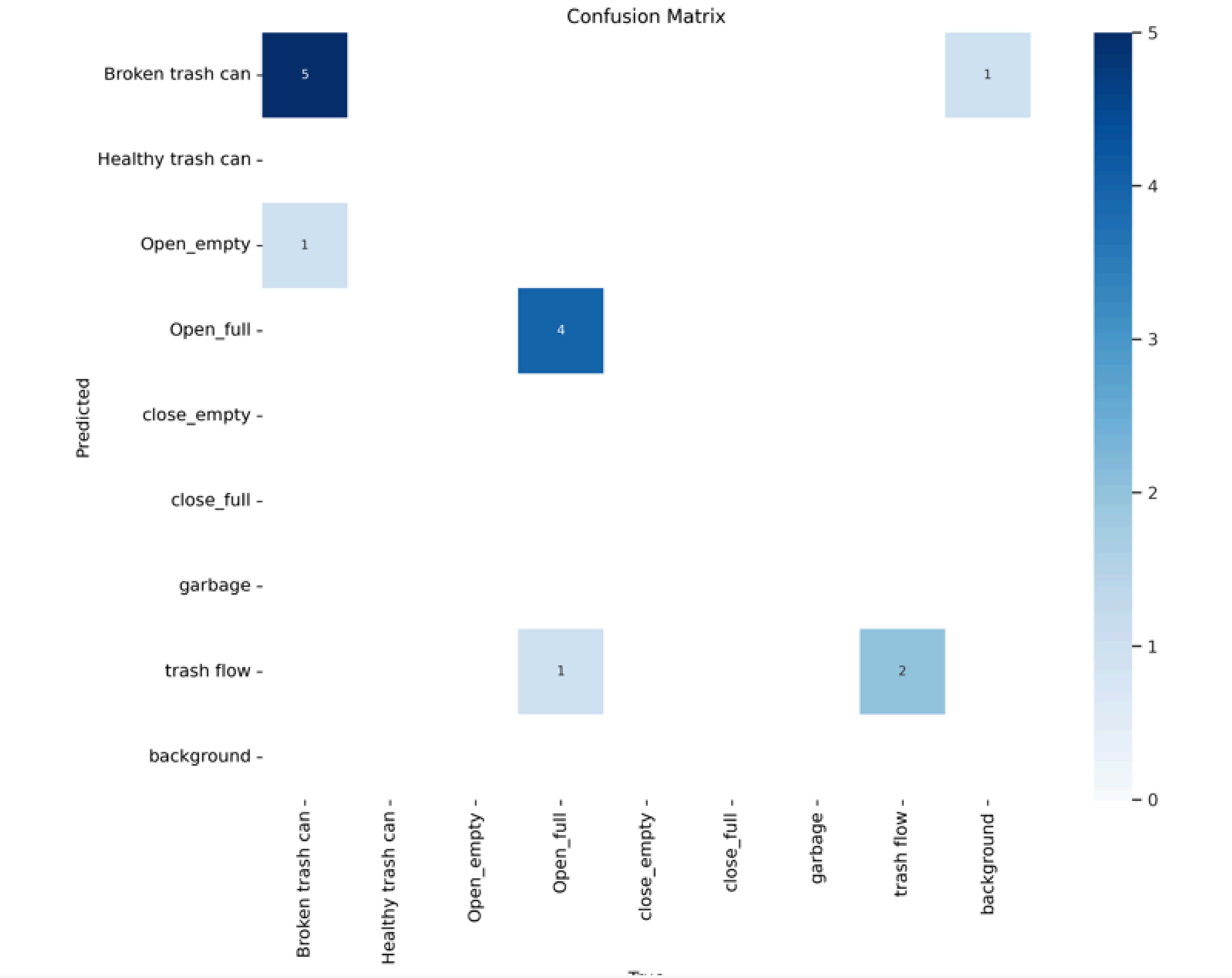
- Total images: 1,683
- Training: 99%
- Validation: 1%

Training:

- 20 epochs to allow iterative learning and performance improvement.

Evaluation Metrics:

- Precision: 91.2% – Low false positives.
- Recall: 87.5% – High detection accuracy.
- mAP@0.5: 89.3% – Strong overall performance at IoU threshold 0.5.



	Broken trash can	Healthy trash can	Open_empty	Open_full	close_empty	close_full	garbage	trash flow	background
Broken trash can	5								1
Healthy trash can									
Open_empty	1								
Open_full				4					
close_empty									
close_full									
garbage									
trash flow				1				2	
background									

Fig: Confusion matrix

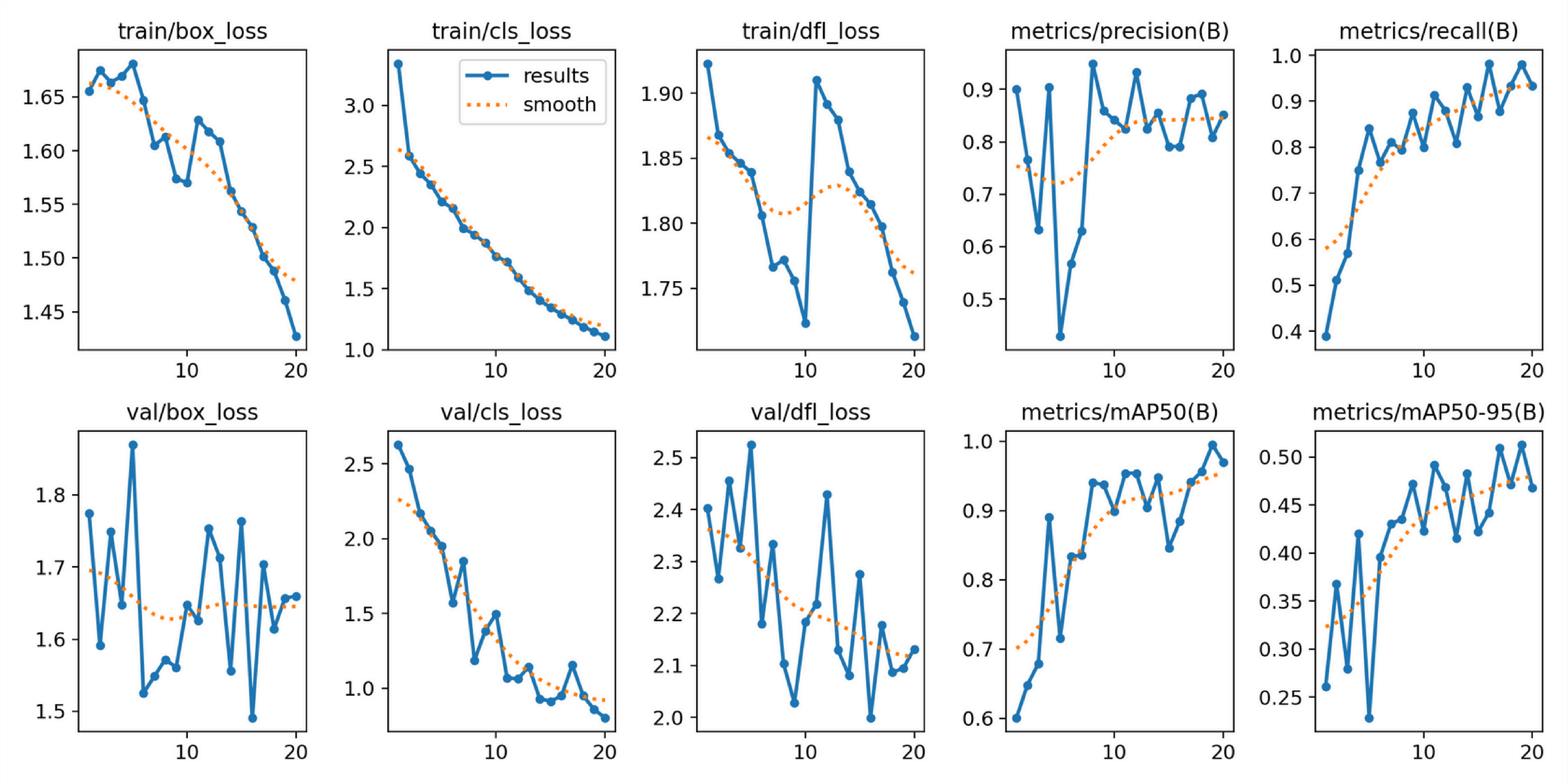

train/box_loss
train/cls_loss
train/dfl_loss
metrics/precision(B)
metrics/recall(B)
results
smooth
val/box_loss
val/cls_loss
val/dfl_loss
metrics/mAP50(B)
metrics/mAP50-95(B)
1.65
1.60
1.55
1.50
1.45
3.0
2.5
2.0
1.5
1.0
1.90
1.85
1.80
1.75
0.9
0.8
0.7
0.6
0.5
1.0
0.4
1.8
1.7
1.6
1.5
2.4
2.3
2.2
2.1
0.50
0.45
0.40
0.35
0.30
0.25
10
20

Fig: Result

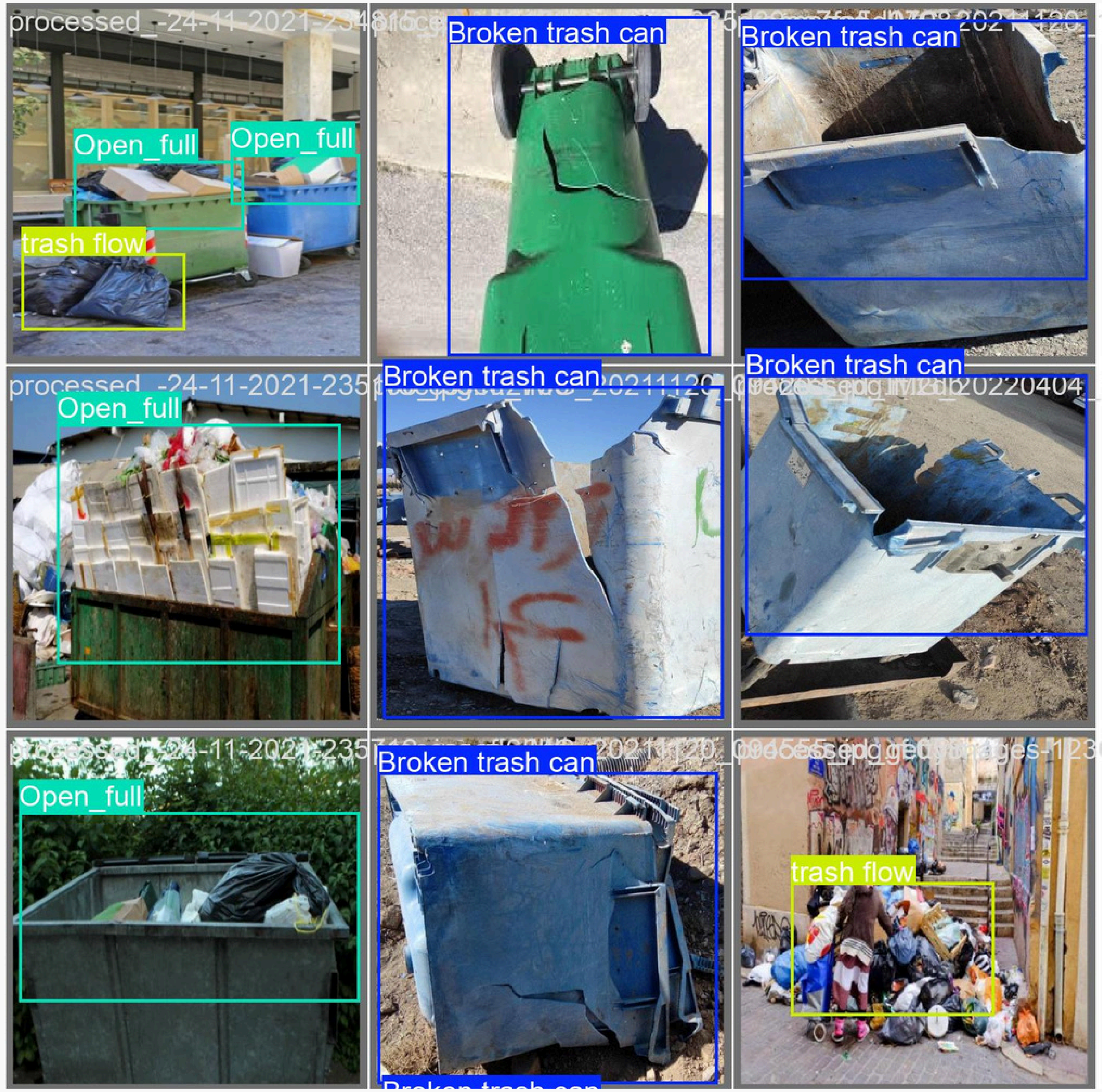


Fig: Detected images

STEP 5:

- Flask + OpenCV is used for uploading a video and to detect the video.
- Flask is a lightweight Python web framework used to create a web application.
- OpenCV is a powerful computer vision library used for image and video processing.
- Flask handles user input (file upload or webcam selection).
- OpenCV captures video frames and processes them using YOLOv8.
- Flask streams the processed frames to the browser as a live video.
- Flask allows stopping the detection, preventing resource wastage

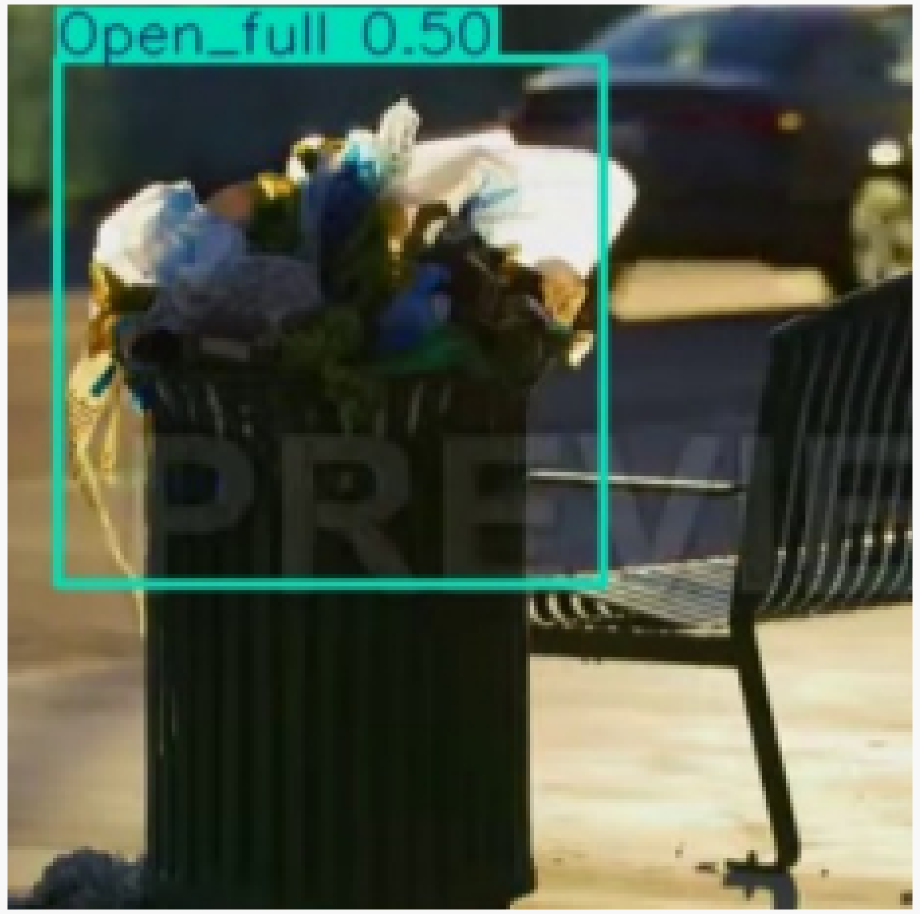


Fig: Detected image

RESULT AND DISSCUSSION

RESULTS

- We extract features to focus on what matters, reduce noise to improve clarity, and
- compress images to save space and make processing fasterEvaluation Metrics:
    - Precision: 91.2% – Low false positives.
    - Recall: 87.5% – High detection accuracy.
    - mAP@0.5: 89.3% – Strong overall performance at IoU threshold 0.5.
- Strong classification performance observed for:
    - Broken trash can
    - Open full
- Misclassifications occurred mainly between:
    - Open empty and Trash flow, likely due to visual similarity.Steady decline in box loss, classification loss, and DFL (Distribution Focal Loss) over 20 epochs.
    - Indicates effective learning with minimal overfitting.

- Precision and Recall:
    - Precision exceeded 0.9 in later epochs, showing model confidence in correct detections.
    - Recall approached 1.0, reflecting very few missed detections.
- mAP (Mean Average Precision):
    - mAP@0.5 peaked around 0.95, confirming high detection accuracy.
    - mAP@0.5:0.95 exceeded 0.5, indicating good performance across stricter IoU thresholds.

DISCUSSIONS

- The model demonstrates robust generalization, even with a small validation set.
- SVD preprocessing helped reduce image noise, improving feature clarity.
- Minor class confusion can be addressed through:
    - Class balancing
    - More training images
- Enhanced augmentation strategies

FUTURE WORK

- Edge Deployment: Port model to embedded devices for real-time city-scale use.
- Alert System Enhancement: Add real-time GPS tagging and mobile notifications to authorities.

THANK YOU