Garbage Detection using YoloV8

Garbage detection project with YOLOv8 addresses practical waste management and environmental monitoring challenges by automating object identification in Waste recycling plant. It is specifically used for sorting waste according to 28 distinct class of recyclable items.

GitHub Link: [Akshat1661/Waste-Detection-in-WaRP-using-yolov8 (github.com)]

1. bottle-blue
2. bottle-green
3. bottle-dark
4. bottle-milk
5. bottle-transp
6. bottle-multicolor
7. bottle-yogurt
8. bottle-oil
9. cans
10. juice-cardboard
11. milk-cardboard
12. detergent-color
13. detergent-transparent
14. detergent-box
15. canister
16. bottle-blue-full
17. bottle-transp-full
18. bottle-dark-full
19. bottle-green-full
20. bottle-multicolorv-full
21. bottle-milk-full
22. bottle-oil-full
23. detergent-white
24. bottle-blue5l
25. bottle-blue5l-full
26. glass-transp
27. glass-dark
28. glass-green

Important Implementation:

1. Definition selection

After exploring various field of subjects, we decided to work with waste recycling system as waste recycling is a critical aspect of sustainable development and environmental preservation. Hence, started exploring for such dataset on Kaggle.

2. Dataset Selection

we found the dataset on Kaggle at [WaRP - Waste Recycling Plant Dataset | Kaggle] and downloaded it. It contains all the required data for the project. This dataset is comprised of three parts:

Warp – D: This folder contains images of garbage for object detection purposes, featuring 28 different classes of garbage products.
Warp – C: The Warp-C dataset contains all classification images.
Warp – S: Warp-S is used for the segmentation of the images.

Since our project focused on object detection, we exclusively utilized the WaRP-D image dataset. This particular dataset consists of 2,452 images in the training sample and 522 images in the validation sample.

3. Uniqueness

we diligently reviewed a wide range of research papers from various sources and databases. Our goal was to gather pertinent information and insights that could contribute significantly to our research project. However, despite our thorough efforts, we did not find any relevant research work already published on this topic specifically.

4. Model Selection

With some prior experience from internship where we worked on yoloV5, we explored various versions but finally decided to work on V8 which is the most current version of YOLO Family with prominent upgradation from previous versions. We implemented the Ultralytics-YOLO (You Only Look Once) algorithm for object detection, specifically YOLOv8, due to its superior accuracy and speed compared to older versions of YOLO. Unlike alternatives such as R-CNN or R-CNN FAST, YOLOv8 excels in computational efficiency, which is crucial for real-time applications requiring rapid image processing. Its single-pass architecture enables it to predict both bounding boxes and class probabilities simultaneously, making it ideal for tasks like garbage detection.

5. Training Process

In the training portion, we divided the 2,452 images into 1,962 images for the training set and 490 images for the validation set. We created a dataset.yaml file and placed it in the training directory for configuration.

Initially, we were training using the YOLOv8s model for 50 epochs. However, this resulted in lower accuracy and uncertainty in the training outcomes, particularly in terms of loss, recall, and precision.

Subsequently, we switched to the YOLOv8m model, which is larger than YOLOv8s, and extended the training to 100 epochs. We performed this training using the Google Colab T4 GPU, which provided us with improved results and more stable performance.

Yolov8m is able to perform classification really well because of its pretraining on the ImageNet dataset (a huge dataset containing millions of images).

6. Result

We achieved some good results from YoloV8m. Some result visuals we achieved by the training the model for 100 epochs, we derived the following results, including a confusion matrix, precision, recall, and loss metrics.