Ground Garbage Detection Using Computer Vision

Frederico Correia Cerqueira
Faculdade de Ciências da Universidade de Lisboa
Campo Grande 016, 1749-016 Lisboa
fc64374@alunos.ciencias.ulisboa.pt

Joana Nunes Chuço
Faculdade de Ciências da Universidade de Lisboa
Campo Grande 016, 1749-016 Lisboa
fc64853@alunos.ciencias.ulisboa.pt

Gonçalo Abreu
Faculdade de Ciências da Universidade de Lisboa
Campo Grande 016, 1749-016 Lisboa
fc64462@alunos.ciencias.ulisboa.pt

Abstract

Urban pollution and the shortage of human resources for street cleaning have increased the demand for innovative technological solutions. This paper presents an automated system for ground garbage detection using deep neural networks and computer vision. Leveraging the YOLOv8 model, an urban waste classifier was trained using the TACO dataset, adapted to five superclasses: plastic, paper, metal, glass, and other. The pipeline includes preprocessing, annotation conversion, data augmentation, and evaluation through metrics such as precision, recall, and mAP. The results show the feasibility of the approach, with an average inference time of approximately 3 ms per image and satisfactory performance on well-represented classes. The study also discusses challenges in detecting small objects and proposes future optimizations to enhance the system's robustness in real-world environments.

1. Introduction

Littering in urban environments poses one of the major ecological and public health challenges of our time. The accumulation of garbage on sidewalks, streets, and public spaces not only degrades the aesthetic value of cities but also threatens ecosystems and public hygiene. However, urban cleaning efforts are often constrained by limited human and financial resources, motivating the need for automation.

In this context, automatic garbage detection on the ground using deep learning and computer vision has emerged as a promising solution. Recent advancements in object detection algorithms, particularly those from the YOLO (You Only Look Once) family, enable real-time detection of various waste types with high accuracy. The latest iteration, YOLOv8, offers an efficient and high-performance alternative for smart environmental monitoring and integration into autonomous cleaning systems.

This paper presents a garbage detection system based on YOLOv8, trained on a customized version of the TACO (Trash Annotations in Context) dataset — an open dataset containing real-world images of garbage with detailed annotations. Inspired by previous work such as the Garbage Detection with TensorFlow project on Kaggle, we propose a semantic simplification of TACO's original fine-grained labels into five broader superclasses: plastic, paper, metal, glass, and other. This grouping reduces annotation complexity and improves generalization performance.

The contributions of this work include the development of a complete training pipeline adapted to YOLOv8, with preprocessing, annotation transformation, and data conversion into the YOLO format. The model is evaluated both qualitatively and quantitatively using precision, recall, and mean Average Precision (mAP). Several techniques are integrated into the pipeline, including data augmentation strategies (e.g., HSV adjustments, flipping, mosaic and copy-paste augmentations) and post-processing filters to eliminate small false positives.

2. Related Works (State of the Art)

Melinte et al. (2020) developed a system capable of detecting and recognizing waste in real time, demonstrating the adaptability of these models in dynamic environments, as intended in this project.

Oliveira and Guedes (2024) applied the YOLOv7 model to detect waste in coastal areas and presented robust results in the segmentation and classification of this type of waste. For these reasons, we chose to use YOLOv8, which is even more robust.

Proença and Simões (2020) proposed TACO, which has been widely used to train waste detection models. As it is a very extensive, varied, and constantly updated dataset, it proved to be suitable for the present project, where its more than 60 categories were reduced to 5 super classes in order to facilitate model training.

3. Methodology

3.1. Model Selection

For the object detection task in this study, we selected the YOLOv8 (You Only Look Once, version 8) model developed by Ultralytics. YOLO models are widely recognized for their high speed and competitive accuracy in real-time object detection tasks, which makes them suitable for practical applications such as mobile or embedded systems.

We considered several alternatives, including Faster R-CNN and SSD (Single Shot MultiBox Detector), but YOLOv8 was chosen due to its improved performance over previous YOLO versions, integrated data augmentation features, ease of training with the Ultralytics API. In addition, YOLOv8 provides multiple model sizes (nano, small, medium, etc.) that allow fine-tuning the trade-off between speed and accuracy according to hardware limitations.

The YOLOv8-nano (yolov8n) model was used as a baseline due to its reduced number of parameters ($\sim$ 3.2 million).

3.2. Dataset and Label Simplification

The dataset used in this project is TACO (Trash Annotations in Context), which contains real-world images of waste in urban environments annotated using the COCO JSON format. The original dataset includes exactly 60 fine-grained categories representing specific waste items, such as *plastic bottle* or *aluminium can*. While detailed, this high number of classes introduces unnecessary complexity and hinders model generalization.

To address this, the original categories were mapped into five broader superclasses based on material type: plastic, paper, metal, glass, and other. This mapping was based on the presence of keywords in the category names. For instance, items with the terms *plastic*, *styrofoam* or *wrapper* were assigned to the plastic class; terms like *can*, *foil*, or *aluminium* were categorized as metal; and categories that did not clearly fit any group were assigned to other. This semantic grouping simplified the classification task and aligned the model with the project's goals.

3.3. Conversion to YOLO Format

The YOLOv8 model, used in this study, requires datasets to be formatted according to the Ultralytics specification. This format consists of one txt file per image, where each line encodes an object using its class ID and normalized bounding box coordinates, as well as a *yaml* file that defines the dataset's structure and class labels.

The conversion process began by generating a new JSON file with the updated superclass annotations. Image file names were standardized to avoid nested directory issues, and all images were reorganized into a single directory structure. The dataset was then split into training and testing sets using an 80/20 ratio. Using this structured data, the annotations were converted to YOLO format, ensuring compatibility with Ultralytics training framework.

This pipeline ensured that the input data was properly structured, semantically simplified, and technically compatible with the chosen object detection model. The preparation of the dataset was essential to avoid annotation errors and inconsistencies that could negatively impact training and evaluation.

3.4. Training Procedure

The model was trained using the Ultralytics YOLOv8 library, focusing on the YOLOv8n variant. Hyperparameters were selected to balance training efficiency and model generalization for the specific task of waste detection.

Training was divided into three distinct phases. In the first phase, training ran for 20 epochs with a learning rate of 1e-3, freezing the first 10 layers of the network to preserve pretrained weights. The second phase consisted of 40 epochs with a cosine annealing learning rate decaying to 5e-4, with freezing reduced to the first 8 layers. In the final phase, training lasted 40 epochs with a learning rate of 1e-4 and no frozen layers, allowing full network fine-tuning.

Input images were pre-processed by resizing to a fixed resolution of 360×360 pixels to ensure compatibility with the model input requirements and uniformity across the dataset. No further normalization or enhancement was applied. A batch size of 16 was used consistently across all training phases to ensure stable gradient estimates and efficient GPU memory usage.

Stochastic Gradient Descent (SGD) was used as the optimizer, combined with the learning rate schedule described above. This setup is already supported in the training process of the UltraRealistic YOLOv8n model. Data augmentation parameters included HSV space transformations with hue shift of 0.015, saturation shift of 0.7, and value shift of 0.4; vertical and horizontal flipping probabilities of 0.5 each; as well as random rotations, scaling, and composite augmentations such as mosaic. These augmentations increase the variability of training samples, enhancing model robustness and reducing overfitting risk.

Performance metrics monitored during training included precision, recall, mAP@50, and mAP@50-95, providing a comprehensive evaluation of predictive accuracy and model generalization capability.

4. Experiments And Results

The primary objective of the initial experiment was to enhance the performance of the YOLOv8n model in detecting challenging waste categories, particularly the glass and other classes, which consist of small or visually complex objects. It was hypothesized that increasing the input resolution to 720×720 pixels, combined with a batch size of 8 and training with Automatic Mixed Precision (AMP), would yield an improvement of 4 to 6 percentage points in the mAP of these classes.

The training procedure was structured into two phases. During the first phase, training was conducted for 20 epochs with a learning rate of 1e-3, while freezing the initial 10 layers of the network to preserve pretrained weights. The second phase consisted of 40 epochs, employing a cosine annealing learning rate schedule decaying to 5e-4, with the number of frozen layers reduced to the first 8 layers, allowing more of the network to be fine-tuned. The results obtained from the initial experiment training are described in Table 1.

Class	Precision (P)	Recall (R)	mAP@50	mAP@50-95
all	0.778	0.510	0.571	0.454
plastic	0.853	0.604	0.720	0.568
paper	0.898	0.781	0.859	0.737
metal	0.904	0.560	0.630	0.522
glass	0.419	0.354	0.257	0.164
other	0.819	0.253	0.392	0.280

Table 1. Results of baseline model

The adjustments in hyperparameters were intended to improve the model's ability to detect small and complex objects. Increasing the resolution allowed the model to process images with finer details, potentially facilitating the identification of smaller objects that might be overlooked at lower resolutions. The chosen batch size aimed to balance training stability with efficient GPU memory utilization, while AMP training was expected to accelerate the process and enhance generalization by using more precise numerical representations.

However, direct comparison with the baseline model, which employed a lower input resolution, revealed that although the other class experienced a modest improvement, the glass class exhibited a significant decline in both precision and mAP. Furthermore, the overall performance of the model with the altered configuration was inferior to the baseline, indicating that the resolution increase did not provide the anticipated benefits and may have adversely affected the model's generalization capabilities.

Table 2 shows the results obtained after adjusting the hyperparameters of the baseline model (Experiment A).

Class	Precision (P)	Recall (R)	mAP@50	mAP@50-95
all	0.477	0.378	0.379	0.246
plastic	0.577	0.546	0.542	0.339
paper	0.586	0.552	0.567	0.393
metal	0.558	0.482	0.504	0.337
glass	0.267	0.133	0.080	0.0406
other	0.400	0.175	0.204	0.120

Table 2. Results of Experiment A

Several factors may explain these results: The increased resolution amplified the model complexity and optimization search space, potentially requiring more data or longer training to achieve better generalization. The relatively small batch size may have limited gradient update stability, affecting convergence. Additionally, while AMP generally benefits training, it can sometimes introduce numerical instabilities, particularly in compact architectures such as YOLOv8n, which might negatively impact performance.

4.1. Inference

During inference, the model demonstrated near-perfect performance on images captured directly from ground level. The system accurately identified most waste items, although occasional false positives occurred, particularly misclassifications of plastic on non-plastic objects. It was also observed that detection accuracy improved when images were taken from a more distant viewpoint. The average inference time per image was approximately 3 milliseconds, highlighting the system's efficiency.

4.2. Visualization

The visualization and subsequent classification of images captured from ground level goes through several stages. The image is captured and then segmented to detect any object of interest to the model. As can be seen in images A and B in Figure 1, the detected objects are delimited by boxes to highlight the object and where the classification and the model's confidence in that prediction are made.

Figure 1 and Figure 2 shows two predictions, one correct and one incorrect. In Figure 1, the actual label would be metal, and the model correctly predicted *metal*. In Figure 2, the object belongs to the *glass* class, but the model incorrectly classified it as belonging to the *plastic* class, representing the model's limitation in correctly classifying glass.

In addition, the model shows some overdetection in objects found in the background of the image. As shown in Figure 3, the main object, which would be the soda can (class *paper*), was misclassified. Furthermore, secondary

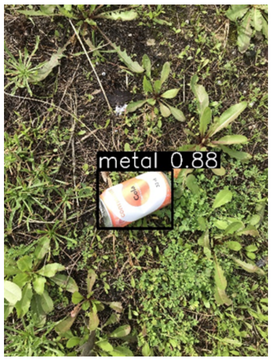

Figure 1. Correct prediction.

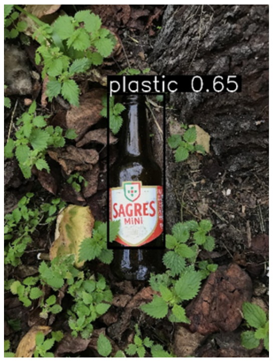

Figure 2. Incorrect prediction.

objects were detected in the image that are irrelevant to the model's objective. Furthermore, these secondary objects were very small in size, revealing that the model has high sensitivity.

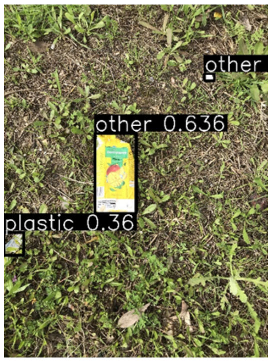

Figure 3. Incorrect prediction.

5. Conclusion and future work

Throughout the experiments, the model showed increasingly better results. However, it still has some limitations, such as detecting smaller objects or even misclassifying glass with other materials. These challenges will be addressed, and more robust results will be achieved, capable of handling any case.

In the future, the goal is to improve the model and make it more accurate, so that it can identify the objects it detects on the ground with as little error as possible. To this end, the data will undergo additional processing, which will allow the model to reduce the error rate, i.e., increase accuracy, recall, and, consequently, the F1-score. Another strategy to make the model more practical and integrable would be to use Edge AI to make prediction responses faster, as it focuses on local devices rather than cloud-based servers.

Once the model is further optimized, it will move on to the integration phase in a real system. Using mobile robot technologies, the artificial intelligence model for detecting garbage on the ground will be integrated into a robot, using sensors, which will patrol the streets and recycle all the garbage it finds.

This project will contribute to environmental sustainability by promoting cleaner cities and reducing the costs of cleaning urban spaces. However, it is thought that this project may serve as a complement to urban cleaning services and not as a replacement for them, only contributing, for now, to the cleaning of waste that has been left behind.

References

[1] Ceunen, B. (2021). *Garbage Detection with TensorFlow*. Obtido de Kaggle: https://www.kaggle.com/code/bouweceunen/garbage-detection-with-tensorflow

[2] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., & Berg, A. (2016). SSD: Single Shot MultiBox Detector. Em *Computer Vision – ECCV 2016* (pp. 21–37). Springer. 10.1007/978-3-319-46448-0$_2$

[3] Mandhati, S. R., Deshapriya, N. L., Mendis, C. L., Gunasekara, K., Yrle, F., Chaksan, A., & Sanjeev, S. (2024). *pLitterStreet: Street Level Plastic Litter Detection and Mapping*. https://arxiv.org/abs/2401.14719

[4] Melinte, D., Travediu, A.-M., & Dumitriu, D. (2020). Deep Convolutional Neural Networks Object Detector for Real-Time Waste Identification. *Applied Sciences, 10(20)*, 7301. 10.3390/app10207301

[5] Oliveira, M., & Guedes, E. (2024). Detecção de Lixo em Áreas Costeiras: Uma Aplicação de Segmentação com R-CNNs da Família YOLO. *Anais do WCAMA*, 11-20. 10.5753/wcama.2024.1902

[6] Proença, P., & Simões, P. (2020). TACO: Trash Annotations in Context for Litter Detection. *arXiv*.

[7] Valente, M., Coelho, A., & Assis, R. (2024). Sistema de detecção de resíduos em rios e nascentes utilizando visão computacional e inteligência artificial. *Revista de Gestão e Secretariado, 15(11)*, e4497. 10.7769/gesec.v15i11.4497

[8] Zhang, A., Lipton, Z., Li, M., & Smola, A. (2023). *Dive into Deep Learning*. Cambridge University Press.