FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Detection and classification of polluting waste in public spaces

Tamires Ribeiro Viegas

Mestrado em Engenharia de Software

Supervisor: Prof. António Fernando Vasconcelos Cunha Castro Coelho

Second Supervisor: Prof. Nuno Feixa Rodrigues

June 28, 2024

Detection and classification of polluting waste in public spaces

Tamires Ribeiro Viegas

Mestrado em Engenharia de Software

June 28, 2024

Abstract

This master's thesis addresses the problem of detecting and classifying hazardous waste in urban spaces. Urban environments are becoming increasingly loaded with hazardous waste, posing significant environmental risks in addition to those related to public health.

Continuous technological advances in object detection have the potential to detect and categorize this waste automatically and can serve as a tool for waste management purposes, including pollution prevention. In the following dissertation, we investigate object detection approaches using the YOLOv8 model trained with a custom dataset from images collected from the Open Images V7 and COCO datasets. In addition to training with the YOLOv8 model, hazardous waste detection was tested on a pre-trained Detectron2 model and custom training using images from the COCO dataset.

To create a custom dataset for hazardous waste detection, images were collected from two different datasets with classes that helped detect some objects. The datasets used were Open Images V7 and COCO.

Several experiments were carried out throughout the research with customized datasets from the COCO and Open Images V7 datasets to detect the waste sought here in this research. However, the results did not achieve the objective, leaving gaps and doubts about why hazardous waste was not found in the images collected in the municipality of Funchal.

The dataset stored on the Tidy City project server has thousands of images, but it is not impossible to say whether any of them contain hazardous waste. It is also impossible to say that there is no hazardous waste due to the low accuracy of some trained models.

The first experiment carried out with YOLOv8 and images collected from Open Images V7 obtained an mAP of 0.5 due to the lack of balance with the classes placed in the dataset but obtained an mAP of 0.78 for the microwave class, whose detection was suitable for images tested on random waste.

The second experiment performed with YOLOv8 and images collected from COCO obtained a mAP of 0.4 and failed to balance the classes. Even though the overall mAP was low, the individual mAP for the laptop class was 0.68, and the detection of random trash in the image was more assertive.

Experiments were also performed with Detectron2 pre-trained with a mAP of 0.9. Even though it was good, it caused confusion in some objects, not detecting all the laptops in the trash in the test image, only the sharper ones, as were the models trained here in the experiment.

We are concluding the need for a more worked dataset in outside images with several versions and angles.

Acknowledgements

Os agradecimentos por esse trabalho vão para os orientadores, familiares e amigos.

Exigiu muita dedicação e tempo para que fosse alcançado os resultados. Não foram os melhores, mas abriram portas para explorar mais o mundo de IA.

Gostava de agradecer também aos professores orientadores pela paciência com a entrega desse trabalho devido a gravidez que tive no caminho da entrega.

Tamires Ribeiro Viegas

"The greatest threat to our planet is the belief that someone else will save it."

Robert Swan

Contents

List of Figures

List of Tables

Abbreviations and Symbols

AI	*Artificial Intelligence*
AP	*Average Precision*
COCO	*Common Objects in Context*
CNN	*Convolutional Neural Network*
CV	*Computer Vision*
DL	*Deep Learning*
F1	*F1-Score*
FN	*False Negative*
FP	*False Positive*
GPU	*Graphics Processing Unit*
mAP	*Mean Average Precision*
PR	*Precision-Recall*
R-CNN	*Region-based Convolutional Neural Network*
ROI	*Region of Interest*
TP	*True Positive*
WWW	*World Wide Web*
YOLO	*You Only Look Once*
YOLOv8	*You Only Look Once version 8*

Chapter 1

Introduction

1.1 Problem and Context

When we think of environmental problems, a number of extremely important issues come to mind. One of them, which consequently ends up interfering in others, is the problem we have with waste management in urban spaces.

In public places we see more and more garbage in the wrong places, outside the garbage cans and garbage that can be lifethreatening. It's hard for people to be careful about the type of garbage they put in and out of the garbage can and they don't realize how dangerous this lack of care can be. The importance of understanding the waste we have and categorizing it in a more assertive way has become increasingly important in dealing with it Çipi (2023).

There's another problem that's also important to take into account when we talk about responsibility for the waste we throw away. The truth is that it's not just the citizens' fault for not being careful. The government also doesn't advertise very well where people can throw some types of unconventional waste in the garbage cans Neofotistos et al. (2022). For example, when you have a battery at home and it needs to be disposed of, people usually have no idea where they can throw this type of waste, which is very dangerous for the soil, and because they don't know, they end up throwing it in the wrong places, often going into ordinary garbage and contaminating the soil when thrown into landfills Çipi (2023); Neofotistos et al. (2022). Studies carried out in Portugal show that a very high percentage of waste is still sent to landfill Apambiente (2022).

This shows how much waste is not properly classified. The study also revealed a low percentage of hazardous waste studied, illustrated in figure 1.1, which raises the question of how this type of waste is being managed. And why is it important to pay attention to this type of waste.

Waste such as household appliances and electronics can release highly harmful substances and endanger the soil, contaminate the water and put the lives of people living near these areas in danger Rajesh et al. (2022).

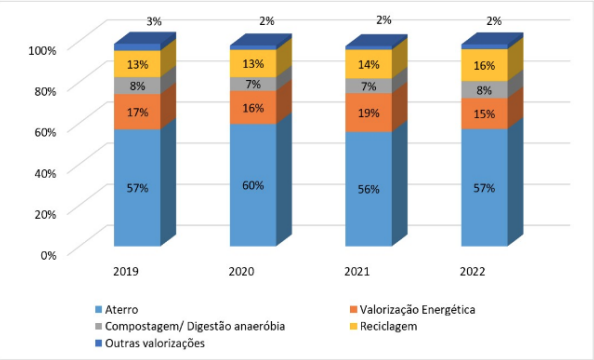

Figure 1.1: Graph showing the classification of waste in Portugal in 2022.

1.1.1 Difficulty

The problem of waste management is a much talked about issue in every country, none of which is exempt from the problem, not least because it is a global problem. The planet belongs to everyone, and the lack of care we take with the waste we generate puts people's survival at risk. Without good soil, where will food be produced? These questions have been debated many times, and it is not easy to find a solution to them.

The amount of waste and types of waste dumped is enormous, and sorting and separating it correctly is a very complex task, even for an AI. The environment can deform garbage and can also have various shapes. So, how can we build an algorithm capable of identifying any type of waste on the streets and categorizing it assertively?

The images in the datasets available for building an algorithm are not assertive, making it very difficult to detect hazardous waste. For this reason, the problem becomes increasingly difficult to solve. To solve the problem, the dataset must have several images of each type of waste and deformities registered and annotated to achieve good detection.

1.1.2 Initiative

The project was suggested by the INESC TEC institute (Institute for Systems and Computer Engineering, Technology and Science), which has been dedicated to solving real-world problems for more than 30 years INESC TEC (2022). The theme meets a very important need to help improve our planet through waste management. The institute has a partnership with the Tidy City project, from which images will be extracted to carry out training in detecting hazardous waste.

1.2 Motivation and goals

A sharp increase in hazardous electronic, chemical, and toxic materials accompanies the rapid urbanization and population growth that cities produce. If not properly disposed of, they can emit toxic materials into the atmosphere, polluting land, water, and air. Additionally, the inhalation of these substances by people can become dangerous for public health as they cause skin and

respiratory diseases, up to cancer. Thus, the early detection of these wastes in urban centers is essential to serve as a basic rule for reducing them and avoiding conflicts promoting health.

The primary goal of this project is to create a computer vision model that will be able to identify different hazardous waste types in urban areas with high precision. Deals with hazardous wastes like e-waste and other such waste, which are dangerous for the environment and human health. The model should be sensitive to the capacity to identify such wastes under various extramural conditions and complex city landscapes.

1.3 Classification

To achieve this dissertation's objective and detect hazardous waste in urban spaces, customized datasets were created and focused only on objects considered dangerous to the environment and people's lives. Images were collected from the Open Images V7 and COCO datasets to assemble a customized dataset with images of laptops, TVs, toasters, microwaves, cell phones, hair dryers, ovens, tablets, and refrigerators.

1.4 Document's Structure

Chapter 2: In this chapter, the dissertation explores detection techniques and deep learning models. It also presents evaluation metrics and benchmark datasets.

Chapter 3: Related Research Work. This chapter explores other work that has contributed to this research's advancement and future work goals. It also contains a survey of existing and available datasets.

Chapter 4: This chapter is dedicated to the state of the art. It discusses which datasets and models were used by the authors, what results were obtained, and the pros and cons.

Chapter 5: This section presents the methodology used in this dissertation. How the training and validations were carried out. Which models were used, and how was the customized dataset for training the models constructed.

Chapter 6: This section discusses the results obtained from the training and the losses and gains.

Chapter 7: Future work, discussion of this model and possible improvements on the algorithm or architecture, theories in optimization techniques

Chapter 2

Object detection models and architectures explored

Object detection is a vital sub-stream of Computer Vision that deals with identifying or recognizing objects (instances) within an image. The ability to effectively detect and classify images separates the success from failure for many Artificial Intelligence creations in real-world application areas. Object detection is a core problem in many applications, e.g security management systems, autonomous driving system, environmental monitoring, etc Redmon et al. (2016a).

The state of the fast-progress deep learning library has prevailed to support contemporary object detection and is thus predestined for optimal accuracy, but we have to train a new AI system from scratch. However, there is plenty of room to continue improving in both aspects (accuracy and computation efficiency) for optimal performance Lin et al. (2017).

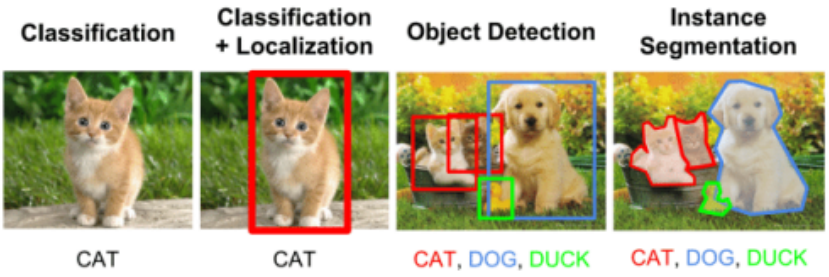

Figure 2.1: Computer vision techniques

Generally, object detectors are divided into two families: one-stage and two-stage. One has greater precision and the other is more agile in detecting objects.

One-stage detectors: One-stage detectors are known for their faster and more resource-efficient. Doing one-step object detection (Locate the place of objects on a piece and classify these pieces together with their label) using this model.One-stage detectors work even faster, that is very suitable for object detection under the real-time in one stage.

YOLO (You Only Look Once): YOLO is also among the most common one-stage detectors. Therefore it divides the image into a grid, and predicts each bounding box simultaneously with

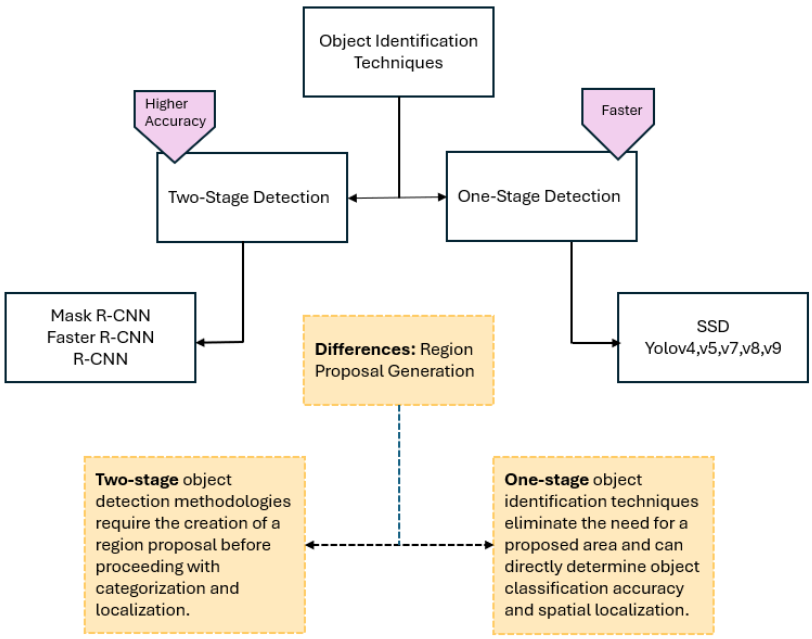

Figure 2.2: Schema two and one stages

class probabilities for each object in that cell. This technique is very fast, and its accuracy may be lower than that of two-stage detectors. Earlier versions were criticized for their accuracy in front of speed trade-off, but the latest ones, such as YOLOv8, are better on both axes and applicable to a wider range of use cases Redmon et al. (2016a).

RetinaNet: Focal Loss, which we briefly talked about in the section on one-stage detectors is proposed to handle the problem of class imbalance where objects from smaller classes can be under detected. RetinaNet is a one-stage detector that is faster than other such detectors in the market while offering better accuracy, hence making it suitable for use-case where we care more about speed Lin et al. (2017).

Two-Stage Detectors As the name implies, two-stage detectors split object detection into two major stages: first, image region proposals are proposed (Region Proposals), and then the actual objects inside these regions are detected by classifying them.

Faster R-CNN: Faster R-CNN is one of the famous two-stage detectors. In the first phase, a Region Proposal Network (RPN) is used to identify candidate bounding boxes. These suggestions are then further polished, and they are categorized as well to reveal the objects in an image. While Faster R-CNN achieves high accuracy, its longer run time also makes it less suitable for real-time scenarios than the one-stage detectors Ren et al. (2015).

Mask R-CNN: Mask R-CNN proposes a simple framework that can be used to speed up and improve several Computer Vision tasks, including localization (object detection), segmentation, and reading stuff. Besides object detection and classification, Mask R-CNN also provides a binary mask in which the exact outlines of each detected object are marked. It is helpful in tasks that

require high segmentation quality (such as object detection in complicated or congested environments) He et al. (2017b).

2.1 Performance evaluation and benchmark datasets

Chapter to present some basics of performance evaluation metrics and evaluations of how things work according to each model explored here in this dissertation.

Computer vision is a subfield of artificial intelligence that will allow machines to see the world as they actually understand and act upon visual info simply like humans. Computer vision is a field that studies how to make computers understand the contents of images from the real world, as well as acquire and identify data. This field of application has applications in the fields of security, healthcare, transportation, and manufacturing. Zhao et al. (2019)

2.1.1 Performance Evaluation in Computer Vision

Efficient performance evaluation of computer vision models is necessary to guarantee high standard functioning in real-time scenarios Zhao et al. (2019). Evaluation usually means comparing the actual model predictions with ground-truth labels of data. Evaluation metrics allow us to measure how well the model is doing Lin et al. (2017).

2.1.2 Performance metrics

Some of the main performance evaluation metrics for computer vision are:

Accuracy: The percentage of correct predictions on all predictions made by the model. Though accuracy is a straightforward metric, it can be misguided if we work with imbalanced datasets Lin et al. (2017).

$$\text{Precision} = \frac{\text{True Positive} + \text{True Negative}}{\text{Total Samples}}$$

Figure 2.3: Accuracy

Precision of Positive class: indicates the percentage of correct optimistic predictions. This is useful when the cost of a false positive is high, as in medical diagnostics He et al. (2017a).

$$\text{Positive Class Precision} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Positives}}$$

Figure 2.4: Precision of Positive class

Recall (True Positive Rate): The percentage of positive samples the model correctly identified. It is helpful in cases where we need to ensure that all true positives have been captured—there are a reasonable number of false optimistic predictions Girshick (2015).

$$\text{Recall} = \frac{\text{True Positive}}{\text{True Positive + False Negatives}}$$

Figure 2.5: Recall

F1 Score: This is the harmonic mean between precision and recall, which helps to balance both. In Girshick (2015), this is especially useful when precision and recall are competing.

$$\text{F1 Score} = 2 \times \frac{\text{Positive Class Precision x Recall}}{\text{Positive Class Precision + Recall}}$$

Figure 2.6: F1 Score

2.1.3 Applications of Metrics

Metrics like these are how the evaluation is performed per task and apply depending upon which task-specific context. In medical diagnostics, for example, false negatives can be more serious than false positives He et al. (2017a), as making mistakes of missing a disease is a higher cost than one patient receiving the wrong diagnosis. On the other hand, false positives might be more acceptable in a product recommendation system Lin et al. (2017).

For computer vision to be effective in security, it needs to decrease false negatives, which means we cannot miss any threat Zhao et al. (2019). However, it is also essential to reduce false positives to avoid unnecessary alarms, such as warnings and wasted resources Redmon and Farhadi (2018).

Analysis: Counting True Positives and False Positives

True Positives (TP): Indicates the cases in which the model correctly predicted positive class. A true positive would be — for example, a model correctly predicting an intruder in something like an intrusion detection system Zhao et al. (2019).

False Positives (FP): The model incorrectly predicted the positive class. For instance, still based on the same example, a case of false positive would be this model indicating that behind our closed door, there is an intruder and consequently generating an unnecessary alarm He et al. (2017a).

This is equivalent to both True Negatives and False Negatives :

True Negatives (TN): This is when the model correctly predicts the negative class. For this reason, the model is right not to assume there is an intruder Lin et al. (2017).

False Negatives (FN): When the model wrongly predicts Sarabchian and Mühlhäuser (2022) a True Positive class as Negative. In this case, an example of a false negative might be if the model failed to detect an attacker and the attack slipped under their notice Girshick (2015).

2.1.4 Example of confusion matrix

	Predicted Positive	**Predicted Negative**
Actual Positive	TP = 50	FN = 10
Actual Negative	FP = 5	TN = 35

Table 2.1: Confusion Table

True Positives (TP) = 50: The model correctly predicted 50 cases as positive.
False Negatives (FN) = 10: The model predicted 10 cases as negative, but they were positive.
False Positives (FP) = 5: The model predicted 5 cases as positive, but they were negative.
True Negatives (TN) = 35: The model correctly predicted 35 cases as negative.

2.1.5 Cross-validation

Cross-validation is essential in machine learning to test the performance of models with more confidence and avoid overfitting. Cross-validation performs in a way for the data distribution where it trains and fits the model using multiple divisions as opposed to training-testing split, compelling hyperparameters values being tested Hastie et al. (2009).

The simplest cross-validation method is k-fold-cross validation. Data is split into k subsets (a.k.a. "folds") of approximately equal size in this method. The model is fitted with the training set k times, wherein each run, one of the k folds from unsegmented data as a test dataset was treated alternately. Finally, the performance of the model is determined by averaging evaluation metrics over each architecture for k runs Kohavi (1995).

For example, say k = 5 (k-fold cross-validation), and the data is divided into five sets of trains in four parts and tests in the other part. Repeat this process five times, each time with a different part test set.

2.2 YOLO - You Only Look Once

Real-time Object Detection is an exciting area in computer vision, and YOLO (You Only Look Once) is one of the most famous real-time object detection systems. Mainly proposed by Joseph Redmon and others in 2016 Redmon et al. (2016a), YOLO is distinct from many popular methods, such as Faster R-CNN for being a single-stage object detection scheme, without the need for multiple stages to find a bounding box (BB) followed by classifying at its inside. The idea behind YOLO is very straightforward. The model detects and classifies objects in a single pass, so instead of object detection over multiple passes or stages, it executes both localization and recognition

simultaneously. This allows for a high-speed detector with relatively higher accuracy Redmon et al. (2016a).

YOLO works by splitting the input image into a grid (e.g., 13x13 or 19x19, depending on the architecture). The cells in the grid will predict a bounding box (during training, every cell is responsible for predicting two boxes) and its class Redmon et al. (2016a). YOLO considers object detection to be a single regression problem where it predicts both bounding box coordinates and corresponding class probabilities simultaneously.

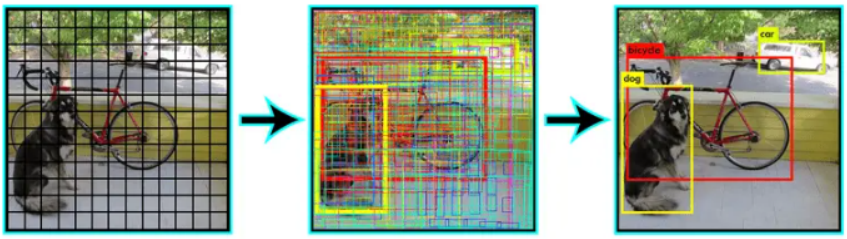

Figure 2.7: YOLO algorithm

Description of output: The number of grid cells in a single image (let us consider this as S) is divided among different bounding boxes such that each cell predicts at most B predictions, where 2 less B less equal 5. For each such predicted box, the model returns the following information:

Bounding box coordinates (x, y, w,h)

The certainty about the box storing an object

The probabilities for each object follow the class score.

Based on this answer, the model will combine all these predictions to get the final list of bounding boxes with high-confidence p-scores and then remove low-confidence detections (using, e.g., Non-Maximum Suppression NMS), which removes overlapping boxes for stuff like an input video or image that only has a few actual images in it towards basically trying to keep just whatever box is most confident above some threshold Redmon et al. (2016a).

2.2.1 Evolution of YOLO Versions

The architecture has come a long way since the first version of YOLO, improving accuracy, speed, and computational efficiency.

YOLOv5: YOLOv5 has good accuracy, but newer versions outperform it on more complex tasks or with smaller-scale objects contained within an input image. However, it remains more competitive for accuracy/speed trade-offs, especially in industrial/commercial environments Ultralytics (2020).

YOLOv7: Is more powerful and accurate than YOLOv5, especially on the COCO dataset. The model performs best in mAP, especially for small object detection and robustness against adverse environments Wang et al. (2022)

YOLOv8: The YOLOv8 model is the most accurate of all versions, and it detects complex objects, i.e., shadows or partially hidden objects, very well. Including new capabilities like instance

segmentation to the YOLOv8 makes it one of the most versatile and powerful computer vision tools Ultralytics (2023).

In terms of speed:

YOLOv5: It gained its name due to the speed it offers in terms of training and inference. This makes it perfect for use cases needing the lowest latency possible, such as real-time applications Ultralytics (2020).

YOLOv7: Though YOLOv7, then again, confirmation disguised improved exactness, absolutely protects exceptionally aggressive speed, making it reasonable to be great for constant applications. However, there could be a little increased latency compared to YOLOv5, especially for high-resolution images Wang et al. (2022).

YOLOv8: YOLOv8 was briefly architected for greater accuracy and less speed. This means that this model can run on models with low performance, but should it be characterized enough to take advantage of resource constraints Ultralytics (2023).

2.3 Detectron2

Detectron2 is PyTorch's open-source, high-quality object detection (instance segmentation) codebase. It implements state-of-the-art Computer Vision tasks like Object Detection and Instance Segmentation Transform Labs (2024) with it is its rich API to easily interact with machine learning models using JSON format datasets, significantly reducing coding effort to develop Machine Learning Models. Detectron2 is the second version of Facebook AI's open-source object detection library, released recently, and represents an evolution of Detectron that has been implemented using PyTorch. This design allows for computing research experiments with fewer lines and better performance during deployment compared to libraries built from previous frameworks, such as Caffe 2, which was used in early versions.

Detectron2 is a Computer Vision solution that consists of several Computer Vision algorithms. It distinguishes itself by providing the end user with great flexibility in that models can be created and tailored to target detection and segmentation tasks. Detectron2 is rooted in convolutional networks (CNNs) architecture, a type of deep neural network iot (2024).

At the high level, Detectron2 architecture consists of three main components.

Backbone: The backbone is the Convolutional network, which will extract features of the input image. Some famous backbones are ResNet and FPN(Functional et al.). This allows the model to capture fine details and global structures of the image He et al. (2016) by feature extraction in multiple layers of abstraction performed (backbone).

Detection Head: Generates bounding boxes and classes of the objects in the image after extracting features. Detectron2 has out-of-the-box support for many detection heads, including Faster R-CNN for object detection and Mask R-CNN instance segmentation Ren et al. (2015).

Segmentation Head: For instance, in segmentation, Detectron2 also employs a standard per-pixel binary classification ask over the object bounding boxes. The binary mask specifies exactly

which pixels in the image make up an object; these masks are beneficial for applications requiring very accurate contouring, such as medical analysis and Industrial automation He et al. (2017b).

Detectron2 is exceptionally well equipped to develop high accuracy on all computer vision tasks. It has been trained extensively using the COCO (Common Objects in Context) and Mapillary Vistas benchmarks, performing with more excellent precision/recall than other known object detection libraries Lin et al. (2014).

For object detection, Detectron2 uses Faster R-CNN, which falls under the category of two-stage approaches where it first generates regions-of-interest proposals followed by region classification. Although slower than one-stage detectors, two-stage detections are more accurate, especially in the case of complex scenarios where objects might be occluded or at different scales Ren et al. (2015).

Instance Segmentation: This is more challenging than object detection because it requires predicting the location and class and an additional accurate running mask that can delineate each object. Task detectron2, on the other, is one of the latest and should have high accuracy and an efficient architecture in terms of computer resources since it supports Mask R-CNN He et al. (2017b).

Detectron2 is more accurate than other computer vision libraries (like YOLO) anyway, and specifically, for instance, segmentation police, it works better compared to the rest. However, This greater precision comes with a higher inference time price, which can be cumbersome in real-time applications like security systems and autonomous driving Liu et al. (2016).

2.3.1 Limitations and Challenges

Finally, although Detectron2 is a great tool, it has some drawbacks. One of the biggest challenges is how computationally expensive it has been to train and infer with these complex models. In contrast, two-stage methods like Faster R-CNN and Mask R-CNN have a higher computational cost compared to one-stage method, which is computationally expensive edge devices Ren et al. (2015).

This may not be the best choice for real-time detection, in which inference time is a critical factor, e.g., Detectron2 March 2020 with the error of class) These models are much faster. Choose a model like YOLO or SSD if you need more speed (but potentially less accuracy). For use cases where the relative clinking FPS is concerned, Detectron2 could be a good option, and it went better than Yolo Redmon et al. (2016b).

While Detectron2 is highly flexible, it can also be cumbersome for those new to computer vision. There is some cost in requiring hyperparameter optimization to understand the underlying architecture and further requires a training procedure to be designed; these are barriers for most users. Nevertheless, comprehensive documentation and an enthusiastic developer community provide means to overcome these pitfalls Paszke et al. (2019).

2.4 Mask R-CNN

Mask R-CNN is a simple and effective Single Shot MultiBox Style method that adds a reshaping branch for segmentation to the existing Faster RCNN framework, thus extending it with the third task of instance-based dense-segmentation identification. Mask R-CNN, developed by Kaiming He and collaboration, further tackles a more complex problem in computer vision – instance segmentation, which demands object detection that outputs the bounding boxes with corresponding classes and accurate binary masks delineating the objects precisely.

2.4.1 How Mask R-CNN Works

Mask R-CNN and Faster R-CNN are two-stage object detectors. In the first stage, we use a RPN Region Proposal Network to produce ROIs that might contain objects. At the next level, these proposals are refined and used to predict object class and bounding box. The core of Mask R-CNN lies in generating an ROI for each object while simultaneously extracting a pixel (binary) mask that appears as the objective and making these predictions He et al. (2017b).

Mask R-CNN adds a branch to its architecture that is responsible for segmentation in parallel with the classification and box regression branches to accommodate this extra layer. The segmentation module is a tiny, Fully Convolutional Network (FCN) that predicts a segmentation mask pixel by pixel for each ROI He et al. (2017b). We desire the model to produce an accurate mask for every detected object without class, location, or scale dependency.

2.5 Faster R-CNN

Faster R-CNN is arguably one of the most influential models in computer vision, and has revolutionized object detection. Faster R-CNN: The Faster R-CNN Ren et al. (2015) was created by Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun, covers the successive models of region-based CNNs, i.e., Fast RCNN (a better version of its predecessor-RCN), but most crucially it is responsible for bringing about an end to racing in real-time object detection: a Region Proposal Network. In this paper, the Faster R-CNN model will be reviewed in a critical way to show how it works, its accuracy and characteristics, and its strengths and weaknesses in detail.

2.5.1 How Faster R-CNN Works

The RPN is efficient and can be shared between multiple detection and segmentation tasks to improve performance while saving on inference time. This adaptation of Faster R-CNN makes it versatile enough to be used for a wide variety of applications, from real-time ones such as pedestrian detection in autonomous vehicles Ren et al. (2015) to tasks requiring high accuracy on platforms that operate with NLPs running at lower frame rates or have a more complex environment around them.

After the RPN generates the ROI proposals, it proceeds to a second stage where a classification and regression head predicts the class for each object and adjusts bounding boxes so that they align better with the edges of objects. This process is vital to finetune the RPN proposals suggested in the first stage for better model precision Ren et al. (2015).

Faster R-CNN employs a Region of Interest Pooling (ROI) layer by which proposals with different sizes are downsampled to fixed sizes before being fed into the classification head. This is important for the model to be scale-agnostic, as it makes a general approach in handling proposals of multiple scales Ren et al. (2015).

2.5.2 Accuracy and Performance

Faster R-CNN is well known to accomplish accurate object detection on benchmarks like COCO. Faster R-CNN has shown state-of-the-art results in the context of mean average precision (mAP), and its generalization performance is excellent when high accuracy detection for small or partially occluded objects is needed Lin et al. (2014). Moreover, Faster R-CNN enhances its detection performance because it uses a region proposal network (RPN), which generates high-quality proposals resulting in fewer false positives and negatives in the final prediction Ren et al. (2015).

Unfortunately, while the time it takes to do so is significantly longer due to this high accuracy, faster R-CNN is not the state-of-art among single-step detectors, i.e., you can find much faster tools, for instance, YOLO or SSD. It means it cannot be used in situations requiring very low latency — e.g., autonomous vehicles or real-time security surveillance systems Redmon et al. (2016b).

Chapter 3

State-of-the-art Analysis

3.1 Datasets

Datasets play a crucial role in designing and implementing such new classification algorithm for garbage containing image data which is already well-chunkified along with the annotations so that all could be reused on an as-needed basis during the course of study. Any research may practically count on a diversity of datasets designed to serve specific purposes, yet they are used concurrently with the same aim that would be advancing knowledge and application in their sectors.

We consider, in this context for selecting the datasets well tailored to detect and classified hazardous waste within those limits. Thanks to these and other specialized respositories AgaMiko (2023), where a comprehensive compilation of such resources can be found, we are able to pinpoint the most optimal ones for codebase validation testing on hazardous waste identification models. These repositories help to identify available datasets, with a way for researchers to filter these by specific criteria (e.g., waste type, annotation coverage and relevance of the dataset).

Choosing sets of such datasets that have been aggregated in a careful manner ensures not only enriched quality results but also guarantees better tuning and precision enablement within the model trained to identify or classify hazardous waste from its safe-to-environment counterparts, which is imperative in reducing environmental impacts by developing smarter methods for more efficient waste management.

TrashCan 1.0 - The TrashCan dataset is a collection of images I have annotated (currently 7,212) which depict multiple instances for various objects such as trash or ROVs and many types underwater flora fauna. The dataset annotations in this case took the form of instance segmentation annotations: mappings where, for each object id present in an image at least once across all training and test images, there is a bitmap containing both foreground/binary mask as well as background class that differentiates which pixels contain each oak canopy. TrashCan is provided with the imagery from J-EDI (JAMSTEC E-Library of Deep-sea images). of Minnesota (2018).

TACO - Here is the latest live version of this dataset as it gets updated. At the time of writing this article, the total number of images on our server is 1500 with 4784 annotations and still many more to be annotated from new uploaded image files=3918. These annotations fall under

60 classes that are part of 28 super categories Here you can see the counts for both category and super-category annotations. Dataset (2024c).

UAVVaste - The UAVVaste dataset consists to date of 772 images and 3718 annotations. The main motivation for the creation of the dataset was the lack of domain-specific data. Therefore, this image set is recommended for object detection evaluation benchmarking but also for developing solutions related to UAVs, remote sensing, or even environmental cleaning. The dataset is made publicly available and will be expanded Kraft et al. (2021).

Trashnet - The dataset consists of six classes: glass, paper, cardboard,plastic,metal and trash. There are 501 glass, 594 paper, 403 cardboard, and metal images (totaling to a count of 1500), while already established datasets only cover around that amount Thung and Yang (2016). The images are low resolution as they were captured on cell phone cameras. The photos are 512 x 384.

TrashBox - The dataset contains images of the following classes: glass, paper cardboard, plastic (waste), metal and a few pictures record for transparency. The dataset has 501 glass, 594 paper, 403 cardboard, 482 plastic ,410 metal and137 trash from the TrashBox with total amount of2527 images Kumsetty et al. (2022). The high resolution is not there due to the fact that it was taken by a cell phone camera. Photos are 512 x 384 Trash Object Dataset For Garbage Classification And Detection These are image data with labels gathered from waste objects, they have 17785 samples divided into classes :glass, plastic,metal, e-waste (wires etc.), cardboard,paper and medical waste. Sub-categories, based on class Hospital waste: syringe, surgical gloves, mask and medicine each with a sum of 2010 images; E-waste: It is related to electronic products such as computer chips, laptops (And so on.), Apple accessories,Optical cables for TV set-top box power supplies and NVC remote controls plus other electrical appliances and cigarettes class among plastic classification count. This class includes a combined amount of 2669 images and the items in them are plastic waste: bags, buttons,bottle caps,cups, electronic cigarettes. Paper waste: paper, food packaging paper cups and so on, 2695 images. Metal waste: cans, metal construction trash and other metals (2586 images) It also has a glass class, which had 2528 images yesterday and cardboard with an image count of 2414.

COCO - Common Objects in Context (COCO) is a dataset Dataset (2024a) widely used by researchers for initial model analysis. The goal of this dataset is to support object detection, instance segmentation and image. Over 200,000 annotated COCO images. The 80 common object operations, including people and animals, cars and other means of transportation, from appliances to chairs and tables, apples in the refrigerator, different epistemic versions. In COCO, there are multiple image instances, each with varied objects to create a rich scene for developing AI algorithms in benchmarking.

Open Images Dataset - Open Images Dataset V7 Dataset (2024b) is one of the biggest and most annotated image datasets. Its 9 million annotated images situate it as a robust resource for anyone involved in AI research, product development or machine learning work of any sort. The size of the dataset is not its onlyistingu characteristic. The annotations are of high quality. It has more than 16 million bounding boxes for 600 classes which makes Object detection in this model is high accurate and precise. The Open Images Dataset V7 also features the world's first instance

segmentation, checkpoints and visual relationships between objects annotations.

3.2 Versions of YOLO modules for object detection

Garbage management in urban areas is to segregate waste into different categories using CNN implementation for object detection. Urban waste is, in turn, a growing challenge as more and more of the world's population clusters into cities. Inevitably, this new model will be asked about (challenged) as well. However, there will be a need for a significantly more optimized model with better waste detection techniques.

In the viewing section Li et al. (2024), Li, Xu, and Lui develop the latest advancements for technique improvement and optimization regarding resource-constrained computational. They are turning the YOLOv8 model even more accurate and faster with the CG-HGNetV2 framework and MSE-AKConv module. The CG-HGNetV2 can use local, contextual, and global information hierarchically, similar to MSE-AKConv. It has an adaptive attentive mechanism that helps the model facilitate itself for complicated backgrounds with different features. All of these technical alterations are crucial in the case of increment detection under challenging conditions or model optimization for edge devices.

Models	P (%)	R (%)	Params (M)	GFLOPs	mAP@0.5 (%)	Inference (ms)	Old Clothes	Beverage Cans	Cartons
YOLOv3-tiny	38.1	39.9	8.77	13.0	34.8	2.1	0.431	0.426	0.288
YOLOv5s	65.5	56.8	7.13	16.1	61.5	6.4	0.600	0.663	0.809
YOLOv6s	74.3	61.7	16.31	44.1	68.8	6.6	0.622	0.911	0.880
YOLOv7-tiny	65.4	51.7	6.12	13.4	55.6	6.8	0.613	0.529	0.887
YOLOv8s	74.2	62.7	11.14	28.5	69.6	7.1	0.643	0.670	0.854
Enhanced YOLOv8	79.0	62.8	10.41	28.4	70.9	9.6	0.861	0.942	0.950

Figure 3.1: Performance Comparison Between YOLO Models

In general, the method proposed in the paper is quite fine. We also use a series of feature fusion mechanisms and data augmentation techniques, besides proposing a novel loss method called MPDIoU to enhance the bounding box regression. As shown in the published study, this strategy definitely increases the accuracy levels of results when applied to the "Huawei Cloud" dataset. This method can be further enhanced by exploring the challenges of the solution, like very expensive computational consumption and the time required for AI training.

While the paper shows some significant improvements, we should not avoid context limitations as well. The first is that the requirement for certain data from a dataset produced by Huawei Cloud; thus, generalizing it to other contexts or types of residues can be impeded. Further, higher complex frameworks such as the proposed CG-HGNetV2 may result in slower inference on devices having very limited resources, which might not be applicable for all the application scenarios.

3.3 Waste management in Bangladesh

This study by Roy, Islam, and Rafi Roy et al. (2024) with the title "Detection and Classification of Non-Biodegradable Plastic Waste Using Deep Learning: Environmental Scenario of Bangladesh"

is one great achievement among studies on solid waste management scenario from depth to detection and classification practicum in SSWM based upon most occurring pollutions called plastic wastes. Given the environmental situation, this study is very important. Plastic waste is a global issue, but the context of that problem and how important (or urgent) it may be in one local are so entirely different from another or even two sides within an ocean.

To mitigate the issue urgently, especially gasoline in countries with high populations and low amounts of garbage management (e.g., Bangladesh is a very perfect example as it often has poor rating among nations on this subject), The paper uses YOLOv8 to give an estimate for detection of non-biodegradable plastic waste. Such waste accumulation has a catastrophic effect on the environment and local communities Roy et al. (2024).

The method the study uses is very well established and proven. For their deep learning model, the authors chose to use YOLOv8 which an efficient and accurate object detector with high inference speed. The dataset used for training the model was gathered from Bangladesh as described in Roy et al. (2024). The dataset is formed by images of plastic waste photographed in different conditions from rivers, streets or landfills. Therefore, the model can be useful for addressing a variety of environmental challenges found in Bangladesh and due to this versatility it has good generalisability.

In addition, one of the pros of the article is that the authors have applied a few data augmentation techniques; hence, the model generalizes from a small dataset. Even though we trained the model on reconstructions from nearly all lighting and view conditions, it should still generalize fine under similarly realistic lighting/viewing parameters that are likely to be outside of testing data.

The results from the article look promising. This model has been shown to detect and classify non-biodegradable plastic waste with much higher accuracy compared to the previously used methodologies, which opens for its further use on a larger scale. In environmental aspects, plastic waste segregation is a breakthrough, and it leads to a huge scope for revolution in solid waste management in Bangladesh, which also allows the application in different countries.

That being the case, certain limitations require noting recognized, nonetheless positive interpretations. The experiment was held in a specific location, so the input photos are all images that originated there, making it more constrained. However, scenarios related to varied cities, nature, etc., were neglected. However, the model is of no use outside that location; its training has already been tailored to particular requirements (in spite of the augmentation technique).

3.4 Data Augmentation

The research paper Swathi et al. (2023) has filled a vacuum in the deep learning arena through an exhaustive representational analysis of image augmentation method. It is important because in an image a single object can look like infinite different things with it changing from one perspective to the other, under lighting conditions and as per context.

In addendum, the work covers a variety of data augmentation techniques, from simple to more complex ones (e.g., rotation/flip vs longitudinal shifting).

Common, simple data augmentation techniques: Rotate translate flip brightness300 × 80Add localized deformation and apply maximal distortion to increase the invariance of images. These methods are simple to implement and, indeed, utilized in virtually all computer vision applications because they offer straightforward benefits without additional computational expense. These methods do achieve their purpose, but according to the authors have trade-offs. They may, for example, add some noise to the data when used inappropriately.

3.4.1 Rotation and Scaling

Rotation and scaling are two fundamental geometric transformations that we use to change the orientation of an object as well as the shape of an image.

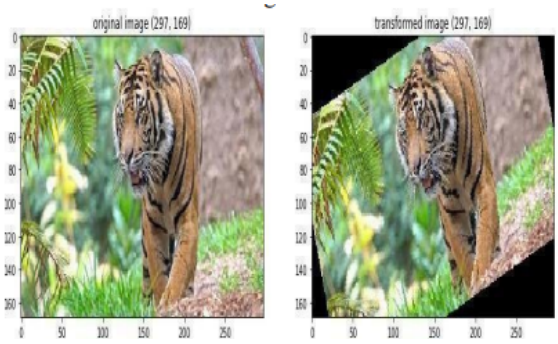

Figure 3.2: Rotation and Scaling

3.4.2 Flipping and Cropping

Turning the image either on the left or right doesn't change its classification in most cases Swathi et al. (2023). Flipping and cropping are some of the simplest methods to tweak an image's perspective or, more precisely, how it tells a story.

3.4.3 Changing Color

Basic Color Space Transformation Techniques, changes in image colors (brightness-contrast-saturation adjustments).

3.4.4 Image Noising

This is a kind of data augmentation, and the purpose is simply to generalize your model towards undesired variations by directly adding noise into images.

Figure 3.3: Image Noising

3.4.5 Image Blurring

One simple technique is applying blur to the images, simulating out-of-focus or other imperfections commonly found in image capture.

Figure 3.4: Image Noising

3.4.6 Geometric Transformation

Analogous to the aforementioned canonical transformations, great geometric operations may help in rotation and scaling but are likely going to be more complex distortions that need access to some model of the image feature space.

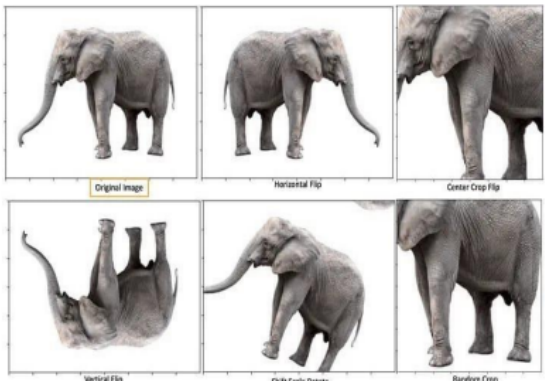

Figure 3.5: Geometric Transformation

3.4.7 Color Space Transformation

However, a color space transformation in itself can be as complex as converting from one type of Color model to another (such as RGB —> HSV), allowing fine grained adjustments at the level of each image element inside this color spectrum.

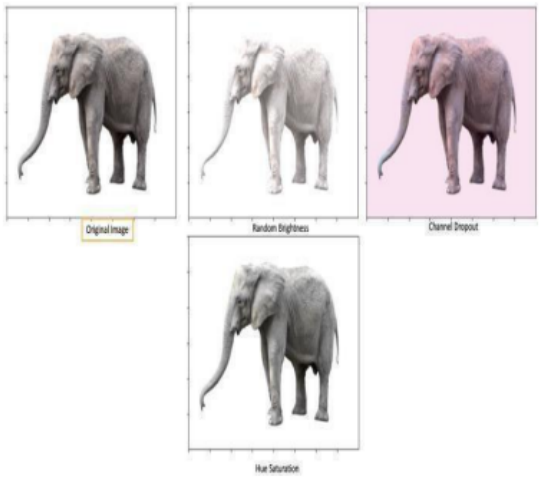

Figure 3.6: Color Space Transformation

3.4.8 Kernel Features

This method uses kernels to find certain features in an image (such as edges or textures) and is considered sophisticated due to the complex nature of multivariate kernel sectioning.

3.4.9 Random Erasing

It is an advanced technique where training images are randomly erased by masking blocks of the snapshot with random numbers to increase the robustness of the model over occlusions and other variabilities.

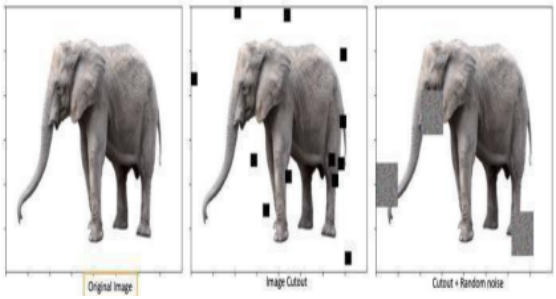

Figure 3.7: Random Erasing

3.5 Dataset Consideration

Li, Xu, and Lui [26] used an upgraded YOLOv8 to recognize waste in urban areas. The dataset was Huawei Cloud, a waste in the city's construction. The dataset is notable for its high image quality and the range of settings, which helped the model be able to recognize various types of waste in a variety of scenarios.

A study by Roy, Islam and Rafi that was interested in the detection of non-biodegradable plastic waste using Bangladesh as its model. The dataset they used was directly obtained in the country and had images of garbage, which were located inside rivers, on streets, or dumpsites. This dataset could be particularly pertinent to the environmental context of Bangladesh. However, generalization to other geographic contexts is likely limited.

3.6 Models Consideration

3.6.1 Accuracy

When incorporating the CG-HGNetV2 framework with the MSE-AKConv module, our model achieved an mAP of 87.4 percent. The high accuracy was mainly due to the various feature fusion and data augmentation techniques used, which made it robust enough for variability in different kinds of data. Although high mAP performance, it also brings the extra computational cost for the segmentation step and may not be suitable for processing in resource-constrained devices. Additionally, the mAP50 was about 94 percent, indicating that it worked in easier-to-detect cases, but during more difficult scenarios, such as partially hidden residues, performance dropped.

The mAP of 82.5 percent on plastic waste detection was achieved by the YOLOv8 model. Data augmentation techniques were used to increase accuracy while handling a small dataset. The receptive field was extended using images taken at various lighting conditions and different angles, further helping in boosting the model performance. However, the mAP50 was 90 percent, which means that in optimal conditions, it worked very well (to localize waste correctly). At the same time, performance degrades under more stressful circumstances, such as low lighting or inter-class confusion with other objects.

3.6.2 Speed

The speed of the YOLOv8 processing, as shown in Li et al. (2024), is about 45 FPS (Frames Per Second), which can be considered acceptably fast for most Real-time applications and scenarios. Although the use of advanced feature fusion techniques also increased, it is narrower than before, e.g., in comparison with YOLOv5. The authors mentioned this trade-off between speed and accuracy while targeting applications on resource constraint devices which is one of their primary contributions in the review.

The prediction speed of the model was about 38FPS - a bit slower than that reported in (Roy et al., 2024) because they needed different tuning to fit an extremely diverse set of environmental conditions present on their Bangladesh dataset.

3.6.3 Pros and Cons

Training these models was the most lengthy process; each model had strengths and weaknesses. A comparison of all of them can be found here. Indeed, it depends heavily on what you want to achieve or where you would like to use object detection.

For the paper of Li, Xu, et al. 2024 Pros: Great generalization performance for complex urban waste with high accuracy. Handles varying lighting and shaping conditions. Cons: Feature fusion techniques increase the inference time and computational resource consumption, which can be an issue in mobile or embedded devices.

Bangladesh Roy et al. (2024) Pros: Excellent adaptability, withstands data augmentation, and is more precise. Cons: Cannot generalize because the dataset is too small. Not being as fast on inference speed can be a bottleneck for some use cases with high processing rates.

Chapter 4

Methodology

The research aims to detect hazardous waste in the municipality of Funchal-Madeira, Portugal, through images captured from a car with a standard camera in the municipality of Funchal. Therefore, this chapter will be dedicated to the methodology applied to creating a customized dataset for hazardous waste in urban areas and the training of models such as Yolo and Detectron2 to detect these hazardous wastes. The formation of the customized dataset and how the training and validation of the detection models occurred will be discussed.

4.1 Custom Dataset for Hazardous Waste

To create the custom dataset focused on hazardous waste, images were collected from two existing datasets, the Open Images V7 and the COCO dataset, which have thousands of images containing various objects, including those sought in this research. Objects were selected from both datasets but trained separately to evaluate the results.

In the first experiment, using images from the Open Images V7 dataset, only the "Laptop" category was selected, with 1501 training images. The model was trained with these images, and some analyses were performed to proceed with the creation of the new custom dataset.

In the second experiment, using images collected from the Open Images V7 dataset, 2666 images were selected for training and 266 for model validation. The proposed custom dataset was created in the following categories: Laptops, microwaves, tablets, hair dryers, and heaters.

For the third experiment with images collected from the COCO dataset, a total of 4092 images were selected for training and 599 images for model validation. The proposed custom dataset was created using the following categories: Laptops, microwaves, tablets, hair dryers, keyboards, toasters, kitchen ovens, TVs, and cell phones.

The custom datasets were trained and validated using YOLOv8n (nano), which has somewhat low-performance metrics, as exemplified in table 4.1 and 4.2. The datasets were not trained on other models with better mAPVal, such as YOLOv8x, as a machine with a GPU would be required to continue training the data.

Model	**Size (pixels)**	**mAPval 50-95**	**Speed CPU ONNX (ms)**	**Speed A100 TensorRT (ms)**	**Params (M)**	**FLOPs (B)**
YOLOv8n	640	37.3	80.4	0.99	3.2	8.7

Table 4.1: Performance metrics of the model used in this project for detecting hazardous waste using images from the COCO dataset.

Model	**Size (pixels)**	**mAPval 50-95**	**Speed CPU ONNX (ms)**	**Speed A100 TensorRT (ms)**	**Params (M)**	**FLOPs (B)**
YOLOv8n	640	18.4	142.4	1.21	3.5	10.5

Table 4.2: Performance metrics of the model used in this project for detecting hazardous waste using images from the Open Images V7 dataset.

4.2 YOLOv8 Model and Open Images V7

The YOLOv8n model, as described in the previous section, was used to train the customized dataset. YOLOv8 (You Only Look Once, version 8) was chosen for this research because it is faster than two-stage models and has a high accuracy rate for object detection. Since the project aims to validate captured images and apply the model quickly and easily, YOLOv8 is a good solution for the proposed problem of detecting hazardous waste.

4.2.1 Custom Dataset Classification

YOLOv8n uses the bounding box format to locate an intended object in an image containing its location and store it in coordinates in a txt file with the same name as the image. Within this txt file, the first position of the text also contains the number of the class of the object being specified. In our case, the classes “Microwave” were chosen as 0, “Laptops” as one, “Tablet” as two, “Hairdryer” as three, and “Heater” as four.

When downloaded, the Open Images V7 dataset does not contain the annotations in the format expected by YOLOv8. The classifications are placed on the labels with the names of the objects. For this reason, it was necessary to develop a code in Python 3 to convert all the labels of the images to numbers.

The laboratory used to create the custom dataset was Google Colab, using a simple CPU for the task.

4.2.2 Training with YOLOv8 model

For training the model, 2666 images from four classes (Laptops, microwaves, tablets, hairdryers, and heaters) were used. The division of the images by classes is illustrated in Figure 4.1.

The machine configuration used for training was a 4-core, x64, i7 hp CPU for developers. The model has its settings in the config.yaml file with the location of the customized dataset along

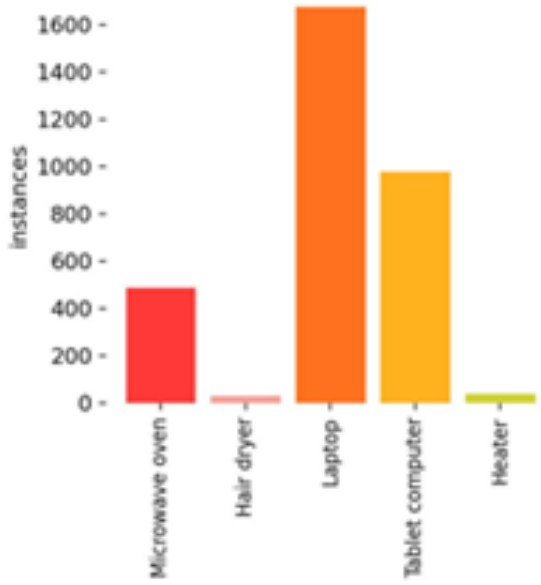

Figure 4.1: Image division by classes for the Custom dataset trained with YOLOv8n.

with the labels, separated by training and validation. The model used does not use conventional cross-validation like k-fold, it uses a simple approach, this means that the train does not divide the data into several parts and the training is done multiple times in different divisions. The dataset is divided into two parts: Training (train) and validation (val), 90 percent of the data is used for training and 10 percent for validation. One hundred epochs were used to reach an overall mAP of 0.5. The classes have different mAPs due to the number of images, which helps the model be more assertive. For example, the Tablet class has a mAP of 0.81.

4.2.3 Validating Model

To validate the model, 266 images with four classes (Laptops, Microwave, Tablet, Hairdryer, and Heater) were used.

In the results of Figure 4.2, it is possible to notice that the model's detection has not yet reached the final objective. Some images have satisfactory accuracy, others have low accuracy, and some also have errors in object detection.

4.2.4 Applying model training in the proposed Funchal scenario

A code was developed in Python to validate the images collected in the municipality of Funchal and to check whether it was possible to find any hazardous waste in the images. The code created analyzes the images in real time using the tidyCity project API, downloading the images, analyzing the content, and applying the trained model. However, even after changing the precision and accepting any prediction, no hazardous waste was found in the collected images on the server.

Figure 4.2: Validation of the YOLOv8n model with the customized dataset using images from the Open Images V7 dataset, trained with 100 epochs.

In order to test the model and verify why no images with the trained waste were found, a code was created in Python to validate input images to check the image output and ensure that the model learned something and could find some trained object.

The results may be a false negative, as it is impossible to truly state that no images are collected in Funchal with hazardous waste on the streets. The person responsible for the set of images collected in Funchal was also asked, and he was also unable to say whether there were images with the proposed objects (hazardous waste) to be found.

In Figures 4.3 and 4.4, it is possible to see that the proposed objects were found, but their precision was not good.

4.3 Model YOLOv8 and COCO

For this new experiment, the same YOLOv8n model is used, but with training and validation, another customized dataset using images from the COCO (Common Objects in Context) dataset is created. The COCO dataset includes objects in a wide context of scenes, ranging from indoor to outdoor, which makes it more balanced and suitable for object detection training. The COCO dataset is among the most popular reference datasets for object detection and segmentation.

4.3.1 Custom Dataset Classification

As described in section 5.2.1, YOLOv8 uses bounding boxes to localize objects within the image. For a new training session with a dataset of classes slightly different from the previous ones, images were collected from the COCO dataset.

The images were downloaded from the COCO dataset website using Python code to download only the desired images. Another Python code was also created to transform the labels file in JSON format into Txt files with the names of the corresponding images downloaded. The program searches this JSON for the image category and reads the parameters of the bounding boxes, thus

Figure 4.3: Test to validate accuracy and detection of microwave object with trained YOLOv8n model.

creating the corresponding txt. Once the data had been prepared with labels and ready images, they were separated into training and validation. The separation took place as follows:

- **Training Set:**
  - TV: 713 images
  - Toaster: 217 images
  - Refrigerator: 704 images
  - Oven: 718 images
  - Microwave: 698 images
  - Laptop: 747 images
  - Keyboard: 712 images
  - Hair Dryer: 189 images
  - Cell Phone: 705 images
- **Validation Set:**
  - TV: 154 images
  - Toaster: 8 images
  - Refrigerator: 101 images
  - Oven: 115 images
  - Microwave: 54 images
  - Laptop: 166 images
  - Keyboard: 106 images
  - Hair Dryer: 9 images
  - Cell Phone: 143 images

Figure 4.4: Testing to validate accuracy and detection of Laptop object with trained YOLOv8n model.

4.3.2 Training with YOLOv8 model

To train the model, 4092 images were used in nine classes (Laptops, microwaves, TVs, hair dryers, toasters, refrigerators, ovens, keyboards, and cell phones). The division of the images by class is illustrated in Figure 4.5.

The machine configuration used for training was a T4 GPU with the Google Colab platform. The model has the settings in the config.yaml file with the location of the customized dataset next to the labels, separated by training and validation. The machine configuration used for training was a T4 GPU with the Google Colab platform. The model has its settings in the config.yaml file with the location of the custom dataset next to the labels, separated by training and validation. The model uses conventional k-fold cross-validation, since the scenario sought here in the study is very specific to electronic objects and does not require a broad generalization of data. In addition, cross-validation, especially in complex models such as YOLOv8, can be computationally expensive and time-consuming.

The training uses a simple approach, which means that the training does not split the data into multiple parts. The dataset is divided into two parts: 90 percent of the data is used for training and 10 percent for validation.

One hundred epochs were used to arrive at an overall mAP of 0.4. The classes have different mAPs due to the number of images, which helps the model to be more assertive. For example, the TV class has the best mAP of 0.62, and the cell phone and hair dryer classes have low mAPs of 0.13 and 0.

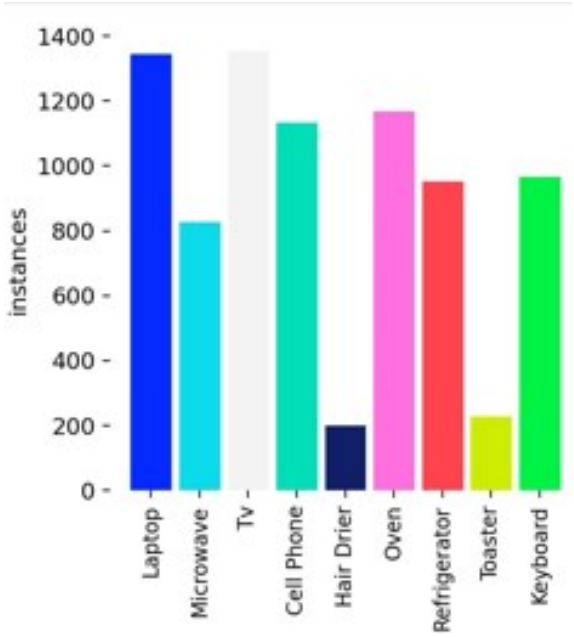

Figure 4.5: Image division by classes for the Custom dataset with images collected from the COCO dataset and trained with YOLOv8n.

The training results used 168 convolutional and normalization layers, 3,007,403 trainable parameters, and 8.1 GFLOPs, showing that the model performs 8.1 billion floating point operations per second.

The total evaluation metrics covering all classes were 599 images used for validation, 1144 instances detected, an average detection precision of 0.67, recall of 0.35, mAP50 of 0.4, and mAP50-95 of 0.2.

Figure 4.6 shows all the results separated by class.

4.3.3 Validating Model

To validate the model, 599 images were used in nine classes (Laptops, microwaves, TVs, hair dryers, toasters, refrigerators, ovens, keyboards, and cell phones).

In the results of Figure 4.7, it can be seen that the model's detection has not yet reached the final goal, with some images having satisfactory accuracy, others having low accuracy, and some with errors in object detection.

4.3.4 Applying model training in the proposed Funchal scenario

As mentioned in section 5.2.4, a Python code was developed to fetch images from the tidyCity project server, but even applying the model trained with the new customized dataset with images from the COCO dataset, it was still not possible to find images with hazardous waste.

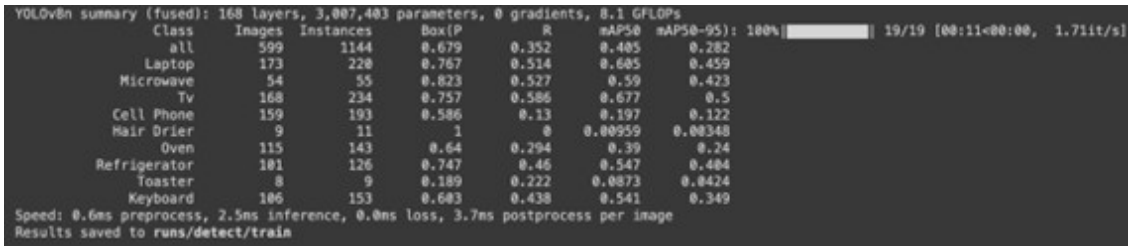

Figure 4.6: Summary of training results using the custom dataset based on images collected from the COCO dataset using the YOLOv8n model.

However, for testing purposes, they were tested on the same image used in section 5.2.4 to check the accuracy values that resulted in finding the desired object in the image, which, in the case of Figure 4.8, was to find the laptops in the rubbish bin.

It is possible to observe in Figure 5.8 an improvement compared to the model using images from the Open Images V7 dataset (Figure 5.4). Even though the mAP is lower for the laptop class in this training with the images from the COCO dataset, the accuracy is higher, and there is no longer any confusion with the image.

4.4 Detectron2 and COCO

To enrich the research and validate the two types of image detection, both one-stage and two-stage, Detectron2 was used to analyze the results in hazardous waste detection.

Detectron2, developed by Facebook AI Research (FAIR), Detectron2 is a flexible modular library for state-of-the-art deep learning object detection, instance segmentation, and image classification tasks Abdusalomov et al. (2023).

This chapter of the thesis will now discuss how instance segmentation was performed on a standard setup using the COCO (Common Objects in Context) dataset, Mask R-CNN architecture with ResNet-50 backbone, and Feature Pyramid Network (FPN). The configuration was done using the Detectron2 Model Zoo library, which offers pre-trained and well-tuned models for different computer vision tasks.

This section will also demonstrate an attempt to train the Detectron model with a customized dataset for hazardous waste only in order to analyze the detections, results, and metrics.

4.4.1 Pre-trained model

First, the default configuration was tested to understand how it worked and how accurate it was for finding objects, in this case, hazardous waste.

The Google Colab Platform is used, using a T4 GPU, it is also necessary to make some installations such as torch and the Detectron2 library in the environment so that the pre-trained Detectron2 model can be used.

Other settings for applying the template to the image are also used, such as configuring the file with the setup described in paragraph 3 at the beginning of this section.

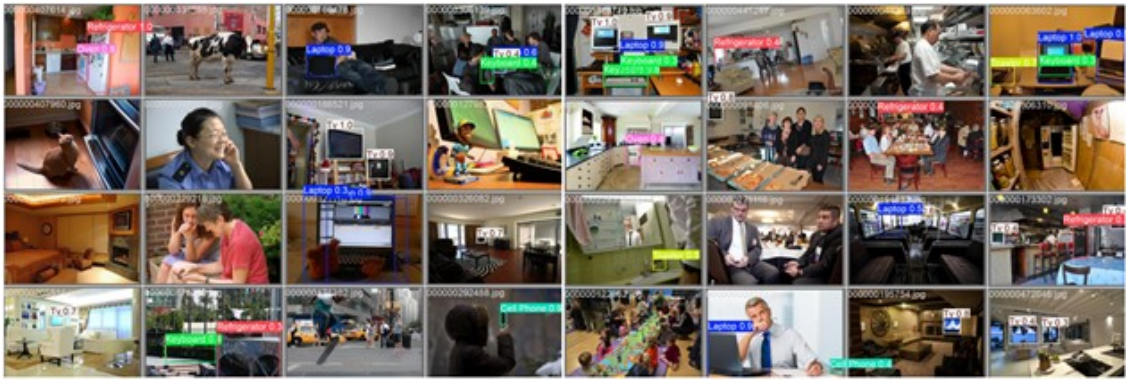

Figure 4.7: Validation of the YOLOv8n model with the custom dataset using images from the Open Images V7 dataset, trained with 100 epochs.

To validate Detectron2's pre-trained model, the same test figure was used for training with the YOLOv8 model. Figure 4.9 shows a significant improvement in object detection accuracy. A notable example is the 'Laptop' object, which, when trained with the YOLOv8 model, achieved an accuracy of 0.74. Using the pre-trained Detectron2 model, this accuracy increased to 0.99, demonstrating substantially better performance.

It can also be seen that the pre-trained model has limitations, as it failed to detect other laptops in the image, resulting in unidentified objects. This shows that, even with a pre-trained model, the dataset used is still insufficient to detect hazardous waste on the streets. These results reinforce the hypothesis that the main problem identified in this research lies in the limitations of existing datasets, which lack more comprehensive scenarios and more diverse images, compromising the effective training of the model.

Figure 4.10 also shows the confusion that the model causes with some very large images. The detection confused what a TV and a keyboard are, and the same confusion was found with the models trained with YOLOv8.

4.4.2 Custom Model

For testing purposes, the Detectron2 model was trained with a custom dataset with images collected from the COCO dataset. The model was trained in 6000 iterations using the Google Colab platform with a T4 GPU. No cross-validation techniques were applied, as was the case for YOLOv8 training, for the same reason mentioned in section 5.3.2 to optimize the training and evaluation of the model, prioritizing adaptation to the specific problem, instead of seeking a broad generalization that would not be applicable or relevant to the objective of the study, simple validation was considered sufficient to ensure the quality of the model training.

The customized dataset contains only laptop images to test detection for this object. The mAP was 0.82.

In Figure 4.11, we also used the same images from other tests to assess the accuracy of the trained model. It was also applied to detect hazardous waste in the Funchal images, but without

Figure 4.8: Test to validate accuracy and detection of the Laptop object with the YOLOv8n model trained with the new custom dataset using images from the COCO dataset.

success, no images of hazardous waste objects in the rubbish bins were returned.

Figure 4.9: Test to validate accuracy and detection of the Laptop, keyboard and mobile phone object with a pre-trained Detectron2 model that uses the COCO dataset.

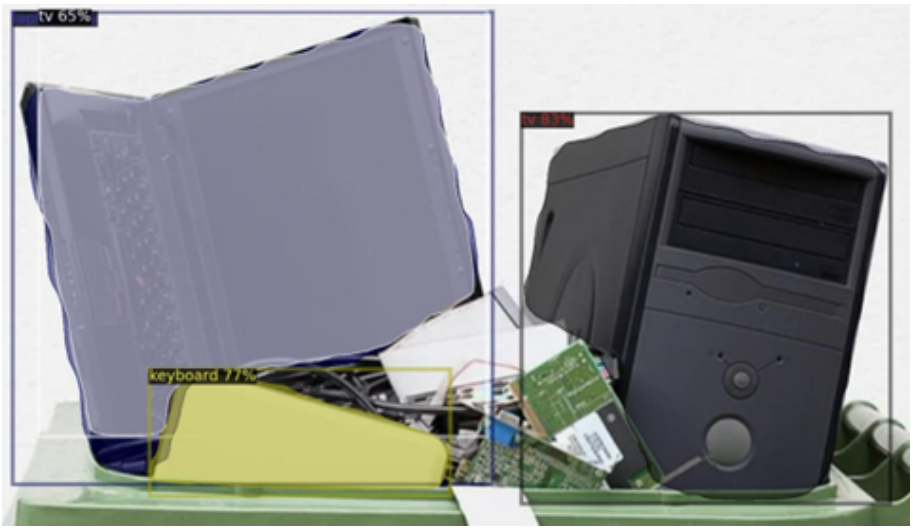

Figure 4.10: Test to validate accuracy and detection of the Laptop and mobile computer tower object with a pre-trained Detectron2 model that uses the COCO dataset.

Figure 4.11: Testing to validate accuracy and detection of Laptop object with trained YOLOv8n model.

Chapter 5

Results and Discussion

This session evaluates and discusses the results obtained from training with YOLOv8 and the images collected from the Open Images V7 dataset. Graphs, model learning curves, results achieved, evaluation of the trained models, failures and successes, and why the model didn't achieve a higher result were analyzed.

5.1 YOLOv8 and Open Images V7 Dataset

Once the model was trained, many performance metrics were observed to check how effectively the method performed. The graphs created are quite explanatory regarding how the model has been doing and where its limitations have fallen.

The loss graphs (box loss, classification loss, and DFL loss) figure 5.1 show how much the model was learning over time during training and validation.

Box Loss: Because only box loss decreases in both train and Val, the model is learning to predict bounding boxes of objects from images. The reduction loss proves that the model better predicts where dangerous waste was dumped.

Decrease in Classification Loss: We can see that over time, loss for classification has also decreased, which means our model can associate classes with detected images correctly. The classes with fewer instances, like hair dryers, did not perform as well; we will analyze more on that in the Precision and Recall graphs.

DFL Loss: The distribution focal loss (DFL) decreases, too, over several epochs of training, which makes it seem like the model is better at predicting the shapes and sizes of objects in images.

In this bar chart figure 5.2, it is easy to notice the imbalance in our dataset as we have way more instances for the laptop class compared to other classes like hair dryers and heaters. Such an imbalance affects generalization across all classes of the model. They even had poor performance in most cases as hair dryers, for example, as shown in recall and precision graphs. There are techniques like data augmentation that could help, or we might collect more images from the underrepresented classes.

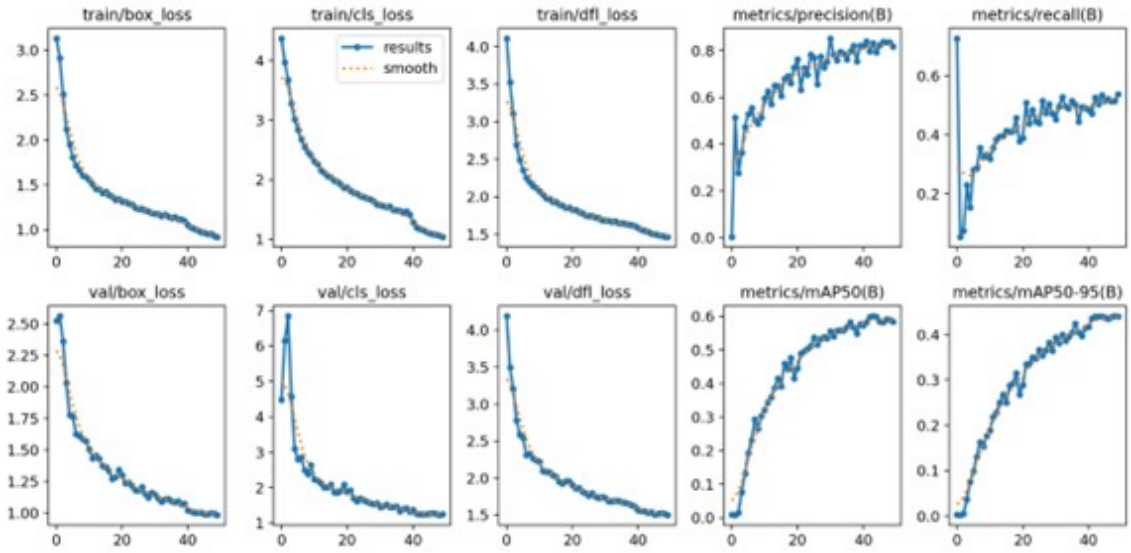

Figure 5.1: Loss graph to evaluate the performance of the model trained with the custom dataset from images collected from the Open Images V7 dataset.

We can see from the graph of overlapping bounding boxes that many datasets tend to center objects in their images. This positional bias could be an artifact of the way objects are positioned in their original dataset Open Images V7 most of them seem to be located around the image center. This factor could have affected the realization of its performance for images that deviate from this kind of pattern, as in the more irregular disposition also found when we are talking about outdoor shots with illegal garbage dumps.

In the graph figure 5.3, we have recall and model confidence. Recall is a measurement of how complete the results are whether relevant objects have been returned to the list of detections or not.

When classes are something like laptops and tablets, memory remains high with different levels of confidence, and laptops can be detected quite easily.

The hair dryer class, on the other hand, has a low recall at all confidence levels, showing that this class is underrepresented in the dataset. Namely, the model simply decides not to include hairdryers in many images.

We can see that although the model's performance is good in some classes, it does not generalize well to all of them (due to having fewer instances).

The F1 score is the harmonic mean of precision and recall. The curve illustrates in figure 5.4 that the laptop and microwave classes have top F1 scores at modest assurance amounts, indicating an excellent compromise between precision and recall for these kinds.

The hair dryer, as before, has a low F1 score, again showing that the model is still struggling to correctly identify this class. The tablet class also boasts a comparatively high F1-score, however one that is lower than the laptop and microwave.

These are results that suggest there is still a significant problem with class imbalance and reinforce the benefit of crafting better balanced datasets to improve detection performance on unseen classes.

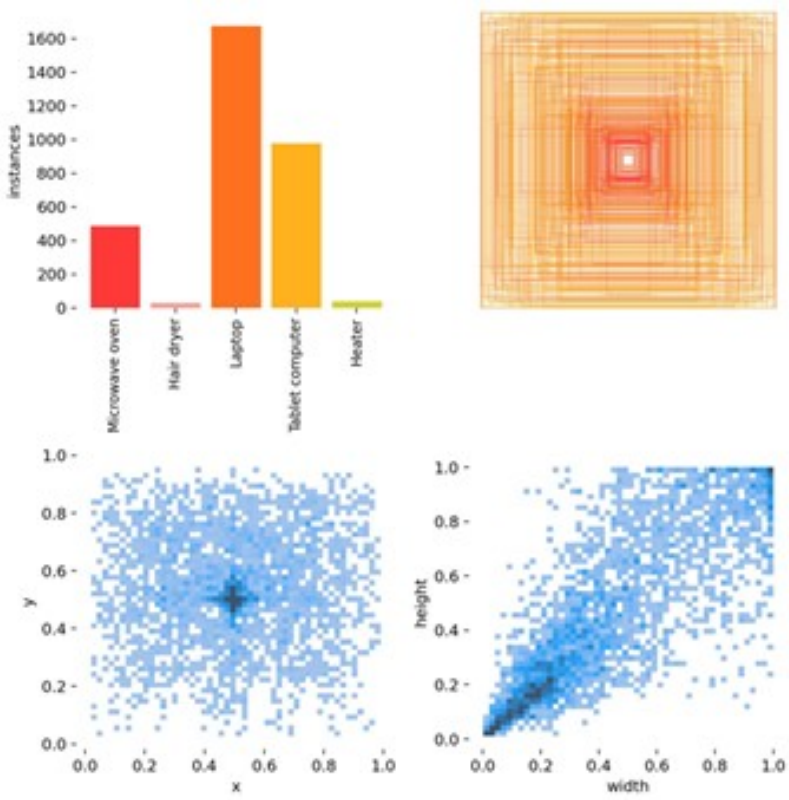

Figure 5.2: Explanatory graphic to illustrate the positioning of images and the unbalanced number between the classes used in the model.

Accuracy is the ratio of correct predictions to the total number of predictions. This means that when a model is more confident about predictions, its accuracy will be at its peak.

The high-accuracy classes for the laptop and tablet have nice sharp peaks figure 5.5. They are accurate at higher confidence levels. The hair dryer class has a very low level of accuracy in other words, the model often makes incorrect predictions with this sort, and so it is particularly difficult to predict from small amounts.

This graph figure 5.6 captures the precision and recall scores. This is especially true in imbalanced datasets like the one we created for this experiment. The precision curve of the laptop is moderately flat, which shows that this class gets input from a good-performing model. The hair dryer shows hardly any curve at all, again enforcing how poorly the model handles with this class.

We obtained a mAP value of 0.599 for all classes, which indicates quite low impact as well. The main failure is the bad generalization, especially in the classes with a small number of instances, and the lack of diversity for images (being indoors). We can notice the lack of balance in the confusion matrix figure 5.7.

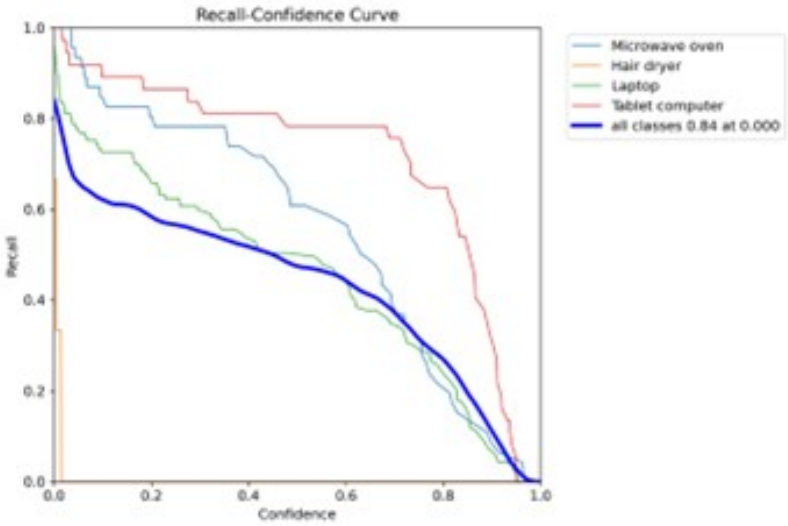

Figure 5.3: Recall and confidence curve of the model trained with images from the custom dataset with images collected from the Open Images V7 dataset.

5.2 YOLOv8 and COCO Dataset

A confusion matrix is a comparison between the true labels of objects and the predicted ones to understand how well our model works visually. In this case, a normalized confusion matrix was used which makes sure that each cell in the same line has equal sum.

It is seen in figure 5.9 that the model performs well in categories such as TV, Laptop, and Microwave, where 0.5 or greater of predicted labels almost match true label values, which indicates strong predictions for subjects within these bags. For example, laptops were 0.54 correctly predicted, and microwaves in 0.53.

On the other hand, objects like toasters and hair dryers (which had much fewer training samples than chairs) were more problematic for this model. These categories had a low count and caused more wrong classifications.

For instance, confusion was observed in keyboard and background objects with hair dryers and toasters, respectively, hinting that the model struggled to decide between similar-looking or little localizing items.

The other noteworthy point is the graph that shows in figure 6.9 the majority of F1-scores in most categories peak around a confidence level between 0.3 and 0,6. TV, laptop, and microwave had the highest F1 scores among object categories, which means that it's easier to predict these objects with this model.

As mentioned previously, the toaster and hair dryer categories have low F1 scores because of the true difficulty with their smaller amount of training examples.

The model slightly underperformed for the cell phone category but received fair scores, presenting higher variance in its F1-score, perhaps due to differences regarding how portable telephones are displayed among images as well.

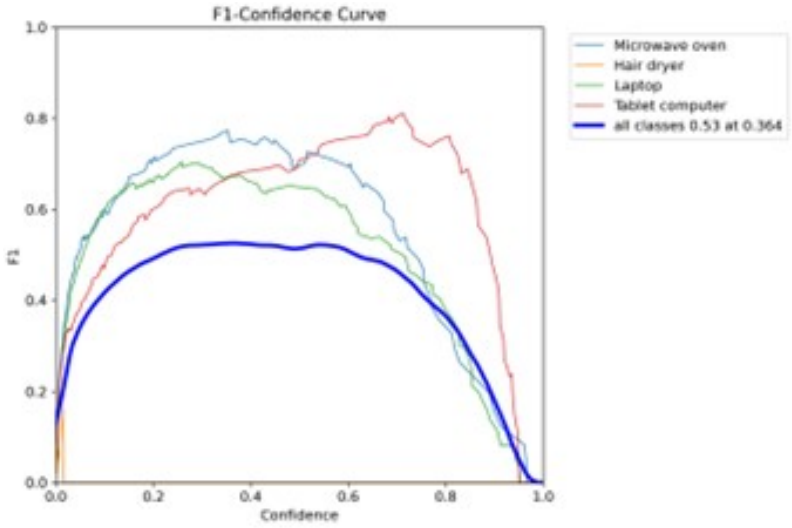

Figure 5.4: Graph referring to the harmonic mean of precision of the model trained with images from the customized dataset with images collected from the Open Images V7 dataset.

The precision rises for objects, laptops, and TVs like we can see in figure 5.10 which is also consistent with the model mostly predicting positive when only making highly confident predictions.

Categories with fewer training examples, such as toaster and hair dryer, are more variable, with lower precision at all confidence levels. This implies that the model makes wrong predictions on these objects which probably doesn't have enough number of images available for training.

Both TV and laptop again have decent precision-recall curves figure 5.11, which result from a trade-off between high recall (find objects correctly) at the expense of ranking faults less important, giving many false positives while predicting.

Toasters, on the other hand, and hair dryers are reflected in very sub optimal precision-recall curves as signs of wrong classifications missed classification, or incorrect bounding boxes that often do not contain these objects.

The mAP of all classes was 0.405, a medium result that also highlights the model's good performance in finding usual categories and its common issues with rare items.

Although no surprise, TVs, laptops, and microwaves have a high recall at the lower confidence thresholds figure 5.13. While it might not come as much of a shocker, most stuff like TVs or Laptops is easily detected even if the minimum confidence level is set to a low limit.

For toasters and hair dryers, even at high confidence levels, the recall is very low, this indicates that our model misses the detection of these objects a lot.

Over the training of our model, we observe a steadily decreasing trend for both box loss and classification loss which means that our model is learning better predictions figure 6.12. Since the loss continues decreasing on training and validation datasets, can argue that this model generalizes well (there are however still some limitations to other classes).

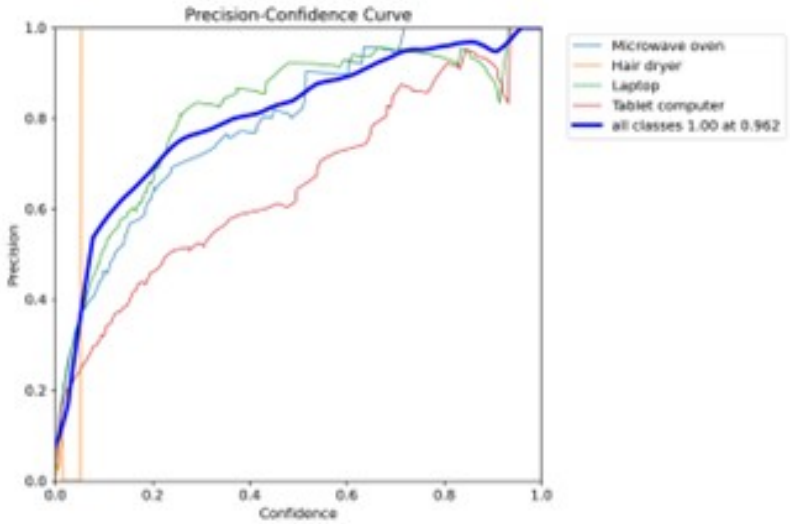

Figure 5.5: Graph referring to the proportion of correct accuracies of the model trained with images from the customized dataset with images collected from the Open Images V7 dataset.

The decline in loss overall indicates that the model is predicting where objects are as well as what they are more effectively over time.

Using the COCO dataset at 640 pixels with YOLOv8n produced better scores for common categories, which showed it could adequately generalize different environments and objects. But the difficulties with underrepresented classes suggest that a more balanced dataset is needed, or additional data augmentation techniques.

Additional improvements might be to better balance the classes with more data, improve the diversity of collected data, and try larger image resolutions or more complex augmentation techniques for all object categories.

It is also possible to notice the improvement in the same images compared to another Open Images dataset in test image waste.

5.3 Detectron2 and COCO Dataset

The blue curve shows a model's training losses per iteration (or epochs). This loss will be high at the start of training, meaning that our model is making many mistakes in its predictions. As the number of iterations continued to increase, the training loss gradually decreased. This tells us that the model has learned to fit the training data and has done well with correct predictions in examples.

The orange curve represents the model's loss on a separate validation dataset used to assess the model's performance on previously unseen data. Thousands of iterations, the validation loss peaks and falls, as illustrated below, throughout indicating that the model is mostly improving its ability to generalize on unknown data. However, after a certain point (around 2,000 iterations), the validation loss stops decreasing and stabilizes, characterizing a possible overfitting of the model.

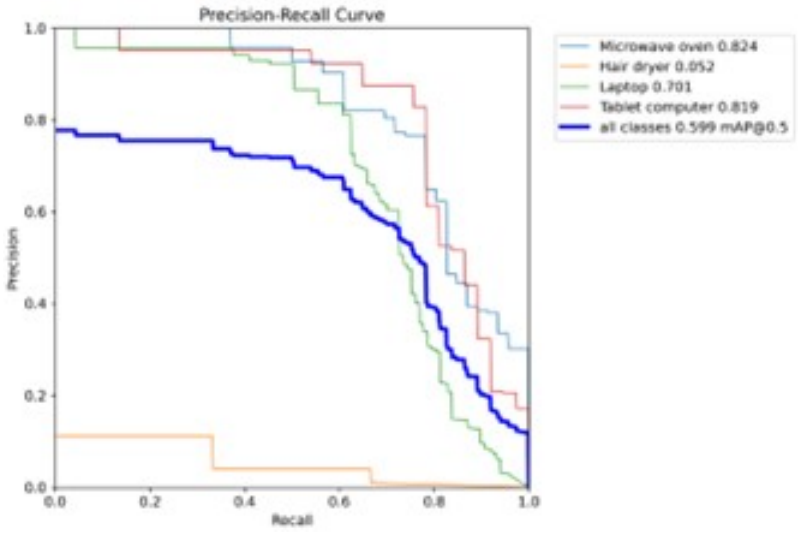

Figure 5.6: Graph referring to precision and recall of the model trained with images from the customized dataset with images collected from the Open Images V7 dataset.

5.4 Discussion

The results in training YOLOv8 with Open Images and COCO based created dataset also shows that there are still some problems to fix.

The main factors when it came to the model failing were:

Class imbalance: The model was not able to generalize well over all the classes because it had too many examples for some, like laptops, and very few examples from other classifications, such as hairdryer.

Nature of images: Several scenes with most likely indoor locations could not capture an outdoor scenario where hazardous material may be disposed of, leaving the model incapable of detecting the possibility.

Few instances for critical classes: The low item numbers in some of the delicate items, such as the hair dryer and heater, made our model very terrible in these cases with an unequal recall together precision, which indicates we need more data towards these categories.

These difficulties also suggest that the dataset should be improved by using data augmentation or collecting more images in outdoor conditions because hazardous waste can usually be found outside.

This experiment was significant for the detection of hazardous waste, because it helped to understand the importance of forming a more specific data set for the problem, requiring images collected from waste in urban spaces and also the need to better work the images for the proposed scenario.

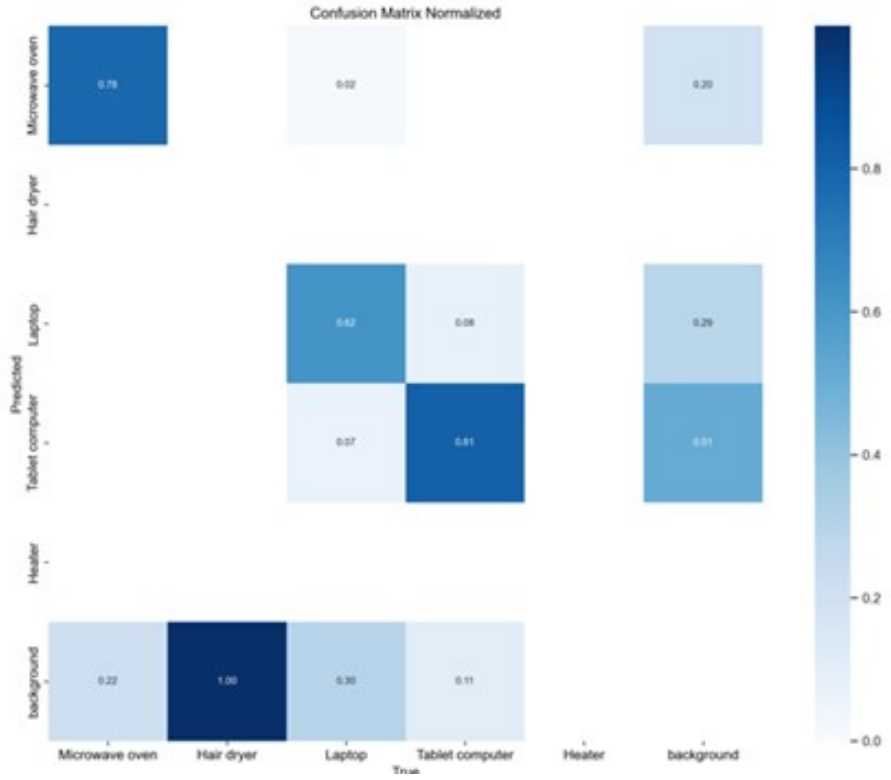

Figure 5.7: Graph referring to precision and recall of the model trained with images from the customized dataset with images collected from the Open Images V7 dataset.

Figure 5.8: Graph referring to the normalized confusion matrix of the model trained with images from the customized dataset with images collected from the COCO dataset.

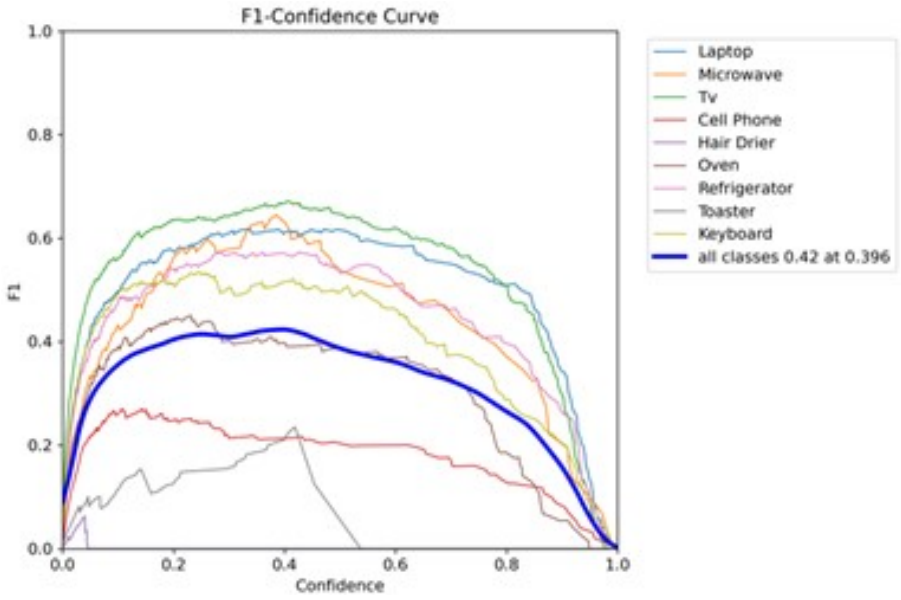

Figure 5.9: Graph referring to the harmonic mean of precision of the model trained with images from the customized dataset with images collected from the COCO dataset.

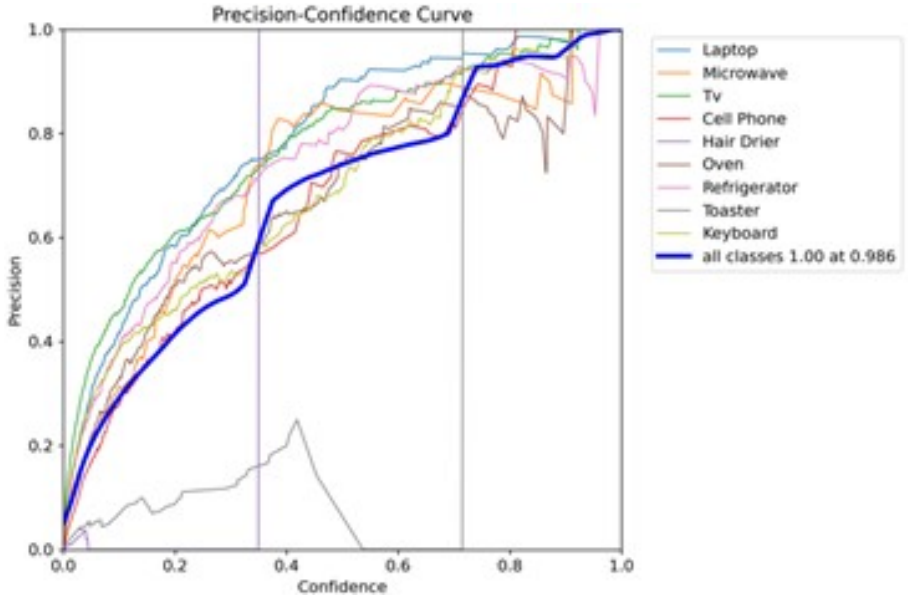

Figure 5.10: Graph referring to the harmonic mean of precision of the model trained with images from the customized dataset with images collected from the COCO dataset.

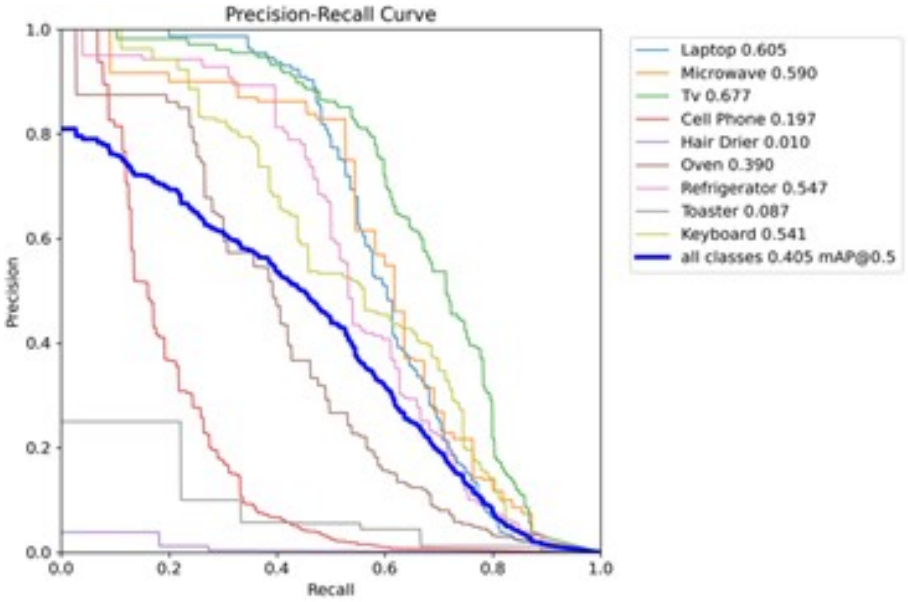

Figure 5.11: Graph referring to precision and recall of the model trained with images from the customized dataset with images collected from the COCO dataset.

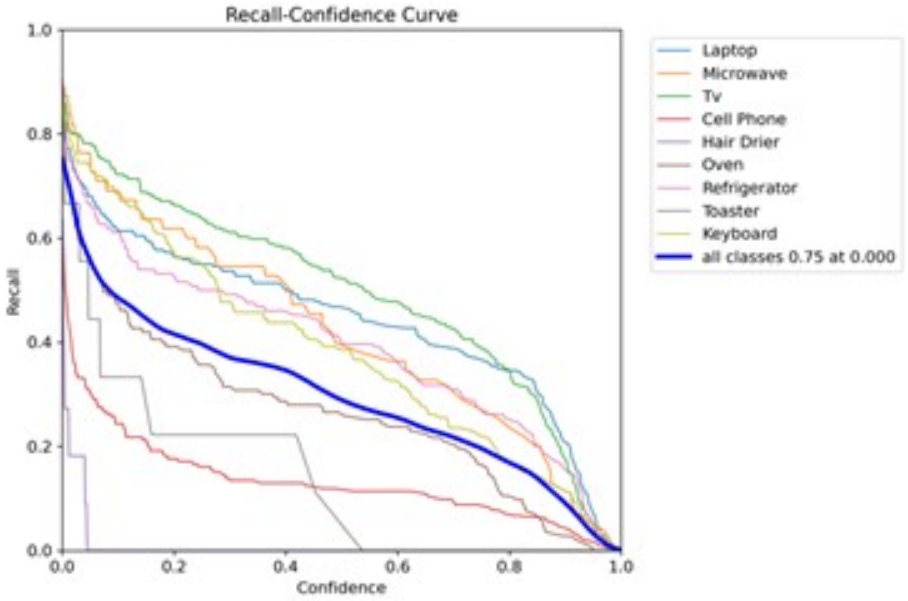

Figure 5.12: Recall and confidence curve of the model trained with images from the customized dataset with images collected from the COCO dataset.

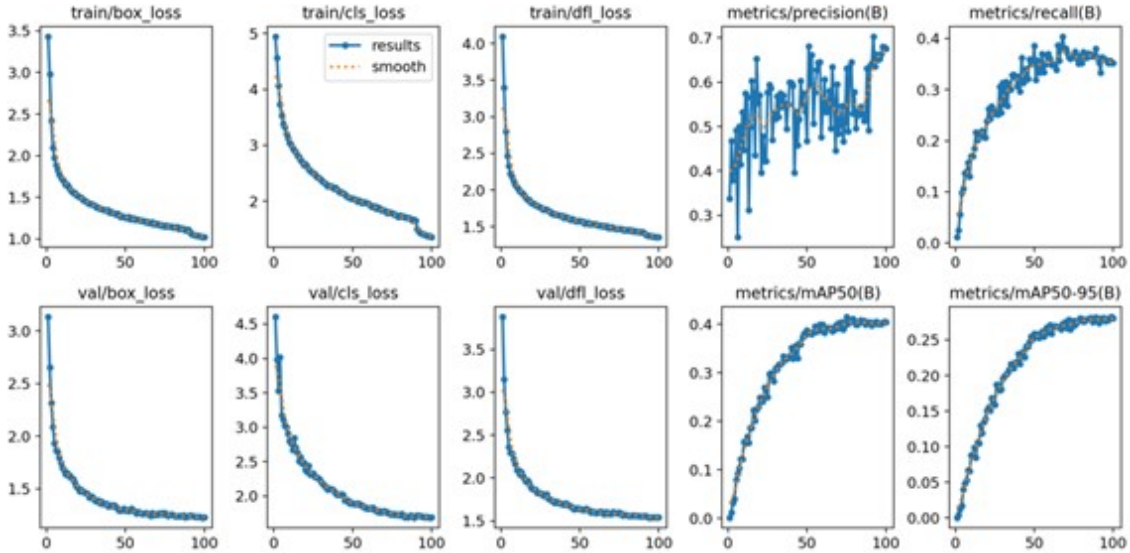

Figure 5.13: Loss graph to evaluate the performance of the model trained with the custom dataset from images collected from the Open Images V7 dataset.

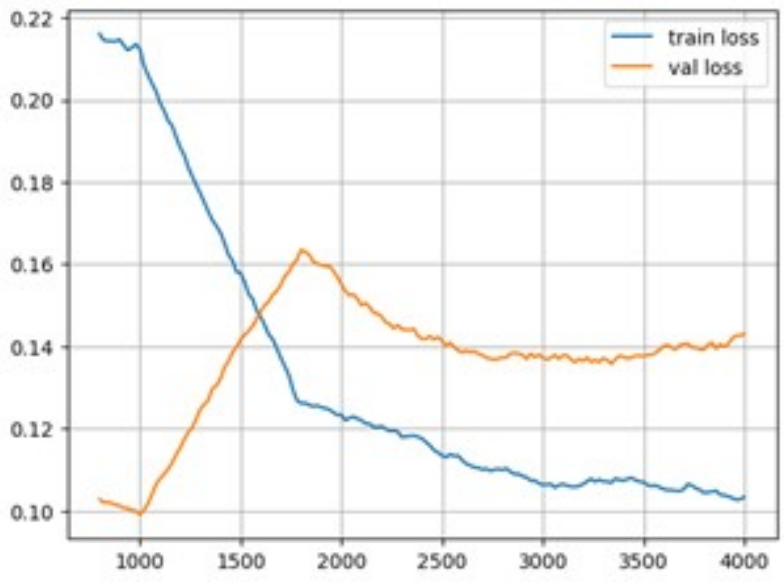

Figure 5.14: Loss graph to evaluate the performance of the model trained with the custom dataset from images collected from the Open Images V7 dataset.

Chapter 6

Conclusions and Future Work

6.1 Satisfaction of Objectives

Over the course of this project, we have learned so much by researching, writing, and practicing these computer vision models. Though we could not get the results as expected in terms of performance metrics like mAP(Mean Average Precision) it was a very enriching experience. This exploration to get better results became a challenge in itself and allowed me to improve my knowledge of object detection along with the various configurations/factors that affect the performance of models doing deep learning of this kind.

It was really rewarding to be able to get our hands on the latest and greatest of models, such as YOLOv8 and Mask R-CNN, through Detectron2. That was no easy, but rewarding labour: data management, dataset making and model hyperparameter finetuning. It turns out the lesson of learning to work with well-structured datasets like COCO provided was a valuable one when it comes to data quality and diversity. The most important learning from this process is that in machine learning, a balanced dataset can make or break your model.

It was to not just get the perfect metrics but how every nook and corner of pipeline can affect machine learning. I learned about data preprocessing, inference techniques and post-processing necessary to build a reliable computer vision model. Also, on one side was the experience-building, like data augmentation and hyperparameter tuning, which is skillful when it learns to fit a better, careful model.

The satisfaction was in learning how and what to measure, and even while the numbers weren't great at first (nor are they today), we made progress throughout development. It was also absolutely essential to have the ability to identify where we could further grow, as these inputs will fuel those improvements in future works.

An important goal of this work was to test the ability for object detection and instance segmentation models to learn from a basic dataset, and generalize. The mAP was not as high or lower than expected (due to the simple dataset used), and overall, despite this, it showed amazing training capabilities. The problem here wasn't the model's inability to learn but rather the constraints set by the dataset and the lack of some important techniques for better performance.

A very big bottleneck was that there were no external images, so the model could not learn more. The majority of the training images were taken indoors, which limited the model to generalize across different settings. For other real use cases where object detection/waste identification occurs in an outdoor environment, lighting conditions are different, as well as backgrounds and objects' positioning. This leads to the model being less capable of more diverse situations and, therefore, is also reflected in validation results.

6.2 Future Work

In future work, I will also focus on the enrichment and augmentation of the dataset used, along with a more efficient training procedure for our model. One of the greatest difficulties encountered in our project was that, due to few images present in the dataset, it had a worse performance than expected, considering feature extraction and generalization with different contexts. The next step is to employ data augmentation and seek a variety of images (preferably outdoor) in order to beat these challenges.

Data augmentation can be very useful to boost datasets (in terms of size and variety in the images), which is an essential step. In this technique, new images are generated from the present dataset through transformations like rotation, flipping, as well brightness and contrast adjustments. These transformations help us train our model so that it can deal with different lighting, viewing angles, and some other variations that are present in the real world. This will allow the model to learn with more stability and adapt better when presented with new images, especially for outdoor-type ones that were less used in this work.

Besides, one of the main focuses for any further development is to pull in outdoor environments like roads and parks or urban places new images. This is important because much of what the model will need to detect in real-world usage (outdoors) and outdoor lighting/conditions are very different from indoors. To do this, I will use real images taken directly on the street with mobile cameras and drones or from public databases that contain photographs of outdoor sites. It will also be necessary to combine these images with other data obtained from the internet since this is how we can get a wider variety of scene variations.

References

IoT-enabled Convolutional Neural Networks: Techniques and Applications. https://www.riverpublishers.com/book_details.php?book_id=1012, 2024. Accessed: 2024-08-13.

A. Abdusalomov, Bappy MD Siful Islam, Rashid Nasimov, Mukhriddin Mukhiddinov, and T. Whangbo. An improved forest fire detection method based on the detectron2 model and a deep learning approach. *Sensors*, 23(3):1512, 2023. doi: 10.3390/s23031512. URL https://doi.org/10.3390/s23031512.

AgaMiko. Waste datasets review, 2023. URL https://github.com/AgaMiko/waste-datasets-review. Accessed: 2024-08-13.

Apambiente. Study on waste management in portugal. *Portuguese Environmental Agency*, 2022. URL https://www.apambiente.pt/_zdata/strategies/waste-management-study.pdf.

Amali Çipi. Integrated urban waste management (case study, vlora municipality). *LIMEN Conference*, 2023. doi: 10.31410/limen.2023.381. URL https://dx.doi.org/10.31410/limen.2023.381.

COCO Dataset. Common objects in context (coco), 2024a. URL https://cocodataset.org/#home. Accessed: 2024-08-06.

Open Images Dataset. Open images dataset: Facts and figures, 2024b. URL https://storage.googleapis.com/openimages/web/factsfigures_v7.html. Accessed: 2024-08-06.

TACO Dataset. Taco dataset statistics, 2024c. URL http://tacodataset.org/stats. Accessed: 2024-08-06.

R. Girshick. Fast r-cnn. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2015. doi: 10.1109/ICCV.2015.169.

Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, New York, NY, USA, 2nd edition, 2009. ISBN 0387848576, 978-0387848570.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. doi: 10.1109/CVPR.2016.90.

Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017a. doi: 10.1109/ICCV.2017.322.

Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross B. Girshick. Mask R-CNN. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 2961–2969, 2017b. doi: 10.1109/ICCV.2017.322.

INESC TEC. Inesc tec. Disponível em: https://www.inesctec.pt/pt/instituicao#structure. Acesso em: 07 de janeiro 2023, 2022.

Ron Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. *International Joint Conference on Artificial Intelligence (IJCAI)*, 14(2):1137–1145, 1995. URL https://dl.acm.org/doi/10.5555/1643031.1643047.

Marek Kraft, Mateusz Piechocki, Bartosz Ptak, and Krzysztof Walas. Autonomous, onboard vision-based trash and litter detection in low altitude aerial images collected by an unmanned aerial vehicle. *Remote Sensing*, 13(5), 2021. ISSN 2072-4292. doi: 10.3390/rs13050965. URL https://www.mdpi.com/2072-4292/13/5/965.

Nikhil Venkat Kumsetty, Amith Bhat Nekkare, Sowmya Kamath S., and Anand Kumar M. Trashbox: Trash detection and classification using quantum transfer learning. In *2022 31st Conference of Open Innovations Association (FRUCT)*, pages 125–130, 2022. doi: 10.23919/FRUCT54823.2022.9770922.

Pan Li, Jiayin Xu, and Shenbo Liu. Solid waste detection using enhanced yolov8 lightweight convolutional neural networks. *Mathematics*, 12(2185):1–26, 2024. doi: 10.3390/math12142185.

Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross B. Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: Common objects in context. In *European Conference on Computer Vision (ECCV)*, pages 740–755, 2014. doi: 10.1007/978-3-319-10602-1_48.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 2980–2988, 2017. doi: 10.1109/ICCV.2017.324.

Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C. Berg. SSD: Single shot multibox detector. In *European Conference on Computer Vision (ECCV)*, pages 21–37, 2016. doi: 10.1007/978-3-319-46448-0_2.

Menelaos Neofotistos et al. A real-world scenario of citizens' motivation and engagement in urban waste management through a mobile application and smart city technology. *SpringerLink*, 2022. doi: 10.1007/s43615-022-00155-z. URL https://dx.doi.org/10.1007/s43615-022-00155-z.

University of Minnesota. E-waste recycling systems, 2018. URL https://conservancy.umn.edu/items/6dd6a960-c44a-4510-a679-efb8c82ebfb7. Accessed: 2024-08-06.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy,

Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 8024–8035, 2019.

K. Rajesh et al. Hazardous waste and its impact on soil and water. *Environmental Research Letters*, 2022. URL https://doi.org/10.1088/1748-9326/ac3501.

J. Redmon and A. Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018. URL https://arxiv.org/abs/1804.02767.

Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016a. doi: 10.1109/CVPR.2016.91.

Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 779–788, 2016b. doi: 10.1109/CVPR.2016.91.

Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 91–99, 2015. doi: 10.5555/2969239.2969250.

Pronoy Kanti Roy, Md. Riadul Islam, and Md.Jubayar Alam Rafi. Non-biodegradable plastic waste detection and classification using deep learning: Bangladeshi environmental scenario. In *2024 International Conference on Electrical Engineering and Information Communication Technology (ICEEICT)*. IEEE, 2024. doi: 10.1109/ICEEICT62016.2024.10534312.

Mojtaba Sarabchian and Max Mühlhäuser. Improving wearable-based activity recognition using image representations. *Sensors*, 22(5):1840, 2022. doi: 10.3390/s22051840.

S. Swathi, M. Rajalakshmi, and Vijayalakshmi Senniappan. Deep learning: A detailed analysis of various image augmentation techniques. In *Proceedings of the 2023 International Conference on Networks, Wireless and Communications (ICNWC)*, 2023. doi: 10.1109/ICNWC57852.2023.10127343.

Nhat-Dang Thung and Matthias Yang. Classification of trash for recyclability status, 2016. URL https://cs229.stanford.edu/proj2016/report/ThungYang-ClassificationOfTrashForRecyclabilityStatus-report.pdf. Accessed: 2024-08-06.

Transform Labs. Computer Vision: Beyond Image Classification. https://awh.net/blog/computer-vision-beyond-image-classification, 2024. Accessed: 2024-08-13.

Ultralytics. YOLOv5: A scalable object detection framework, 2020. URL https://github.com/ultralytics/yolov5. Accessed: 2024-08-13.

Ultralytics. YOLOv8: Next-generation object detection, 2023. URL https://github.com/ultralytics/ultralytics. Accessed: 2024-08-13.

Chien-Yao Wang, Alexey Bochkovskiy, and Hong-Yuan Mark Liao. YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. *arXiv preprint arXiv:2207.02696*, 2022. URL https://arxiv.org/abs/2207.02696. Accessed: 2024-08-13.

Z. Q. Zhao, P. Zheng, S. T. Xu, and X. Wu. Object detection with deep learning: A review. *IEEE Transactions on Neural Networks and Learning Systems*, 30(11):3212–3232, 2019. doi: 10.1109/TNNLS.2018.2876865.