EasyChair Preprint

№ 14850

A Two-Stage YOLOv8 Approach for Waste Detection and Classification in Cognitive Cities

Ahmad Nayfeh, Sadam Al-Azani and Hussein Samma

September 24, 2024

Available online at www.sciencedirect.com

ScienceDirect

Transportation Research Procedia 00 (2024) 000–000

Transportation Research Procedia

www.elsevier.com/locate/procedia

The 1st International Conference on Smart Mobility and Logistics Ecosystems (SMiLE)
September 17-19, 2024, KFUPM, Saudi Arabia

A Two-Stage YOLOv8 Approach for Waste Detection and Classification in Cognitive Cities

Ahmad Nayfeh[a,b], Sadam Al-Azani[a,*], Hussein Samma[a]

[a] *SDAIA-KFUPM Joint Research Center for Artificial Intelligence, King Fahd University of Petroleum & Minerals, Dhahran 31261, Saudi Arabia*
[b] *Department of Electrical Engineering, King Fahd University of Petroleum & Minerals, Dhahran 31261, Saudi Arabia*

Abstract

Waste, as a primary cause of visual pollution, not only impacts public health but also has significant economic implications, particularly in tourism. Visual pollution from waste or trash encompasses various types that require classification. Cognitive cities are beginning to develop automatic systems to classify these types, but the task is challenging due to the similarity among different types of waste and the common features of most elements. To address this issue, we propose an innovative two-stage methodology using YOLOv8 for object detection. This advanced approach is designed to detect and classify 16 different types of waste objects. The proposed approach is compared to the traditional YOLOv8 to evaluate its performance. The experimental results highlight the effectiveness of the modified YOLOv8 approach, which integrates YOLOv8 for detection and the Swin Transformer for classification. Notably, when applied to larger image sizes, this enhanced method achieved a significant improvement in the F1-score, underscoring the viability and robustness of the proposed framework[1].

Keywords: Deep Learning; Computer Vision; YOLOv8; Waste Detection and classification; Visual Pollution; Cognitive Cities

1. Introduction

The rapid and often unchecked pursuit of development comes at a cost to our environment. While progress is commendable, certain forms of development, particularly unplanned and unregulated growth, lead to significant consequences, including pollution. As highlighted in (Ukaogo et al., 2020), human activities related to urbanization, industrialization, mining, and exploration are at the forefront of global environmental pollution. Understanding the magnitude of the danger posed by pollution, (Alharbi and Rangel-Buitrago, 2022) emphasizes that pollution's impact on health surpasses that of war, terrorism, diseases like malaria, HIV, tuberculosis, and even substances like drugs

[1] This paper was awarded the Best Paper Award in the "Sustainable Cognitive Cities" track at the conference.

* Corresponding author.
E-mail address: sadam.azani@kfupm.edu.sa

and alcohol. Surprisingly, the number of deaths caused by pollution is comparable to those resulting from smoking. Pollution, in essence, encompasses harmful activities caused by human actions that negatively affect the environment. Recognizing waste accurately in cognitive cities is crucial for reducing visual pollution, protecting public health, and minimizing economic losses, particularly in the tourism sector. In light of these concerns, it becomes imperative to seek solutions that can mitigate this issue. The utilization of object detection and deep learning concepts offers a promising avenue for addressing visual pollution. By employing these technologies, we can identify instances of potential visual pollution and take appropriate actions to reduce and eventually eliminate its impact on the environment.

Machine Learning (ML) has brought transformative change to diverse sectors (Al-Azani et al., 2022) like energy, autonomous vehicles, cybersecurity, and environmental sciences. Unlike traditional programming, ML involves training the machine by providing input-output pairs, allowing it to learn correlations and construct algorithms. However, this process requires data to be preprocessed into a suitable format, often involving feature extraction. Deep Learning (DL), a subset of ML, takes this further. DL's representation learning methods enable machines to learn from raw data directly, bypassing the need for extensive preprocessing (LeCun et al., 2015).

Object detection, a specialized technique beyond image classification, serves to identify and categorize objects within images by enclosing them in bounding boxes. This technique has found significance in diverse fields such as autonomous vehicles, pedestrian detection, facial recognition, medical applications, video surveillance, and more. The progress of object detection has generally gone through two historical periods: the traditional object detection period (before 2014) and the deep learning-based detection period (after 2014) (Zou et al., 2023). The methods used in DL for object detection can be broadly divided into two categories based on the detection process and the network structure (Zou et al., 2023; Wang et al., 2021).

The first category is 2-stage object detection (Carranza-García et al., 2020; Chen et al., 2019) which consists of two stages starting with the first stage "region proposal" where the model tries to identify the locations of possible objects surrounding it by a box (Region of Interest ROI). Multiple methods can be used for region proposal including selective search, CPMC, multi-scale combinatorial grouping, and RPN (Chen et al., 2020). In the second stage, the model will try to refine the localization of the boxes and classify the proposals. Common 2-stage models (Chen et al., 2020) include R-CNN released in 2015 (Girshick et al., 2014) and other improved versions following it such as Fast R-CNN (Girshick, 2015).

Conversely, the second category, 1-stage object detection, a direct approach that swiftly classifies and localizes objects using candidate anchor boxes. Its first exposure to the world began with YOLO (You Only Looks Once) released in 2016 (Redmon et al., 2016). The name reveals the principle behind it which was to do both localization and classification using only one convolution neural network. After that, many improved versions of YOLO were introduced starting from the first version to YOLOv8 (Jocher et al., 2023) that was released in January 2023. Generally, two-stage detectors achieve higher accuracy compared to one-stage detectors, though this comes with a greater computational cost (Kraus and Dietmayer, 2019), (Carranza-García et al., 2020). However, this outcome is significantly influenced by the choice of convolutional backbone network and the hyperparameter configuration, which is a complex and nuanced process (Wang et al., 2021)

In this paper, we present a dual-stage approach utilizing the YOLOv8 detection model to detect and classify 16 types of waste. For the detection task, Yolov8 has been utilized, while for the classification task, different CNN-based architectures and a Swin transformer have been evaluated.

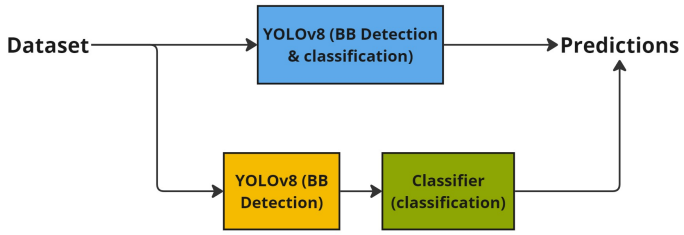

Fig. 1: High-level architecture of the proposed method

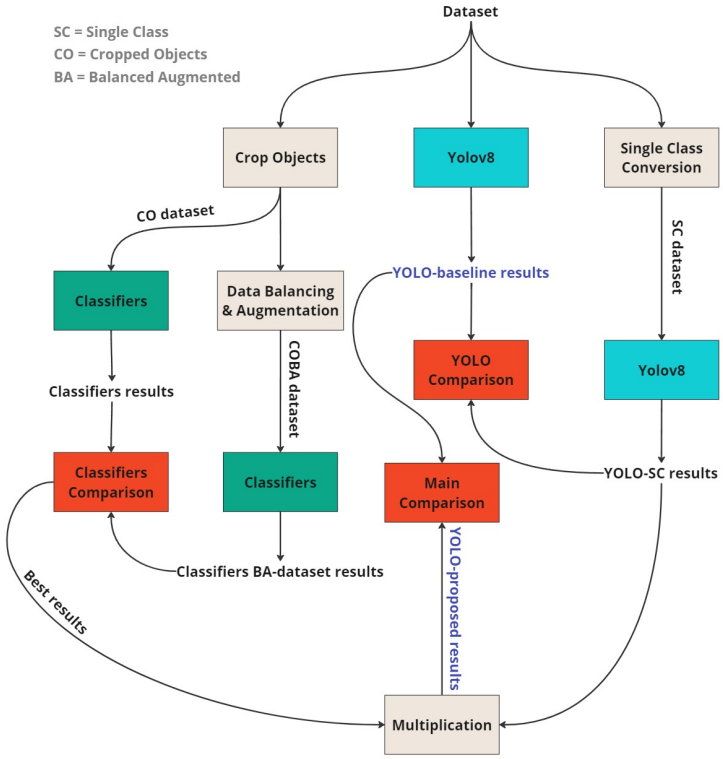

Fig. 2: Detailed flowchart illustrating the applied methodology

2. Methodology

We propose a method that involves using YOLOv8 for object detection, followed by a classifier to categorize the objects predicted by YOLOv8, as depicted in Figure 1.

In Figure 2, we present two distinct lines: the baseline and our new approach. This configuration facilitates a comparative analysis between the outcomes of our proposed method and those of the original YOLOv8 results. The baseline operates conventionally. We employ a dataset comprising training, validation, and test sets. We train YOLOv8 on the training and validation sets, utilizing the model to predict classes and bounding boxes for all objects across test set images.

Conversely, the new approach entails altering the dataset structure into a single-class dataset by modifying the yaml file with 'nc = 0'. We then adjust labels in the training and validation sets to have a class index of '0', while keeping the test set unchanged. Subsequently, we train another YOLOv8 on these modified training and validation sets, employing the model to predict bounding boxes for objects within the test set images. Returning to the original dataset, we crop objects from the train and validation set images, organizing them by class. This process generates a suitable dataset for training a classifier. Multiple classifiers are trained and tested, with the best one selected. Moving forward, we take the prediction results from the YOLOv8 trained on the single-class dataset and input them into the classifier, image by image and object by object. The classifier's predictions overwrite the YOLOv8's original predictions for the single-class dataset. The underlying assumption that motivated this approach is that training a robust classifier on objects from the train and validation set images, then using this classifier to predict object classes in test set images, could produce better outcomes compared to the original YOLO classification results.

3. Experiments

3.1. Environment Setup

The experiments were conducted using a NVIDIA GeForce GPU RTX3090 with CUDA Version 12.0. The environment utilized Linux as the operating system, Python Version 3.8.17, PyTorch Version 2.0.1+cu117, and Torchvision Version 0.15.2+cu117. The available memory for the experiments was 26Gi. These specifications provided the necessary computational resources for our study.

3.2. Dataset and preprocessing

The dataset utilized in this study originates from Roboflow which was developed by Technological Institute of the Philippines (Roboflow, 2023), specifically identified as "YoloV7 - Trash Dataset V3 - 04/01/2023 Computer Vision Project". It is composed of 16 classes, with a total of 10543 images. 9,740 of the images are used for training, 748 images for validation, and 55 images for testing. The original split ratio did not evenly distribute the data for training, validation, and testing. To ensure more reliable results, we adjusted the split to 70% for training, 10% for validation, and 20% for testing.

To prepare for classifier training, we converted the dataset into a suitable structure by cropping all objects from the images and organizing them into their respective class folders. The resulting dataset exhibited imbalances across some classes, as shown in Table 1. Figure 3 depicts examples of the considered waste objects. To address the imbalance issue, we established specific criteria: each class in the training folder would contain a maximum of 1,000 cropped objects, while the validation folder would include up to 100 cropped objects per class. When a class exceeded this limit, we randomly removed objects until the threshold was met. Conversely, if a class fell short, we duplicated images strategically to achieve the desired count, prioritizing those that had not been duplicated previously. This approach effectively employed oversampling for minority classes and undersampling for majority classes.

Following the object cropping and dataset balancing steps, we applied data augmentation techniques to enrich the dataset's diversity. These techniques randomly apply transformations—such as RandomHorizontalFlip, RandomVerticalFlip, RandomRotation (30 degrees), and GaussianBlur (kernel size 3)—to each image. By augmenting the dataset with varied transformations, we aimed to create a more resilient and adaptable dataset. This process enhances the model's capacity to generalize effectively across different scenarios, ultimately improving its performance in real-world applications.

Table 1: Dataset Statistics

Class	**Train Instances (objects)**	**Valid Instances (objects)**	**Test Instances (objects)**
Cans	1216	174	472
Cardboard	1865	300	556
Face Mask	1883	298	643
Glass Bottle	1022	165	308
Paper Bag	3472	499	1001
Paper Cup	834	106	234
Paperboard	498	123	181
Peel	288	58	98
Pile of Leaves	732	117	226
Plastic Bag	1436	171	469
Plastic Bottle	2877	389	680
Plastic Cup	861	128	298
Plastic Wrapper	4700	730	1174
Rags	329	48	86
Styrofoam	759	87	223
Tetra Pak	2196	331	671
Total	**24968**	**3724**	**7320**

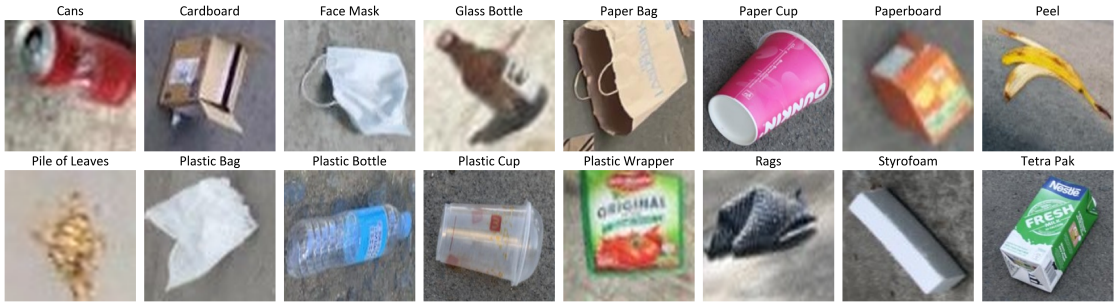

Fig. 3: Examples of the waste objects considered in this study

3.3. Selection of YOLOv8 Configuration

We conducted an analysis of the performance of the nano, small, and medium YOLOv8 versions before and after transforming the dataset into a single-class dataset. We set the batch size to 32, epochs to 100, optimizer to 'auto', and image size to 640. Table 2 provides a clear picture of our findings. Across all model versions, there is a notable improvement of approximately 10% in mAP50 and Recall, alongside an approximate 8% increase in Precision. Moreover, the mAP50-95 metric exhibits a growth of about 6% to 7%.

Interestingly, the medium model demonstrates a more pronounced improvement in terms of mAP50 convergence when utilizing the single-class dataset. This trend is distinct from the results observed in the small and nano versions. Based on these insights, we have chosen to focus on the medium model of YOLOv8 for our study.

To further investigate the effects of reducing the image size on the metrics improvements when transforming the dataset to a single class, we experimented by changing the image size to 160. As can bee seen in Table 3, there is an increase of mAP by approximately 3%, but the convergence is slower and the overall performance lags behind when using an image size of 640 by an approximation of 7%. Therefore, we decided to use the medium version with an image size of 640 in our experiment.

3.4. Classifier Selection

In this comprehensive experimentation, we conducted an in-depth evaluation of various classifiers, including `EffecientNet`, `ResNet50`, `ResNetx50`, and `Swin-transformer`, with selecting the hyperparameters systematically.

The dataset used for these experiments comprised cropped objects extracted from the dataset, mentioned previously. We conducted two distinctive scenarios to evaluate classifier performance utilizing the original cropped dataset and the augmented cropped dataset, presented in Table 4. In the first scenario, we utilized the original cropped image dataset, as comprehensively presented in the left part of Table 4. Subsequently, we delved into the second scenario, where we applied data balancing and augmentation techniques to the cropped image dataset. The results of this augmentation are presented in the right part of the same table.

In the context of classifying performance using the original cropped dataset, the EfficientNet model stands out with remarkable achievements. It recorded the highest training accuracy of 91.43% and secured the top position in validation accuracy at 89.19%, both accomplished when employing the SGD optimizer with a learning rate (LR) of 0.001. Upon analyzing the optimizers, a consistent pattern emerges: the SGD optimizer consistently outperforms the Adam optimizer. This trend is evident in the validation accuracy values. In summary, the combination of the SGD optimizer with LR and weight decay of 0.001 consistently yielded the best validation results. In this configuration, the EfficientNet model excelled with a high accuracy of 89.19%, followed by ResNet50 achieving 87.83%, ResNetx50 with 87.04%, and finally Swin transformer with 86.35%.

On the other hand, the augmented balanced dataset yielded the following highest validation accuracies: EfficientNet achieved 86%, ResNet50 demonstrated 85%, ResNetx50 neared 85%, and Swin transformer reached 82%.

Table 2: YOLOv8 Performance Metrics of YOLOv8 Using Image Size 640

Model Size	Dataset	Metric	Last Epoch Value	Best Value	Best Epoch
Nano	Original	Precision	0.81567	0.83152	92
		Recall	0.68631	0.72474	70
		mAP50	**0.76001**	**0.77547**	66
		mAP50-95	0.44599	0.45512	80
	Single-class	Precision	0.90882	0.90885	95
		Recall	0.80531	0.82176	77
		mAP50	**0.86639**	**0.87884**	77
		mAP50-95	0.51405	0.52534	78
Small	Original	Precision	0.82066	0.82141	98
		Recall	0.70704	0.75839	82
		mAP50	**0.76963**	**0.80379**	58
		mAP50-95	0.47889	0.50934	63
	Single-class	Precision	0.90417	0.90928	78
		Recall	0.82901	0.85028	54
		mAP50	**0.88447**	**0.89747**	77
		mAP50-95	0.54818	0.56013	65
Medium	Original	Precision	0.82062	0.83886	81
		Recall	0.76164	0.76861	60
		mAP50	**0.806**	**0.81669**	81
		mAP50-95	0.51611	0.51995	89
	Single-class	Precision	0.91832	0.91832	99
		Recall	0.83509	0.85962	52
		mAP50	**0.89461**	**0.90157**	52
		mAP50-95	0.58377	0.58639	89

Table 3: YOLOv8 Performance Metrics of YOLOv8 Using Image Size 160

Model Size	Dataset	Metric	Last Epoch Value	Best Value	Best Epoch
Medium	Original	Precision	0.8127	0.84869	78
		Recall	0.65216	0.65712	96
		mAP50	**0.70883**	**0.70961**	78
		mAP50-95	0.44322	0.44322	99
	Single-class	Precision	0.87967	0.89029	95
		Recall	0.75442	0.75442	99
		mAP50	**0.82003**	**0.82003**	99
		mAP50-95	0.51646	0.51646	99

By looking at these results, we found that models trained on the original, unbalanced, and non-augmented dataset outperformed those trained on the balanced and augmented dataset. This performance drop can be due to a couple of factors.

One possible factor is the data balancing, we reduced the number of objects to a maximum of 1,000 per class for training and 100 per class for validation. As a result, the sample sizes for validation before and after balancing differ significantly, which may be a major contributor to the variations observed in the results. In addition, reducing the training samples likely led to a loss of valuable information and reduced the variety of training samples. This reduction left the model with fewer diverse examples to learn from, impacting its ability to generalize. Another possible factor is the process of data augmentation, while it was intended to enhance dataset diversity, it introduced redundancy and noise. Duplication of images to balance classes resulted in the model overfitting to these repetitive samples rather than learning new patterns. Additionally, the augmented variations might not have accurately represented real-world scenarios, further degrading the model's performance.

Table 4: Classifiers Performance Using Original and Augmented Cropped Datasets

Model	Optimizer	Original Cropped Datase				Augmented Cropped Dataset			
		Train Acc. (%)	Val. Acc.(%)	Train Loss	Val. Loss	Train Acc. (%)	Val. Acc.(%)	Train Loss	Val. Loss
EffcientNet	ADAM	80.23	78.13	0.61	0.65	78.4	76.63	0.68	0.77
	SGD	91.43	89.19	0.26	0.36	89.48	86.13	0.32	0.5
ResNet50	ADAM	84.3	81.1	0.49	0.56	76.26	72.69	0.74	0.8
	SGD	92.03	87.83	0.25	0.41	90.39	85.25	0.3	0.5
ResNetx50	ADAM	77.1	75	0.7	0.77	77.38	76.5	0.7	0.75
	SGD	90.98	87.04	0.28	0.44	88.22	84.75	0.37	0.54
Swin	ADAM	70.8	66.02	0.91	1.05	70.61	63.19	0.93	1.13
	SGD	91.63	86.35	0.26	0.47	84.52	82.25	0.47	0.59

4. Results & Discussion

The proposed approach is evaluated using different evaluation measures including Mean Average Precision (mAP), Precision, Recall, and F1.

To assess the performance of the modified YOLOv8 approach, we start presenting the results of the original YOLOv8m baseline, trained on both image sizes 640 and 160, on the test set. Subsequently, we examine the outcomes of the modified YOLOv8m, also trained on both image sizes, using two distinct calculation methods. For the modified YOLOv8m, its performance is evaluated by multiplying the classifier F1-score with the Yolo single class F1-score. The performance of the trained classifiers on the original cropped dataset using SGD optimizer with LR and weight decay of 0.001 is presented in Table 5.

The consolidated outcomes for the modified YOLOv8m, trained on image sizes 640 and 160, are illustrated in Table 6. This table includes the results for both the modified YOLOv8m and the baseline original YOLOv8m on both image sizes 640 and 160. It provides a comprehensive overview of the performance metrics for the different scenarios.

Table 7 provides the final results when comparing the baseline approach with the new approach using the F1-score. It can be seen that the new approach is better when considering larger image sizes, also it can be seen that we achieved almost 2% more F1-score for the modified version. This can be justified due to that YOLO's simultaneous balancing of localization and classification tasks can lead to suboptimal performance, especially with closely packed or overlapping objects, unlike single-class-based classifiers that focus on one class at a time.

Table 5: Classifiers Performance on Test Set

Model	**Accuracy %**	**Precision %**	**Recall %**	**F1-score %**
EffecientNet	90.77	91.58	90.77	91.17
ResNet50	89.23	90.31	89.23	89.77
ResNetx50	89.23	91.40	89.23	90.30
Swin-transformer	91.54	94.07	91.54	92.79

Table 6: YOLOv8m Performance on test Set

Image Size	**Dataset**	**Precision%**	**Recall%**	**F1-score%**	**mAP50%**
640	Original	93.0	75.0	83.04	83.8
	Single Class	92.2	91.5	91.85	95.0
160	Original	88.9	74.5	81.07	78.9
	Single Class	89.0	74.7	81.23	85.2

Table 7: Model Performance Comparison Based on F1-score

Image Size	YOLO Baseline%	YOLO Single Class%	Classifier (Swin)	YOLO Modified (Multiplication)%
640	83.04	91.85	92.79	85.23

5. Conclusion

This study presented a novel approach to object detection using YOLOv8, by extending it into a two-stage framework through the incorporation of a classifier for refining its predictions. From the conducted analysis, it can be observed that the new approach demonstrates superior performance with larger datset. Specifically, it achieves nearly a 2% increase in F1-score compared to the baseline model, indicating a significant improvement in Precision and Recall metrics for object detection tasks. Future efforts may include adopting mean Average Precision (mAP) as a performance metric alongside the F1-score. Additionally, exploring more advanced classifiers or optimizing hyperparameters could further enhance classification performance and elevate the overall accuracy and robustness of our proposed approach. Furthermore, comparing our YOLO model with other 2-stage object detection models would allow for an assessment of computational efficiency and accuracy trade-offs, guiding the selection of the most suitable model for practical applications.

6. Acknowledgements

The authors would like to thank King Fahd University of Petroleum & Minerals (KFUPM), Saudi Arabia, for all its support. They would like to extend their sincere appreciation to SDAIA-KFUPM Joint Research Center for Artificial Intelligence (JRC-AI) and the Uxplore program at KFUPM for providing the resources and facilities that were crucial to the completion of this study.

References

Al-Azani, S., Sait, S.M., Al-Utaibi, K.A., 2022. A comprehensive literature review on children's databases for machine learning applications. IEEE Access 10, 12262–12285.

Alharbi, O.A., Rangel-Buitrago, N., 2022. Scenery evaluation as a tool for the determination of visual pollution in coastal environments: The rabigh coastline, kingdom of saudi arabia as a study case. Marine Pollution Bulletin 181, 113861.

Carranza-García, M., Torres-Mateo, J., Lara-Benítez, P., García-Gutiérrez, J., 2020. On the performance of one-stage and two-stage object detectors in autonomous vehicles using camera data. Remote Sensing 13, 89.

Chen, K., Li, J., Lin, W., See, J., Wang, J., Duan, L., Chen, Z., He, C., Zou, J., 2019. Towards accurate one-stage object detection with ap-loss, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 5119–5127.

Chen, K., Lin, W., Li, J., See, J., Wang, J., Zou, J., 2020. Ap-loss for accurate one-stage object detection. IEEE Transactions on Pattern Analysis and Machine Intelligence 43, 3782–3798.

Girshick, R., 2015. Fast r-cnn, in: Proceedings of the IEEE international conference on computer vision, pp. 1440–1448.

Girshick, R., Donahue, J., Darrell, T., Malik, J., 2014. Rich feature hierarchies for accurate object detection and semantic segmentation, in: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 580–587.

Jocher, G., Chaurasia, A., Qiu, J., 2023. Yolo by ultralytics (version 8.0. 0)[computer software]. YOLO by Ultralytics (Version 8.0. 0)[Computer software] .

Kraus, F., Dietmayer, K., 2019. Uncertainty estimation in one-stage object detection, in: 2019 ieee intelligent transportation systems conference (itsc), IEEE. pp. 53–60.

LeCun, Y., Bengio, Y., Hinton, G., 2015. Deep learning. nature 521, 436–444.

Redmon, J., Divvala, S., Girshick, R., Farhadi, A., 2016. You only look once: Unified, real-time object detection, in: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 779–788.

Roboflow, 2023. Yolov7 - trash dataset v3 -04/01/2023 dataset. https://universe.roboflow.com/technological-institute-of-the-philippines/yolov7-trash-dataset-v3-04-01-2023. Visited on 2024-05-22.

Ukaogo, P.O., Ewuzie, U., Onwuka, C.V., 2020. Environmental pollution: causes, effects, and the remedies, in: Microorganisms for sustainable environment and health. Elsevier, pp. 419–429.

Wang, Z., Jiao, B., Xu, L., 2021. Visual object detection: A review, in: 2021 40th Chinese Control Conference (CCC), IEEE. pp. 7106–7112.

Zou, Z., Chen, K., Shi, Z., Guo, Y., Ye, J., 2023. Object detection in 20 years: A survey. Proceedings of the IEEE 111, 257–276.