Proceedings of Engineering and Technology Innovation, vol. 33, 2026, pp. 106-119

Detection and Classification of Floating Waste on Water Surfaces Using YOLO-Based Algorithms

Boonchoo Jitnupong, Jirawan Charoensuk, Supaporn Bundasak, Supakrit Somritjinda, Nonpawit Silabumrungrad, Jaruwan Suraseing*

Department of Computer Science and Information, Faculty of Science at Sriracha, Kasetsart University, Sriracha, Thailand

Received 26 December 2025; received in revised form 06 April 2026; accepted 10 April 2026

DOI: https://doi.org/10.46604/peti.2026.16023

Abstract

Floating waste on water surfaces is one of the causes in environmental pollution. Foam, plastics, and glass bottles are not readily biodegradable and adversely affect aquatic life. This research focuses on detecting, classifying, and counting non-biodegradable and hard-to-decompose waste floating on water surfaces using object detection techniques. Five classes of waste, cans, foam, plastic bags, plastic bottles, and miscellaneous items, are evaluated in this study. Detection and classification are performed by using six models-YOLOv5, YOLOv7, YOLOv8, YOLOv9, YOLOv10, and YOLOv11-which can identify overlapping objects. This study utilizes two datasets with varying resolutions, as well as two model sizes and two batch sizes. In the experimental evaluation, the YOLOv11 model outperforms the other models with a precision of 83% and mAP50 of 78%. Plastic bags and plastic bottles are classified more accurately by YOLOv11 than by YOLOv5, with improvements of 22% and 15%, respectively.

Keywords: floating waste detection, image processing, object detection, YOLO, environmental monitoring

1. Introduction

The floating waste problem is one of the environmental crises with severe consequences for public health [1], aquatic ecosystems, and the economy, including declines in tourism and fisheries [2]. Most of this waste comes from human activities, including household, industrial, and commercial activities. Waste found in water sources consists of plastics, such as water bottles, plastic bags, foam boxes, aluminum cans, food containers, and decomposable food scraps and organic materials. Allowing this waste to float in water sources for a long time causes changes in water chemistry and reduces dissolved oxygen levels. Therefore, managing floating waste has become an urgent necessity in many countries, especially those with large river and canal networks [3]. However, monitoring and managing floating waste on a large scale is challenging and requiring substantial resources [4]. The use of human labor or traditional garbage collection boats is often limited in terms of efficiency, coverage, and cost [5]. In addition, physical factors, such as currents, heavy rain, or constantly changing water surface reflections, make the detection of debris by human eyes prone to inaccuracies [6].

With the advancement of technologies, especially in the fields of artificial intelligence (AI) and deep learning, new opportunities have been opened to deal with complex environmental problems. Object detection is one of the widely used techniques in image processing [5], enabling the real-time identification, tracking, and classification of objects in images and videos without continuous human supervision [7]. Within this broader context, the detection of floating debris in aquatic environments represents a critical yet challenging sub-domain. YOLO (You Only Look Once) is an object detection

* Corresponding author. E-mail address: jaruwan.sur@ku.th

architecture developed to increase efficiency in both speed and accuracy by separating object search and classification. The YOLO model can separate object bounding boxes and identify object types simultaneously in a single iteration, making it suitable for real-time applications, especially when installed on drones or autonomous vehicles.

This research faces limitations in terms of resources, specifically the inability to input high-resolution data and the limited performance of the GPU used for processing. Future work involves the deployment of devices across diverse locations to detect waste with acceptable accuracy while minimizing installation and processing costs. Other popular models, such as Faster R-CNN and Detection Transformer, require significant VRAM for processing, and with low-performance GPUs, processing speeds are slow. Furthermore, image pixelation or blurriness can lead to incorrect object detection, as both models demand high image resolution. However, YOLO can process and maintain good accuracy even with low-performance GPUs and VRAM.

In this research, the results of a comparison of floating debris detection and classification methods using six versions of the YOLO model: YOLOv5, YOLOv7, YOLOv8, YOLOv9, YOLOv10, and YOLOv11, are presented, which have been improved to detect small objects and perform well. The types of waste specified in the classification are plastic bottles, metal cans, foam boxes, plastic bags, and other types of waste. This waste takes a long time to decompose or cannot decompose at all. The operation involves importing data, and all image data are manually framed using the labeling tool and subjected to further processing, such as image rotation, brightness adjustment, and image flipping, to increase the diversity of the training data. However, YOLO is a popular model that is constantly being developed and now has many versions. Therefore, when using a YOLO model in object detection processes with limited input and processing resources, it is necessary to establish guidelines for selecting the most appropriate YOLO version based on the resource constraints of each task.

2. Literature Review

Developing efficient and realistic methods for detecting and classifying objects on the water's surface requires a study of relevant theories, concepts, and research, particularly in image processing and object detection using deep learning techniques such as YOLO, which is widely used in real-time applications. Therefore, the following section provides a literature review of the fundamental concepts of object detection and prior research on floating waste detection.

2.1 Object Detection Using Deep Learning

Object detection is a research area that has gained much attention and plays a significant role in various applications [8] such as autonomous driving, surveillance, medical diagnosis, and traffic monitoring [9-11]. Object detection can process both images and videos in real time [12]. In the past, object characteristics were manually classified by humans, which was slow and prone to errors. Therefore, the traditional method was replaced by deep object detection techniques [11, 13]. Today, computer vision is used to identify and locate objects in images, providing flexible and efficient detection through deep learning methodologies [14-15]. Deep learning architectures such as R-CNN, Fast R-CNN, Faster R-CNN, and YOLO are commonly used for object detection [16]. Although many tools are available, traditional detection algorithms often suffer from low efficiency and poor robustness [9], prompting the development of new algorithms. However, each algorithm has its own characteristics, making it difficult to select the most suitable one. It is therefore necessary to study various deep learning architectures and determine appropriate parameter values suited to the specific conditions of each domain.

2.2 Floating Waste Detection Using YOLO

The aquatic environment is increasingly polluted by various types of litter, such as bottles, plastics, and cans, which float on the water surface and pose a significant threat to marine ecosystems [17]. As a result, the application of object detection

techniques for identifying and counting floating litter has attracted significant research attention. However, detecting floating litter presents several challenges, including small object size, varying viewing angles [17-18], low pixel density, and complex backgrounds, all of which increase the likelihood of false detections.

To address these challenges, various versions of the YOLO (You Only Look Once) algorithm have been employed for floating litter analysis. The objects detected typically include plastic bottles, aluminum cans, plastic bags, foam, and plastic containers. YOLOv5, YOLOv7, and YOLOv8 have been effectively used to identify such litter in river environments, demonstrating high detection accuracy [20-21]. In addition, input data is typically collected via CCTV cameras, and different model sizes of YOLOv5-Nano, Small, Medium, and Large have been tested to evaluate their respective performance in terms of model accuracy [22].

Due to resource limitations in this research, particularly regarding data input equipment (a low-resolution camera) and processing power (which is limited), and despite previous research applying YOLO to surface debris detection, the varying processing resource requirements necessitated further experimentation to find a processing model optimized for the limited resources available in this study.

2.3. Multiclass Detection Using YOLO

In object detection, the class must be defined, which refers to the type of object. Sometimes, in detection, there are many types of objects. The more similar the characteristics of each class, the more difficult it is to detect. Research has shown that as the number of answer classes increases, accuracy tends to decrease. The experiment used the same algorithm for testing [23]. Currently, object detection is used to classify many types of classes, such as detecting heavy traffic by identifying eight classes of vehicle types: cars, buses, trucks, combis (micro-buses), moto-taxis (auto-rickshaws), taxis, motorcycles, and bicycles [24]. In detecting objects with remarkably similar characteristics, such as recognizing pests that involve many classes with similar features, classification becomes more difficult. However, improving the efficiency of the model results in faster and more accurate processing [25].

3. Methodology

This research presents four main stages: data collection, data preprocessing in a specified format (including image color transformation, rotation, and resolution adjustment), followed by model training and performance evaluation for efficient object detection. The overall procedure is illustrated in Fig. 1.

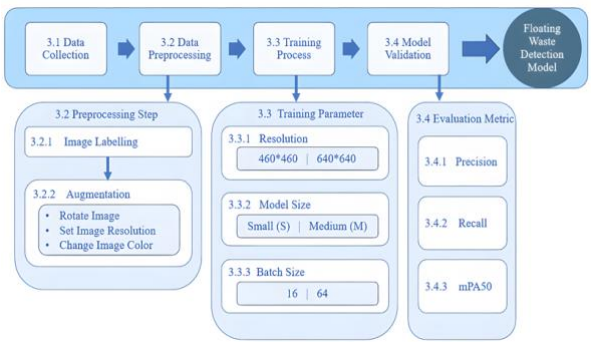


Fig. 1 Overview of Floating Waste Detection

Fig. 1 shows the entire work process of this research, consisting of four main steps. The first step is data collection, which involves collecting photos of several types of floating waste on the surface of the water. The second step is data preprocessing, which consisted of three sub-steps: image labeling, augmentation, and separate data training and testing. The third step is the

training process, which imports data through the data preprocessing process and performs training and testing to obtain a model that is accurate in detecting objects, including an experiment conducted to adjust the values of three parameters: image resolution, model size, and batch size to find the optimization values. Finally, the last step is accuracy verification of the model used to detect objects. The details of each step are explained below.

3.1 Data Collection

Although several public datasets, such as FloW [26] or RiSID [27], are available, they primarily consist of high-resolution images captured under controlled conditions [28]. In contrast, the target application of this research is designed for resource-limited embedded systems equipped with low-specification cameras. As a result, these datasets are unsuitable for training and evaluation under the actual operating conditions of this research system. To address this limitation, a dedicated dataset is collected to closely match the hardware specifications and deployment environment of the embedded platform.

In this research, a custom image dataset of floating garbage is collected from the sea, urban canals, and other waterways. All images are captured using a camera mounted on a fixed support approximately 3 m above the water surface, with the optical axis tilted downward at about 45° to emulate the intended deployment setup. Image acquisition is conducted during daytime (09:00–18:00) under natural illumination at multiple locations with varying water colors, backgrounds, and lighting conditions, ensuring that the dataset captures diverse real-world scenarios. The captured scenes contain various types of floating debris, including clearly visible objects such as colored foam boxes and plastic bottles, as well as partially submerged or visually degraded items such as crumpled plastic bags and incomplete floating objects. In total, A total of 225 images are collected, stored at the camera's native low resolution, and resized to 640 × 640 and 460 × 460 pixels to match the target system input specifications. The dataset used in this study is publicly accessible at https://github.com/jitnupong-b/marine-debris640. Representative examples of the collected images are shown in Fig. 2.

Fig. 2 Examples of Floating Waste on Water Surfaces

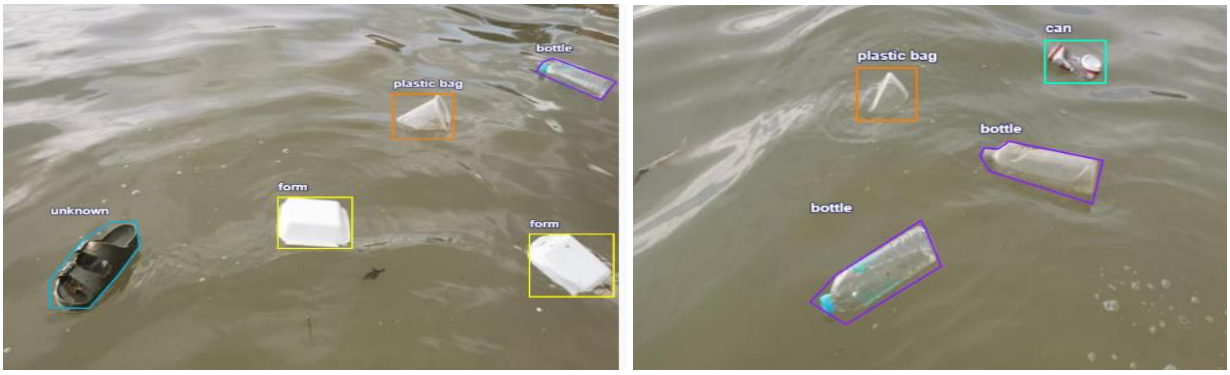
3.2 Data Preprocessing

Data preprocessing is a crucial step in the proposed methodology as it directly impacts model performance. Since images taken in water-surface environments often exhibit issues such as lighting variation, reflections, and low resolution, appropriate data processing techniques are necessary to improve data quality before model training. The detailed data processing steps used in this work will be described in the next section.

3.2.1 Image Labelling

The object type definition is a key step in data preparation for model training, particularly in object detection tasks such as floating waste detection. Roboflow is used to support this step. Five object categories are defined: can, foam, plastic bag,

bottle, and unknown. The 'unknown' category includes objects that do not belong to the specified classes, such as shoes, coconuts, and bamboo. In this step, Roboflow provides boundary tools for defining object regions. Square bounding boxes are used for regular-shaped objects such as foam, while the Polygon tool is applied to non-square objects, such as long bottles, for more accurate labeling. An example of image labeling is shown in Fig. 3.



(a) Object Classes: Unknown, Foam, Plastic Bag, Bottle (b) Object Classes: Bottle, Plastic Bag, Can

Fig. 3 Example of Image Labelling

A total of 225 images are imported and labeled into five object categories: can (128 instances), foam (149 instances), plastic bags (169 instances), bottles (500 instances), and unknown (269 instances). Among these, plastic bottles are the most prevalent, reflecting their common occurrence as floating waste in the study area. The detailed distribution of each object category is presented in Table 1.

Table 1 Number of each Type of Object Counted

Class	Name	Number of Pieces
1	Can	128
2	Foam	149
3	Plastics Bag	169
4	Bottle	500
5	Unknown	269

3.2.2 Image Augmentation

Image Augmentation is a method that increases the diversity of the dataset used to train the model without collecting additional data. To improve the robustness and generalization of the detector, this research applies standard image augmentation techniques, including random flipping, rotation, cropping, and brightness adjustment, to synthetically increase the diversity of the training set. These augmentations simulate variations in camera pose, illumination, and background clutter that may occur in real deployments, and thus help mitigate overfitting when training on a relatively small dataset [29-30]. In this research, three types of image adjustment techniques are used:

(1) Image: This technique allows the model to learn to detect waste from different viewpoints, which makes the model more flexible, by rotating each image by 90 degrees, as shown in Fig. 4.
(2) Set Image Resolution: Two resolution levels, 460 × 460 and 640 × 640, are used to evaluate the effect of image resolution on object detection accuracy.
(3) Change Image Color: This technique involves image brightness and color tone adjustment to help the model recognize garbage images in different lighting and color conditions. An example of adjusted image brightness is shown in Fig. 5.

After image augmentation, the dataset increases from 225 to 3,864 images. Of these, 3,012 are used for training, 753 for validation, and 99 for testing, as summarized in Table 2.

(a) Original Image Orientation

(b) 90º Rotation of the Image

(c) 180º Rotation of the Image

(d) 270º Rotation of the Image

Fig. 4 Example of Image Rotation

(a) Image with Natural Colors

(b) Grayscale Images

Fig. 5 Example of Image Colour Change

Table 2 Number of Data Training and Testing

Data Set	Number of Images
Training dataset	3,012
Validation dataset	753
Test dataset	99

3.3 Training Process

To train the model, different versions of YOLO are used: YOLOv5, YOLOv7, YOLOv8, YOLOv9, YOLOv10, and YOLOv11. The learning rate is set to 0.01, the optimizer is Stochastic Gradient Descent, and the number of epochs is set to 100. To evaluate the practical model, it is deployed on a Raspberry Pi 5 (8GB RAM) with a Broadcom BCM2712 Quad-core ARM Cortex-A76 processor at 2.4 GHz. The inference is performed using ONNX Runtime to optimize performance for edge computing. This setup represents a realistic scenario for low-cost, portable marine debris monitoring systems. To examine the parameter values that affect the performance of YOLO in different versions, three different parameter values are defined as follows:

3.3.1 Resolution

Resolution is a measure of the level of detail that can be displayed in an image or digital display. It is usually measured in terms of the number of pixels or dots. Higher resolution values provide more detailed images, but it requires more storage space and may also require higher processing efficiency. In this research, two levels of resolution are set: 460 x 460 and 640 x 640. This is a relatively low resolution, as this research focuses on data processing using low-efficiency computing resources in order to reduce operating costs to investigate the appropriate resolution for object detection to achieve the highest accuracy.

3.3.2 Model Size

In the processing of YOLO in different versions, the size of the processing model is specified, which refers to the complexity of the processing architecture. In each version, the model size is specified in five levels:

(1) Nano is the smallest processing size, which has the fastest processing, but it is the least accurate.
(2) Small is a small-scale process that has a good balance and can be used on mobile devices because it does not require many resources.
(3) Medium is medium size, so the processing has a medium level of efficiency.
(4) The large model offers high accuracy but slower processing
(5) XLarge is the highest accuracy processing, which has many calculations and requires extremely high processing resources.

The advantage of the small model is that it is fast and does not require many processing resources, such as RAM and GPU, but the accuracy is quite low, especially if small objects are to be inspected. The advantage of the large model is that it has better accuracy in detecting small objects, but the processing is slow and requires a lot of memory.
In this research, the model size is specified in two levels, S and M. The garbage detection model might not invest much in resources in this part of the work, so the large-sized model is not included in the processing. However, the nano-sized model is not chosen because some garbage photos might have been taken from a distance, resulting in a small size of the objects. If the nano-sized model is chosen for processing, it might result in lower processing accuracy.

3.3.3 Batch Size

Batch size is the number of images imported for training in each round. The higher the value, the more data is imported for training at a time, resulting in faster training. However, there are limitations in terms of resources used for processing, such as GPU and memory. If the image is large and many images are sent for training, the machine's resources may be insufficient for processing. The batch size can be set at several levels: 8, 16, 32, and 64. Setting a small batch size makes training slower.

On the other hand, setting up a large batch size makes training faster, but it also requires higher resources. In this research, two levels of batch size were set: 16 and 64, to compare the performance between small and large batch sizes.

3.4 Model Validation

Model validation is a crucial step in evaluating model reliability. Models developed using YOLO must be validated using previously unseen validation and testing datasets to ensure their applicability in real-world scenarios. In this research, object detection evaluation metrics used include precision, recall, and Mean Average Precision (mAP).

3.4.1 Precision

Precision is a measure of the model's accuracy, which ranges from 0 to 1. The closer the value is to 1, the more accurate the model is. However, the closer the value is to 0, the more the model mis predict the Positive class. The equation for calculating the precision value is shown in Eq. (1).

$$Precision = \frac{TurePositive(TP)}{TurePositive(TP) + FalsePositive(FP)} \tag{1}$$

True Positives (*TP*)*:* Correct positive predictions

False Positives (*FP*)*:* Incorrect positive predictions

3.4.2 Recall

Recall is the ratio of true positive detections to all actual positive instances. The Recall value ranges from 0 to 1, with higher values being better. When the Recall value is close to 1, it indicates that the model is very capable of identifying the correct Positive Class because the model can fully find all positive classes available in the data. However, if the Recall value is close to 0, it indicates that the model is less capable of identifying the correct positive class. The equation for calculating the Recall value is shown in Eq. (2).

$$\mathrm{Re}\,call = \frac{TruePositive(TP)}{TruePositive(TP) + FalseNagative(FN)} \tag{2}$$

True Positives (*TP*)*:* Cases where the model correctly predicted the positive class.

False Negative (*FN*)*:* Cases where the model missed a positive class.

3.4.3 mPA50μ

Mean Average Precision (mAP) is a widely used metric for evaluating model performance. It represents the mean of the average precision values across all classes. The Intersection over Union (IoU) threshold is typically set above 0.5; in this study, an IoU threshold of 0.5 is used. The formula for calculating mAP is presented in Eq. (3).

$$mAP = \frac{1}{N} \sum_{i=1}^{N} AP_i \tag{3}$$

N: Number of Classes

AP_i*:* Average Precision for class i

4. Experiment and Result

In this research on surface debris detection, six YOLO versions are evaluated to determine which is most suitable for detecting surface debris. The YOLOs used in the analysis were YOLOv5, YOLOv7, YOLOv8, YOLOv9, YOLOv10, and YOLOv11. Three factors that may affect the accuracy of the processing are considered: resolution, model size, and batch size. The test results are shown below:

4.1 Experimental Results with Different Parameters

(1) Resolution levels are set 460x460 and 640x640

Various measurement values are used to measure the performance of all six versions of YOLO. The experimental results are shown in Table 3. In image processing, increasing the resolution typically leads to improved accuracy compared to using lower resolutions. However, as shown in Table 3, YOLOv5 demonstrates impressive performance even at lower resolutions, whereas more advanced versions of YOLO achieve better results at higher resolutions. Notably, the average precision value remains consistent at 0.80 for both resolution settings. These findings suggest that resolution, whether high or low, has a comparable impact on the model's accuracy.

(2) Inference

There are two resolution options: 460x460 and 640x640. The experimental results in Table 4 indicate that image resolution influences processing speed. Therefore, for fast real-time processing on the target device, a resolution of 460x460 combined with the YOLOv11 model is the most suitable configuration, as it achieves the lowest inference time and the highest FPS.

Table 3 Results of Different Resolution

YOLO Version	Resolution					
	460x460			640x640		
	Precision	Recall	mAP50	Precision	Recall	mAP50
YOLOv5	0.84	0.73	0.77	0.69	0.61	0.66
YOLOv7	0.75	0.64	0.68	0.84	0.76	0.80
YOLOv8	0.83	0.70	0.76	0.82	0.76	0.79
YOLOv9	0.76	0.68	0.73	0.81	0.71	0.77
YOLOv10	0.81	0.70	0.76	0.83	0.76	0.80
YOLOv11	0.82	0.71	0.77	0.84	0.73	0.79
Average	0.80	0.69	0.75	0.80	0.72	0.77

Table 4 Inference Time and FPS of Different Resolutions

YOLO Version	Resolution			
	460x460		640x640	
	Inference Time (ms)	FPS	Inference Time (ms)	FPS
YOLOv5	108	7.8	201	5.1
YOLOv7	172	5.8	305	3.1
YOLOv8	125	7.3	257	3.5
YOLOv9	149	6.8	268	3.4
YOLOv10	102	8.9	190	4.8
YOLOv11	99	9.5	180	5.2

(3) Model sizes and Batch sizes in YOLO

In the analysis of Model size and Batch size, another principal factor is determined. In this research, two Model size values are determined: Medium (M) and Small (S). In addition, two Batch size values are determined: 16 and 64.From the experimental results in Table 5, it is found that Model Size S and M did not significantly affect the model accuracy. However, setting the Batch Size with varied sizes of 16 and 64 led to differences in model accuracy. Setting the Batch Size to a larger size resulted in higher model accuracy.

(4) Each type of object (In this research, five types of objects are detected: can, foam, plastic bag, bottle, and unknown.)

Table 5 Result in Different Model Sizes and Batch Sizes

YOLO Version	Model Size and Batch Size			
	M		S	
	16	64	16	64
YOLOv5	0.70	0.84	0.69	0.83
YOLOv7	0.85	0.84	0.73	0.82
YOLOv8	0.83	0.84	0.81	0.82
YOLOv9	0.77	0.76	0.80	0.80
YOLOv10	0.82	0.83	0.82	0.81
YOLOv11	0.84	0.84	0.81	0.83
Average of Precision	0.80	0.83	0.78	0.82

Table 6 Result on The Accuracy of Detection of Each Type of Object

YOLO Version	Can	Foam	Plastic Bag	Bottle	Unknown
YOLOv5	0.92	0.94	0.63	0.58	0.76
YOLOv7	0.83	0.87	0.81	0.72	0.74
YOLOv8	0.89	0.94	0.83	0.73	0.73
YOLOv9	0.80	0.92	0.81	0.69	0.69
YOLOv10	0.86	0.93	0.85	0.72	0.74
YOLOv11	0.87	0.94	0.85	0.73	0.77
Average of Precision	0.86	0.93	0.80	0.69	0.74

Table 6 shows the accuracy of each object detection. It is found that Foam detection gives the highest accuracy of YOLO in all versions, with an average precision of 0.93, which shows that all models can detect foam objects most accurately. Bottles are a challenging object to detect because they have different characteristics and colours, making it difficult to detect. The bottle has an average precision of 0.69. Nevertheless, an imbalance in the number of objects in the training is shown that bottle is significantly more prevalent, but the accuracy of the result for the bottle is still less than that of the other, this is because of the variety of bottle styles and colors. Unknown group, with an average precision of 0.74, which is reasonable because these objects are very diverse. In terms of model quality, YOLOv5 can detect objects such as cans and foam very well. However, it performs poorly on plastic bags and bottles, with later versions of YOLO showing more balanced performance across categories, showing improvements in object detection across versions, especially plastic bags, where the accuracy increases from 0.63 in YOLOv5 to 0.85 in YOLOv11.

(5) Each type of object (in this research, comparing each object using a confusion matrix is shown in Fig. 6.)

(6) Overall, every object in YOLO in every version (When comparing the performance of YOLO in all versions in object detection, using three metrics, the comparison is shown in Table 7.)

The confusion matrix shows correct and incorrect predictions across classes. The 'bottle' class is predicted most frequently (365 instances) due to dataset imbalance but has the lowest accuracy, with misclassifications into 'plastic bag' (52) and 'unknown' (45), likely due to variability in appearance. In contrast, 'foam' achieves the highest accuracy, with only four misclassifications, owing to its consistent shape and distinctive color. Table 7 shows a clear trend in the improvement of YOLO across different versions. YOLOv5 performed the worst across all metrics, with a precision of 0.76, a recall of 0.67, and an mAP50 of 0.71. In comparison, YOLOv11 demonstrated the best overall performance, with a precision of 0.83, a recall of 0.73, and an mAP50 of 0.78. There is a continuous improvement from YOLOv5 to YOLOv11, but with a degradation in YOLOv9, with a precision of 0.78, a recall of 0.69, and mAP50 of 0.75. YOLOv8, v10, and v11 had remarkably similar performances.

4. 2 Waste Detection Model Implementation

This research has developed the results from the experiment into a web application and has added object counting functionality from the images, which impacts the application in real-life scenarios. Because the waste analyzed in this research is non-degradable or may take a long time to decompose, if it is detected from images taken by Raspberry Pi 5 (8GB RAM) and there is a large amount of waste, officers should be dispatched to collect the waste to prevent further environmental damage. In developing this web application, the study results of YOLO in various versions are analyzed, and it is found that YOLOv11 has the most efficient performance in detecting waste on the water surface. Therefore, YOLOv11, which uses images with a resolution of 640×640, model size M, and batch size 64 as the analysis model to obtain the best results, is used to develop this web application. An example of the web application page is shown in Fig. 7.

Fig. 7 shows the interface, with the left panel illustrating data input and the process of importing image data into the system. When the start button is pressed, the detected objects are displayed with bounding boxes, and the count of each object type is shown on the screen.

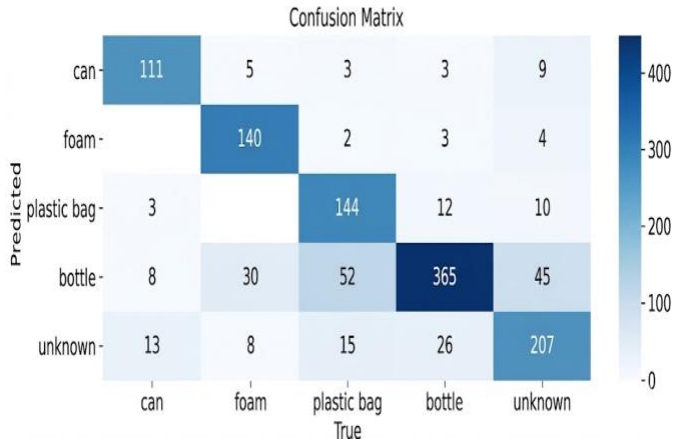


Fig. 6 Confusion Matrix to Predict the Accuracy of Each Object

Table 7 Overall Experimental Results of Different YOLO Versions

YOLO Version	Average of Precision	Average of Recall	Average of mAP50
YOLOv5	0.76	0.67	0.71
YOLOv7	0.80	0.70	0.74
YOLOv8	0.82	0.73	0.78
YOLOv9	0.78	0.69	0.75
YOLOv10	0.82	0.73	0.78
YOLOv11	0.83	0.73	0.78

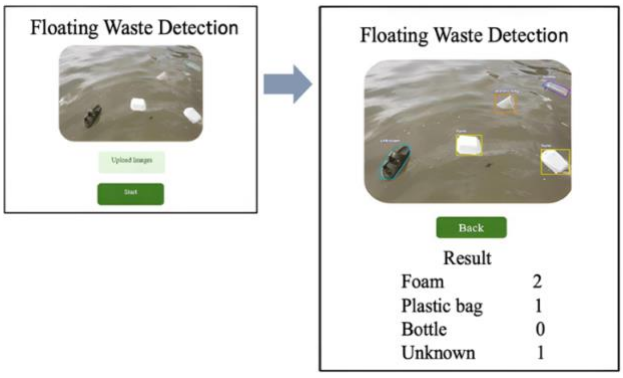


Fig. 7 Web Application for Floating Waste Detection

5. Conclusions

This study applied image processing techniques to detect floating waste on water surfaces, particularly non-biodegradable or slowly degradable waste. Object detection was performed using YOLO, a widely used real-time detection framework. Since multiple YOLO versions and parameter settings were available, experiments were conducted to evaluate their performance under different configurations. Six YOLO versions, YOLOv5, YOLOv7, YOLOv8, YOLOv9, YOLOv10, and YOLOv11, were tested using variations in image resolution, model size, and batch size. The key findings are summarized as follows:

(1) YOLOv11 achieves the best performance, with a precision of 83% and an mAP50 of 78%, outperforming YOLOv5 in floating waste detection.
(2) For specific objects, YOLOv11 improves the detection accuracy of plastic bags by 22% and bottles by 15% compared to YOLOv5.
(3) Image resolution and model size have minimal impact on accuracy, while higher batch sizes improve detection performance.
(4) The best-performing model is deployed in a web application for automated waste counting, supporting environmental monitoring and waste management.

The developed system supports environmental monitoring through automatic waste detection and counting in real time. It can identify areas with high waste accumulation more efficiently. The system also assists waste management officers in optimizing waste collection operations. By improving monitoring and collection efficiency, the system enhances waste management practices.

Conflicts of Interest

The authors declare no conflict of interest.

References

[1] Y. Zheng, X. Nong, L. Chen, and D. Long, “A Novel Deep Learning-Based Floating Garbage Detection Approach and Its Effectiveness Evaluation in Environmentally Sustainable Development,” Journal of Environmental Management, vol. 38, pp. 125154, 2025.
[2] A. Nahman, L. Haywood, and S. Oelofse, “The Economic Impact of Marine Plastic Debris in South Africa,” Marine Pollution Bulletin, vol. 212, pp. 117461, 2025.
[3] S. Ashok, V. A. J. Venita, E. G. Shankari, M. Ranjana, and V. Prabhu, “Autonomous Watercraft for Cleanup of Floating Waste in Water Bodies Using YOLO,” Premier Journal of Science, vol 15, pp. 100144, 2025.
[4] M. Liedermann, D. González-Fernández, F. Mendrik, L. Biermann, and T. H. van Emmerik, “Contributions to River Plastic Monitoring Across Scales, Volume II,” Frontiers in Earth Science, vol. 13, pp. 1636075, 2025.

[5] J. P. Q. Tomas, J. E. E. Tupas, M. T. Soniel, C. H. M. E. Caruz, and D. B. Babar, "Real-Time Detection of Floating Debris in Waterways Using YOLOv8," Proceedings of the 2024 14th International Workshop on Computer Science and Engineering (WCSE), Phuket Island, Thailand, pp. 19–21, 2024.

[6] Y. Zhong, J. Wan, M. Cao, Z. Tan, Y. Qiu, L. Zhang, et al. "Lightweight Robust Detection of Anthropogenic Floating Debris in Turbid and Dynamic Aquatic Environments Via Enhanced Feature Fusion," Scientific Reports, vol. 16, article no. 1373, 2025.

[7] A. Konya and P. Nematzadeh, "Recent Applications of AI to Environmental Disciplines: A Review," Science of The Total Environment, vol. 906, pp. 167705, 2024.

[8] L.Jiao, F.Liu, S.Yang, Z. Feng, and R.Qu , "A Survey of Deep Learning-Based Object Detection," Advances in Engineering Technology Research, vol. 7, pp. 128837-128868, 2019.

[9] J. Meng, S. Shen, J. Wang, and C. Zhou "Object Detection Algorithms Based on Deep Learning: A Review," Asian Journal of Research in Computer Science, vol. 17, no. 8, pp. 1-12, 2024.

[10] B. Ganga, Lata B.T., and Venugopal K.R., "Object Detection and Crowd Analysis Using Deep Learning Techniques: Comprehensive Review and Future Directions," Neurocomputing, vol. 597, article no. 127932, 2024.

[11] Z. Q. Zhao, S.T.Xu, X.Wu, "Deep Learning Object Detection," IEEE Transactions on Neural Networks and Learning Systems, vol. 30, pp. 3212–3231, 2019.

[12] G. Pavithra, J. J. Jose, and T. A. Chandrappa, "Real-Time Color Classification of Objects from Video Streams," Proceedings of The 2nd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT), IEEE Press, pp. 1683-1686, 2017.

[13] B. Fan and X. Song, "A Review of Deep Learning Based Object Detection Algorithms," Journal of Engineering Research and Reports, vol. 26, no. 11, pp. 88-99, 2024.

[14] Y.H. Lee and W.B. Lee, Object Detection and Tracking Based on Deep Learning., Switzerland: Springer Nature, 2020.

[15] P. K. Hebbar and P. K. Pullela, "Deep Learning in Object Detection: Advancements in Machine Learning and AI," Proceedings of International Conference on the Confluence of Advancements in Robotics, Vision and Interdisciplinary Technology Management (IC-RVITM), IEEE Press, pp. 1-7, 2023

[16] S. Doshi, K. Desai, and K. Mehta, "Various Approaches to Object Detection Using Deep Learning," International Journal for Research in Applied Science and Engineering Technology, vol. 11, no. 7, pp. 1799-1806, 2023.

[17] M. R. Arjasubrata, A. M. Bakir, G. N. Irene S, S. A. Sari, and M. D. Sulistiyo, "Trash Detection in Surface Waters Using YOLOv8," Proceedings of the 11th International Conference on Information Technology and Electrical Engineering (ICITACEE), IEEE Press, pp. 332-337, 2024.

[18] Y. Li, R. Wang, D. Gao, and Z. Liu, "A Floating-Waste-Detection Method for Unmanned Surface Vehicle Based on Feature Fusion and Enhancement," Journal of Marine Science and Engineering, vol. 11, no. 12, article no. 2234, 2023.

[19] C. Peng, B. He, W. Xi, and G. LIN, "Improved YOLOv7 Algorithm for Floating Waste Detection Based on GFPN and Long-Range Attention Mechanism," Wuhan University Journal of Natural Sciences, vol. 29, no. 4, pp. 338-348, 2024.

[20] B. S. Devi, D. Gupta, and R. P. Singh, "Yolo-Based Deep Learning Techniques for Identifying Floating Bottles in Inland Water: A Comprehensive Analysis," Proceedings of International Conference on Computing, Communication, and Networking Technologies (ICCCNT), IEEE Press, pp. 1-6, 2024.

[21] G. A. Sio, D. Guantero, and J. Villaverde, "Plastic Waste Detection on Rivers Using YOLOv5 Algorithm," Proceedings of International Conference on Computing Communication and Networking Technologies (ICCCNT), IEEE Press, pp. 1-6, 2022.

[22] L. Liu, B. Zhou, G. Liu, D. Lian, and R. Zhang, "Yolo-Based Multi-Model Ensemble for Plastic Waste Detection Along Railway Lines," Proceedings of IEEE International Geoscience and Remote Sensing Symposium, IEEE Press, pp. 7658-7661, 2022.

[23] S. C. Shrawne, J. Sawant, O. Chaubal, K. Suryawanshi, D. Sirwani, and V. K. Sambhe, "Multiclass Fruit Detection Using Improved YOLOv3 Algorithm," International Journal of Advanced Computer Science and Applications, vol. 15, no. 9, pp. 978-988, 2024.

[24] P. I. Montenegro-Montori, J. Camasca-Huamán, and J. Fabian, "Multi-Class Vehicle Detection and Automatic License Plate Recognition Based on YOLO in Latin American Context," Advances in Intelligent Systems and Computing, vol. 1150, Springer, Cham, pp. 264-278, 2020.

[25] S. Dong, J. Zhang, F. Wang, and X. Wang, "YOLO-Pest: a Real-Time Multi-Class Crop Pest Detection Model," Proceedings of 11th International Conference on Graphics and Image Processing (ICGIP 2022), SPIE, pp. 1-7, 2022.

[26] Y. Cheng, J. Zhu, M. Jiang, J. Fu, C. Pang, P. Wang, and Y. Bengio, “Flow: A Dataset and Benchmark for Floating Waste Detection in Inland Waters,” Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), IEEE Press, pp. 10953-10962, 2021

[27] T. Kataoka, T. Yoshida, and N. Yamamoto, “RiSID: River Surface Image Dataset for Instance Segmentation of Floating Macroplastic Debris,” Data in Brief, vol. 63, article no. 112189, 2025.

[28] B. Badams, U. U. Sheikh, N. A. Wahab, S. A. A. Bakar, M. I. Masud, M. Khouj, et al., “A2ANet: Real-Time Detection of Floating Marine Debris Using Atrous Convolution and Channel Attention,” Ecological Informatics, vol. 94, article no. 103620, 2026.

[29] C. Shorten and T. M. Khoshgoftaar, “A Survey on Image Data Augmentation for Deep Learning," Journal of Big Data, vol. 6, pp. 1-48, 2019.

[30] L. N. Smith, “Cyclical Learning Rates for Training Neural Networks,” Proceedings of the 2017 IEEE Winter Conference on Applications of Computer Vision (WACV), IEEE Press, pp. 464-472, 2017.