Ocean Medical Waste Detection for CPU-Based Underwater Remotely Operated Vehicles (ROVs)

Adrian Lee, Benny Jiang, Ingrid Zeng, Michal Aibin *Member, IEEE*
Department of Computing, British Columbia Institute of Technology, Vancouver, Canada
maibin@bcit.ca

***Abstract*—This paper explores the possibility of using computer vision and underwater Remotely Operated Vehicles (ROVs) to detect medical waste, such as masks and gloves in oceans. We use a single-stage detector to train the machine learning approach and then validate the results using the video feed from the tethered ROV.**

***Index Terms*—ROV, computer vision, ocean waste**

I. Introduction

The ongoing COVID-19 disease has negatively impacted the lives of everyone and the country's economy [1]. Therefore, the demand for these products has increased significantly. To control the spread of the virus, governments recommended that PPE (personal protective equipment), such as face masks and plastic gloves, be used daily in public and shared spaces.

Since many of these medical products are disposable, they generate millions of tons of plastics and other plastics derivatives [2], [3]. In the short two-year span, an alarming amount of waste has ended up in the environment by indirect or deliberate actions ranging from waste mismanagement to illegal dumping. Studies have shown that face masks and other PPEs could release chemical pollutants and nanoplastics into the ocean, raising concerns from scientists, environmentalists, and environmental activists.

Face masks and plastic gloves increase the amount of marine debris circulating in the oceans. This is an urgent issue due to ocean currents that disperse litter across all major bodies of water and at all depths of the ocean. Plastic waste also lingers in the marine environment and destroys marine ecosystems. Marine organisms risk suffocation, entanglement, drowning, and physical rupture of internal organs [4]. Furthermore, nanoplastics could be accidentally ingested by these organisms, leading to starvation, hydration, or malnutrition due to false satiation.

As we know, underwater remotely operated vehicles (ROVs) could mitigate the problem by finding and removing marine litter [4]. To achieve this goal, it is essential that ROVs can detect medical waste successfully in underwater environments. To the best of our knowledge, this is the first paper to use ROVs and YOLOv5 to detect medical waste in oceans.

The paper is divided as follows. Section I describes the urgency of the issue we are trying to solve, followed by a discussion of related work. Then we introduce an object detection algorithm, specifically YOLOv5. We then discuss how we annotated our custom dataset for training and testing and trained YOLOv5 to solve our problem in Section III. Finally, we describe how we improved the efficiency of YOLO5 medical waste detection, followed by results, discussion, and conclusions.

II. Related Works

YOLO, or "You Only Look Once", is a family of open-source models. We have found that YOLO has been utilized in many real-time object detection studies through research. We have divided the relevant studies into two categories. One for detecting debris in the ocean. Another for identifying mask wear in public spaces.

YOLOv5, as mentioned, can be easily deployed on embedded devices. An embedded device is an object that contains a special-purpose computing system [5]. Running YOLOv5 on devices such as ROVs or buoys can help detect marine plastic debris more accurately. Therefore, many studies focus on training YOLOv5 for such purposes. Tata et al. used YOLOv5 in their research. The discussion section of their paper suggests improving the data augmentation, object detection algorithm, dataset, and camera to increase classification efficiency. They mentioned data augmentation techniques to improve the model's variability, including grayscale, saturation, and vertical/horizontal flipping. The dataset can also be enhanced by adding more images from different locations with different types of water conditions [4].

Another study by Lin et al. focused on re-designing the YOLOv5 algorithm to detect floating debris [6]. They introduced a feature map attention layer (FMA) at the end of the backbone to improve the ability to extract features. To maintain the number of output channels consistent with the input without increasing the computation time of the neck section, the FMA layer employs a self-attention method to weigh each channel of the top layer feature map and 1×1 convolution to manage the number of output channels. They also applied mosaic data augmentation to improve the detection impact of tiny targets during training and a dataset extension strategy to increase the training dataset from 1920 to 4800 images.

Furthermore, YOLOv5 was successfully applied to identify the mask and unmask person during the COVID-19 pandemic. In one of the studies, we found that Liu et al. improved YOLOv5's default settings by using data augmentation and anchor adjustment without gathering additional data; data augmentation allowed practitioners to expand the diversity of data available for training models significantly. Liu et

al. have implemented methods such as data flipping, data rotation, image scaling, image clipping, image translation, and noise addition [7]. Anchor adjustment involves using K-means to calculate anchors and improve the detection rate of the bounding box [7]. Applying those methods successfully increased the efficiency and accuracy of YOLOv5's models in mask-wearing detection. We will use similar methods discussed above to improve the accuracy of detecting masks and other medical waste in the ocean and then apply the model to the ROV.

III. Problem Statement

YOLOv5 provides multiple training models for different object detection cases. We will compare the YOLOv5n model (nano) with the YOLOv5s (small) model to see which model produces the most accurate results in detecting face masks and plastic gloves in the ocean.

We can compare the performance of each model using the mean average precision (mAP) using the average precision of each class. The higher mAP, the more accurate the model is based on the dataset. In the following equation, $ApMask$ refers to the average precision of the mask class, and $ApGlove$ refers to the average precision of the glove class.

$$mAP = (ApMask + ApGlove)/2 \quad (1)$$

IV. Methodology

A. Dataset

The dataset created to train the neural network includes images found on Google and screenshots taken from Youtube videos of plastic gloves and masks in an ocean environment. Currently, we are only focusing on plastic gloves and blue disposable face masks. The images have masks or gloves on the ocean's surface or submerged. We made sure that the backgrounds varied, for example, the colouring, light levels, and different locations in the ocean. Some examples in our dataset are images of masks submerged in shallow water where there are more light or images where algae slightly cover the object on the ocean floor. We also included images that show a large amount of waste in the ocean. Having a greater variety of backgrounds helps to increase object detection accuracy when looking at masks and gloves in different environments.

B. Object Detection

There are mainly two types of state-of-the-art object detectors. The two-stage detector uses the Region Proposal Network to generate regions of interest. Then, it sends those region proposals down the pipeline for object classification and bounding-box regression. Examples of two-stage detectors are R-CNN (Region-Based Convolutional Neural Networks) and Mask R-CNN. These detectors have the highest accuracy rates but are typically slower, making them less ideal for real-time object detection.

On the contrary, single-stage detectors are excellent trade-offs between accuracy and speed. They take inputs such as images and learn class probabilities and bounding box coordinates in a single step. Examples of single-stage detectors are YOLO and SSD (Single Shot MultiBox Detector) [8].

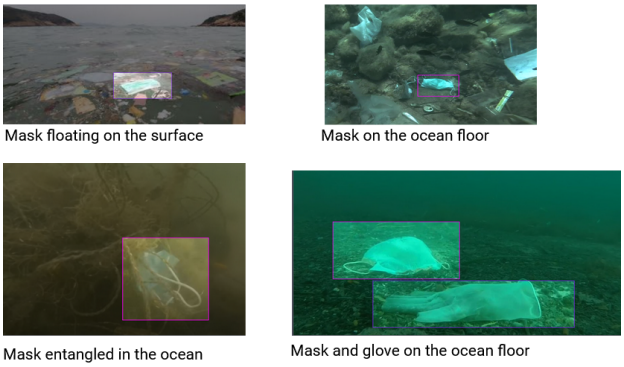

Fig. 1. Examples of images in the dataset

1) YOLO: The YOLO real-time object detection model is renowned for its accuracy and speed. It turns the detection process into a regression problem, simplifying many calculation processes [9]. YOLO was the first object detector to connect the procedure for predicting class labels in an end-to-end differentiable network [10]. YOLO implements its architecture by dividing images into a grid system and having each cell in the grid detect objects within itself.

The YOLO network has three main components: the backbone, neck, and head. The backbone is a convolutional neural network that aggregates and forms image features at different granularities [10]. It is for pre-training and can run on GPU or CPU platforms [11]. The neck is a series of layers between the backbone and the head. These layers are used to collect feature maps from different stages. The neck is composed of several bottom-up paths and several top-down paths. Lastly, the head is used to predict classes and bounding boxes of objects [12]. It can be a one-stage detector for dense prediction or a two-stage for sparse prediction objects [11].

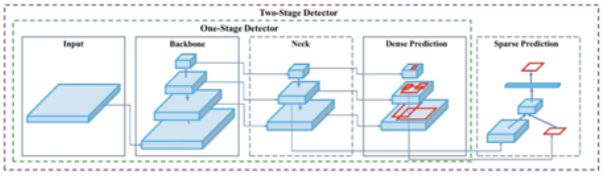

Fig. 2. The object detection process of YOLO

2) YOLOv5: For this project, we will use YOLOv5, the latest version of the YOLO series. It has a model architecture similar to YOLOv4 but has achieved performance improvement due to PyTorch training procedures [10]. YOLOv5 has achieved top performance in two official object detection datasets: Pascal VOC (visual object classes) and Microsoft COCO (common objects in context) [9].

YOLOv5 incorporated a partial cross-stage network (CSP-Net) to formulate the features of the image [9]. CSP models are based on DenseNet, designed to better connect layers in

a convolutional neural network [10]. The CSPNet addresses the duplicate gradient problem in other larger ConvNet backbones. It integrates the gradient changes into the feature maps, supporting feature propagation and encouraging the network to reuse features. It also reduces the number of network parameters and FLOPS (floating-point operations per second) of the model [9], [13]. Ultimately, implementing CSPNet minimizes the need for heavy computing resources and can be cost-saving. Another feature implemented by YOLOv5 is the PA-NET neck for feature aggregation. PA-NET adopts a new pyramidal network feature structure (FPN) and improves bottom-up routes in the neck of YOLOv5 [9], resulting in an improvement in the propagation of low-level features. Moreover, PA-NET with adaptive feature grouping links a feature grid and additional feature levels, allowing helpful information at each feature level to propagate directly to the following subnetwork [9]. Most importantly, PA-NET improves the utilization of accurate localization signals in the lower layers, which can significantly improve the location accuracy of the object.

YOLOv5n6 is the most compact version of YOLOv5. The nano model maintains the depth multiple of YOLOv5s of 0.33 but reduces the width multiple of YOLOv5s from 0.50 to 0.25, resulting in 75% fewer parameters, from 7.5M to 1.9M [14], making it the fastest among other YOLOv5 models. Because they are ideal for mobile and CPU-based solutions, we believe they will be most efficient for real-time medical waste in the ocean with ROVs.

C. Performance Evaluation

The performance of YOLOv5n can be measured by its ability to detect relevant objects accurately (Precision) and find all appropriate cases (Recall). The most common metrics are the average precision (AP) and the mean average precision (mAP). Both are based on the IoU metric, shown in (2).

$$IOU = \frac{area(gt \cap pd)}{area(gt \cup pd)} \tag{2}$$

IoU metric measures the overlap between the ground truth mask (gt) and the predicted mask (pd). It is calculated as the intersection area between gt and pd divided by the union area of gt and pd. IoU can range from 0 to 1. 0 implies that there is no overlap, and 1 means that there is perfect overlap. To evaluate an accurate detection, it is necessary to define a threshold (α). Correct detection occurs when IoU is greater than or equal to α.

Based on the result of IoU, we can build the confusion matrix to create the precision-recall curve. The confusion matrix will then classify the situation of each training result into four events. These four events can be classified into two types, positive and negative events. Positive events mean that YOLO detects the mask, while negative events mean that YOLO does not detect the mask. True means that YOLO is correct, while false means that YOLO is wrong.

Precision is the ratio of true positive events in all events for which YOLO detects the mask (3). Recall is the ratio of true positive events in all events in which there is a mask in the ocean (4). The precision call curve will be calculated based on these events. If we represent recall on the x axis and precision on the y axis, the area under this curve will be the average precision of mask class (1).

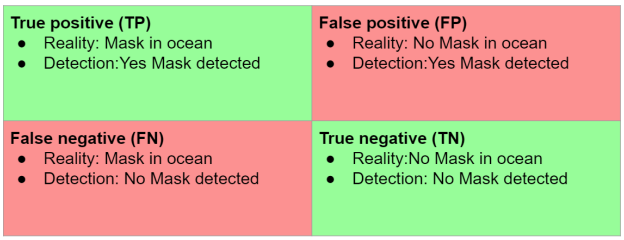

Fig. 3. Confusion matrix details

$$Precision = \frac{TP}{TP + FP} \tag{3}$$

$$Recall = \frac{TP}{TP + FN} \tag{4}$$

AP is calculated individually for each class from their precision-recall curve. $AP@alpha$ means AP at the IoU alpha threshold. Ideally, it is the Area Under the PR curve ($AUC - PR$). A high AUC-PR implies high precision and high recall. mAP can be derived from AP.

V. Training

We used Roboflow to annotate and augment the image set, and the models were trained using Google Colab. We used a small YOLOv5 model in the first experiment (YOLOv5s). In the second experiment, we used a nano YOLOv5 (YOLOv5n). Two models were trained on the same machine and the same set of augmented images. We wanted our image set to be similar to the natural underwater situation in the augmentation. Therefore, we applied 25% images in grayscale, changing the hue of the images to 50% increased or 50% decreased and changing the blurriness to 3.5 times pixels for each image. Moreover, in each image, we applied the Mosaic effect. After augmentation, we applied the 70/20/10% ratio for training/validation/test, with 1200 images. We trained using a CPU of over 200 epochs, as the underwater ROVs do not have the powerful GPUs onboard.

The overall loss of the neural network is based on both the loss of training and validation. The value is from each iteration and helps represent how well the model is performing. The goal is to have the validation loss as low as possible and a ”perfect fitting model”, meaning that the training loss is equal to the validation loss.

A. YOLOv5n

When training the nano model, we noticed that the training loss produces a slightly higher output value than the validation loss, as seen in Figure 4. The graph shows that the model in the training set has an output of 0.04 compared to the validation set, which outputs 0.008. This means that the nano

model is underfitted. If a model is underfitted, the model cannot learn the patterns in the training set. This would be solved by increasing the size of the dataset to help the model identify the trends of each class of objects.

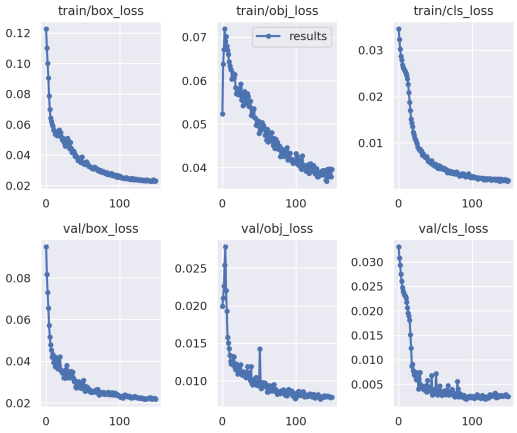

Fig. 4. Yolov5n validation and training loss results

B. YOLOv5s

The small model shows that the training and validation sets produce results similar to the nano model in terms of the loss trends, as seen in Figure 5. The small model is also under-fitted, as shown by the object loss graph. Since the model produces higher values with the training set, it shows a problem with the model not correctly detecting the difference between the object and the background within the box. For loss of objects, the small model produces values for the training set of 0.04, and the validation set is 0.008. An essential difference between the nano- and small-models is that the small model produces much lower values within the validation test.

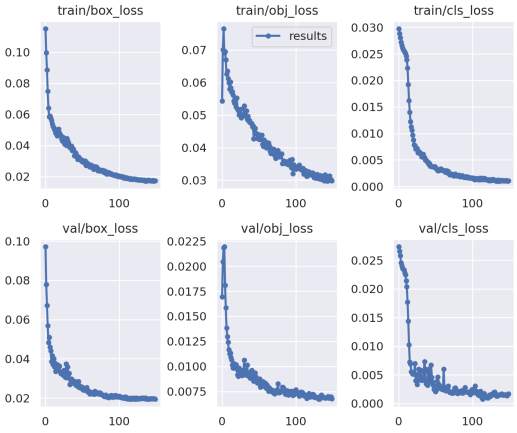

Fig. 5. Yolov5s validation and training loss results

VI. Performance Testing

There are various metrics used to measure the accuracy and performance of an object detection model. We used precision, recall, and mAP (when IOU are at 0.5 and 0.95). The nano model had 1,761,871 parameters and 4.2 GFLOPs, while the small one had 7,015,519 parameters and 15.8 GFLOPs. Both models ran 200 epochs with 213 layers and no gradients.

The precision graph shows the performance similarities between nano and small versions of YOLOv5. The precision for the nano model was 0.971, with the mask class at 0.977 and the glove class at 0.965. On the contrary, the small model had a precision of 0.978, with the mask class at 0.98 and the glove class at 0.956. Both curves maximized their logarithmic growth after 60 epochs and deviated less. High precision indicates a low false-positive rate. This means that the YOLOv5 models did not falsely detect masks or gloves when there were no masks in the images.

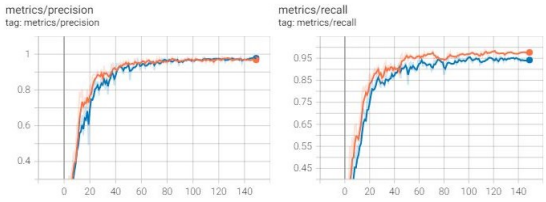

Fig. 6. Precision and Recall (blue - YOLOv5n, orange - YOLOv5s)

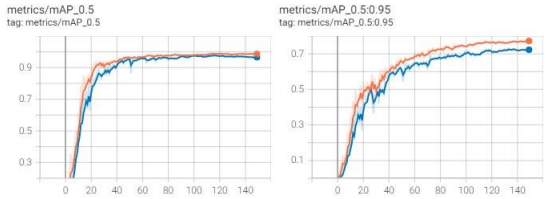

Fig. 7. mAP (blue - YOLOv5n, orange - YOLOv5s)

For recall, the nano model has an overall recall of 0.947, with mask class at 0.924 and glove class at 0.971. On the other hand, the small model has an overall recall of 0.969, with a mask class at 0.957 and a glove class at 0.982. A more significant difference between nano and small was observed from the recall graph. Small, represented by the orange line, had better recall than nano. Moreover, after 80 epochs, both versions fluctuated less. The high recall numbers are related to a low false-negative rate, which means that YOLOv5 did not leave masks or gloves undetected from the images.

The mAP detection metric at IOU = 0.5 shows that both nano and small versions of YOLOv5 are strong detectors. The mAP value for nano is 0.975, with the mask class at 0.972 and the glove class at 0.979. The mAP for small is 0.987, with mask class at 0.987 and glove class at 0.988. And for mAP at IOU = 0.95, the value of mAP for nano is 0.733 with the mask class at 0.704 and the glove class at 0.762. For small, mAP is 0.778 with mask class at 0.77 and glove class at 0.786.

VII. Discussion

After the experiment, we examined the result generated from our set of tests for the nano- and small models using the BlueROV2 ROV. The video was recorded using the 1080p, 30fps wide-angle low-light camera. Example images with detection are shown in Figures 9 and 10. As discussed previously, the trained model performed well, allowing the tethered ROV to use its CPU to detect masks and gloves in the ocean efficiently.

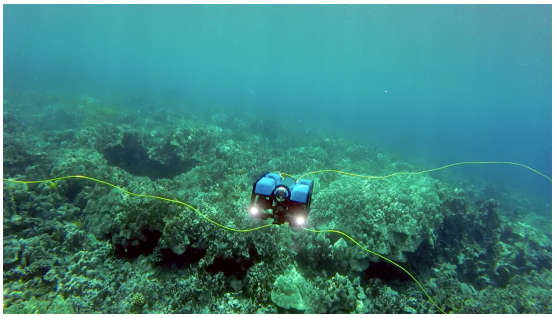

Fig. 8. Image of a BlueROV2, obtained from the official website.

Fig. 9. Mask detection with high confidence level

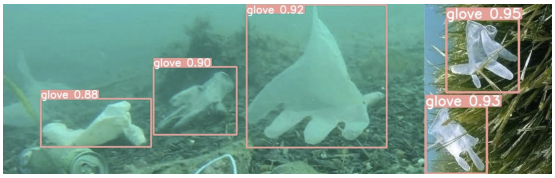

Fig. 10. Glove detection

VIII. Conclusion

In this short paper, we trained two YOLOv5 models, nano and small, using the same images. We then applied two models in ROV, observing no significant differences in mAP, precision, and recall. In future work, we plan to expand our research to detect other types of debris using ROVs, explore various authentication strategies for ROVs [15], as well as to use Autonomous Underwater Vehicles (AUVs) that conduct survey missions without operator intervention [16].

Acknowledgement

This work was supported by statutory funds from the Department of Computing and the Institute Research Grant from the British Columbia Institute of Technology.

References

[1] H. Anantharajah, K. Harika, A. Jayasinghe, and M. Aibin, "Covid-19 contact tracing using ble and rfid for data protection and integrity," in *2021 IEEE 12th Annual Ubiquitous Computing, Electronics Mobile Communication Conference (UEMCON)*, 2021, pp. 0190–0196.

[2] K. Selvaranjan, S. Navaratnam, P. Rajeev, and N. Ravintherakumaran, "Environmental challenges induced by extensive use of face masks during covid-19: A review and potential solutions," *Environmental Challenges*, vol. 3, p. 100039, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2667010021000184

[3] C. Xu, S. Nguyen, J. Whangbo, and M. Aibin, *A Hybrid Model for Sustainable Urban Metabolism in Metropolitan Communities*. Cham: Springer International Publishing, 2020, pp. 119–130.

[4] G. Tata, S.-J. Royer, O. Poirion, and J. Lowe, "Deepplastic: A novel approach to detecting epipelagic bound plastic using deep visual models," *arXiv preprint arXiv:2105.01882*, 2021.

[5] T. Contributor, "What is embedded device?" Apr 2014. [Online]. Available: https://whatis.techtarget.com/definition/embedded-device

[6] F. Lin, T. Hou, Q. Jin, and A. You, "Improved yolo based detection algorithm for floating debris in waterway," *Entropy*, vol. 23, no. 9, 2021. [Online]. Available: https://www.mdpi.com/1099-4300/23/9/1111

[7] Y. Liu, B. Lu, J. Peng, and Z. Zhang, "Research on the use of yolov5 object detection algorithm in mask wearing recognition," *World Scientific Research Journal*, pp. 276–284, 2020.

[8] P. Soviany and R. T. Ionescu, "Optimizing the trade-off between single-stage and two-stage deep object detectors using image difficulty prediction," in *2018 20th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC)*. IEEE, 2018, pp. 209–214.

[9] R. Xu, H. Lin, K. Lu, L. Cao, and Y. Liu, "A forest fire detection system based on ensemble learning," *Forests*, vol. 12, p. 217, 02 2021.

[10] J. Solawetz, "Yolov5 new version - improvements and evaluation," Sep 2021. [Online]. Available: https://blog.roboflow.com/yolov5-improvements-and-evaluation/

[11] S. Gutta, "Object detection algorithm-yolo v5 architecture," Aug 2021. [Online]. Available: https://medium.com/analytics-vidhya/object-detection-algorithm-yolo-v5-architecture-89e0a35472ef

[12] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "Yolov4: Optimal speed and accuracy of object detection," *arXiv preprint arXiv:2004.10934*, 2020.

[13] C.-Y. Wang, H.-Y. M. Liao, Y.-H. Wu, P.-Y. Chen, J.-W. Hsieh, and I.-H. Yeh, "Cspnet: A new backbone that can enhance learning capability of cnn," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, 2020, pp. 390–391.

[14] Ultralytics, "Releases · ultralytics/yolov5." [Online]. Available: https://github.com/ultralytics/yolov5/releases

[15] M. Karimibiuki, M. Aibin, Y. Lai, R. Khan, R. Norfield, and A. Hunter, "Drones' Face off: Authentication by Machine Learning in Autonomous IoT Systems," *2019 IEEE 10th Annual Ubiquitous Computing, Electronics and Mobile Communication Conference, UEMCON 2019*, pp. 0329–0333, 10 2019.

[16] M. Aibin, M. Aldiab, R. Bhavsar, J. Lodhra, M. Reyes, F. Rezaeian, E. Saczuk, M. Taer, and M. Taer, "Survey of RPAS Autonomous Control Systems Using Artificial Intelligence," *IEEE Access*, vol. 9, pp. 167 580–167 591, 2021.

View publication stats