Neural Networks and Deep Learning

Project Title: Underwater Image Instance Segmentation

Aya Zabalawi - g00089004

Khaled Mohamed - b00087968

Johnny Kortbawi - b00088581

1. Description of the Problem

Our research centers on using deep learning methods to tackle the problem of object and scene segmentation in underwater photos. Because of the effects of water, including light attenuation, color distortion, and reduced visibility, underwater imaging poses special challenges. Due to these difficulties, conventional image processing techniques are less successful, thus creating the need for the development of a more sophisticated solution.

Multiple sectors such as underwater robotics, environmental monitoring, marine biology, and other sectors depend heavily on underwater surroundings. However, there are a number of obstacles that we may encounter with underwater image analysis:

1. Water attenuates light by scattering and absorbing it, which reduces contrast and visibility. This causes objects and structures in underwater photos to often appear blurry or faded.
2. Water has the ability to distort color, causing color shifts and a loss of color information. This makes it difficult to distinguish between various materials and to identify objects.
3. Complex backgrounds can be created by marine life, vegetation, rocks, and other elements clogging underwater scenes. This makes it difficult to distinguish the objects of interest from the background.
4. Due to refraction and distortion, the water's depth can vary greatly, changing how objects appear.

Segmentation is a key component for deep learning-based underwater image analysis. By assigning a class or category to each pixel in an image, instance segmentation divides the image into regions that represent various classes while also distinguishing different instances of those classes existing within the same images. This enables us to efficiently differentiate between different objects and elements in the underwater scene.

2. Previous Approaches

Non-neural network-based approaches in the context of underwater image segmentation historically included threshold-based segmentation, edge detection, and region-based approaches. These techniques, while foundational, often struggled with the inherent challenges of underwater imagery, such as poor light conditions, varying degrees of visibility, and the complex, dynamic nature of underwater scenes. However, the recent trend strongly leans towards neural network-based methods due to their superior performance in handling the complexities of underwater imagery as neural networks excel at automatically learning and extracting complex features from images. Therefore, this report is solely focusing on implementing deep learning models to tackle the underwater image segmentation problem in order to explore the tradeoff between model size and model performance.

Paper	Authors	Models	Summary	Strengths	Weaknesses	Metrics
[1]	Paulo Drews-Jr, Isadora de Souza, Igor P. Maurell, Eglen V. Protas & Silvia S. C. Botelho	Semantic segmentation using SegNet and DeepLabv3+	This paper focuses on the task of underwater image segmentation using deep learning techniques. The authors explore fine-tuning and image restoration of both real and simulated data.	They were able to avoid creating a large and labeled dataset by transfer learning.Two models were used here (both DeepLabv3 and SegNet).	The dataset is of a relatively limited size and it relies on simulated data.	mIoU = 91.9%, FPS = 16.54
[2]	Bowen Cheng, Ishan Misra, Alexander G. Schwing, Alexander Kirillov, Rohit Gridhar	Mask2Former	A new architecture capable of addressing any image segmentation task (panoptic, instance, or semantic). The key components include masked attention, which extracts localized features by constraining cross-attention within predicted mask regions	Outperforms previous models like MaskFormer and K-Net in various metrics	Limitations in segmenting small objects and fully leveraging multi-scale features	AP = 43.7, mIoU = 57.7
[3]	Jinkang Wang,Xiaohui He ,Faming Shao,Guanlin Lu,Ruizhe Hu,Qunyan Jiang	ResNet + Encoder-Decoder architecture	The paper aims to address specific challenges associated with underwater image processing and segmentation, such as low-quality original images, blurred target edges, insufficient segmentation accuracy,	The image enhancement will be based on multi-spatial transformation is performed to improve image quality. This method is uncommon. FPS	FPS lower than DFANet. The network segments by pixel region, so some objects of different classes but similar shapes are difficult to distinguish.	mIoU = 68.3, OA = 79.4, FPS = 125

			and poor regional boundary segmentation effects.	higher than Deeplab V3+ and CANet		
[4]	Shijie Lian, Hua Li, Runmin Cong, Suqi Li, Wei Zhang, Sam Kwong	WaterMask R-CNN	The authors propose WaterMask for underwater instance segmentation on the UIIS dataset.	Higher mAP and AP metrics than Mask-R-CNN	WaterMask R-CNN experiences a 0.7AP lag compared to Mask R-CNN	mAP = 25.6, AP50 = 41.7, AP75 = 27.9, APS = 8.8, APM = 21.3, APL = 36.0
[5]	Md Jahidul Islam, Chelsey Edge, Yuyang Xiao, Peigen Luo, Muntaqim Mehtaz, Christopher Morse, Sadman Sakib Enan, and Junaed Sattar	SUIM-Net VGG and SUIM-Net RGB	The paper presents SUIM-Net, a fully convolutional encoder-decoder CNN model to perform semantic segmentation on underwater images.	Boundary Learning Loss was designed to avoid blurred localization of boundaries	The dataset contains matching low-resolution images taken with handheld cameras. It also contains images with significant quality degradation, high saturation, and high contrast.	mAP = 27.2
[6]	Fangfang Liu and Ming Fang	DeepLabv3+ model	It utilizes the DeepLabv3+ network, enhanced with a UCM module for image quality improvement and additional up-sampling layers for better feature and boundary retention.	Additional up-sampling layers in the decoder structure retain more target features and object boundary information.	very computationally expensive	mIoU = 64.65%
[7]	Yang Zhou, Jiangtao Wang, Baihua Li, Qinggang Meng, Emanuele Rocco, and Andrea Saiani	VGG-16 , a popular cnn	The study focuses on enhancing underwater scene semantic segmentation using a deep neural network. It emphasizes the network's effectiveness in faster convergence and training efficiency	Improved decoder architecture for better handling of lower-resolution feature maps.	none	mIoU = 74.4%

[8]	Nezla N. A. . Mithun Haridas T.P. . Supriya M.H. .	A u-net architecture based on CNN	The paper focuses on exploring underwater objects using semantic image segmentation with a U-Net based architecture.The authors highlight the importance of this study in understanding environmental effects and the richness of underwater ecosystems	High performance in terms of accuracy and segmentation quality, as indicated by the evaluation metrics.	High Memory Consumption: The U-shaped architecture, with its skip connections and concatenation of feature maps from the encoder to the decoder, can lead to a substantial increase in the memory footprint	IoU (Testing): 85.83%
[9]	Maryam Rahnemoonfar and Dugan Dobbs	. This model is specifically based on a combination of **dilated convolution, dense module, and inception.** However, the paper does not assign a specific name to this model.	This paper focuses on the development and application of a novel deep learning framework to perform semantic segmentation on underwater sonar imagery. The primary aim is to automate the extraction of potholes in such images, with testing conducted on a collection of images from Laguna Madre in Texas.	Innovative use of a combination of dilated convolution, dense module, and inception in a deep learning framework.	The framework lacks scalability to different types of sonar imagery or varied environmental conditions,	IoU = 78.48%
[10]	Not specified just University of Wuhan	U-Net network	This study focuses on semantic segmentation of coral images in underwater photogrammetry. It involves comparing different neural network models and fine-tuning the U-Net architecture for improved performance in coral image segmentation.	The paper's approach benefits from a comprehensive evaluation of multiple models, allowing for an informed selection of U-Net as the most suitable for fine-tuning	Fine-tuning and optimizing deep learning models, especially those as complex as U-Net, can often lead to increased computational demands, which might be a challenge for real-time or on-field applications.	IoU = 89.2%

[11]	Not specified just University of Toulon	SegNet model.	The study focuses on semantic segmentation for detecting and delineating biofouling regions in underwater imagery. The technique involves the use of synthetic imagery for training deep learning algorithms, followed by region enhancement using support vector machines (SVMs).	Successful detection and delineation of soft biofouling regions in most cases.	The approach may struggle with images containing small floating particles, as evidenced by misclassifications in one of the test images.	(IoU) = 87%
[12]	Andrew King, Suchendra M. Bhandarkar, and Brian M. Hopkinson	convolutional neural network (CNN) and and fully convolutional neural network (FCNN) models	This study compares deep learning methods for semantic segmentation of coral reef images, focusing on both patch-based CNNs and FCNNs. The Resnet152 CNN architecture is identified as the most effective among the CNNs, and Deeplab v2 is highlighted as the best FCNN model for semantic segmentation of coral reef images.	Comprehensive evaluation of multiple CNN and FCNN architectures.	The segmentation task is sensitive to downsampling, affecting the performance of some models.	Resnet152 (CNN), classification accuracy = 90% Deeplab v2(FCNNs), pixelwise accuracy = 67.7%
[13]	Yuanyuan Tian, Luyu Lan, Linna Sun	PCNN, CNN, and FCN	Reviews various sonar image segmentation methods, particularly focusing on threshold-based, Markov Random Field-based, clustering, and deep learning-based approaches, highlighting their applications, strengths, and limitations	Can accept images of any size and avoids repetitive storage and convolution problem	Poor universality and high computational demand	none

[14]	Xinnan Fan, Pengfei Cao, Pengfei Shi, Xinyang Chen, Xuan Zhou, Qian Gong	MA-AttUNet (Multi-level Adversarial - attention)	The paper proposes a transfer learning method called MA-AttUNet for underwater dam crack image segmentation. This method leverages knowledge transfer from a source domain to the target domain (underwater images) and uses an attention mechanism to eliminate noise interference	The multi-level adversarial transfer learning approach significantly reduces the need for labeled underwater samples.	The segmentation method might have limitations in terms of scalability and generalizability to diverse underwater environments or different types of underwater structures.	IoU = 20%
[15]	Peiyuan Jiang, Daji Ergu, Fangyao Liu, Ying Cai, Bo Ma	Yolov2, Yolov3, Yolov4, Yolov5, and Yolo-Lite	Overview of the YOLO algorithm and its advanced versions, discussing the differences and similarities among them and with Convolutional Neural Networks (CNNs), and underscoring the ongoing improvements in YOLO algorithm development	Small model size, fast calculation speed, straightforward structure	Poor performance in detecting objects that are very close together, low recall	Improving YOLO V3's AP and FPS by 10% and 12% respectively
[16]	Rachel Huang, Jonathan Pedoeem, Cuixian Chen	YOLO-Lite	YOLO-LITE is a real-time object detection model optimized for non-GPU computers, achieving significant speed with reasonable accuracy on the PASCAL VOC and COCO datasets	Demonstrates capability of shallow networks for fast non-GPU object detection applications. Achieves higher FPS compared to SSD MobileNet V1 and Tiny-YOLOv2	Significant drop in accuracy compared to original YOLO architecture	mAP = 33.81%, 21 FPS
[17]	Issam H. Laradji, Alzayat Saleh, Pau Rodriguez, Derek Nowrouzezahrai, Mostafa Rahimi Azghadi & David Vazquez	PL-FCN, LCFCN, A-LCFCN, A-LCFCN+PM	Paper proposes a novel segmentation network called Affinity-LCFCN that can be trained on images with point-level annotations. They show that their method outperforms standard point-level supervision	Significantly reduces the human effort needed for training data acquisition by using point-level supervision. A-LCFCN demonstrates better segmentation	Point-level annotations might not be as easily obtainable as image-level annotations, especially in images with many fishes.	mIoU = 0.749

			methods and fully-supervised methods when the annotation budget is fixed. They also evaluate their method on the SUIM dataset.	results than previous point-level segmentation methods.		
[18]	Muduo Xu1, Jianhao Su, and Yutao Liu	SAM, AquaSAM	The purpose of AquaSAM is to adapt the Segment Anything Model (SAM) to the underwater domain and create a versatile method for segmenting various underwater targets. AquaSAM aims to overcome the limitations of SAM in underwater image segmentation and improve segmentation performance in underwater images.	Outperforms the default SAM model	Can lead to incorrect segmentation results when multiple similar instances surround the segmentation target	DSC = 7.13%, mIoU = 8.27%
[19]	Adnan Haider, Muhammad Arsalan, Jiho Choi, Haseeb Sultan, Kang Ryoung Park	EFS-Net, MFAS-Net	Presents two deep learning models, EFS-Net and MFAS-Net, for robust and efficient underwater fish segmentation, demonstrating superior performance and computational efficiency on challenging datasets	Initial feature refinement and multi-level feature accumulation improve segmentation accuracy	None	mIoU = 76.42% on DeepFish and 92.0% on SUIM
[20]	Lin Li, Bo Dong, Eric Rigall, Tao Zhou, Junyu Dong, Geng Chen	ECD-Net	Presents ECD-Net, an effective marine animal segmentation model that outperforms state-of-the-art object segmentation models in complex underwater environments	Network outperforms 10 cutting-edge object segmentation models both qualitatively and quantitatively	None	28.5 FPS
[21]	Hanqi Zhang, Ming Li, Xiaotian Pan, Xinlin Zhang, Jiageng Zhong, Jiangying Qin	VGG-Net and SUIM-Net	A comprehensive survey of deep learning techniques for coral reef monitoring, encompassing random point annotation	Improved accuracy and efficiency in coral reef monitoring compared to traditional	Class imbalance and the segmentation of densely labeled data	F-Score and mIoU within 5% of top scores

			classification, semantic segmentation of sparsely and densely labeled data	methods.		
[22]	P. Muñoz-Benavent, J. Martínez-Peiró, G. Andreu-García, V. Puig-Pons, V. Espinosa, I. Pérez-Arjona, F. De la Gándara, A. Ortega	Faster R-CNN, YOLOv5x, Mask R-CNN	Impact of deep learning techniques on the automatic sizing of bluefin tuna, highlighting significant improvements in fish detection, segmentation, and sizing accuracy, especially using Mask R-CNN and PointRend module	Increases the number of fish measurements by up to 2.45 times with Mask R-CNN and the PointRend module, and by up to 3.5 times in the number of measurements per minute of computing time	None	Faster R-CNN, YOLOv5x, and Mask R-CNN showed AP scores of 0.806, 0.850, and 0.855 respectively
[23]	Jiangtao Wang, Baihua Li, Yang Zhou, Emanuele Rocco, Qinggang Meng	MobileNet V2	Novel underwater segmentation network, optimized for deployment on autonomous underwater vehicles (AUVs), offering a balance between high accuracy and efficient performance on embedded platforms	High segmentation accuracy on two different underwater segmentation datasets (Seagrass and SUIM)	None	mIoU = 88.63, F-Score = 93.93
[24]	Gordon Boer, Rajesh Veeramalli, Hauke Schramm	PSPNet, DeeplabV3	Adapts and evaluates DeepLabV3 and PSPNet for semantic segmentation of fish in challenging underwater conditions, showing that PSPNet, with its limited parameter set and high accuracy, is particularly effective for devices with constrained hardware	Achieves high pixel accuracy with significantly fewer parameter	None	PSPNet Accuracy = 96.8%, IoU = 73.8%. DeepLabv3 Accuracy = 96.2% and IoU = 69.9%
[25]	Pertiwang Sismananda, Maman Abdurohman, Aji Gautama Putrada	YoloV3 and Yolo-Lite	To compare the performance of two object recognition methods, YOLO-LITE and YOLOV3, in terms of accuracy and speed, particularly in the context of their	YOLO-LITE demonstrates much faster processing speeds, making it suitable for real-time applications on low-end systems,	YOLO-LITE has lower accuracy and detection rates compared to YOLOV3, and YOLOV3 is considerably slower and	YOLO-LITE IoU = 0.573. YOLOv3 IoU = 0.867

			implementation on a Raspberry Pi system using MotionEyeOS.	while YOLOV3 shows better accuracy and a higher detection rate	heavier on low-end systems	
[26]	Qi-Chao Mao, Hong-Mei Sun, Yan-bo Liu, Rui-Sheng Jia	YoloV3	The purpose of this paper is to improve onto YoloV3 so that it is suitable to be deployed on an embedded system.	It obtains suboptimal results that are comparable with YOLOV3	The speed is half that of YOLOV3	mAP-50 = 52.1, FPS = 67
[5]	Md Jahidul Islam, Chelsey Edge, Yuyang Xiao, Peigen Luo, Muntaqim Mehtaz, Christopher Morse, Sadman Sakib Enan, and Junaed Sattar	SUIM-Net VGG and SUIM-Net RGB	The paper presents SUIM-Net, a fully convolutional encoder-decoder CNN model to perform semantic segmentation on underwater images.	Boundary Learning Loss was designed to avoid blurred localization of boundaries	WaterMask RCNN experiences a 0.7AP lag compared to R-CNN	mAP = 27.2
[27]	Petr Hurtik, Vojtech Molek, Jan Hula, Marek Vajgl, Pavel Vlasanek, Tomas Nejezchleba	Poly-YOLO	The proposed model, Poly-YOLO, aims to improve the performance and accuracy of object detection by addressing the weaknesses of YOLOv3, such as label rewriting and anchor distribution. It introduces instance segmentation using bounding polygons and is trained to detect size-independent polygons defined on a polar grid.	Poly-YOLO achieves higher precision by using a single scale output with high resolution and aggregating features from a light backbone.	None	FPS = 37.1

3. Data Selection

We have selected our dataset from paper [4]. The reasoning behind this selection is that there is only one paper in the current literature that used this dataset for underwater instance segmentation. Furthermore, this paper fails to compare the Mask RCNN model, as well as their model named Watermask, to different YOLO segmentation algorithms, that have been known to yield great performance results. The Underwater Image Instance Segmentation (UIIS) dataset is comprised of 4628 underwater images that contain pixel-level annotations for seven different classes which are aquatic plants, fish, human divers, reefs, robots, sea-floor, and wreck-ruins. The annotations of the dataset exist as JSON annotations containing the image information such as height, width, ID, and filename. Moreover, they contained the segmentation masks as pixel coordinates along with the corresponding class ID, as well as the bounding box coordinates. This dataset offers us with a sufficient amount of images to execute the task of underwater image instance segmentation accordingly. If needed, augmentations can be performed on the images to increase the dataset size, improve the model's generalization by introducing different image transformations.

4. Data Cleaning and Feature Engineering

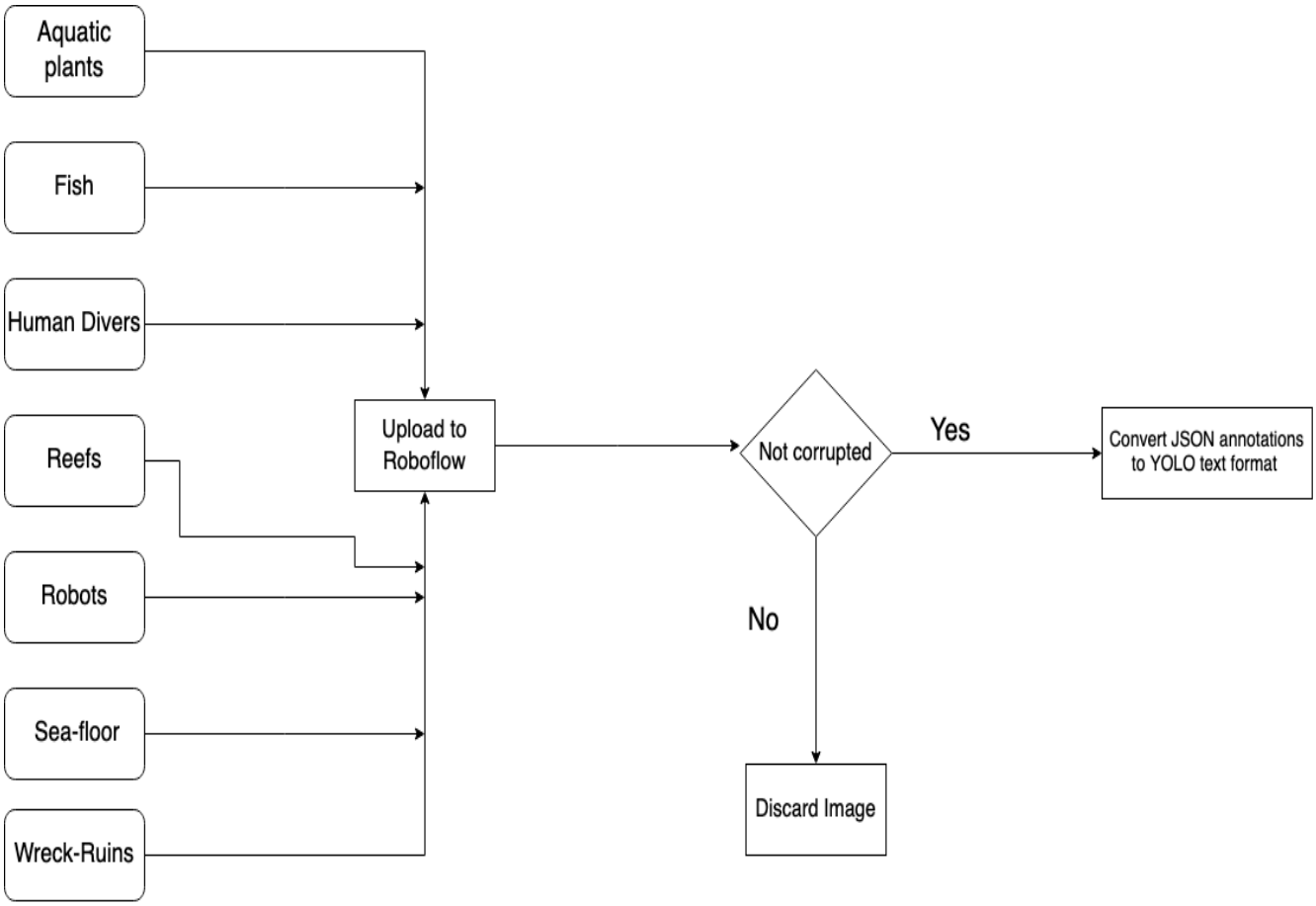
Data Cleaning Process:

1. **Initial Data Assessment:** The original dataset, comprising underwater images, came with annotations in JSON format. The first step involved a thorough assessment of this data for quality and consistency.
2. **Cleaning with RoboFlow:** We utilized RoboFlow for the primary data cleaning process. While the specifics of RoboFlow's internal cleaning mechanisms are not entirely transparent, it generally involves steps such as removing corrupt images, normalizing annotation formats, and filtering out unusable data.
3. **Quality Assurance:** Post-RoboFlow processing, we conducted a manual check to ensure data quality. This involved verifying a subset of images for correct annotations and ensuring no significant data loss or corruption during the cleaning process.

Feature Engineering Techniques:

1. **Annotation Format Transformation:** The original JSON annotations were transformed into YOLO text format. This conversion is crucial as YOLO models require specific annotation formats (classes, bounding box coordinates) that differ from the JSON structure.
2. **Standardization:** Image data was standardized to 640 x 640 to ensure pixel values were scaled appropriately for neural network processing. This step helps in reducing model training time and improves performance.
3. **Data Augmentation:**To enhance the robustness of our models and accurately mimic the diverse conditions encountered in underwater imaging, we employed a series of specialized data augmentation techniques such as blurring and noise,which are common visual distortions in underwater environments. This approach aimed to replicate the typical challenges faced in

underwater image analysis, such as variations in clarity and light diffusion. By incorporating these realistic distortions, our augmentation process significantly diversified our dataset



5. Neural Network Architecture Selection

Selection Rationale: Our project employs YOLO versions 5, 7, and 8 for underwater instance segmentation, specifically focusing on the Underwater Image Instance Segmentation (UIIS) dataset as referenced from the paper by Lian et al. (2023). This dataset is uniquely challenging due to its composition of 4628 underwater images with pixel-level annotations for various classes like aquatic

plants, fish, human divers, reefs, robots, sea-floor, and wreck-ruins. The YOLO models were chosen based on their renowned efficiency in performance and their adaptability to diverse data environments, as seen in previous works.

Previous research has demonstrated the effectiveness of YOLO models in underwater environments. For instance, a comparative study utilized YOLOv3, SSD, and SIFT-based models for target detection in underwater settings, providing insights into the adaptability and accuracy of YOLO models in such challenging conditions. However, most of the work in the literature did not include the use of YOLO in image segmentation, which motivated us to explore it in this problem.This study's results indicate that YOLO models are capable of handling the complexities of underwater imagery, making them a suitable choice for our project.

YOLOv5: Selected for its high-speed performance, making it ideal for real-time detection in variable underwater conditions. Its balance between speed and accuracy suits the diverse and dynamic nature of the UIIS dataset.

YOLOv7: Chosen for its improved accuracy over YOLOv5 and efficient training capabilities. This version strikes a perfect balance between speed and precision, which is vital for processing the complex scenes found in underwater imagery.

YOLOv8: Opted for its state-of-the-art accuracy and advanced features. It provides high precision in segmenting intricate underwater objects, a crucial requirement given the detailed nature of the UIIS dataset.

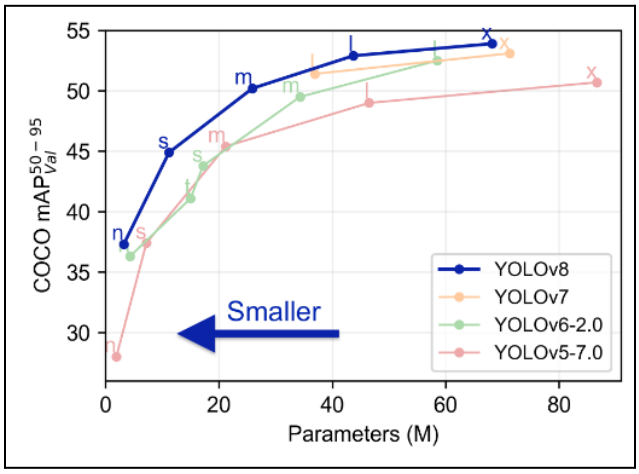


Figure 1 - YOLO mAP vs Size Comparison

Model	Characteristics	Kernel
YOLOv5	While not as accurate as the latest YOLO models, YOLOv5 still offers a reliable balance between speed and accuracy.	[28]
YOLOv7	YOLOv7 offers a significant improvement in detection accuracy over its predecessors, crucial for precise object identification.	https://github.com/WongKinYiu/yolov7
YOLOv8	YOLOv8 is known for its high accuracy, especially in detecting small and intricate objects, crucial for detailed tasks like instance segmentation.	[28]
Mask R-CNN	Mask R-CNN excels in producing pixel-level segmentation, making it ideal for tasks that require precise delineation of object boundaries. It detects objects and also provides a segmentation mask for each instance, distinguishing different objects of the same class.	[29]

6. Why do chosen NN Architectures Work

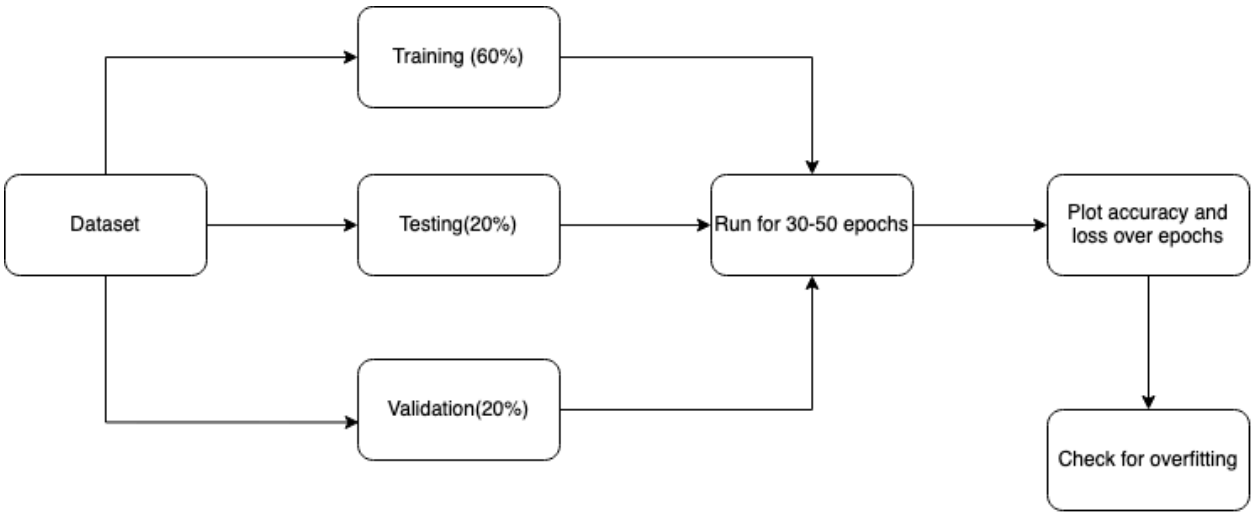
Model	Justification	Kernel
YOLOv5	Suitable for real-time detection in variable underwater conditions. Also, demonstrates strong performance across various environmental conditions, which is essential for dealing with the varying visibility and complex backgrounds found in underwater imagery.	[28]
YOLOv7	YOLOv7 is designed for efficiency, allowing for faster model training and real-time inference. This makes it suitable for processing large datasets like the UIIS dataset without compromising on speed.	https://github.com/WongKinYiu/yolov7
YOLOv8	YOLOv8's enhanced architecture excels at capturing fine details in underwater imagery, crucial for distinguishing subtle variations among similar elements like various aquatic plants or corals.	[28]
Mask-R CNN	Provides high precision in producing pixel-level accurate masks in segmenting intricate underwater objects. Its architecture allows for flexibility in handling various image qualities	[29]

	and complexities often found in underwater environments, like varying lighting conditions, turbidity, and occlusions.	

7. Validation Methodology

Data Split and Preparation: Our project employed a straightforward yet effective validation methodology. The dataset was split into training, validation, and testing sets with a proportion of 60/20/20. This approach ensured that 60% of the data was used for model training, 20% for validation (to tune the models and prevent overfitting), and the remaining 20% for testing (to assess model

performance). This distribution was chosen to maintain a balance between sufficient training data and adequate evaluation capacity.



Training, Validation, and Testing Data Generation:

1. Data Preprocessing: Initially, the entire dataset underwent preprocessing, which included normalization and augmentation techniques to enhance model training.
2. Data Splitting: Following preprocessing, the data was randomly divided into the respective training, validation, and testing sets.

Exclusion of K-Fold Testing: While K-fold cross-validation is a robust method for model validation, it was not employed in this project due to significant time constraints. K-fold testing, which involves dividing the dataset into 'K' folds and using each fold as a test set at different iterations, is a time-consuming process. Given the large size of our dataset and the computational demands of training YOLO models, this method was deemed impractical for our project timeline.

Metrics for Model Evaluation: We focused on several key metrics from **Roboflow** to evaluate the performance of our models:

1. **Mean Average Precision (mAP):** It combines two concepts: precision and recall, and then computes the mean of these values over different classes or over different levels of intersection over union (IoU) thresholds. To calculate mAP, the average precision for each class is calculated separately. Then, the mean of these average precision values across all classes. This calculated mean represents the mAP.
2. **Loss Curves (per Epoch)**: We plotted loss curves for both training and validation phases on an epoch-by-epoch basis. These curves depict the model's loss, or error, over each epoch, providing insights into the learning process. A downward trend in the training loss curve suggests improvement in model performance, while the validation loss curve is crucial for identifying overfitting, particularly if it starts going in one direction while the training loss continues to go in another direction.

3. **Recall:** Recall measures the ratio of correctly predicted positive instances to the total number of actual positive instances

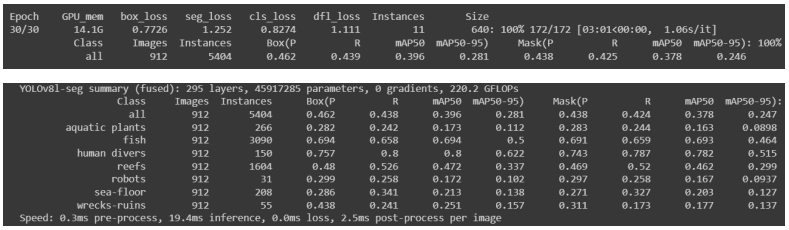
   Recall = $\frac{TP}{TP+FN}$

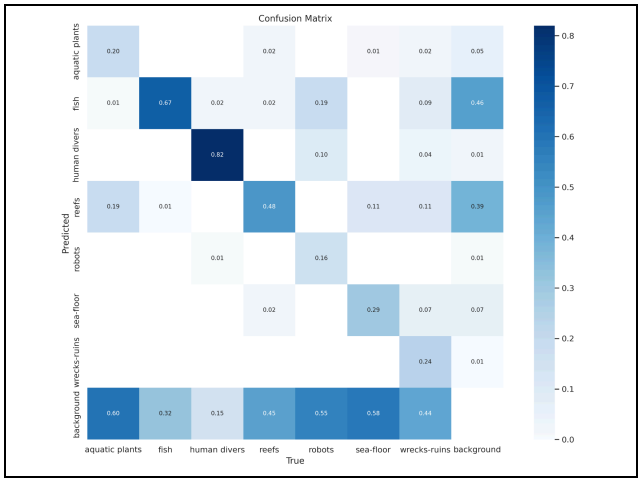
4. **Precision:** Precision measures the ratio of correctly predicted positive instances to the total number of predicted positive instances

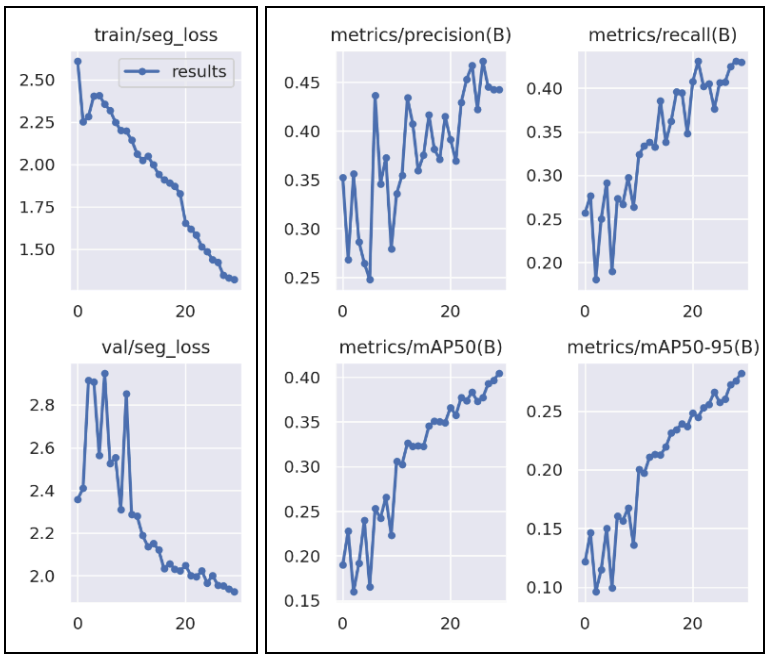
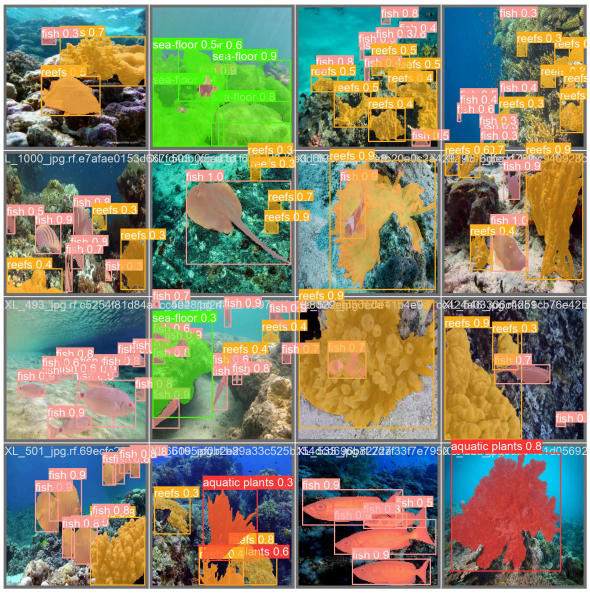
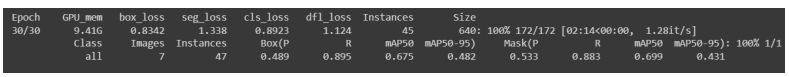
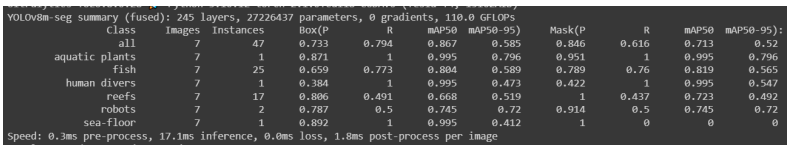
   Precision = $\frac{TP}{TP+FP}$

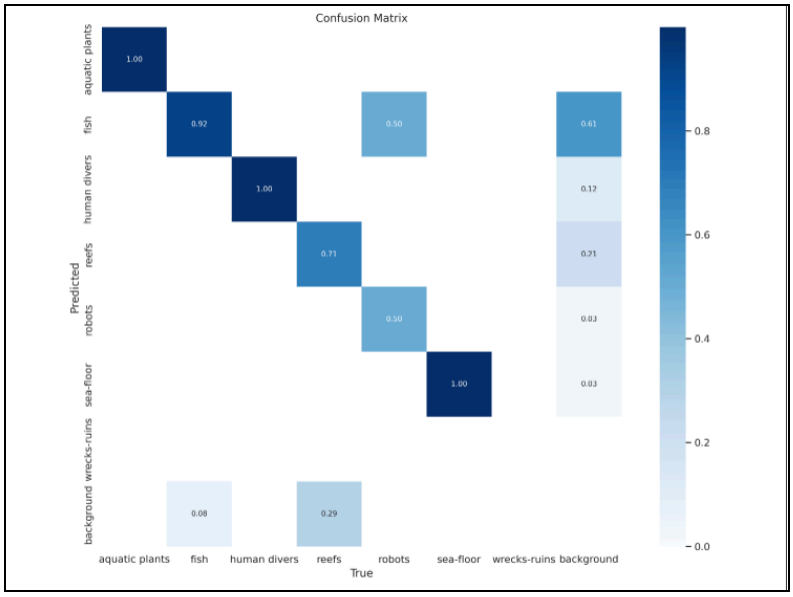
Exclusion of Grid Search: In the initial planning of our project, we recognized the potential benefits of employing grid search for hyperparameter optimization. This technique, known for its systematic approach to testing various combinations of model parameters, could have been very useful in determining the most effective configuration for our YOLO models. Our goal was to explore different settings, such as learning rates, layer configurations, and other hyperparameters, to identify the optimal setup that yields the best performance on our dataset. However, due to significant time constraints and the extensive computational resources required for such an exhaustive approach, we were unable to implement grid search in our validation methodology. This decision was made to prioritize the timely completion of the project, despite the acknowledged advantage grid search could have provided in fine-tuning our models to their best possible performance.

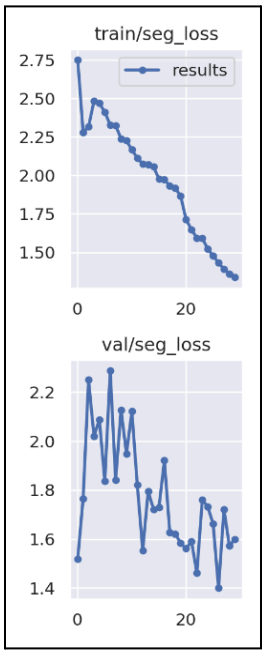
8. Results

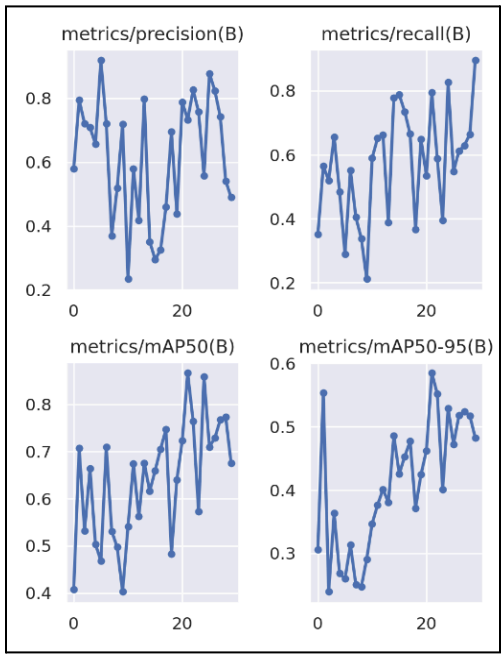
Architecture	Graphs and Results	Explanation
YOLOv8 (without augmented data)		From the second table on the left, it is evident that this model is best at identifying the fish and human diver classes. This can be attributed to their more distinguishable features compared to the other classes such as sea-floor and wreck ruins that have similar features, which are easily mistaken as background. For that reason, the precision, recall of fish and human divers is significantly higher compared to the other classes. Moreover, as illustrated in the confusion matrix the model is more suited for creating masked maps of fish (true positive proportion of 0.67) and human divers (true positive proportion of 0.82). From the bottom row of the confusion matrix it can be deduced that the model confuses the six classes as background. This can be

		due to the similarities in pixel values caused by the way depth is perceived visually in underwater images. From the left most, the loss graphs are decreasing over the epochs indicating that the model is learning and improving its segmenting abilities. The train loss graph is decreasing more steadily while the validation loss fluctuates at the beginning. Despite this, the model does not seem to be overfitting greatly and therefore is able to adequately generalize the different types of underwater image instance segmentation. However, the precision and recall fluctuate to a greater degree over the epochs. Suggesting that the model may be inconsistent with predicting specific cases or rare instances. This fluctuation may also be a result of the model not being able to identify certain classes when they blend with the rest of the background in the images. The final graph on the bottom illustrates what the predicted instance segmentation masks look like.

	 	
YOLOv8l (with augmented data)	 	These results are from using augmented data on the YOLOv8l model. The augmentations include blur and noise. The idea behind them was to replicate underwater visibility. From the tables on the right, it is evident that this model achieved higher mAP50 (0.699), precision (0.533), recall (0.883) when compared to the previously



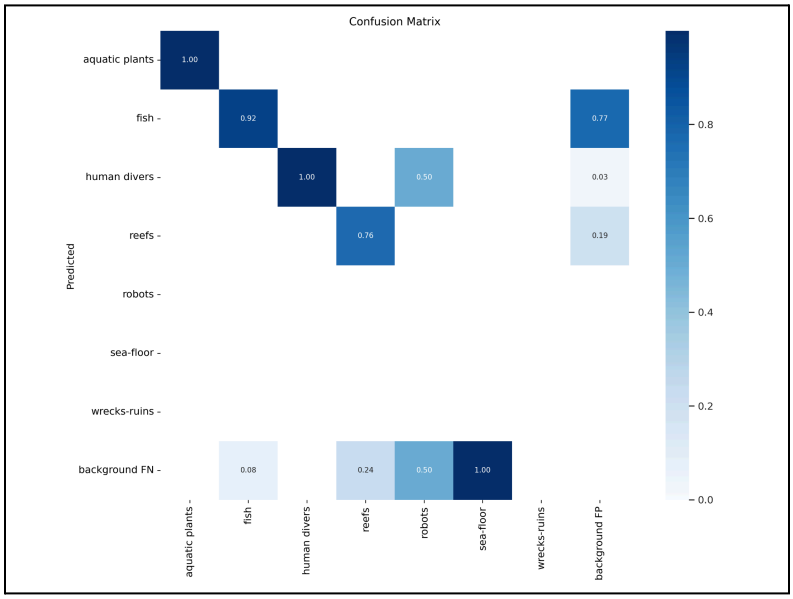


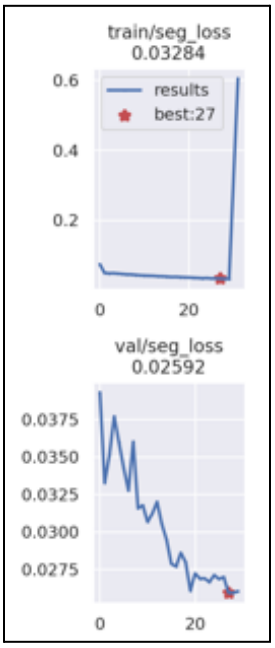


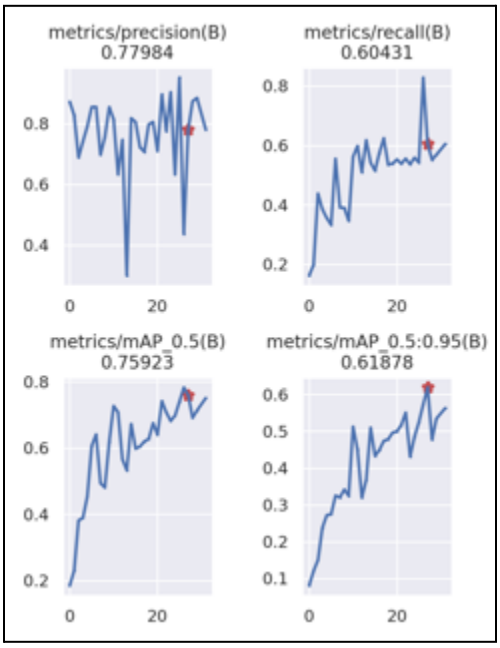
mentioned model. However, this does not imply that the model performed better. From the confusion matrix we can deduce from the diagonal that the classes (except for wreck-ruins) have a strong proportion of true positive predictions. This would be perceived as good however the loss curves below indicate that there may be overfitting.Although the train loss decreases gradually, the validation loss fluctuates throughout the epochs. The behavior of the validation loss deviates from the training suggesting that the model may be overfitting. The same goes for the precision, recall, and mAP curves. While the model achieves a high mAP50 (0.699), precision (0.533), and recall (0.883), the fluctuation in these metrics over the epochs suggests that the model's performance is not stable. This instability could be due to overfitting or could indicate that the model is highly sensitive to certain types of augmentations or variations in the data.

YOLOv7 (augmented data)

```
Epoch    GPU_mem   box_loss   seg_loss   obj_loss   cls_loss  Instances       Size
29/29      13.2G    0.02533    0.03192    0.05393   0.007396        118        640: 100% 172/172 [03:21<00:00,  1.17s/it]
           Class     Images  Instances      Box(P          R      mAP50  mAP50-95)     Mask(P          R      mAP50  mAP50-95): 100% 1/1
             all          7         47      0.885      0.568      0.713      0.534      0.892      0.576      0.719      0.466
```

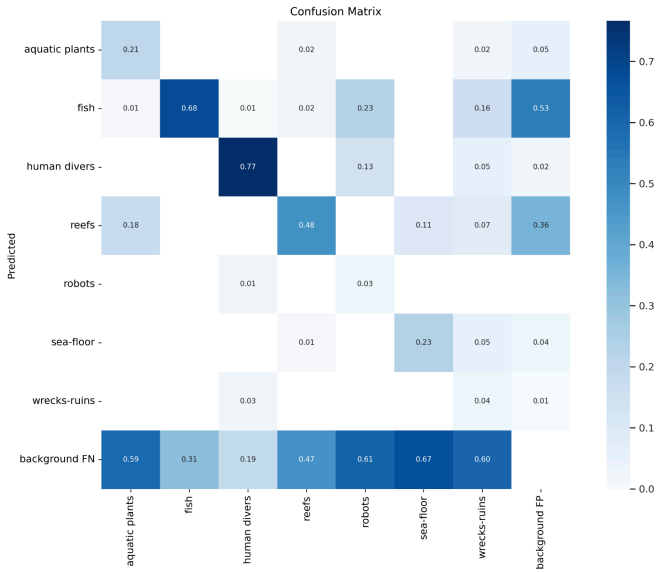
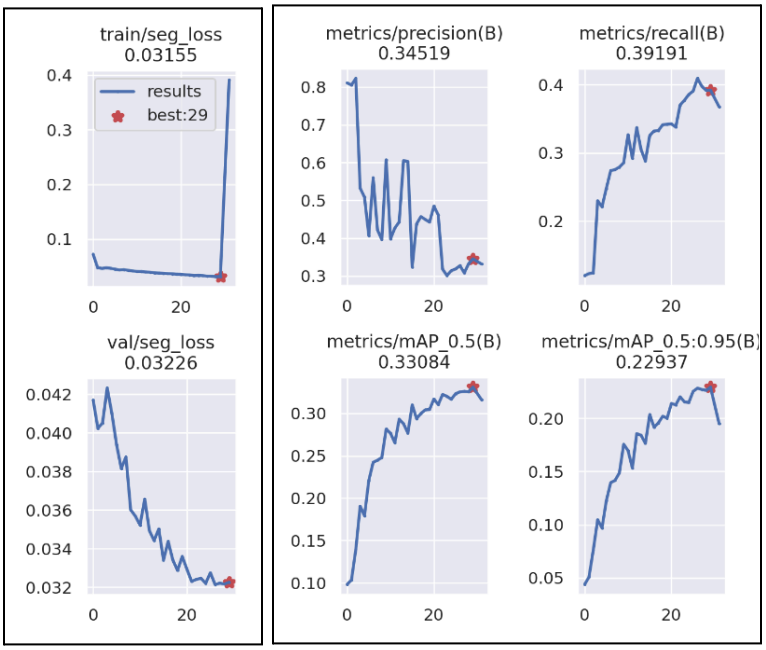


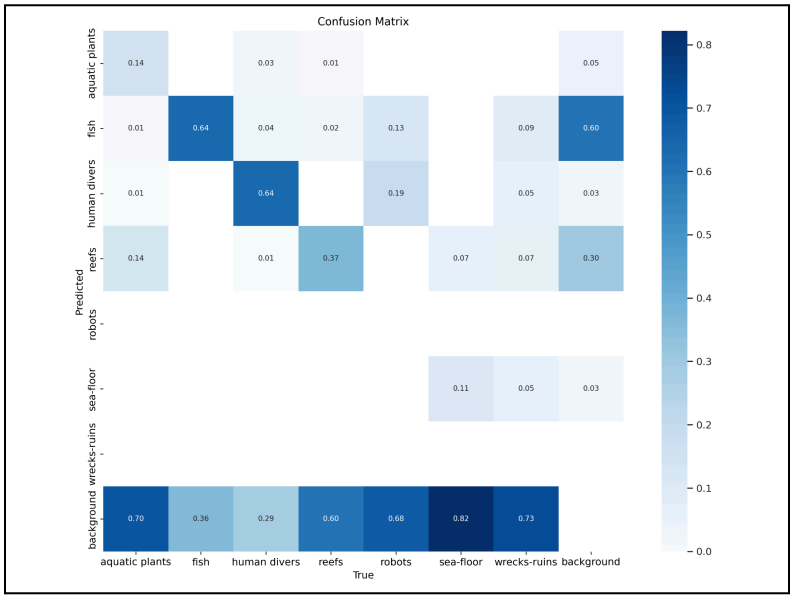


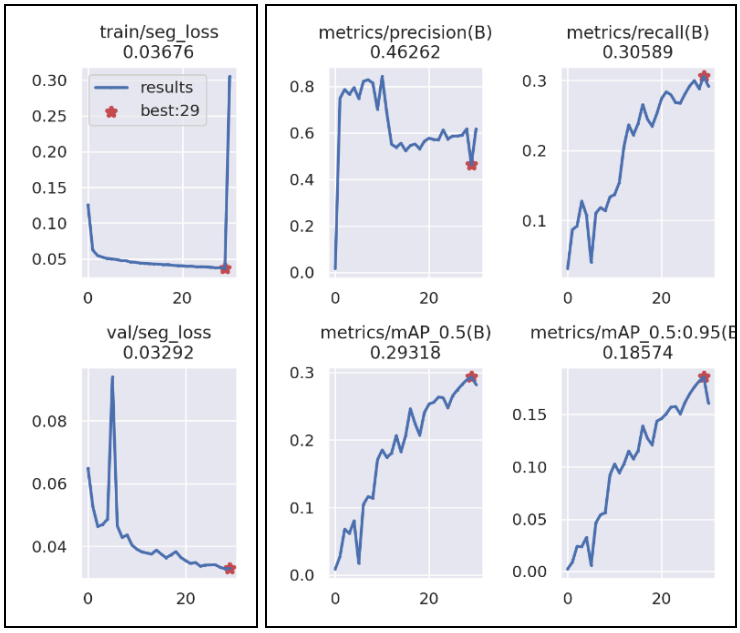


Certain misclassifications are visible in the confusion matrix, such as fish being mistaken for reefs and vice versa. There may be space for improvement in the model's capacity to discern objects from the background, as evidenced by the background false negatives (FN) and positives (FP), which show cases in which the background is mistakenly classified as an object or vice versa. The precision and recall values across various intersections over union (IoU) thresholds, as well as the training and validation losses, are displayed on the graphs. A strong indication of learning is the model's good convergence as the loss is declining, as indicated by the train/seg_loss and val/seg_loss graphs. There are fluctuations in the precision and recall graphs, which could mean that the model performs differently at different IoU thresholds.

Two graphs representing mAP at IoU=0.5 and mAP at IoU ranging from 0.5 to 0.95 are created from the mAP metrics. The mAP values are generally high, indicating good model performance; however, the performance decreases when examining the more stringent IoU threshold, which is between 0.5 and 0.95. All things considered, these findings point to a model that does well on some classes and metrics but may benefit from additional training or hyperparameter tuning to lower misclassification rates and boost accuracy and recall at more strict IoU thresholds.

Yolov7 (unaugmented data)	  Epoch GPU_mem box_loss seg_loss obj_loss cls_loss Instances Size 29/29 13.1G 0.02487 0.03155 0.05455 0.007425 24 640: 100% 172/172 [03:25<00:00, 1.19s/it] Class Images Instances Box(P R mAP50 mAP50-95) Mask(P R mAP50 mAP50-95): all 912 5404 0.345 0.392 0.331 0.229 0.331 0.367 0.316 0.195	The following conclusions can be drawn from the confusion matrix: The 0.59 at the bottom of its column indicates that aquatic plants have a high false negative rate, meaning the model frequently misses them. Fish are often misidentified as background, as evidenced by the 0.53 at the bottom of the fish column, although they are correctly identified 68% of the time. Although the true positive rate for human divers is reasonably high (0.77), there is a noticeable degree of background confusion (0.31). Reefs are frequently mislabeled fish or background, as seen by their respective column values of 0.23 and 0.36. Robots, sea-floor, and wrecks-ruins have a lower presence in the data, but there is still some confusion between these categories and the background. High values at the bottom row indicate that the model often incorrectly labels background as other categories, suggesting the model struggles to distinguish between background and objects. From the graphs, we can deduce that: The validation loss also decreases, indicating the model generalizes well to unseen data. The validation loss also decreases, indicating the model generalizes well to unseen data. Precision fluctuates significantly, which could indicate an imbalance in the dataset or that the model is inconsistent in its predictions across classes. Recall steadily increases, which suggests the model is getting better at identifying relevant objects in the data over time. The mean average precision at an IoU threshold of 0.5 is around 0.33, which is low, suggesting the model is not very accurate in its predictions. The mAP across a range of IoU thresholds from 0.5 to 0.95 is even lower, around 0.22, indicating that

		precision drops further when stricter localization criteria are applied.
YOLOv5s	Epoch GPU_mem box_loss seg_loss obj_loss cls_loss Instances Size 29/29 35.2G 0.03343 0.03676 0.05947 0.009961 517 640: 100% 22/22 [00:30<00:00, 1.40s/it] Class Images Instances Box(P R mAP50 mAP50-95) Mask(P R mAP50 mAP50-95): 100% all 912 5404 0.463 0.306 0.293 0.186 0.617 0.292 0.282 0.161 Model summary: 165 layers, 7414604 parameters, 0 gradients, 25.7 GFLOPs Class Images Instances Box(P R mAP50 mAP50-95) Mask(P R mAP50 mAP50-95): 100% all 912 5404 0.606 0.305 0.293 0.186 0.617 0.292 0.282 0.161 aquatic plants 912 266 0.274 0.207 0.159 0.0893 0.256 0.173 0.14 0.0689 fish 912 3090 0.619 0.651 0.636 0.409 0.631 0.639 0.625 0.374 human divers 912 150 0.585 0.733 0.708 0.482 0.606 0.693 0.681 0.392 reefs 912 1604 0.492 0.405 0.382 0.236 0.507 0.394 0.377 0.212 robots 912 31 1 0 0.0331 0.0174 1 0 0.0316 0.015 sea-floor 912 208 0.276 0.139 0.109 0.0558 0.316 0.144 0.105 0.0554 wrecks-ruins 912 55 1 0 0.0245 0.0105 1 0 0.0152 0.00729  	From the confusion matrix, we can infer that: Aquatic plants are occasionally mistaken for other categories, with a true positive rate of 0.14. Fish are correctly identified 64% of the time, but there is a notable false positive rate with the background (0.60), indicating that background is often misclassified as fish. Human divers are correctly classified 64% of the time, but also have a significant false positive rate with the background (0.29). Reefs are identified correctly 37% of the time, with some confusion with fish (0.14) and the background (0.30). The background class has the highest false negative rates across all categories, particularly with fish, indicating that the model is struggling to distinguish between the background and fish. From the graphs, we can infer that: The model's training and validation loss is decreasing, suggesting that the model is learning and generalizing well. Precision, while fluctuating, is relatively low at around 0.46, indicating that when the model predicts a class, it's correct less than half the time. Recall is also low, at around 0.31, suggesting the model is missing a significant number of positive cases. Mean average precision at an IoU threshold of 0.5 is about 0.29, which is low and suggests the model does not predict bounding boxes very well. mAP over a range of IoU thresholds is even lower at around 0.19, indicating precision drops as the localization criteria get stricter.


train/seg_loss
0.03676
results
best:29
metrics/precision(B)
0.46262
metrics/recall(B)
0.30589
val/seg_loss
0.03292
metrics/mAP_0.5(B)
0.29318
metrics/mAP_0.5:0.95(B)
0.18574

9. Discussion and Future Work

Given that YOLOv8 is the most advanced model released to date and has been refined with Roboflow, it produced the best results when used without augmented data. Our findings show that different YOLO architectures can be used successfully for the task of underwater image segmentation. Specifically, YOLOv8 (without augmented data) performed the best overall, which can be ascribed to its sophisticated architecture and the Roboflow fine-tuning procedure. YOLOv8's high degree of accuracy indicates that it has been able to accurately distinguish between various classes and capture the intricate patterns found in underwater imagery. Even though the outcomes are encouraging, there is still space for development. The fact that the augmentation of data did not result in the anticipated increase in our model performance suggests that we may need to modify or adjust the intensities of our augmentation techniques. The augmentation techniques ought to more accurately represent the various visibility conditions, color distortion, and motion blur that exist in underwater environments. In addition, a more thorough hyperparameter optimization procedure might be advantageous for the models; this could involve applying Bayesian optimization techniques, which have the potential to be more effective than grid search. In the future, we plan to build on these successes by concentrating on YOLOv8 implementation on a Raspberry Pi in the future. This step is a major step towards real-world, practical applications, especially when it comes to implementing AI-driven image segmentation in hardware that is small, affordable, and easily accessible.

Raspberry Pi Deployment:

1. **Optimization for Edge Devices**: Adapting YOLOv8 for efficient performance on the Raspberry Pi, a resource-constrained device. This will involve model optimization techniques like pruning, quantization, and knowledge distillation to reduce the model's size and computational demand without substantially compromising accuracy.
2. **Real-Time Image Processing**: Integrating a camera module with the Raspberry Pi to enable real-time image capture and processing. This setup will allow the Raspberry Pi to collect underwater images and immediately process them through the YOLOv8 model, facilitating instant segmentation and analysis.

In conclusion, our project has laid the groundwork for using YOLO models in underwater image segmentation. The next steps involve refining the data augmentation process, exploring new model architectures, and preparing the models for deployment in field applications. By continuing this line of research, we aim to enhance the precision of underwater object detection and contribute valuable tools to the fields of marine research and robotics.

10. Youtube Video Demo Link

https://youtu.be/H-38VIVHBDE?si=V-D7DkkKhxWZ6TaI

References

[1] P. Drews-Jr, I. D. Souza, I. P. Maurell, E. V. Protas, and S. S. C. Botelho, “Underwater image segmentation in the wild using deep learning,” *J. Braz. Comput. Soc.*, vol. 27, no. 1, p. 12, Dec. 2021, doi: 10.1186/s13173-021-00117-7.

[2] B. Cheng, I. Misra, A. G. Schwing, A. Kirillov, and R. Girdhar, “Masked-attention Mask Transformer for Universal Image Segmentation.” arXiv, Jun. 15, 2022. Accessed: Nov. 23, 2023. [Online]. Available: http://arxiv.org/abs/2112.01527

[3] J. Wang, X. He, F. Shao, G. Lu, R. Hu, and Q. Jiang, “Semantic segmentation method of underwater images based on encoder-decoder architecture,” *PLOS ONE*, vol. 17, no. 8, p. e0272666, Aug. 2022, doi: 10.1371/journal.pone.0272666.

[4] S. Lian, H. Li, R. Cong, S. Li, W. Zhang, and S. Kwong, “WaterMask: Instance Segmentation for Underwater Imagery”.

[5] M. J. Islam *et al.*, “Semantic Segmentation of Underwater Imagery: Dataset and Benchmark,” in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Las Vegas, NV, USA: IEEE, Oct. 2020, pp. 1769–1776. doi: 10.1109/IROS45743.2020.9340821.

[6] F. Liu and M. Fang, “Semantic Segmentation of Underwater Images Based on Improved Deeplab,” *J. Mar. Sci. Eng.*, vol. 8, no. 3, p. 188, Mar. 2020, doi: 10.3390/jmse8030188.

[7] Y. Zhou *et al.*, “Underwater Scene Segmentation by Deep Neural Network,” presented at the UK-RAS19 Conference: “Embedded Intelligence: Enabling and Supporting RAS Technologies,” Mar. 2019, pp. 44–47. doi: 10.31256/UKRAS19.12.

[8] N. A. Nezla, T. P. Mithun Haridas, and M. H. Supriya, “Semantic Segmentation of Underwater Images using UNet architecture based Deep Convolutional Encoder Decoder Model,” in *2021 7th International Conference on Advanced Computing and Communication Systems (ICACCS)*, Coimbatore, India: IEEE, Mar. 2021, pp. 28–33. doi: 10.1109/ICACCS51430.2021.9441804.

[9] M. Rahnemoonfar and D. Dobbs, “Semantic Segmentation of Underwater Sonar Imagery with Deep Learning,” in *IGARSS 2019 - 2019 IEEE International Geoscience and Remote Sensing Symposium*, Yokohama, Japan: IEEE, Jul. 2019, pp. 9455–9458. doi: 10.1109/IGARSS.2019.8898742.

[10] H. Zhang, A. Gruen, and M. Li, “DEEP LEARNING FOR SEMANTIC SEGMENTATION OF CORAL IMAGES IN UNDERWATER PHOTOGRAMMETRY,” *ISPRS Ann. Photogramm. Remote Sens. Spat. Inf. Sci.*, vol. V-2–2022, pp. 343–350, May 2022, doi: 10.5194/isprs-annals-V-2-2022-343-2022.

[11] M. O’Byrne, V. Pakrashi, F. Schoefs, and B. Ghosh, “Semantic Segmentation of Underwater Imagery Using Deep Networks Trained on Synthetic Imagery,” *J. Mar. Sci. Eng.*, vol. 6, no. 3, p. 93, Aug. 2018, doi: 10.3390/jmse6030093.

[12] A. King, S. M. Bhandarkar, and B. M. Hopkinson, “A Comparison of Deep Learning Methods for Semantic Segmentation of Coral Reef Survey Images,” in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, Salt Lake City, UT: IEEE, Jun. 2018, pp. 1475–14758. doi: 10.1109/CVPRW.2018.00188.

[13] Y. Tian, L. Lan, and L. Sun, “A Review of Sonar Image Segmentation for Underwater Small Targets,” in *Proceedings of the 2020 International Conference on Pattern Recognition and Intelligent Systems*, Athens Greece: ACM, Jul. 2020, pp. 1–4. doi: 10.1145/3415048.3416098.

[14] X. Fan, P. Cao, P. Shi, X. Chen, X. Zhou, and Q. Gong, “An underwater dam crack image segmentation method based on multi-level adversarial transfer learning,” *Neurocomputing*, vol. 505, pp. 19–29, Sep. 2022, doi: 10.1016/j.neucom.2022.07.036.

[15] P. Jiang, D. Ergu, F. Liu, Y. Cai, and B. Ma, “A Review of Yolo Algorithm Developments,” *Procedia Comput. Sci.*, vol. 199, pp. 1066–1073, 2022, doi: 10.1016/j.procs.2022.01.135.

[16] R. Huang, J. Pedoeem, and C. Chen, “YOLO-LITE: A Real-Time Object Detection Algorithm Optimized for Non-GPU Computers,” in *2018 IEEE International Conference on Big Data (Big Data)*, Seattle, WA, USA: IEEE, Dec. 2018, pp. 2503–2510. doi: 10.1109/BigData.2018.8621865.
[17] I. H. Laradji, A. Saleh, P. Rodriguez, D. Nowrouzezahrai, M. R. Azghadi, and D. Vazquez, “Weakly supervised underwater fish segmentation using affinity LCFCN,” *Sci. Rep.*, vol. 11, no. 1, p. 17379, Aug. 2021, doi: 10.1038/s41598-021-96610-2.
[18] M. Xu, J. Su, and Y. Liu, “AquaSAM: Underwater Image Foreground Segmentation”.
[19] A. Haider, M. Arsalan, J. Choi, H. Sultan, and K. R. Park, “Robust segmentation of underwater fish based on multi-level feature accumulation,” *Front. Mar. Sci.*, vol. 9, p. 1010565, Oct. 2022, doi: 10.3389/fmars.2022.1010565.
[20] L. Li, B. Dong, E. Rigall, T. Zhou, J. Dong, and G. Chen, “Marine Animal Segmentation,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 32, no. 4, pp. 2303–2314, Apr. 2022, doi: 10.1109/TCSVT.2021.3093890.
[21] H. Zhang, M. Li, X. Pan, X. Zhang, J. Zhong, and J. Qin, “NOVEL APPROACHES TO ENHANCE CORAL REEFS MONITORING WITH UNDERWATER IMAGE SEGMENTATION,” *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.*, vol. XLVI-3/W1-2022, pp. 271–277, Apr. 2022, doi: 10.5194/isprs-archives-XLVI-3-W1-2022-271-2022.
[22] P. Muñoz-Benavent *et al.*, “Impact evaluation of deep learning on image segmentation for automatic bluefin tuna sizing,” *Aquac. Eng.*, vol. 99, p. 102299, Nov. 2022, doi: 10.1016/j.aquaeng.2022.102299.
[23] J. Wang, B. Li, Y. Zhou, E. Rocco, and Q. Meng, “Compact and Fast Underwater Segmentation Network for Autonomous Underwater Vehicles,” in *Computer Vision – ACCV 2020*, vol. 12624, H. Ishikawa, C.-L. Liu, T. Pajdla, and J. Shi, Eds., in Lecture Notes in Computer Science, vol. 12624. , Cham: Springer International Publishing, 2021, pp. 688–703. doi: 10.1007/978-3-030-69535-4_42.
[24] G. Böer, R. Veeramalli, and H. Schramm, “Segmentation of Fish in Realistic Underwater Scenes using Lightweight Deep Learning Models:,” in *Proceedings of the 2nd International Conference on Robotics, Computer Vision and Intelligent Systems*, Online Streaming, --- Select a Country ---: SCITEPRESS - Science and Technology Publications, 2021, pp. 158–164. doi: 10.5220/0010712700003061.
[25] P. Sismananda, M. Abdurohman, and A. G. Putrada, “Performance Comparison of Yolo-Lite and YoloV3 Using Raspberry Pi and MotionEyeOS,” in *2020 8th International Conference on Information and Communication Technology (ICoICT)*, Yogyakarta, Indonesia: IEEE, Jun. 2020, pp. 1–7. doi: 10.1109/ICoICT49345.2020.9166199.
[26] Q.-C. Mao, H.-M. Sun, Y.-B. Liu, and R.-S. Jia, “Mini-YOLOv3: Real-Time Object Detector for Embedded Applications,” *IEEE Access*, vol. 7, pp. 133529–133538, 2019, doi: 10.1109/ACCESS.2019.2941547.
[27] P. Hurtik, V. Molek, J. Hula, M. Vajgl, P. Vlasanek, and T. Nejezchleba, “Poly-YOLO: higher speed, more precise detection and instance segmentation for YOLOv3.” arXiv, May 29, 2020. Accessed: Nov. 23, 2023. [Online]. Available: http://arxiv.org/abs/2005.13243
[28] “Ultralytics,” GitHub. Accessed: Nov. 24, 2023. [Online]. Available: https://github.com/ultralytics
[29] K. He, G. Gkioxari, P. Dollar, and R. Girshick, “Mask R-CNN,” in *2017 IEEE International Conference on Computer Vision (ICCV)*, Venice: IEEE, Oct. 2017, pp. 2980–2988. doi: 10.1109/ICCV.2017.322.