# Underwater Object Detection Using Transfer Learning: Research Project

1st Geethu Girish
ggirish@uwaterloo.ca

2nd Snigdha Kakkar
s2kakkar@uwaterloo.ca

3rd Vinitha Raj RajaGopal Muthu
vr3rajag@uwaterloo.ca

4th Heidi Ye
s22ye@uwaterloo.ca

***Abstract*—Underwater object detection is one of the active areas of research in the past decade. Several countries have been investing in newer technologies to monitor marine life to gauge the level of marine pollution. This is monitored by studying the changing behavioural patterns of marine life. To study the marine life population and their adapting behaviours, it is essential to detect marine objects and classify them. Some of the challenges in marine object detection are: low visibility, increased noise, low contrast, blurred edges, black patches, and color deviation. The aim of this project is to explore and analyze transfer learning techniques to achieve this objective of detecting marine objects and classifying them. This project draws on the brackish dataset proposed by Pederson et al. [1] and uses two unique models: Yolov4 and EfficientDet. Later, the results of the two techniques have been evaluated and compared with each other to draw some interesting insights.**

***Index Terms*—Underwater images, Deep Learning, object detection, Yolov4, EfficientNet, EfficientDet**

## I. Introduction

One of the several necessities of life for humans is water. Owing to the increasing aquatic pollution from the three key sources: industrial, agricultural and urbanization [2], the presence of hazardous material in the water bodies is increasing at high levels. This marine pollution impacts each life form which relies on it for survival including humans. The marine beings such as fish, crabs depict changing behaviors and patterns for survival in such an environment. The Heath [3] indicates that most of these changing behaviors have regular patterns which include rapid swimming, floating, breathing patterns, circuitous frequency, and motor behaviors. These need to be monitored which require some technology to capture, and detect these marine lifeforms.

This project is a step in this direction to detect and classify these marine lifeforms. Although object detection using transfer learning is not a new technology and is actively used for real-time situations, but water as a medium is quite different from air as a medium when it comes to locating objects. The images taken underwater have a blurry effect due to light scattering, absorption and dispersion which occurs within water. Thus, the question we are trying to solve using our transfer learning techniques is: *"How can we detect and classify marine lifeforms in brackish water using existing deep learning techniques?"*

### A. Brackish Water Dataset

Pederson et al. [1] proposed Brackish water dataset in 2019. It is one of the first few datasets that has been made in estuarine environment. This dataset has been captured under the bridge Limfjordbroen in Limfjord (a 170 km shallow section of sea between North Sea and Kattegat). This data was collected using 3 cameras and 3 lights on the whole. We obtained this dataset from Pederson et al. [1], which has also been posted on Kaggle and Roboflow. The dataset comprises six distinct marine lifeforms, namely: Big Fish, Small Fish, Crab, Jellyfish, Shrimp and Starfish. The image frames of these marine lifeforms came from 89 videos recorded in this dataset as depicted on Kaggle. [4]. The annotated images of the six classes have been shown in Figure 1. Some of the challenges in object detection for this dataset were: blur, changing light conditions, dark patches, overlapping objects and animals, occlusions, floating objects such as seaweed, varying angles for animals, camouflaged animals.

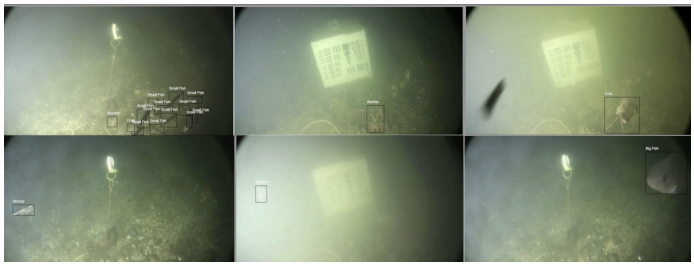

Fig. 1. Six categories in Brackish Water Dataset [1]

### B. Literature Review

As the computer vision and optical imaging has advanced, there has been a considerable amount of work in marine object detection. Autonomous underwater vehicles (AUV's) and Unmanned Underwater Vehicles (UUV's) are a few applications of these technologies. Normal object detection differs from underwater object detection as underwater images suffer a lot of blurriness due to light attenuation within water.

In [5], conventional techniques have been compared with deep learning based methods and it has been found that the deep learning based methods perform better than the conventional CV technique along with HOG and Support Vector Machine classifier. In [6], corals are detected using VGGNet. Another interesting work [7] proposes a new model ZooPlanktoNet (an inspiration from AlexNet and VGGNet). The paper [8] explores the impact of considering image enhancement and deep learning object detectors together for marine object detection.

### C. YOLO

**YOLO (You Only Look Once)** is one of the new deep learning algorithms which was first introduced in 2016 for object detection by Redmond et al. [9] The improved version was introduced in the paper: "YOLO9000: Better, Faster, Stronger" [10] . The latest improved version YOLOv8 in the market was brought by Ultralytics [11]. Now, the question is how is YOLO different from other algorithms for object detection. The answer is quite simple. YOLO makes it possible to use only one neural network for predicting the bounding boxes and class probabilities in just one run over one image. Previous approaches such as R-CNN were mostly object classifiers which were repurposed for object detection. They basically worked on sliding window approach. Complex pipelines were necessary with the previous algorithms as each component needed to be trained separately whereas YOLO could be trained as a simple regression problem.
YOLOv4 has a quite high inference time when compared to YOLOv3 and v2. It is basically a one-stage detector with several components in it as shown in Figure 2.

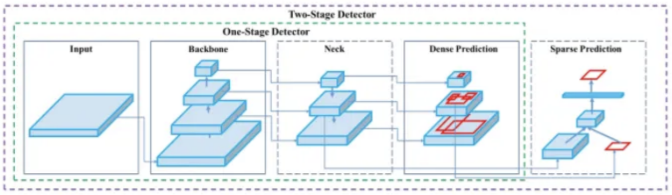


Fig. 2. Components for YOLOV4 [12].

The various building blocks of YOLOv4 are as follows:

1) **Input**: Image, patches, Pyramid
2) **Backbone**: VGG16, ResNet-50, SpineNet, EfficientNet-B0-B7, CSPResNext50, CSPDarknet53.
3) **Neck**:
   - Additional Blocks: SSP, ASPP, RFB, SAM
   - Path-aggregation blocks: FPN, PAN, NAS-FPN, Fully-connected FPN, BiFPN, ASFF, SFAM
4) **Heads**:
   - Dense prediction (one-stage):
     - RPN, SSD, YOLO, RentinaNet (anchor-based)
     - CornerNet, CenterNet, MatrixNet (anchor-free)
   - Sparse prediction:
     - Faster R-CNN, R-FCN, Mask R-CNN (anchor-based)
     - RepPoints (anchor-free)

The best thing for YOLOv4 is that it can be implemented in any combination of input, backbone, neck, and head. The YOLOv4 architecture is shown in Figure 3. In the figure, CSPDarkNet-53 is used as the backbone in YOLOv4 to extract features from input images. SPP and PANet are used as the neck and YOLOv3 as the head.

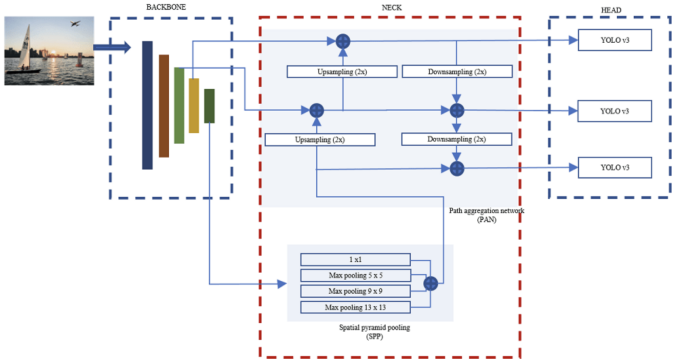


Fig. 3. Architecture for YOLOV4 [13].

### D. EfficientDet

The EfficientDet architecture was developed by combining 3 stages of pre-existing networks [14] [15]- EfficientNet and the Weighted Bi-directional Feature Pyramid Network, BiFPN and the Box/Class Prediction Network. The EfficientNet is used as the backbone network while the BiFPN is used as the feature network. EfficientNets are Object Detection models which themselves are built from pretrained ImageNets [16]. Pretrained EfficientNets from versions EfficientNet-b0 consecutively till EfficientNet-b7 are used in this architecture to leverage on these networks' pretrained checkpoints, making it appropriate for underwater object detection [17] [18] [19]. The BiFPN is an efficient bidirectional cross-scale connections and weighted feature fusion network. As illustrated in Fig. 4, the BiFPN layer is repeated several times to enhance the feature fusion by applying top-down and bottom-up feature fusions.

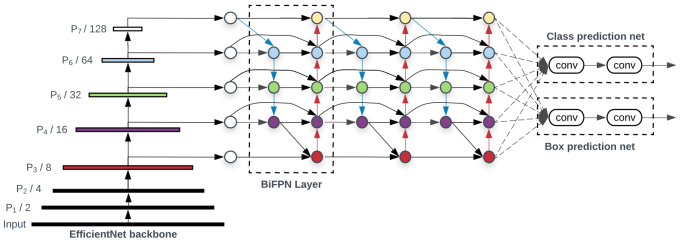


Fig. 4. Architecture for EfficientDet [15].

## II. Approach: Preprocessing & Methodology

### A. YOLO

The entire dataset comprised of 14674 images from those 89 videos captured using the three cameras. The animals are annotated in YOLO v3 Darknet format.
The following pre-processing was applied to each image:

- Auto-orientation of pixel data (with EXIF-orientation stripping).
- Resize to 960x540 (Stretch).
- No image augmentation techniques were applied.

- The split (80/10/10) done in training, valid and test dataset was such that they comprised of 11739, 1467 and 1468 images respectively.
- The number of iterations used for training the model were 13000.
- The default anchors were used for training: learning_rate=0.001, batch=64, max_batches = max(6000, number_of_training_images, 2000*classes).
- The subdivisions were changed to 64 to fasten the training process for YOLOv4.
- The default configuration is trained for 80 classes in COCO dataset which was changed to 6 as there were only 6 categories of animals in the brackish water dataset. The number of filters was also changed to 33 accordingly.

### B. *EfficientDet*

For the scope of this project, the EfficientDet-d0 model from the Google AutoML codebase was used [20] by modifying it to output the mAP values for each run [21]. The same dataset using the same split ration of images for training, valid and test as used with YOLOv4 was used. In order to feed this dataset, a few steps were taken:

*1) Data Preprocessing:* The same preprocessing steps that were applied before being fed into YOLOv4 was applied, therefore theinput images were of dimensions 960x540.

*2) Conversion to TF records:* The images that were used from the dataset were in the PNG format. To be fed into the chosen EfficientDet model, the images were first converted into the TensorFlow TF version, using the Roboflow converter [22].

*3) Hyperparameters:* The hyperparameters that were chosen for EfficientDet are described in Table 1. When a batch size of 64 was used, the model kept terminating and therefore the train and batch size had to be reduced to 2 to ensure that the model ran smoothly.

TABLE I
HYPERPARAMETERS FOR EFFICIENTDET ON UNDERWATER IMAGES [21]

| Hyperparameter | Value |
|---|---|
| Model | EfficientDet-d1 |
| Number of epochs | 100 |
| Max Instances per Image | 100 |
| Learning Rate | 0.08 |
| Optimizer | Stochastic Gradient Descent |
| Evaluation Batch Size | 2 |
| Train Batch Size | 2 |

## III. EXPERIMENTS & RESULTS

### A. *Evaluation measures*

For the purpose of comparing the two methods: EfficientDet and YOLOv4 for Brackish Water dataset, we would need to define certain set of common metrics. For Computer Vision field, specific metrics have been accepted for use in Object detection. One of those key metrics is **Average Precision (AP)**. This metrics has also been used popularly by two very famous datasets: COCO [23] and Pascal VOC [24]. The two building blocks for Average Precision metrics are: **Confusion Matrix** and **Intersection Over Union (IoU)**. IoU can be defined as the ratio of ground truth bounding box to the predicting bounding box [25]. Finally, a threshold is applied to this particular ratio. Pascal Dataset considered this threshold to be 50 percent. For computing this IoU, one should have two components: Ground truth bounding box obtained from test data, Predicted bounding box obtained after learning from the model.

The average precision can be obtained by the area under the Precision-Recall curve. **Mean Average Precision (mAP)** is computed as AP obtained for each class in the dataset divided by the number of classes.

### B. *Results*

The results from YOLOv4 are depicted in Table 2.

TABLE II
RESULTS WITH THE BEST WEIGHTS FOR YOLOV4

| Metrics | Result |
|---|---|
| **mAP** | 83.7 |
| **BigFish** | 89.89 |
| **Small Fish AP** | 92.5 |
| **Shrimp AP** | 82 |
| **Crab AP** | 76.5 |
| **JellyFish AP** | 62.1 |
| **StarFish AP** | 97 |
| **Loss** | 0.32 |
| **Average Recall** | 0.85 |

The loss curve for YOLOv4 is shown in the below figure.

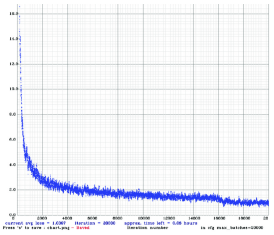

Fig. 5. Loss Curve for YOLOv4.

The results obtained from the EfficientDet-d1 are as below.

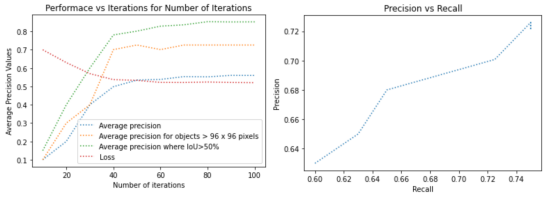


Fig. 6. Performance Metrics for EfficientDet-d1 on Underwater Images Dataset

The results for EfficientDet have also been summarized in Table 3.

The resulting images for YOLOv4 and EfficientDet are shown in Fig. 8, where the trained model is able to identify larger objects such as a fish and a crab with high confidence

TABLE III
RESULTS AT FINAL EPOCH []

| Metrics | Result |
|---|---|
| **mAP** | 55.9 |
| **AP for objects** $< 32^2$ **pixels** | 41.37 |
| **AP** $32^2$ **pixels** $\leq$ **objects** $\leq 96^2$ **pixels** | 66.24 |
| **AP for objects** $> 92^2$ **pixels** | 72.59 |
| **AP, IoU=50**% | 80.2 |
| **AP, IoU=75**% | 58.001 |
| **Loss** | 0.52040815 |
| **Average Recall** | 0.6285396 |

levels, and an AP of 72.59%. However, for smaller objects, the detection accuracy of EfficentDet is much lower, with an AP of 41.37%. Due to this fact, and since the underwater dataset consists of a higher proportion of smaller obejcts such as small fish and starfish, the overall mAP by the EfficientDet is pulled down.

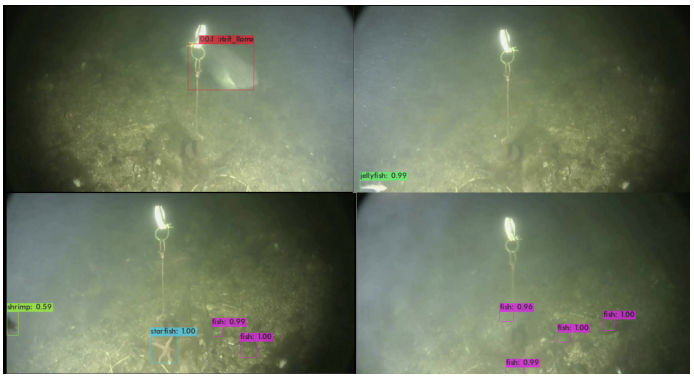

Fig. 7. Resultant Images for YOLOv4.

Fig. 8. Resultant images from trained EfficientDet-d1 on Underwater Dataset

From the results obtained for EfficientDet and YOLOv4 on the brackish water dataset, we can safely conclude that the mAP values and loss were significantly better in YOLOv4. YOLOv4 obtained the highest AP as well as mAP. Thus, YOLOv4 outperformed EfficientDet for object detection.

## IV. LIMITATIONS & FUTURE WORK

There is an improved Brackish water dataset, named as The New Brackish X [25] Dataset as proposed by the same researchers at Aalborg University. This dataset was better than Brackish dataset as it captured frames from a new camera at a completely different angle. The annotation errors in the original brackish water dataset have also been corrected in this new dataset. Thus, one of the future work could be done is to compare the object detection methods on this new dataset. The work can also be improved by comparing the latest versions of YOLO, such as YOLOv7 or YOLOv8 with not only EfficientDet but also with R-CNN, ASFF, NAS-FPN, CenterNet, CornerNet, etc. Due to time constraints we were able to train and compare two state of the art algorithms but for a larger perspective we can train and compare the above mentioned algorithms. Lastly, before drawing conclusions for algorithms' performance, we should try the algorithms on different datasets as one algorithm might outperform another on a unique dataset whereas that might not be true for another dataset.

## V. CONCLUSION

Both YOLOv4 and EfficientDet are state of the art object detection models. They both were easy to use and implement for this dataset using Google Colab. The mAP obtained using YOLOv4 was 83.7 whereas it was 55.9 for EfficientDet for this dataset while utilizing the same hyperparameters. Both the models thus did well on the dataset but YOLOv4 outperformed EfficientDet in terms of accuracy. We found that EfficientDet was slightly faster in terms of training time when compared with YOLOv4. The size of each model in terms of the size of weights when compared, it was evident that EfficientDet was smaller with far fewer parameters (4 million) when compared with YOLO (over 65 million) parameters. To summarize, the EfficientDet model performed decently in terms of speed of training and number of parameters as compared to YOLOv4 but YOLOv4 was better in terms of accuracy and loss outcomes for marine object detection in the Brackish water dataset.

## REFERENCES


[1] Pedersen et al. "Detection of Marine Animals in a New Underwater Dataset with Vary- ing Visibility". In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops. 2019.

[2] Yacout MHM. Sources of Water Pollution and Procedures Necessary for the Prevention of Pollution (A Review). Journal Of Dairy, Veterinary & Animal Research. 2016.

[3] Alan G. Heath. Water Pollution and Fish Physiology. United States of America : CRC Press, 384 p. ISBN 9780262038034. 2019.

[4] Gudiksen, M., & Hassen, L. N. (2023, February 22). The Brackish Dataset. Kaggle. https://www.kaggle.com/datasets/aalborguniversity/brackish-dataset

[5] Villon et at. Coral Reef Fish Detection and Recognition in Underwater Videos by Supervised Machine Learning: Comparison Between Deep Learning and HOG+SVM Methods. ACIVS: Advanced Concepts for Intelligent Vision Systems, Lecce, Italy. ffhal-01374123. 2016.

[6] Mahmood et al. "Coral classification with hybrid feature representations". In: Sept. 2016, pp. 519–523. doi: 10.1109/ICIP.2016.7532411.

[7] Dai et al. "ZooplanktoNet: Deep convolutional network for zooplankton classification". In: OCEANS 2016 - Shanghai. 2016, pp. 1–6.

[8] Xu, S., Zhang, M., Song, W., Mei, H., He, Q., & Liotta, A. (2023). A systematic review and analysis of deep learning-based underwater object detection. Neurocomputing, 527(0925-2312), 204-232. https://www.sciencedirect.com/science/article/pii/S0925231223000656

[9] Redmon et al. "You Only Look Once: Unified, Real-Time Object Detection". In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2016, pp. 779–788. doi: 10.1109/CVPR.2016.91.

[10] Joseph Redmon and Ali Farhadi. "YOLO9000: Better, Faster, Stronger". In: arXiv preprint arXiv:1612.08242 (2016).

[11] Rosebrock, A. (2016, November 7). Intersection over Union (IoU) for object detection. PyImageSearch. https://pyimagesearch.com/2016/11/07/intersection-over-union-iou-for-object-detection/

[12] Bhalerao, C. (n.d.). YOLO v8! The real state-of-the-art? Medium. https://medium.com/mlearning-ai/yolo-v8-the-real-state-of-the-art-eda6c86a1b90

[13] (n.d.). Getting Started with YOLO v4. Mathworks. https://www.mathworks.com/help/vision/ug/getting-started-with-yolo-v4.html

[14] Wednesday, A., & Vision, A. (n.d.). Efficientdet: Towards scalable and Efficient Object Detection. Retrieved March 29, 2023, from https://ai.googleblog.com/2020/04/efficientdet-towards-scalable-and.html

[15] Tan, M., Pang, R., & Le, Q. V. (2020). EfficientDet: Scalable and efficient object detection. 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). doi:10.1109/cvpr42600.2020.01079

[16] Zylo117. (n.d.). ZYLO117/yet-another-efficientdet-pytorch: The PYTORCH re-implement of the official efficientdet with Sota Performance in real time and pretrained weights. Retrieved March 29, 2023, from https://github.com/zylo117/Yet-Another-EfficientDet-Pytorch

[17] Jia, J., Fu, M., Liu, X., & Zheng, B. (2022). Underwater object detection based on improved Efficientdet. Remote Sensing, 14(18), 4487. doi:10.3390/rs14184487

[18] Li, J. (2020, November 03). Retraining Efficientdet for high-accuracy object detection. Retrieved March 29, 2023, from https://blog.ml6.eu/retraining-efficientdet-for-high-accuracy-object-detection-961e906cae8

[19] Kumar, A. (2020, April 14). Custom object detection using efficientdet-the simplest way. Retrieved March 29, 2023, from https://pub.towardsai.net/custom-object-detection-using-efficientdet-the-simplest-way-32749fb93359

[20] Google. (n.d.). Automl/efficientdet at master · google/automl. Retrieved March 29, 2023, from https://github.com/google/automl/tree/master/efficientdet

[21] Xuannianz. (n.d.). XUANNIANZ/Efficientdet: EfficientDet (scalable and efficient object detection) implementation in Keras and tensorflow. Retrieved March 29, 2023, from https://github.com/xuannianz/EfficientDet

[22] How to convert tensorflow object detection CSV to tensorflow TFRecord. (n.d.). Retrieved March 29, 2023, from https://roboflow.com/convert/tensorflow-object-detection-csv-to-tensorflow-tfrecord

[23] Lin et al. "Microsoft COCO: Common Objects in Context". In: Computer Vision - ECCV 2014. Springer International Publishing, 2014, pp. 740–755. isbn: 978-3-319-10602-1.

[24] Everingham et al. The PASCAL Visual Object Classes (VOC) Challenge. 2009.

[25] Stranovsky, L. (2020). Classification and Localization of Sea Animals in Brackish Waters with Diverse Visibility Using Deep Learning [Master's Thesis, Aalborg University]. https://projekter.aau.dk/projekter/files/333575561/VGIS10_Lukas_Stranovsky.pdf