Computer Vision CS-GY 6643 - Final Project - FathomNet Fine-Grained Visual Categorization

Chhatrapathi Sivaji Lakkimsetty cl7203@nyu.edu, Suemy Inagaki si2324@nyu.edu,
Felipe de Oliveira fd2264@nyu.edu, Gordon Lu gl1589@nyu.edu

Codebase :
Wavelet transformation for Image processing
Training pipeline and Out of sample detetction

1 Introduction and background

Although machine learning models already exist to classify data from underwater cameras, these models have limited ability to generalize to new situations. Some possible causes include changes in lighting, the use of different cameras, the emergence of unknown organisms, and variations in the appearance of the seafloor. (*The FathomNet2023 Competition Dataset* 2023).

In this context, we used the underwater image dataset made available by *The FathomNet2023 Competition Dataset* 2023, which contains annotated and unannotated data from several marine species and from different ocean depths. The dataset is described in the next section. Our ultimate goal is to correctly classify animals in the images, either by identifying them in existing categories or by detecting samples outside of these previously established categories.

To support our work, we explored related literature in three main areas. First, we searched for studies that address the classification of challenging oceanic or biological images, with a focus on the identification of marine organisms in complex and dynamic environments. We then investigate research dealing with image classification under different lighting conditions, one of the issues for improving the robustness of models in underwater scenarios where light varies significantly with depth. Finally, we review studies dealing with the classification of previously unseen species, with an emphasis on techniques for detecting samples outside the training distribution.

1.1 Classification of biological images

Fish recognition using convolutional neural network, (Ding et al. 2017): This work explores the use of convolutional neural networks (CNNs) for fish recognition. The authors present the challenges due to distortions, overlaps, and occlusions present in digital images. They designed several CNN architectures to address these challenges and performed a series of experiments to find the best performing configuration. Although specific to fish, our problem presents similar challenges.

Deep Learning for Marine Species Recognition, (Xu et al. 2019): This study provides a comprehensive review of the use of deep learning in marine species recognition, covering everything from classification to species detection in images. The authors compare traditional machine learning techniques with the latest deep learning-based approaches, discussing the advantages and limitations of each. One focus of the article is the analysis of the challenges inherent in marine datasets involving corals, algae, plankton and fish, highlighting the need for segmentation and image quality enhancement to improve the accuracy of computer vision techniques. This study is particularly relevant in highlighting how deep learning can be used to overcome the difficulties of analysis in dynamic and visually complex environments, such as the ocean.

Vision-Based Classification of Mosquito Species: Comparison of Conventional and Deep Learning Methods, (Okayasu et al. 2019): Although focused on a different area of biology, this study provides a comparison between conventional computer vision methods and deep learning approaches for mosquito species classification. The authors show that while traditional techniques based on handcrafted features achieve an accuracy of 82.4%, deep neural networks using data augmentation achieve an accuracy of 95.5%.

1.2 Image classification under different lighting conditions

Neural networks for wood species recognition independent of the colour temperature of light, (Martinka 2021): This study focuses on wood species recognition, addressing the specific challenge of variations in the colour temperature of light. Unlike many neural network systems that recognize objects based on their contours, wood species recognition relies primarily on surface structure and texture, features that are strongly influenced by lighting. The authors developed a neural network trained to be independent of colour temperature, using different settings and testing with images captured under various lighting conditions (2700K, 4000K, 6500K, among others). The networks were trained with colour and greyscale images, adjusted to highlight surface textures. This study is relevant to our problem, where lighting varies significantly.

Animal Species Recognition with Deep Convolutional Neural Networks from Ecological Camera Trap Images, (Binta Islam et al. 2023): In this work, the authors investigate the use of deep convolutional neural networks to classify herpetofauna species (amphibians and reptiles) captured by camera traps in challenging environmental conditions. One of the main contributions of the study is the use of preprocessing and data augmentation techniques to deal with imbalanced and low-quality biological images, something common in natural environments. This work also shows how preprocessing adjustment and data manipulation can significantly improve the performance of neural networks in variable lighting and image quality conditions, common characteristics in underwater environments.

1.3 Classification of previously unseen species

Recognition of Unseen Bird Species by Learning from Field Guides, (Rodríguez et al. 2024): This study addresses the recognition of bird species that were not seen during training, using knowledge transfer techniques based on discriminative properties of previously known species. The proposed approach involves two main strategies: contrastive coding and the use of illustrations, which are structurally closer to photographs than other forms of secondary data. The first strategy allows the system to learn subtle differences between known species, which is essential for transferring this knowledge to unseen species. The second technique uses illustrations from field guides to fill knowledge gaps, taking advantage of the visual similarity between illustrations and photographs. This approach stands out for integrating complementary information and for its ability to expand the applicability of the model to new categories, something we are looking for in our problem.

Towards Recognizing Unseen Categories in Unseen Domains, (Mancini et al. 2020): This work aims to solve the challenge of recognizing unseen visual categories in domains that were also not observed during training. Traditionally, machine learning systems suffer significant performance degradation when exposed to new classes or domain changes, but the CuMix technique presented in this study aims to overcome these limitations. CuMix combines concepts from Zero-Shot Learning (ZSL) and Domain Generalization (DG) to create more robust training samples, simulating unknown scenarios by mixing images from different domains and categories during training. Despite using a pre-trained model, the study carried out by the authors will serve as a basis for our project.

2 Dataset

Figure 1: Sample image

We used the FathomNet2023 dataset, a selection of images sourced from the large-scale FathomNet repository. This repository was specifically developed to store, classify, and work with annotated underwater images, offering a wide range of data that presents considerable challenges for machine learning problems.

For FathomNet2023, the dataset was carefully curated to explore an important challenge related to ocean depth. The images used for training were captured exclusively in shallower waters, in the upper ocean, while the validation set contains images from deeper regions. This imbalance was intentionally designed to investigate how models handle significant environmental shifts between training and inference, simulating a distribution shift.

Additionally, the dataset curation ensured that longitudinal variations were avoided, meaning all images were collected from geographically consistent locations. It was also ensured that the same equipment was used across samples, minimizing any bias introduced by technological differences. As a result, the dataset contains 290 categories of marine species, providing a broad diversity of classes for supervised learning.

In terms of depth, the training images were collected at depths between 0 and 800 meters, while the validation images span a range from 0 to 1300 meters. This depth distribution was designed to examine how models cope with changing visual conditions as they move into deeper waters, where factors such as lighting and visibility change considerably. Figure 2 shows the cumulative distribution of Ids and Figure 3 shows the histogram of classes in the Trainig dataset. Note that the dataset is highly imbalanced.

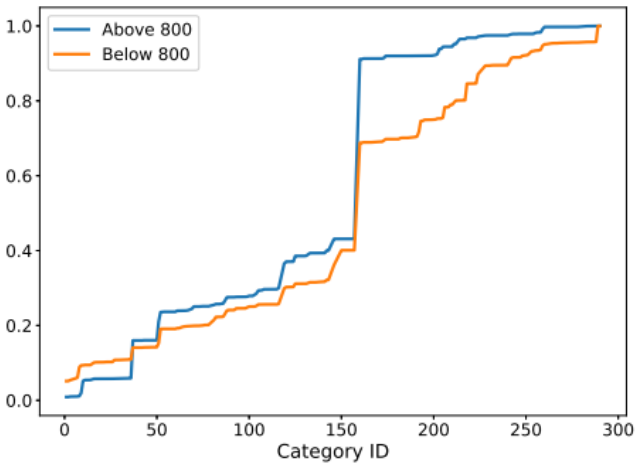

Figure 2: Statistics provided by authors: Cumulative distribution of categories in the training and evaluation datasets for FathomNet2023. *The FathomNet2023 Competition Dataset* 2023

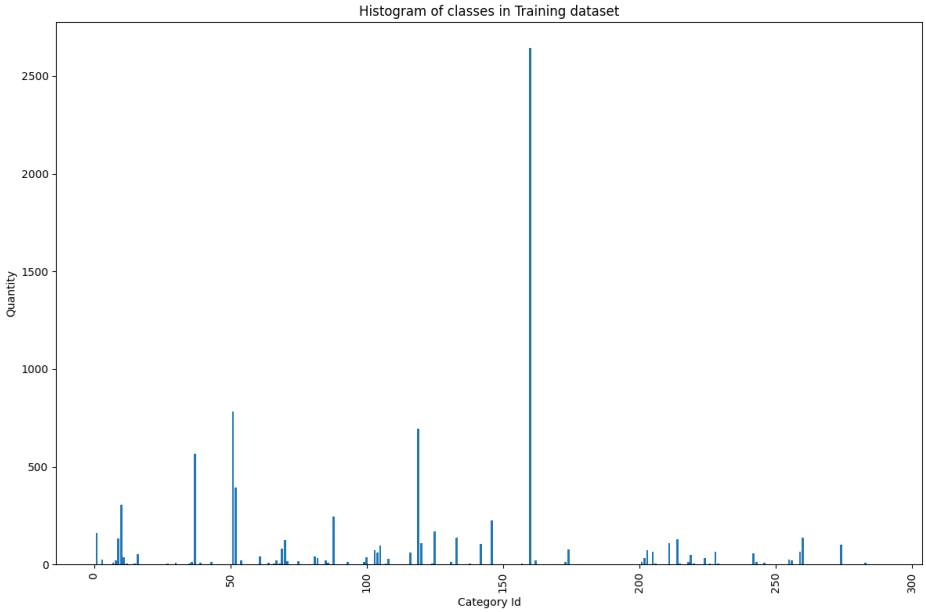

Figure 3: Histogram of Classes in Training dataset. Note that the dataset is highly imbalanced.

Each of the 290 categories belongs to one of 20 supercategories. As illustrated in Figure 4 the training data is highly imbalanced, with "Urchin" emerging as the most frequently occurring supercategory. This imbalance could pose challenges for model training and generalization

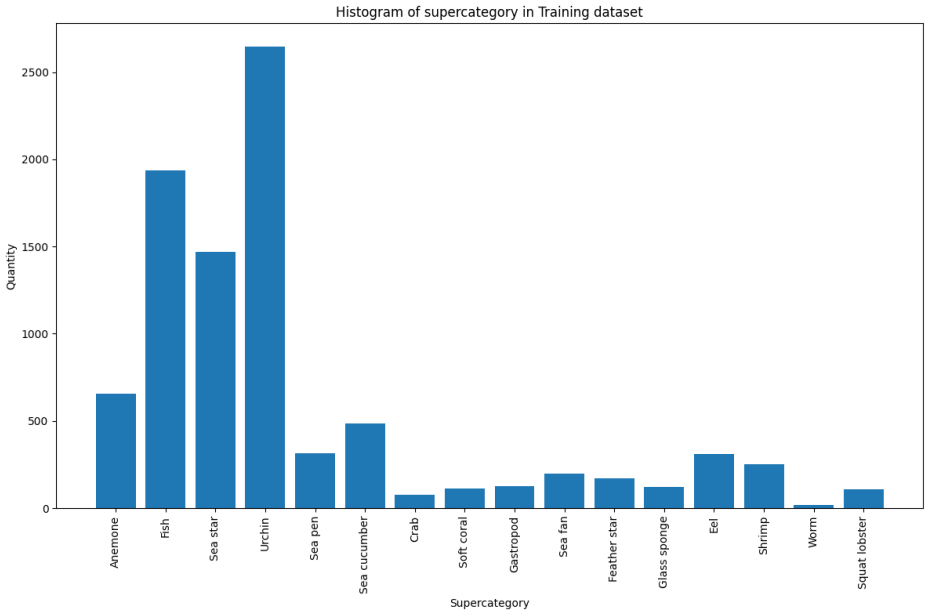

Figure 4: Histogram of superclasses in Training dataset.

Another challenge is that not all classes are represented in the training data, as shown in Figure 5, In such cases, the task becomes identifying that the object belongs to an out-of-distribution (OOD) class, adding an extra layer of complexity to the prediction process.

Figure 5: Example of class that is present on Validation dataset but not in the Training. *The FathomNet2023 Competition Dataset* 2023

Furthermore, even within the same class there are large variations, such as pose, distance from the camera, blurred or incompletely annotated images. This scenario is common in the real world, where annotating data is extremely costly and can lead to noisy annotations.

3 Pre-processing

Underwater images often suffer from severe degradation due to selective light absorption and scattering caused by suspended particles. Common issues include low visibility, reduced contrast, blur, and color distortion, particularly a bluish-green color cast due to the rapid attenuation of red light. These problems make underwater image processing challenging and necessitate effective methods for enhancement and color restoration.

According to Wang et al. 2019, these methods can be organized in two main categories: Underwater Image Enhancement and Underwater Color Restoration.

3.1 Underwater Image Enhancement

Image enhancement is an approach based on the direct manipulation of pixel intensity values to improve image visibility and contrast. Enhancement methods do not explicitly consider the physical principles of underwater image formation. These techniques redistribute intensity values in the spatial or transformed domain, seeking to correct features such as low illumination, reduced contrast, and distorted tones, often caused by light scattering and selective absorption. The goal is to produce an image that is more visually appealing to the human observer.

We used the following methods:

- **CLAHE (Contrast Limited Adaptive Histogram Equalization) Pizer et al. 1990:** Adjusts image contrast by applying adaptive histogram equalization with limits to avoid excessive noise amplification.
- **GC (Gamma Correction):** Corrects brightness by adjusting the power-law relationship between input and output intensity levels, compensating for uneven illumination.
- **HE (Histogram Equalization, Hummel 1975):** Redistributes pixel intensities to enhance contrast and visibility, especially in low-light images.
- **RDist (Rayleigh Stretching, Ghani et al. 2014):** Uses the Rayleigh distribution to stretch intensity values, enhancing contrast and reducing overexposed or underexposed areas.
- **Wavelet (Ma et al. 2022)** : Uses the discrete wavelet transform (DWT) to decompose an image into frequency bands, generating an image structure with low frequency and high frequency details. The approach incorporates two subnetworks: a multi-color space fusion network, which corrects colors from different color space representations, and a detail enhancement network, which improves details using high frequency bands.

Figure 6 shows a comparison between image enhancement methods.

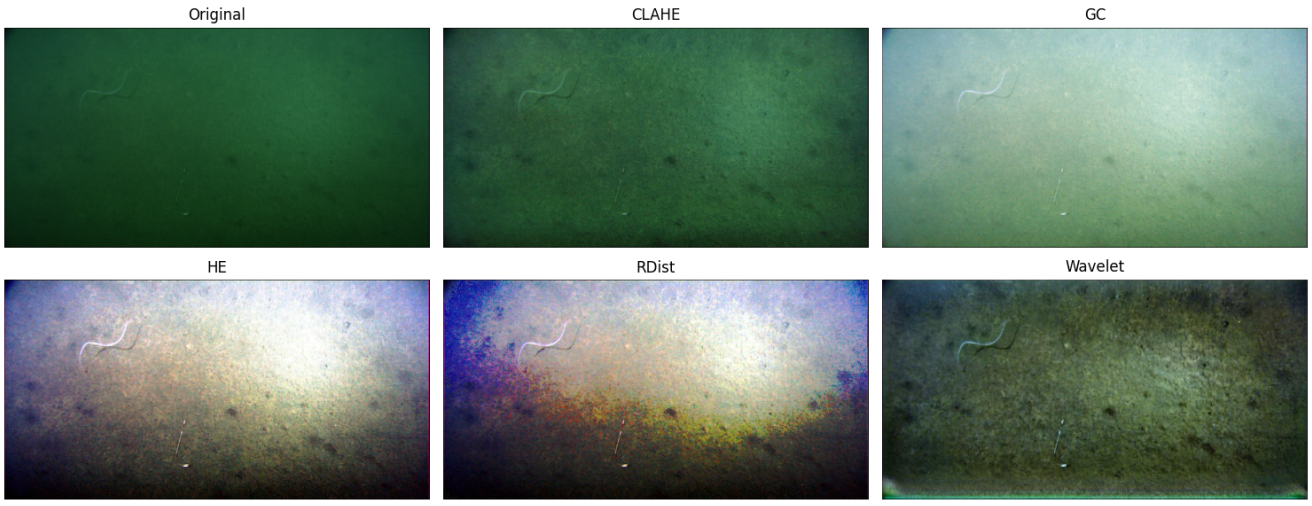

Figure 6: Comparison between image enhancement methods

3.2 Underwater Color Restoration

Color restoration uses models based on the physical principles of underwater image formation. This approach analyzes the selective absorption of light (mainly red) and scattering caused by particles suspended in the water. Restoration methods aim to reconstruct the true colors of the scene, compensating for color loss and eliminating scattering. This is accomplished by estimating parameters such as background light and transmission maps, based on hypotheses or physical models, restoring the natural appearance of the underwater scene.

We used the following methods:

- **DCP (Dark Channel Prior, He et al. 2009):** Uses the dark channel concept (pixels with low intensity in at least one color channel) to estimate transmission and correct scattering and absorption.
- **GB-RC (Blue-Green Channels Dehazing and Red Channel Correction, Li et al. 2016):** Removes haze and corrects the red channel by considering selective light attenuation underwater.
- **IBLA (Image Blurriness and Light Absorption, Peng et al. 2017):** Restores colors based on image blurriness and light absorption properties, reducing scattering effects.
- **LC-DCP (Low Complexity Dark Channel Prior, Yang et al. 2011):** A simplified version of DCP with reduced computational complexity.
- **MIP (Maximum Intensity Prior, Carlevaris-Bianco et al. 2010):** Estimates scene depth and corrects color distortion based on differences between maximum channel intensities.
- **nom (New Optical Model, Wen et al. 2013):** A modified optical model considering scattering and selective attenuation for color restoration.
- **RoWS (Removal of Water Scattering, Chao et al. 2010):** Removes scattering effects caused by suspended particles.
- **UDCP (Underwater Dark Channel Prior, Drews Jr et al. 2013):** Adapts DCP to underwater conditions using only blue and green channels to minimize red channel distortion.
- **ULAP (Underwater Light Attenuation Prior, Song et al. 2018):** Estimates scene depth based on the difference between red, green, and blue channel intensities, enabling precise restoration.

Figure 7 shows a comparison between the methods mentioned and the method we chose (Wavelet).

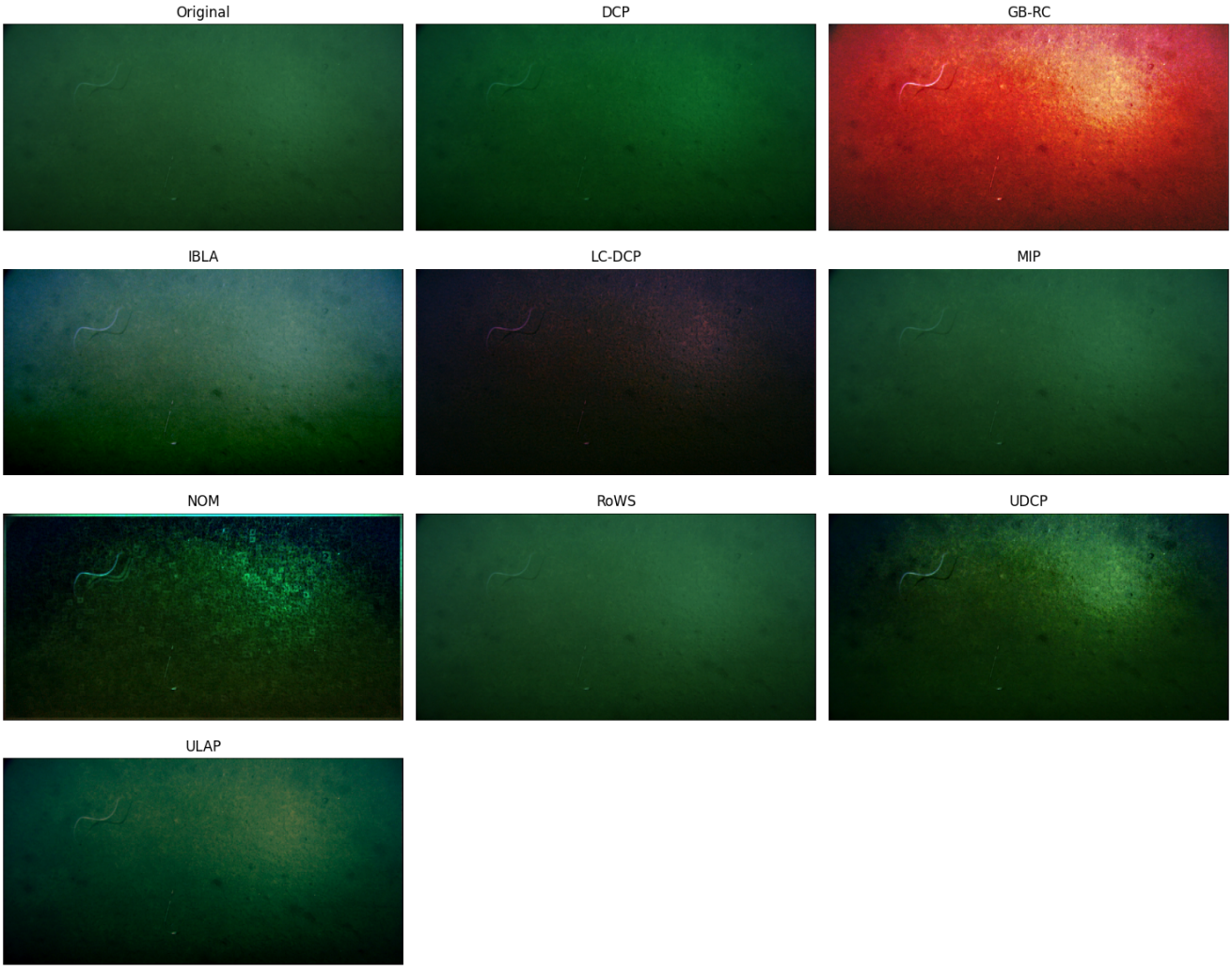

Figure 7: Comparison between Color Restoration methods with the chosen method (Wavelet)

After an exhaustive comparison of the images in our dataset, we chose to pre-process the images using the Wavelet method, which we will describe in the next subsection. We will not explain all the other methods, please refer to Wang et al. 2019 for a detailed explanation.

3.3 Wavelet Method

Methodology

The article proposes a wavelet-based dual-stream network to enhance underwater images, addressing color correction and detail enhancement separately. The methodology consists of the following steps:

Decomposition into Sub-Bands

The input image is decomposed into multiple frequency bands using the Discrete Wavelet Transform (DWT). This process generates:

- A low-frequency structural image.
- High-frequency detail images (vertical, horizontal, and diagonal).

Dual-Stream Network

- **Multi-Color Space Fusion Network:** Processes the low-frequency structural image to correct color distortions by leveraging representations in multiple color spaces (RGB, HSV, Lab).
- **Detail Enhancement Network:** Enhances image details using the high-frequency sub-band images.

Reconstruction

The processed sub-band images are integrated and reconstructed to the original resolution using the Inverse Discrete Wavelet Transform (IDWT). Structural and detail losses are optimized using an adversarial loss based on Generative Adversarial Networks (GANs).

Datasets

The model was trained using synthetic images derived from the NYU-v2 dataset (Nathan Silberman et al. 2012) following the procedure proposed in , comprising 20,000 images generated under varying turbidity and lighting conditions. Validation was performed on the following datasets:

- **NYU-v2:** 3,000 synthetic images.
- **UIEB:** 890 real-world underwater images.
- **ColorChecker:** 7 images collected with standard color calibration.

Figure 8 show images extracted from the training dataset.

Figure 8: Images extracted from the training dataset. Image extracted from (Anwar et al. 2018)

Results

According to the authors, the proposed model effectively addresses color cast correction and blur removal with low computational complexity. It outperforms both physics-based and learning-based methods in metrics such as UIQM, UCIQE, and CIE2000, producing visually realistic images free of artifacts.

Figure 9 shows the result of this pre-processing method on some images from our dataset. Note that it helps to remove the blue tone from the images, but some still have blue artifacts on the edges. We believe this is due to the process used to generate the synthetic dataset.

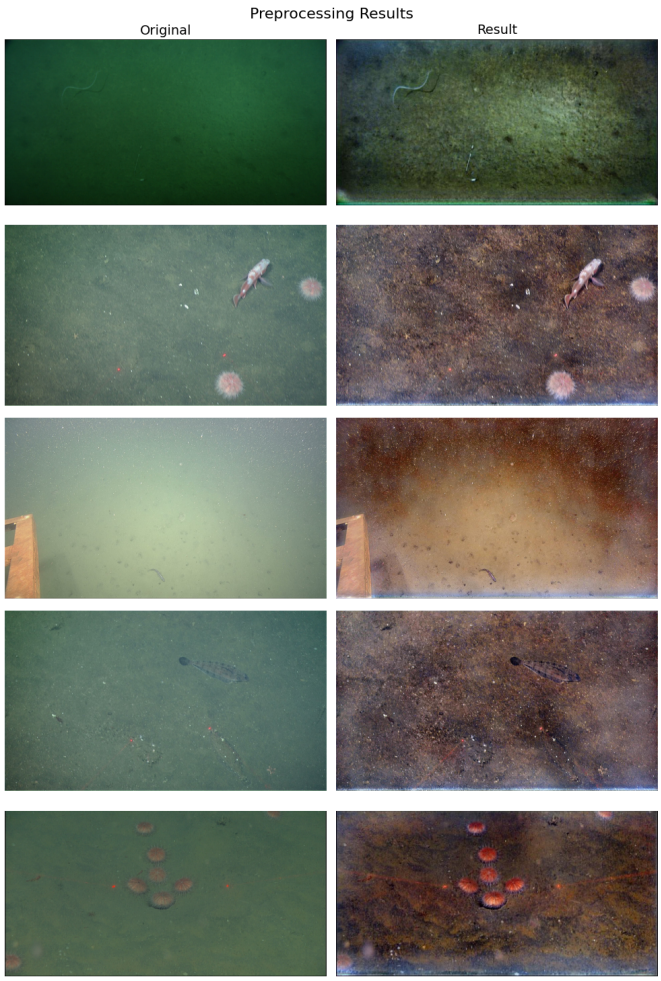

Figure 9: Result of wavelet method on some images from our dataset

All the preprocessing code is available at our Github. More specifically, The process for applying the wavelet method can be found in this notebook, while the other methods can be found in the following folder

4 Methods

4.1 Model Architecture

The model architecture for this project is an advanced iteration of the YOLO (You Only Look Once) object detection series, termed YOLO. This model is a deep convolutional neural network designed specifically for speed and accuracy, with improvements to handle diverse and challenging datasets like FathomNet2023 effectively.

4.1.1 YOLOv8 Network Design

YOLOv8's architecture is an evolution of previous YOLO models, utilizing a convolutional neural network divided into three main parts: the backbone, the neck and the head. The backbone is based on a modified version of the CSPDarknet53 architecture, consisting of 53 convolutional layers enhanced with cross-stage partial connections.The neck uses a series of convolutional layers to refine the feature maps from the backbone, employing techniques such as spatial pyramid pooling to enhance the receptive field and capture multiscale objects effectively. The head comprises multiple convolutional layers followed by fully connected layers responsible for predicting bounding boxes, objectness scores, and class probabilities. Notably, YOLOv8 integrates a self-attention mechanism in the head of the network and a feature pyramid network for multi-scaled object detection, enabling it to focus on various parts of an image and detect objects of different sizes and scales. Below is the diagrammatic representation of the YOLOv8 architecture:

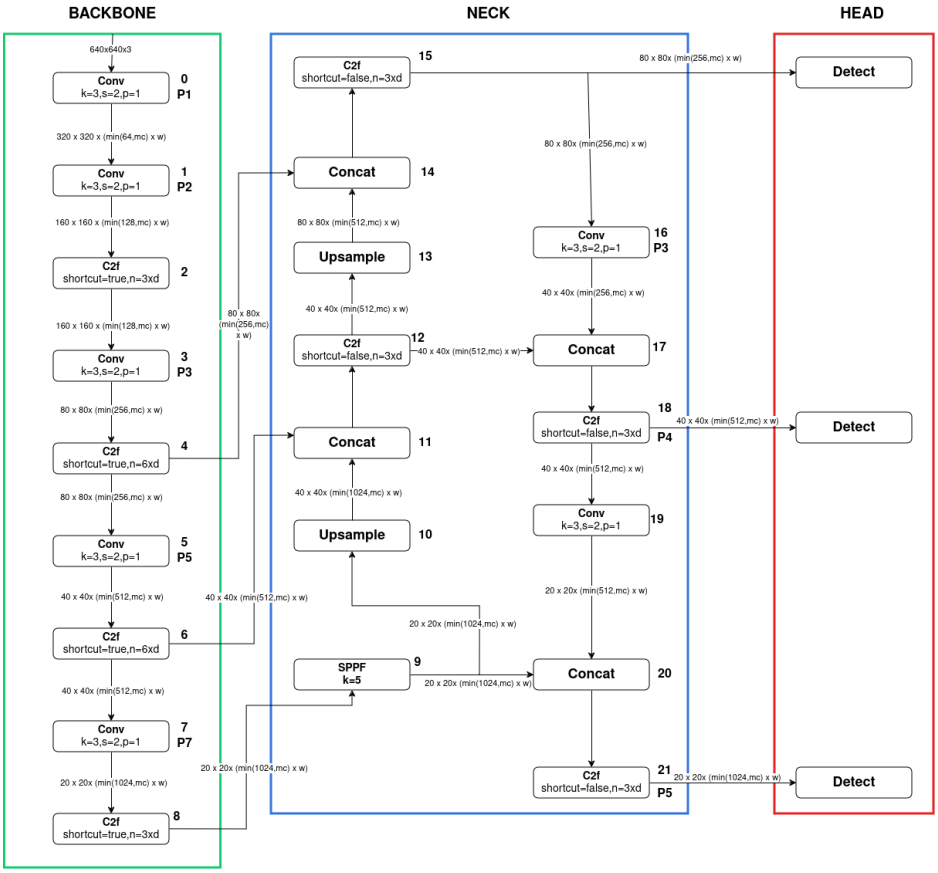

Figure 10: The YOLOv8 Model Architecture.

The YOLOv8 network architecture is structured as follows:

- **Backbone:** The backbone of YOLOv8 uses a modified CSPDarknet53, which is a deep residual network known for its ability to extract rich feature maps from input images. This backbone is optimized for fast feature extraction.
- **Neck:** The neck uses a series of convolutional layers to refine the feature maps from the backbone, employing techniques such as spatial pyramid pooling to enhance the receptive field and capture multiscale objects effectively.
- **Head:** The head of the network, which is responsible for making the final predictions, includes separate branches for bounding box regression, objectness scores, and class probabilities. It employs anchor boxes optimized for the specific scale and aspect ratios observed in the training dataset.

4.1.2 Feature Integration

Integration of features from different layers is achieved through a feature pyramid network (FPN), which allows the model to detect objects at various scales effectively. This is crucial for underwater imagery where objects can vary significantly in size and appearance due to varying distances and lighting conditions.

4.1.3 Loss Function

The loss function used in YOLOv8 is a composite of three terms:

- **Box Regression Loss:** For the accurate localization of objects.
- **Objectness Loss:** To determine whether a box contains an object.
- **Classification Loss:** Cross-entropy loss for classifying the objects within the box.

This composite loss ensures that the model learns to optimize both localization and classification tasks simultaneously.

4.1.4 Training Procedure

Training of YOLOv8 involves the following steps:

- **Data Augmentation:** To make the model robust to variations in input data, extensive data augmentation techniques such as rotation, scaling, and color adjustment are applied.
- **Batch Normalization:** Applied after each convolutional layer to help stabilize the learning process and accelerate convergence.
- **Transfer Learning:** The model is pre-trained on a large dataset (such as COCO) to help in faster convergence and improved generalization on the target dataset.

4.1.5 Inference

During inference, YOLOv8 processes images in real-time, providing bounding boxes and class predictions. Non-maximum suppression is used to refine the bounding boxes, ensuring that each object is detected only once.

4.2 YOLOv11

YOLOv11 represents a significant evolution in the YOLO family of models, emphasizing accuracy, computational efficiency, and versatility. It builds on the foundation laid by YOLOv8 and introduces several improvements in architecture and optimization techniques.

4.2.1 Enhanced Accuracy with Fewer Parameters

YOLOv11 achieves higher **mean Average Precision (mAP)** scores compared to previous versions, such as YOLOv8m, on standard benchmarks like the COCO dataset. This improved accuracy translates to better object detection, instance segmentation, and classification performance, particularly in challenging datasets.

In terms of parameter efficiency, YOLOv11 uses **22% fewer parameters** than YOLOv8m, significantly reducing memory consumption and speeding up computations. The lightweight design makes it ideal for deployment on **resource-constrained devices** such as mobile phones, drones, and edge devices, without compromising detection performance.

4.2.2 Architectural Improvements

- **Efficient Feature Extraction**: YOLOv11 incorporates advanced feature extraction layers, capturing more detailed information from images while maintaining computational efficiency. The architecture is optimized for better spatial and contextual understanding, enabling the detection of small and occluded objects.
- **Optimized Processing Pipelines**: Redundant operations are reduced, and layer connections are streamlined, enhancing throughput without increasing computational demands. This leads to faster inference speeds, even on low-power devices.

4.2.3 Versatility Across Tasks

YOLOv11 extends support across a variety of computer vision tasks, making it highly flexible:

- **Object Detection**: Identifies and localizes objects in images with high precision.
- **Instance Segmentation**: Assigns pixel-level masks to objects for precise segmentation.
- **Pose Estimation/Keypoints**: Detects and tracks keypoints for human poses or other objects requiring skeletal representation.
- **Classification**: Efficiently classifies objects in images, even in complex scenarios.

The versatility ensures YOLOv11 can be applied to real-world problems across domains such as healthcare, surveillance, robotics, and autonomous systems.

4.2.4 Computational Efficiency

Optimized for Edge Deployment: With fewer parameters and reduced memory requirements, YOLOv11 is designed for real-time performance on devices with limited computational power. It achieves faster inference speeds compared to its predecessors, making it suitable for time-critical applications like autonomous vehicles and real-time video analytics.

Balanced Performance: The model strikes an excellent balance between accuracy and speed, ensuring practical utility across diverse scenarios.

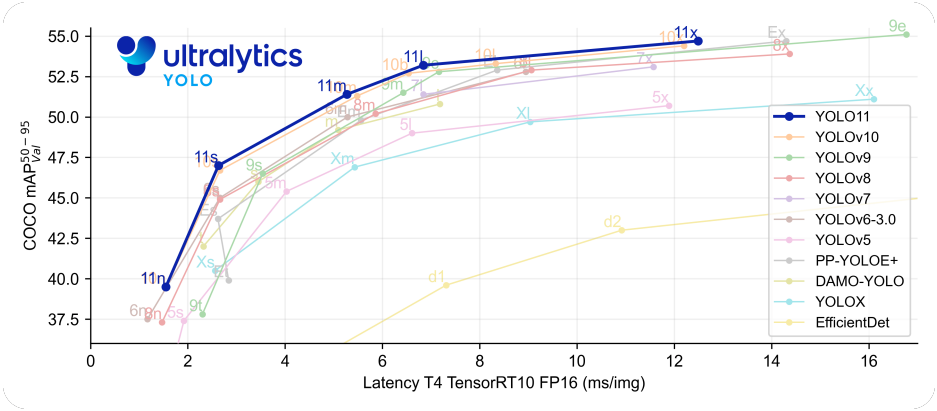

Figure 11: The YOLOv11 vs other versions

5 Evaluation metrics

We will use the mAP50 and mAP50-95 metrics to evaluate the performance of our model. mAP50 and mAP50-95 are two important metrics used to evaluate object detection models, and they give us a clear idea of how well the model is doing in terms of both detecting and localizing objects.

5.1 mAP@50 Metric

mAP50 measures the average precision at a 50% overlap between predicted and true bounding boxes. It's a good, general measure of how accurately the model is detecting objects, even if the bounding boxes aren't perfect. It's widely used because a 50% overlap is often considered a reasonable balance between being too lenient and too strict.

5.2 mAP@50-95 Metric

mAP50-95 takes it a step further by measuring average precision at multiple overlap levels, ranging from 50% up to 95%. This gives a more detailed picture of how well the model performs across different levels of accuracy. If a model performs well here, it means it not only detects objects but also localizes them very precisely.

6 Training Pipeline

The training pipeline begins with the establishment of a baseline using the YOLOv8m model. The baseline was trained on the full dataset, consisting of 5950 images, for 5 epochs. This provides a point of comparison for assessing the performance of different model configurations and training setups. In addition to the baseline, we trained and evaluated three distinct variants of the YOLO models to analyze their performance under varying conditions. These variants include changes in data preprocessing, sample size, and model architecture, as detailed below.

6.1 Baseline: YOLOv8m Model

The YOLOv8m model serves as the baseline and was trained on the complete dataset without any preprocessing. The training was performed with a 60:40 split into 3570 training images and 2380 validation images. The training process involved:

- **Dataset Size**: 5950 images, split into training and validation sets of 60/40 (3570 train, 2380 validation images).
- **Number of Epochs**: 200 epochs, sufficient for initial convergence.
- **Loss Behavior**: The training loss exhibited rapid stabilization, and the mAP metrics reached reasonable levels, showcasing the effectiveness of YOLOv8m on unaltered data.

6.2 Variant 1: YOLOv8m Model on Preprocessed Data + Finetuning

The second variant applied the Wavelet-based preprocessing method discussed earlier to the full dataset. This preprocessing removed hues, illumination, and blueness—common features in underwater images—resulting in a dataset that challenges the model to focus on structural features rather than color cues. The training details are as follows:

- **Dataset Size**: Full dataset (5950 images), preprocessed using the Wavelet method, split into training and validation sets of 60/40 (3570 train, 2380 validation images).
- **Number of Epochs**: 200 epochs, sufficient for initial adaptation to the new data representation.
- **Challenges and Observations**:
  - The preprocessing significantly altered the dataset, forcing the model to learn features unrelated to the ocean's typical hues and illumination.
  - Initial loss values (train/boxloss, train/clsloss, and train/dfl loss) were higher compared to the baseline but exhibited a declining trend.
  - mAP scores were lower initially, reflecting the increased difficulty in detecting objects without color and illumination cues.
  - Loss functions converged after approximately 40 epochs, showcasing that the model is struggling to learn after the hueness was removed.
- **Model Adaptation**: The increasing mAP trend indicated that the model was gradually adapting to the new data distribution, though convergence required more epochs.

This experiment highlights the trade-off between dataset size and model accuracy, showing that smaller datasets require extended training to achieve comparable results.

6.3 Variant 2: YOLOv8m Model on a larger split

Following the preprocessed model, and thorough data analysis of the distribution of labels, we decided to try an 80/20 split, so that the model had more data to learn over and not be susceptible noise altering the accuracy.

- **Dataset Size**: Full dataset (5950 images), preprocessed using the Wavelet method, split into training and validation sets of 80/20 (4760 train, 1190 validation images).
- **Number of Epochs**: 200 epochs, allowing the model sufficient time to converge despite the smaller dataset.
- **Performance Insights**:
  - Loss functions (box, classification, and dfl loss) converged steadily over the epochs.
  - mAP50 and mAP50-95 scores, performed significantly better than the baseline, ranging from 80% to approximately 100%, confirming the hypothesis that the results were being influenced by the distribution of labels.

6.4 Variant 3: YOLOv11 Model on varying split size of original data

The YOLOv11 model is highly sophisticated that it gave very similar and better results on different sample sizes, trained on the original data set, with 200 epochs. This variant is adaptable to different sample sizes, which makes it perfrom stable and very effective even on the different sample sizes:

- **Sample sizes**: 60-40 split and a 80-20 split have been used in this variant to understand the difference in the sample size, which was very clearly seen in the v8. But, V11 has shown very effective results on both the samples.
- **Number of Epochs**: 200 epochs allowed the model to train well and adaptable to the changes.
- **Performance Insights**:
  - Loss convergence started to happen at earlier epochs and the uniformity of the curve, shows how the model training is very stable and reliable.
  - The mAP50 and mAP50-95 scores immensely improved to almost perfection of 1, compared to all variants and the baseline, indicating better generalization and detection capabilities and a benchmark solution.
- **Computational Efficiency**: Although it's computationally extensive, it performed many models in terms of reliability, adaptability and effectiveness.

6.5 Variant 4: YOLOv11 Model on the Preprocessed Dataset

The fourth variant introduced the YOLOv11 model, trained on the same wavelet-preprocessed data set as the first variant. This variant aimed to leverage YOLOv11's architectural advancements and efficiency for better performance:

- **Model Architecture**: YOLOv11's efficient feature extraction and processing capabilities provided an edge in learning from challenging data distributions.
- **Performance Insights**:
  - Loss functions converged after approximately 40 epochs, showcasing YOLOv11's ability to efficiently learn from preprocessed data.
  - The mAP50 and mAP50-95 scores are similar to that of the first variant, highlighting that the model still struggles with learning fine-grained details after the removal of hueness.

6.6 Variant 5: YOLOv11 Large Model on an 80/20 Split

Building off of the results from variant 2 and 3, a larger model was attempted. This variant aims to see if there will be any significant improvement compared to variants 2 and 3.

- **Number of Epochs**: 200 epochs allowed the model to train well and adaptable to the changes.
- **Performance Insights**:
  - Loss convergence started to happen at earlier epochs and the uniformity of the curve, shows how the model training is very stable and reliable.
  - The mAP50 and mAP50-95 scores immensely improved to almost perfection of 1, compared to all variants and the baseline, indicating better generalization and detection capabilities and a benchmark solution.
- **Computational Efficiency**: Although it's computationally extensive, it performed many models in terms of reliability, adaptability and effectiveness.

6.7 Comparative Analysis

A comparative summary of the three variants and the baseline is provided below:

- **Baseline**: The YOLOv8m model trained on the full dataset on a 60/40 split without preprocessing showed rapid convergence but relied heavily on the dataset's original features.
- **Variant 1**: Preprocessed data introduced additional challenges, with initial performance dips due to the absence of ocean-specific features. However, increasing mAP trends showed the model's potential with extended training.
- **Variant 2**: The YOLOv8m model trained on an 80/20 split resulted in significant improvements in the loss curve and the mAP50 and the mAP50-95 indicating the noise that minority count categories had on the training.
- **Variant 3**: The YOLOv11m model trained on a 60/40 split yielded similar results to variant 2, highlighting the ability of YOLOv11 to adapt to new images, with less training images and smaller number of parameters.
- **Variant 4**: The YOLOv11m model trained on a 60/40 split yielded similar results to variant 2, highlighting the ability of YOLOv11 to adapt to new images, with less training images and smaller number of parameters.

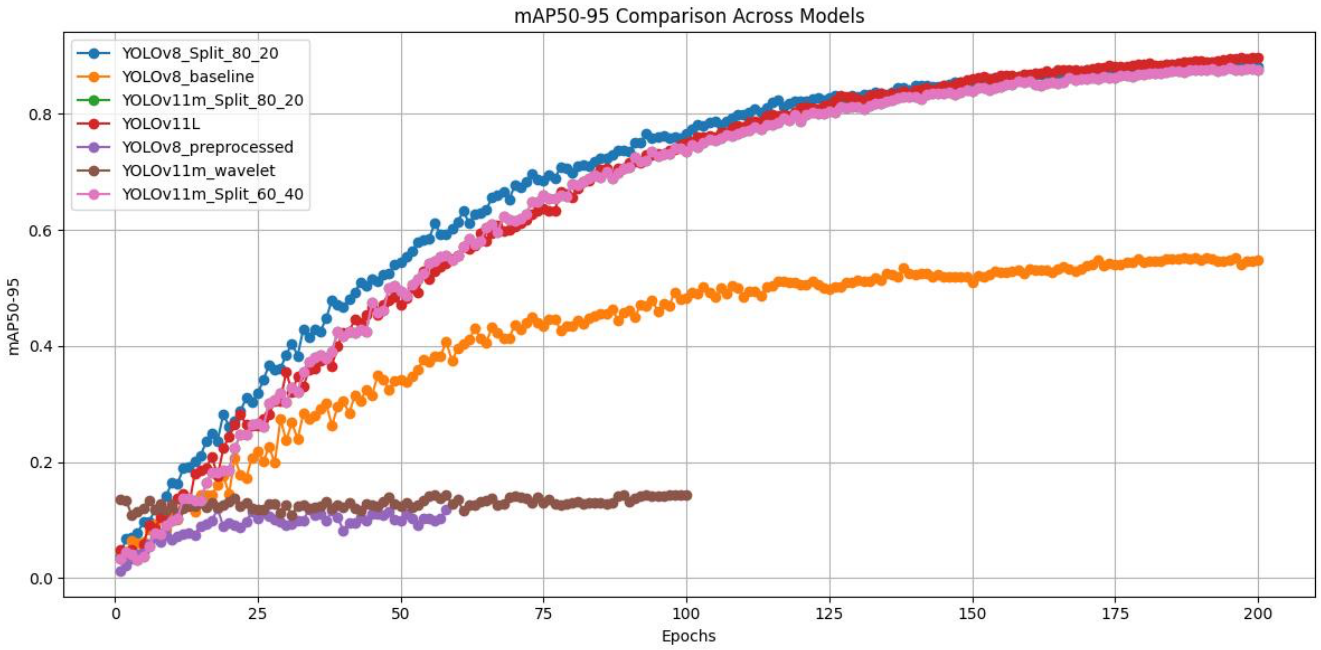

Figure 12: Model Comparision

The numerous model plot demonstrates the performance of the mAP50-95 across multiple variants. As we can see, the preprocessed models are performing the worst due to the models struggling to adapt and learn fine-grained details from each preprocesed image. In addition, the YOLOv11 are the top-contendors along with the YOLOv8m model trained with an 80/20 split highlighting the importance of data in its sampling to adapt to minority classes. The performance of the YOLOv11 models highlight the effectiveness of the up-to-date model to adapt to new images better than the YOLOv8 model with a smaller number of parameters, leading us to prefer to use the YOLOv11m model with a 60/40 split. In addition, the losses decrease much faster to lower values using the 80/20 splits on the YOLOv8 and YOLOv11 models and the 60/40 split on the YOLOv11 models, indicating the improvement in performance of the optimizer with the better training models and more effective, up-to-date models.

This training pipeline demonstrates the impact of dataset size, preprocessing, and architectural improvements on model performance. While the baseline provided quick convergence, the experiments with preprocessed data and YOLOv11 highlight the importance of extended training and robust architectures

for challenging datasets. With its optimized design and efficient feature extraction, YOLOv11 proves to be a superior choice for resource-constrained applications requiring high precision and adaptability.

7 Inference

Note: The metrics are computed on the validation set.

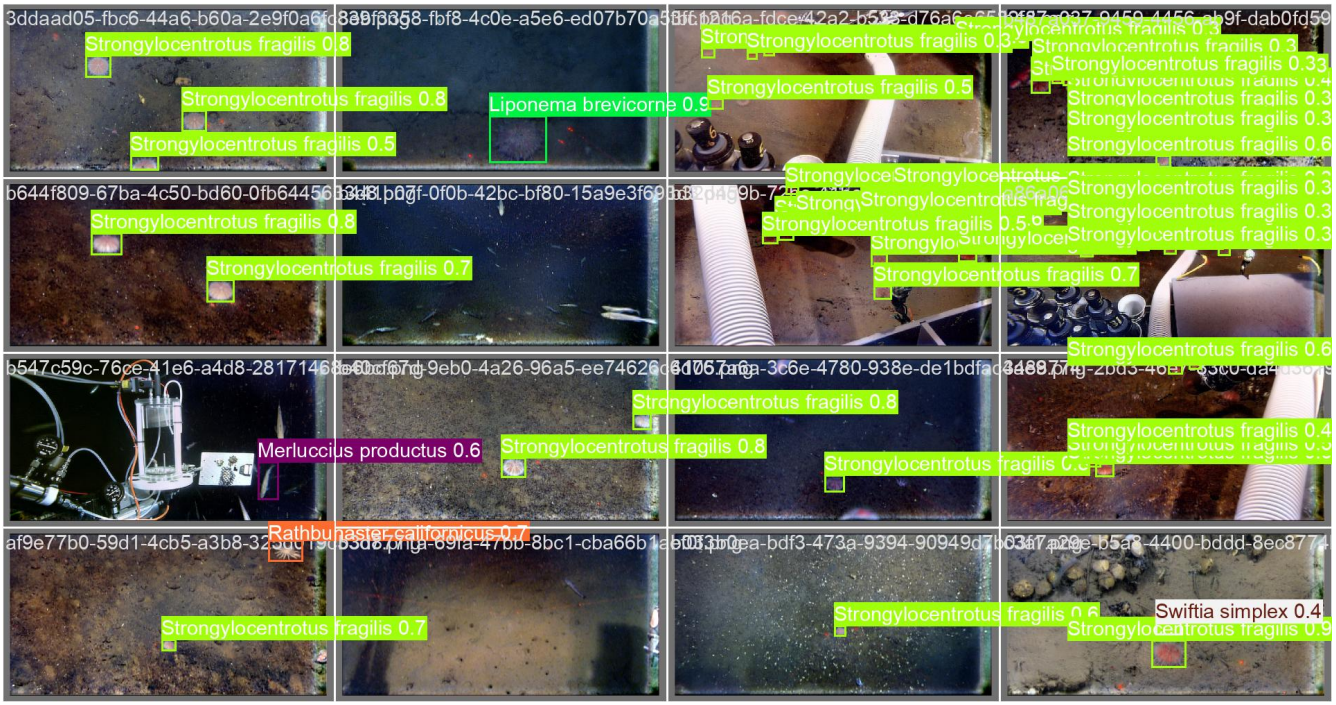

Figure 13: Annotated predictions on Processed data on YOLOv11

Figure 14: Annotated predictions on raw data on YOLOv11

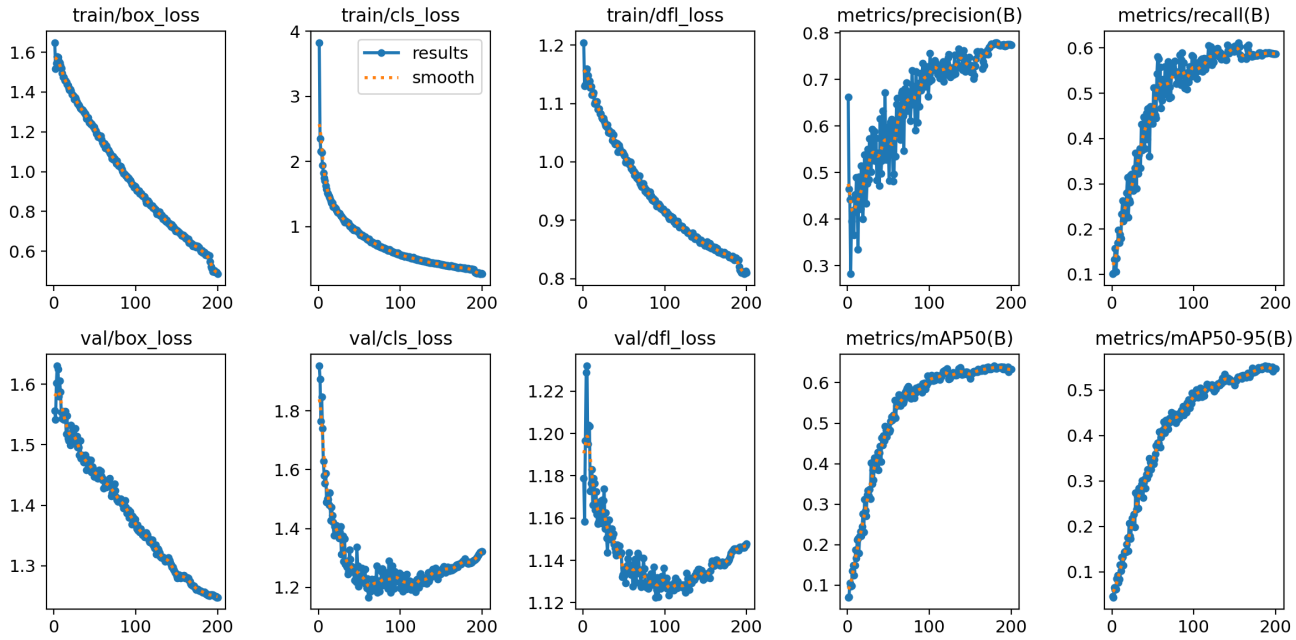

Figure 15: Evaluation metrics on the baseline YOLOv8 model using a 60/40 split (200 epochs)

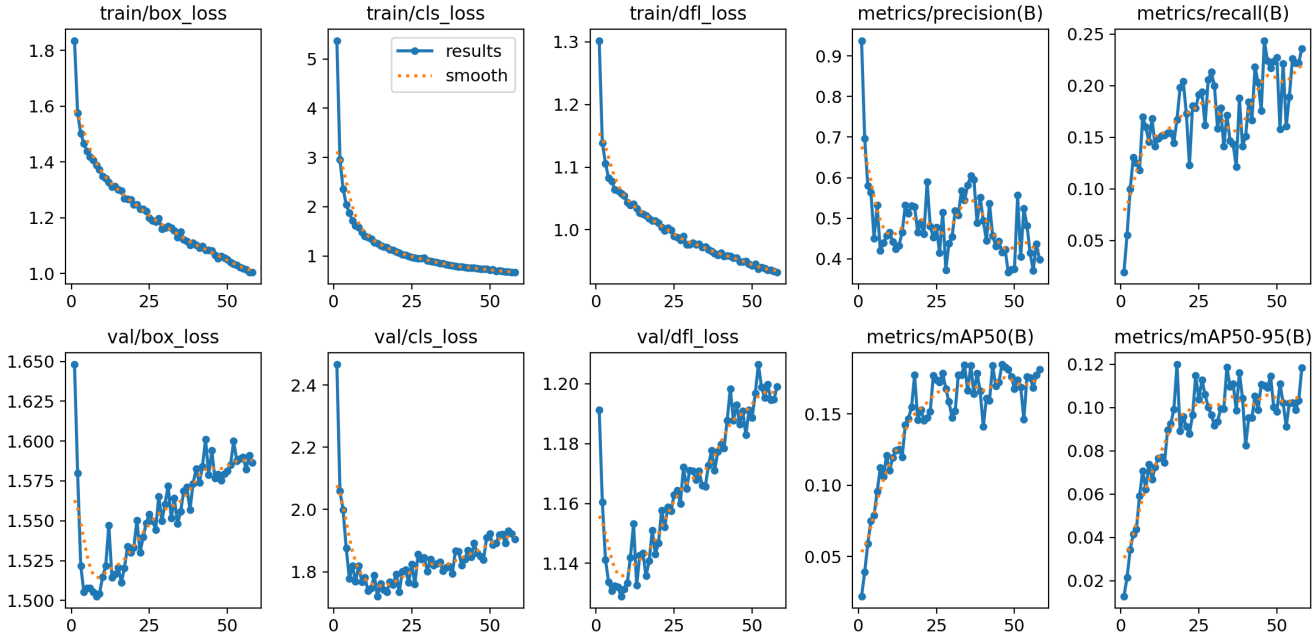

Figure 16: Evaluation metrics on YOLOv8 model using a 60/40 split on the preprocessed dataset (200 epochs)

For the YOLOv8 model trained using the raw dataset for a training split of 60/40 and 200 epochs, the training behavior indicates that the model is performing around the same as randomly flipping a coin, with 50% mAP50 and mAP50-95. The convergence for the loss curves is quite unstable, and is rapidly fluctuating, but is generally decreasing, which indicates some struggle in the model to accurately detect some objects, likely those in the minority label count classes. In looking at the preprocessed data, the loss curves are much more unstable, with an eventual constant linear increase in the losses and rapid fluctations in the mAP50 and mAP50-95 values, indicating there is severe instability in the model to adapt in the images, due to the fine-grained details in the images, leading to misclassifications.

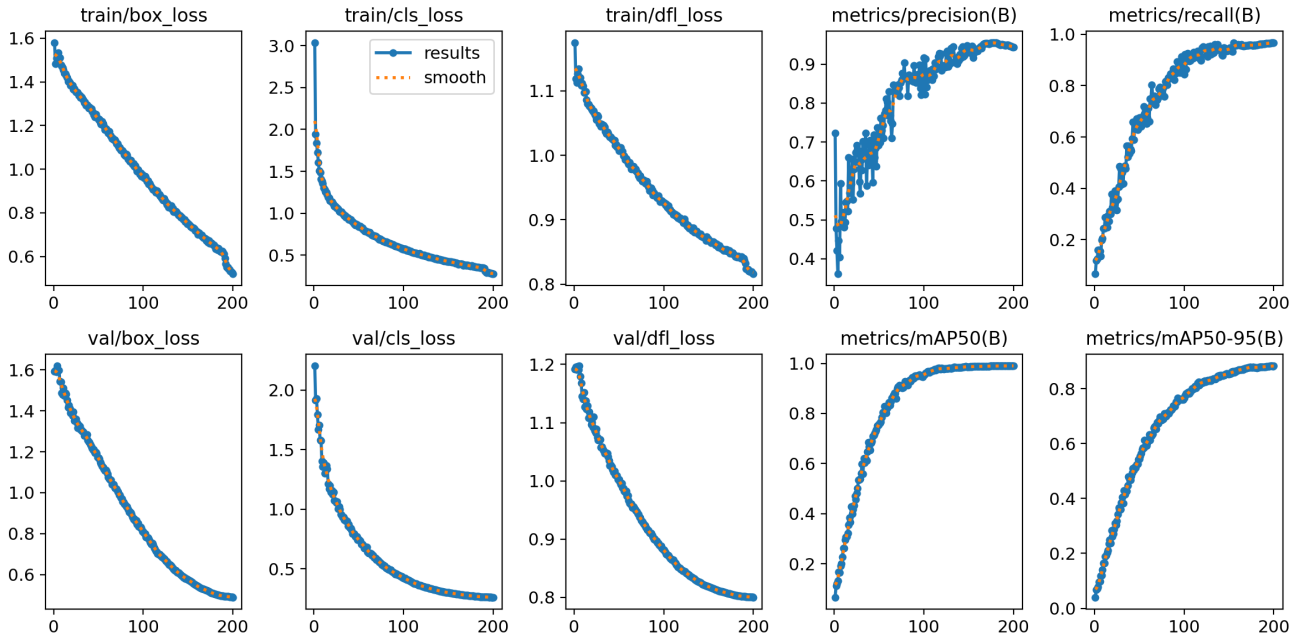

Figure 17: Evaluation metrics on the baseline YOLOv8 model using an 80/20 split (200 epochs)

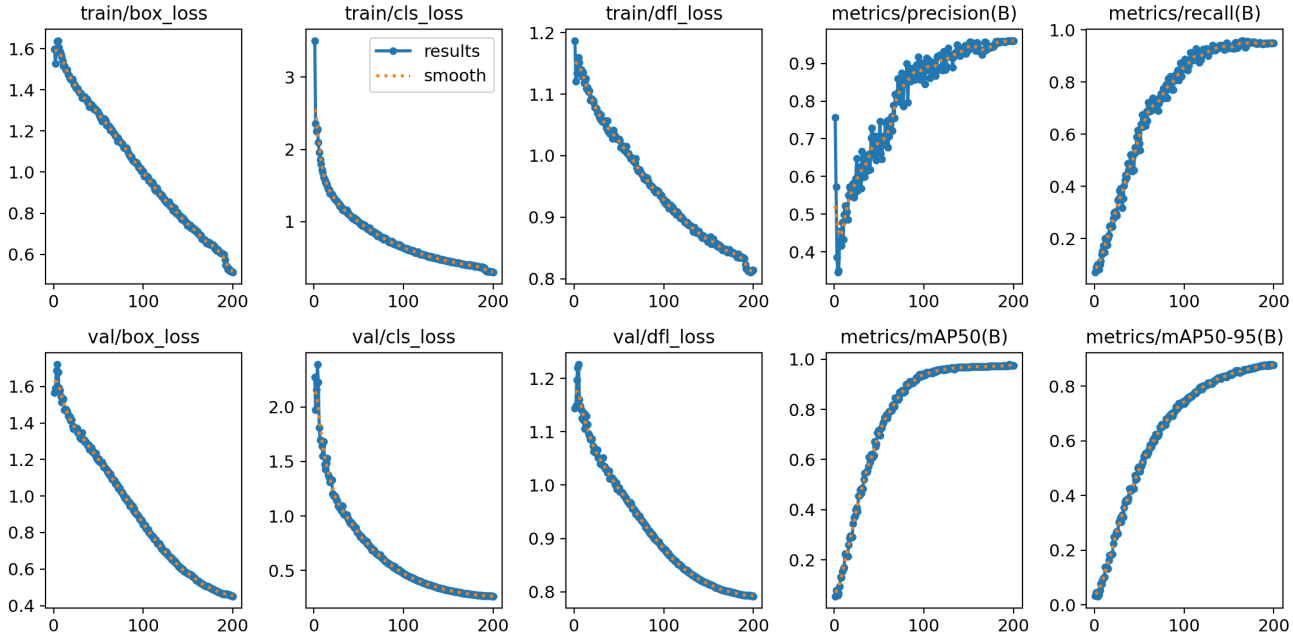

Figure 18: Evaluation metrics on the baseline YOLOv11 model using a 60/40 split (200 epochs)

In comparing the YOLOv8 model trained using the raw dataset for a training split of 80/20 and 200 epochs, the training behavior indicates that the model is converging to near perfect mAP50 and high mAP50-95 scores of around 80%, which is a significant improvement compared to the YOLOv8 model trained using a 60/40 split. This supports the hypothesis that the minority label counts contribute to misclassifications, and with more training data, the YOLOv8 model will improve and generalize to new data more effecitively. With the YOLOv11 model trained using the raw dataset for a training split of 60/40 and 200 epochs, this yields similar behavior to the YOLOv8 model with an 80/20 split, likely due to the effectiveness of the more up-to-date YOLOv11 model to generalize to new data. Not only that, but since the YOLOv11 model has less parameters, and needs less data than the YOLOv8 to produce similar results, YOLOv11 is more effective and efficient in object detection.

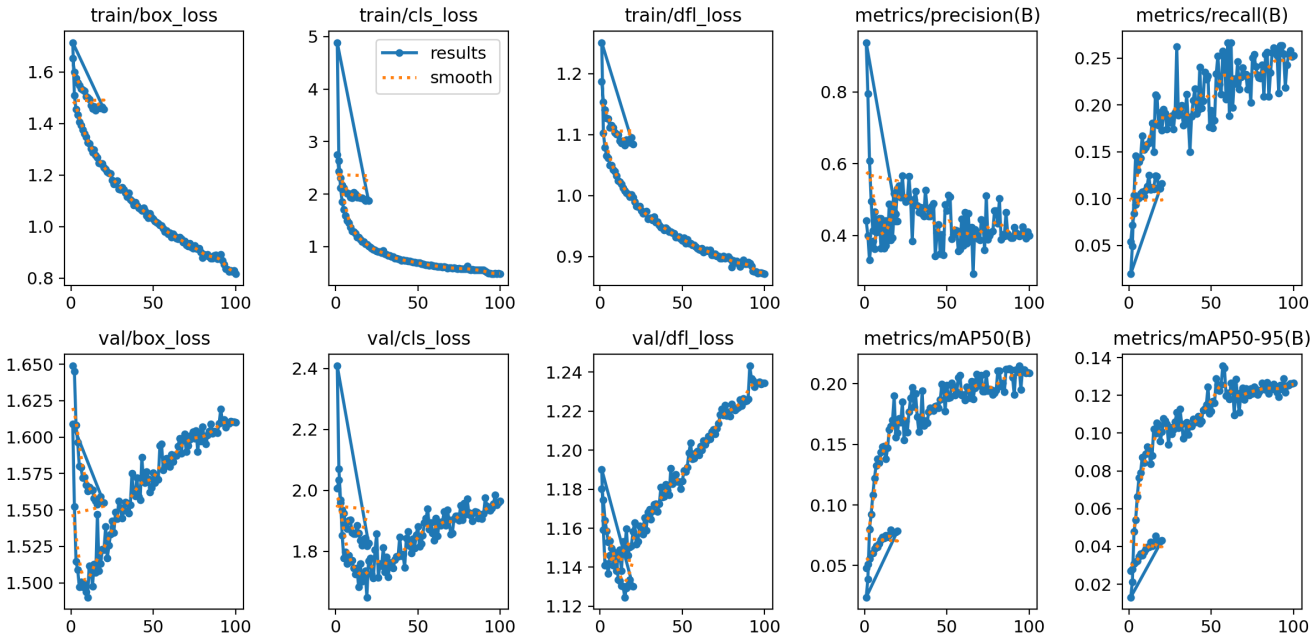

Figure 19: Evaluation metrics on the pre-processed dataset using YOLOv11m (100 epochs)

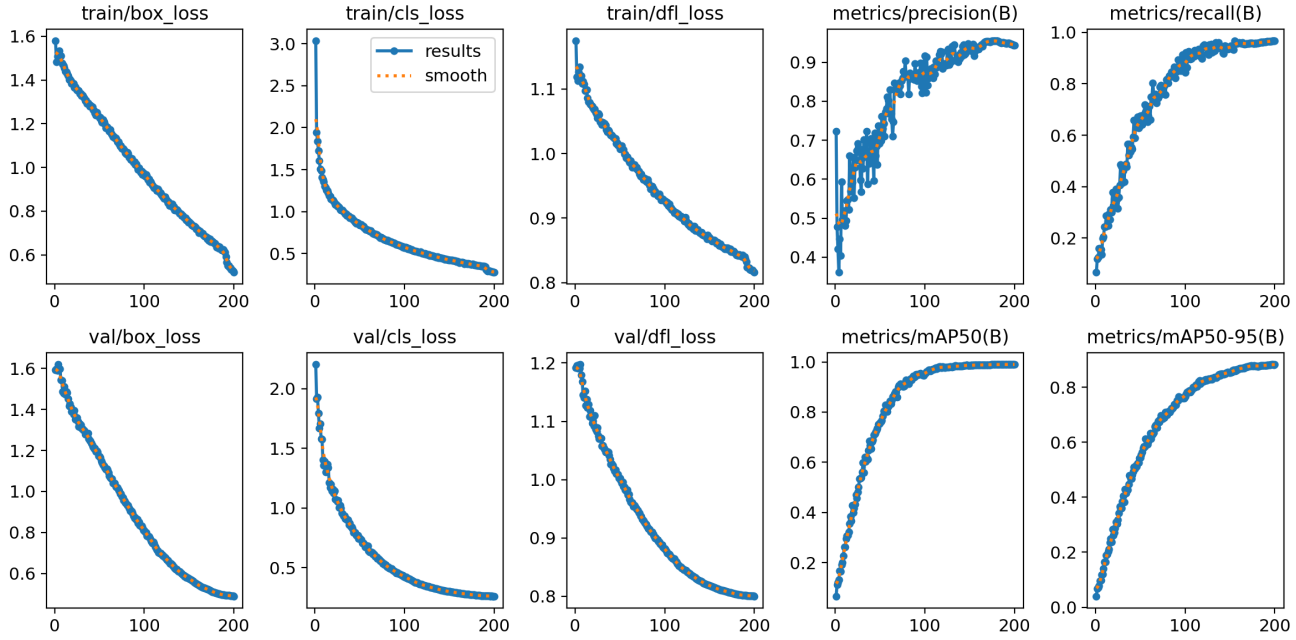

Figure 20: Evaluation metrics on the raw dataset for an 80/20 split using YOLOv11m (200 epochs)

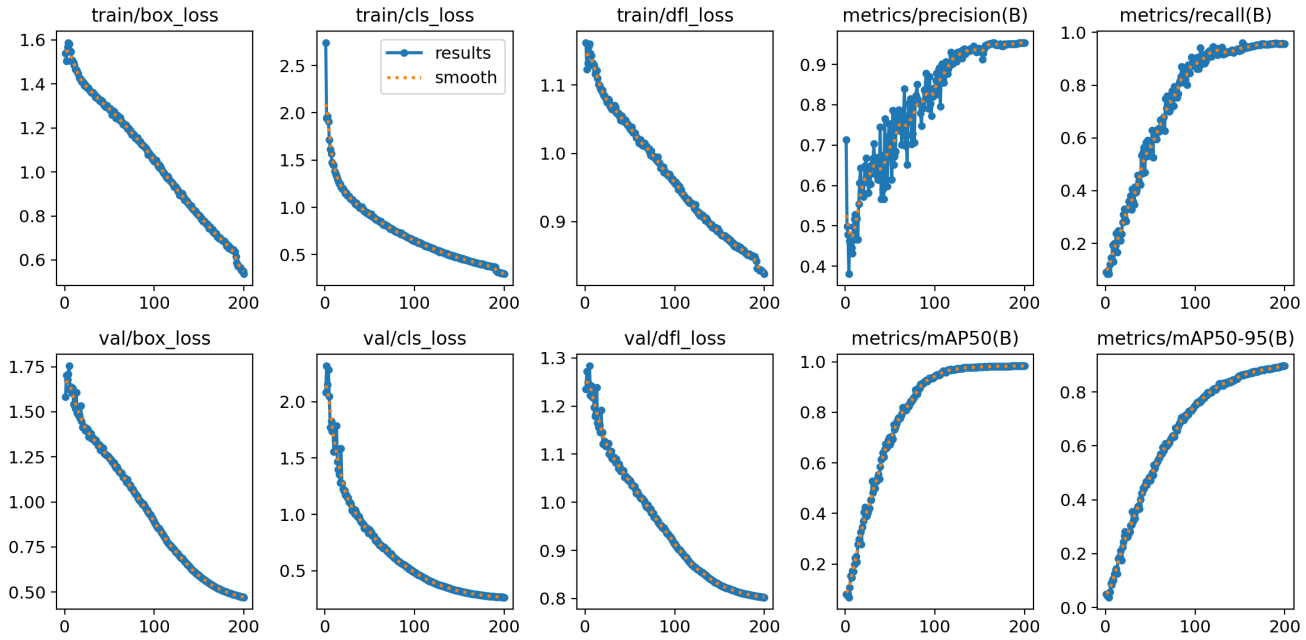

Figure 21: Evaluation metrics on the baseline YOLOv11 model using a 60/40 split (200 epochs)

For the YOLOv11 model trained using the raw dataset for 200 epochs, the training behavior indicates that the model is in the perfect condition and easily adaptable to the raw data. It is very stable and converged to the perfect statistics, which indicates that the YOLOv11m architecture is far ahead of all the proposed variants and it's perection in terms of the Object detection, efficiency, as well as adaptability,considering the fact that v11 has fewer parameters, but still the convergence is better stable over the number of epochs. YOLOv11 is computationally efficient and more accurate than the baseline and any other proposed models so far, on raw data. Although the training on the preprocessed data is very extensive, but training on a fewer epochs clearly indicates that the convergence and model performance are very unstable and not at all converging, which indicates that the pre-processed images are way too fine-graded and leads to the misclassification. In addition, the YOLOv11L model was trained on an 80/20 split and yields similar results to that of the YOLOv11m model on an 80/20 split which has results akin to the YOLOv8 model using an 80/20 split, which highlights that introducing more parameters / complexity to the model has no significant to the training results. This further supports the effectiveness of the YOLOv11 model at adapting to new images in object detection, as it does not require more parameters to detect objects.

Thus, the YOLOv11 model with a 60/40 split is preferred due to its versatility, and adaptability to images without the need for more training images and complex parameters, as shown through the multiple training results from various models.

8 Out-of-Sample Detection

The second task was to do out-of-sample detection. To achieve this, we utilized a pretrained ResNet50 model for feature extraction and a Random Forest classifier for detecting out-of-sample (OOD) images. The key steps involve extracting high-level features from images, training a classifier to distinguish between in-distribution and OOD samples, and evaluating the model's performance using classification metrics and unsupervised clustering.

The pipeline is highlighted below.

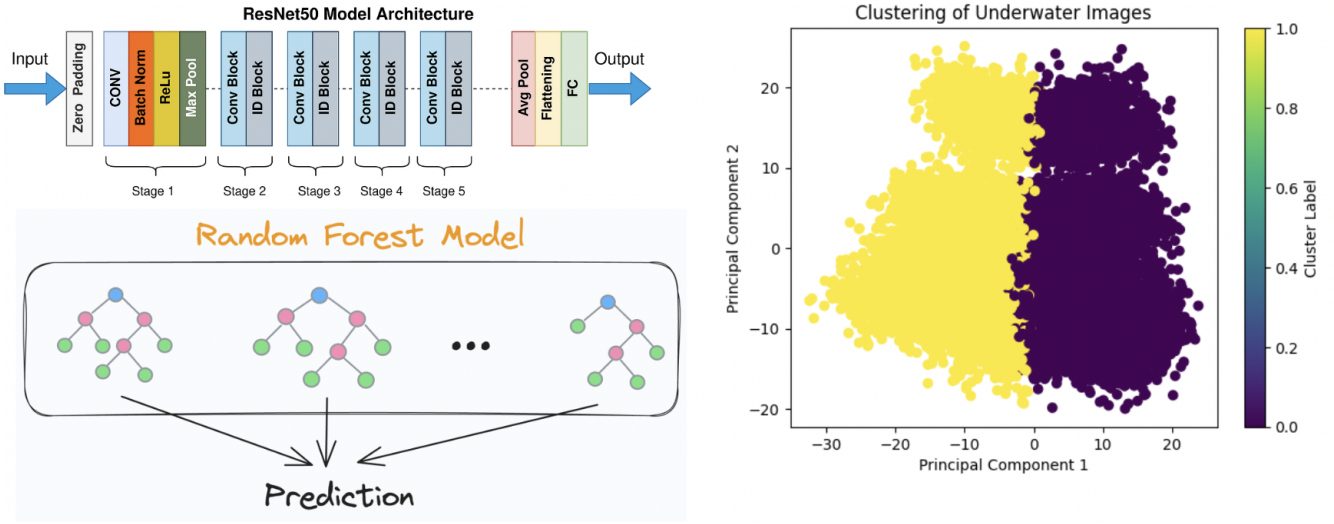

Figure 22: Out-of-Sample Detection pipeline

9 Out-of-Sample Detection Methodology

1. Feature Extraction

- **Model**: ResNet50 pretrained on ImageNet is used as a feature extractor (with fully connected layers removed).
- **Preprocessing**: Images are resized to 224×224 pixels and preprocessed using ResNet50's preprocessing pipeline.
- **Feature Output**: ResNet50 produces a 2048-dimensional feature vector for each image via global average pooling.

2. Dataset Preparation

- Two datasets are created:
  - **In-distribution (0–800m images)** labeled as 0.
  - **Out-of-distribution (0–1300m images)** labeled as 1.
- Features and labels are combined and split into training (70%) and testing (30%) sets.

3. Classification

- A Random Forest classifier is trained on the extracted features to predict whether an image is OOD.
- The classifier's performance is evaluated using:
  - **AUC-ROC**: Measures the ability to distinguish between in-distribution and OOD samples.
  - **Rescaled AUC (sAUC)**: AUC is rescaled to $[-1, 1]$ using $sAUC = 2 \times AUC - 1$.
  - **Accuracy**: Assesses classification performance on the OOD dataset.

4. Clustering Analysis

- **K-Means Clustering**: Features are clustered into two groups to validate separability between in-distribution and OOD samples.
- **Visualization**: Principal Component Analysis (PCA) reduces feature dimensions to 2D for visualization of cluster separation.

Key Outputs

- **AUC-ROC and sAUC Scores**: Quantify the classifier's ability to detect OOD samples.
- **Confusion Matrix**: Provides insights into true/false positive and negative predictions.
- **Clustering Visualization**: Demonstrates natural separability of in-distribution and OOD features.

Results

The process overall yielded the following clusters:

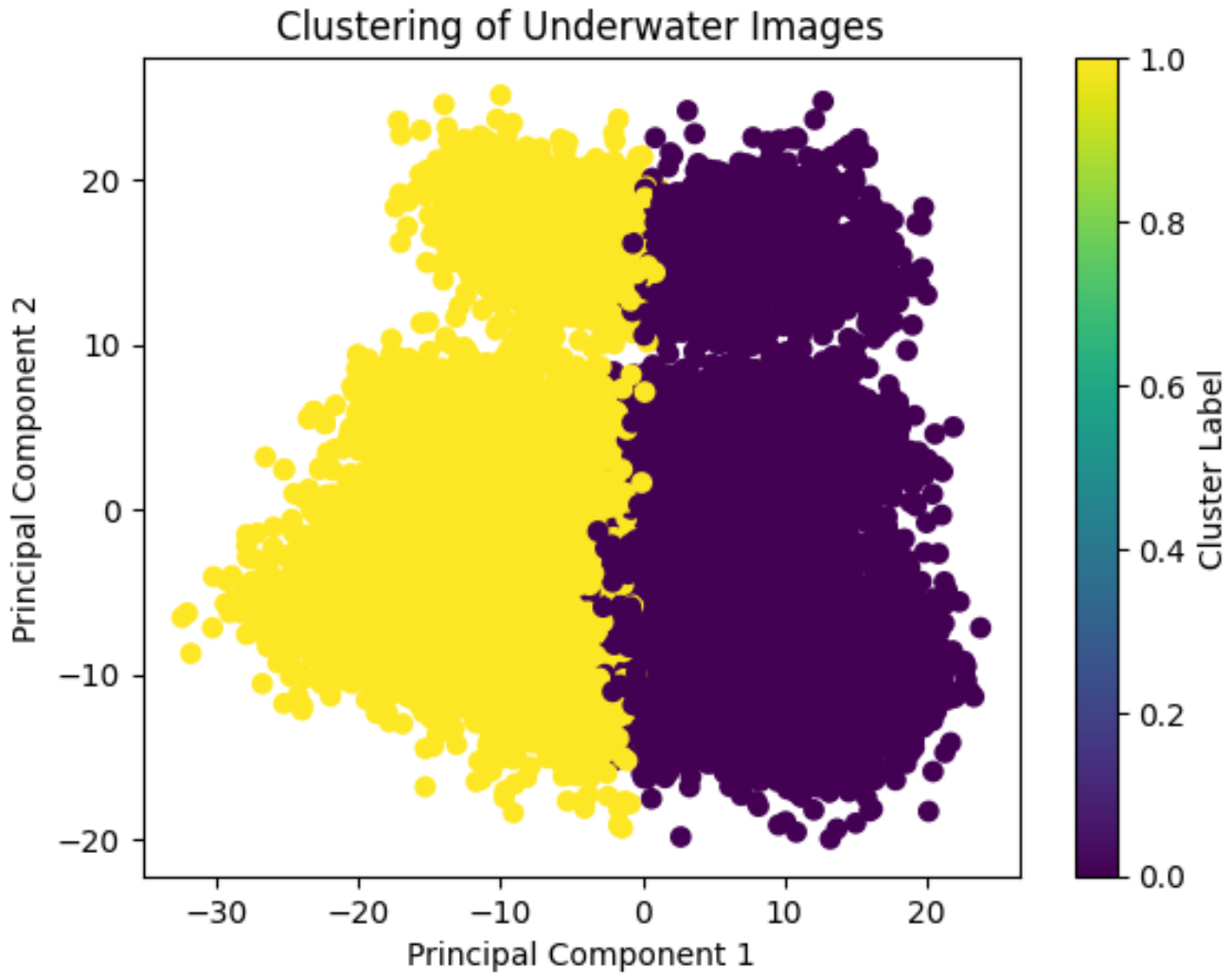

Figure 23: Out-of-Sample Detection Clustering

This resulted in accuracy of about 97.12%, and sAUC of around 91%.
An example of images classified as 0-800m and one as outside this range are plotted below:

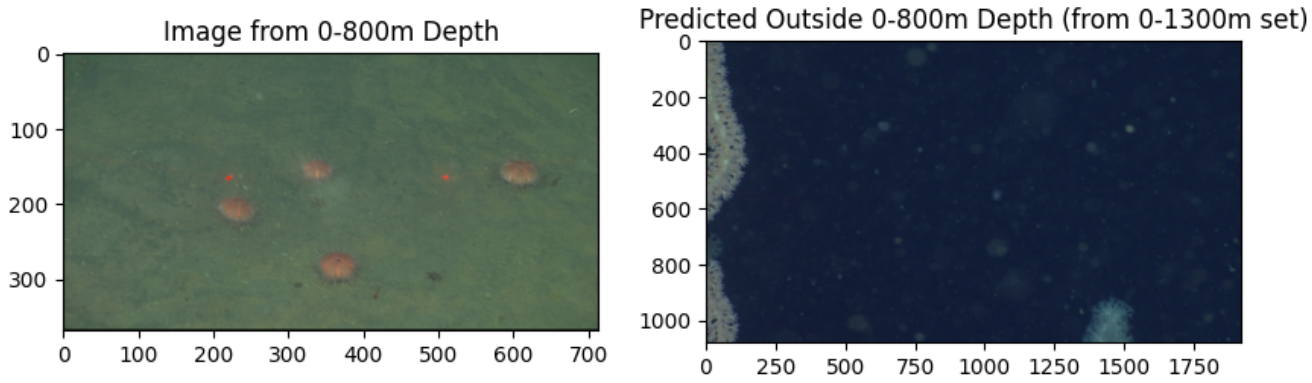

Figure 24: Out-of-Sample Prediction

Here, the detection was quite successful, as evident by the hue of the blue in the right image. The right image has a much darker hue, which is prevalent at lower depths.

10 Conclusion

The marine ecosystem remains one of the least explored and underutilized domains in terms of technological advancements, making FathomNet a groundbreaking initiative in integrating technology with the underwater world. Our study on FathomNet, inspired by research presented at an annual conference bridging computer vision with marine science, has successfully fulfilled its objectives.

The baseline model, published in 2023, utilized the YOLOv8 architecture to classify marine animals in underwater images by either associating them with existing categories or detecting novel samples outside these predefined categories. However, the baseline model exhibited several limitations, including misclassifications, false positives, and inconsistent confidence scores. These challenges were further amplified by the model's difficulty in distinguishing between biological and non-biological objects, frequent overlapping bounding boxes, and inadequacies in localization and non-maximum suppression. These issues underscored the need for improved feature extraction, enhanced training data representation, and fine-tuned model parameters to achieve higher precision and reliability in diverse underwater environments.

In our research, we analyzed an extensive dataset of underwater images and proposed innovative modifications to the YOLOv8 and YOLOv11 architectures, coupled with advanced underwater image enhancement techniques such as wavelet transformations. Among the evaluated models, our proposed YOLOv11 architecture emerged as the benchmark, delivering near-perfect performance statistics. Its robust training on a highly diverse set of images ensures its readiness for real-world applications, such as integration with underwater cameras for species detection and real-time learning from new samples.

Additionally, we tackled the critical task of out-of-sample detection, which aligns with advancements in leveraging depth-based image characteristics. Incorporating depth as an additional dimension enables the model to capture more contextual information, improving detection accuracy across varying marine environments. This advancement also holds promise for studying species variations at different depth levels, offering valuable insights into marine biodiversity.

Our work delivers a reliable, efficient, and highly sophisticated object detection system capable of accurately identifying objects in marine environments. By incorporating depth information and leveraging cutting-edge methodologies, this system sets a new standard in underwater computer vision and offers significant contributions to oceanographic research, advancing the study of marine ecosystems in unprecedented ways.

11 Author contributions

- **Research and experimentation of pre-processing using Underwater Color restoration techniques- Suemy**
- **Feature Extraction, Network Architecture and Model Training - Chhatrapathi Sivaji**

- **Research and experimentation of pre-processing using Underwater Image Enhacement techniques - Felipe**
- **Model Training, Evaluation Metrics and Out-of-Sample Prediction - Gordon Lu**

References

Anwar, S., C. Li, and F. Porikli (2018). *Deep Underwater Image Enhancement*. arXiv: 1807.03528 [cs.CV]. URL: https://arxiv.org/abs/1807.03528.

Binta Islam, S. et al. (2023). "Animal species recognition with deep convolutional neural networks from ecological camera trap images". In: *Animals* 13.9, p. 1526.

Carlevaris-Bianco, N., A. Mohan, and R. M. Eustice (2010). "Initial results in underwater single image dehazing". In: *OCEANS 2010 MTS/IEEE SEATTLE*, pp. 1–8. DOI: 10.1109/OCEANS.2010.5664428.

Chao, L. and M. Wang (2010). "Removal of water scattering". In: *2010 2nd International Conference on Computer Engineering and Technology*. Vol. 2, pp. V2–35–V2–39. DOI: 10.1109/ICCET.2010.5485339.

Ding, G. et al. (2017). "Fish recognition using convolutional neural network". In: *OCEANS 2017-Anchorage*. IEEE, pp. 1–4.

Drews Jr, P. et al. (2013). "Transmission Estimation in Underwater Single Images". In: *2013 IEEE International Conference on Computer Vision Workshops*, pp. 825–830. DOI: 10.1109/ICCVW.2013.113.

Ghani, A. S. A. and N. A. M. Isa (2014). "Underwater image quality enhancement through composition of dual-intensity images and Rayleigh-stretching". In: *2014 IEEE Fourth International Conference on Consumer Electronics Berlin (ICCE-Berlin)*, pp. 219–220. DOI: 10.1109/ICCE-Berlin.2014.7034265.

He, K., J. Sun, and X. Tang (2009). "Single image haze removal using dark channel prior". In: *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1956–1963. DOI: 10.1109/CVPR.2009.5206515.

Hummel, R. A. (1975). "Image Enhancement by Histogram transformation". In: *Computer Graphics and Image Processing* 6, pp. 184–195. URL: https://api.semanticscholar.org/CorpusID:122868774.

Li, C. et al. (2016). "Single underwater image restoration by blue-green channels dehazing and red channel correction". In: *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1731–1735. URL: https://api.semanticscholar.org/CorpusID:1869509.

Ma, Z. and C. Oh (2022). *A Wavelet-based Dual-stream Network for Underwater Image Enhancement*. arXiv: 2202.08758 [cs.CV]. URL: https://arxiv.org/abs/2202.08758.

Mancini, M. et al. (2020). "Towards recognizing unseen categories in unseen domains". In: *European Conference on Computer Vision*. Springer, pp. 466–483.

Martinka, J. (2021). "Neural networks for wood species recognition independent of the colour temperature of light". In: *European Journal of Wood and Wood Products* 79.6, pp. 1645–1657.

Nathan Silberman Derek Hoiem, P. K. and R. Fergus (2012). "Indoor Segmentation and Support Inference from RGBD Images". In: *ECCV*.

Okayasu, K. et al. (2019). "Vision-based classification of mosquito species: Comparison of conventional and deep learning methods". In: *Applied sciences* 9.18, p. 3935.

Peng, Y.-T. and P. C. Cosman (2017). "Underwater Image Restoration Based on Image Blurriness and Light Absorption". In: *IEEE Transactions on Image Processing* 26.4, pp. 1579–1594. DOI: 10.1109/TIP.2017.2663846.

Pizer, S. et al. (1990). "Contrast-limited adaptive histogram equalization: speed and effectiveness". In: *[1990] Proceedings of the First Conference on Visualization in Biomedical Computing*, pp. 337–345. DOI: 10.1109/VBC.1990.109340.

Rodríguez, A. C. et al. (2024). "Recognition of Unseen Bird Species by Learning from Field Guides". In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 1742–1751.

Song, W. et al. (2018). "A Rapid Scene Depth Estimation Model Based on Underwater Light Attenuation Prior for Underwater Image Restoration". In: *Pacific Rim Conference on Multimedia*. URL: https://api.semanticscholar.org/CorpusID:52303790.

The FathomNet2023 Competition Dataset (2023). arXiv: 2307.08781 [cs.CV]. URL: https://arxiv.org/abs/2307.08781.

Wang, Y. et al. (2019). *An Experimental-based Review of Image Enhancement and Image Restoration Methods for Underwater Imaging*. arXiv: 1907.03246 [eess.IV]. URL: https://arxiv.org/abs/1907.03246.

Wen, H. et al. (2013). “Single underwater image enhancement with a new optical model”. In: *2013 IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 753–756. DOI: 10.1109/ISCAS.2013.6571956.

Xu, L. et al. (2019). “Deep learning for marine species recognition”. In: *Handbook of deep learning applications*, pp. 129–145.

Yang, H.-Y. et al. (2011). “Low Complexity Underwater Image Enhancement Based on Dark Channel Prior”. In: *2011 Second International Conference on Innovations in Bio-inspired Computing and Applications*, pp. 17–20. DOI: 10.1109/IBICA.2011.9.