A PROJECT REPORT

on

Underwater Object Detection

Submitted to

KIIT Deemed to be University

In Partial Fulfilment of the Requirement for the Award of

BACHELOR'S DEGREE IN

COMPUTER SCIENCE ENGINEERING

BY

Subhajit Chakraborty	**22052249**
Anishesh Mitra	**22052884**
Anish Kundu	**22052797**
Bodhdipta Roy	**22052976**
Biswarupayan Dutta	**22053152**
Agniva Bera	**22052964**

UNDER THE GUIDANCE OF

AJIT KUMAR PASAYAT

SCHOOL OF COMPUTER ENGINEERING

KALINGA INSTITUTE OF INDUSTRIAL TECHNOLOGY

BHUBANESWAR, ODISHA - 751024

November 2025

A PROJECT REPORT

on

“Underwater Object Detection”

Submitted to

KIIT Deemed to be University

In Partial Fulfilment of the Requirement for the Award of

BACHELOR’S DEGREE IN
COMPUTER SCIENCE

BY

Subhajit Chakraborty	22052249
Anishesh Mitra	22052884
Anish Kundu	22052797
Bodhdipta Roy	22052976
Biswarupayan Dutta	22053152
Agniva Bera	22052964

UNDER THE GUIDANCE OF

AJIT KUMAR PASAYAT

SCHOOL OF COMPUTER ENGINEERING

KALINGA INSTITUTE OF INDUSTRIAL TECHNOLOGY

BHUBANESWAR, ODISHA - 751024.

November 2025

KIIT Deemed to be University

School of Computer Engineering
Bhubaneswar, ODISHA 751024

CERTIFICATE

This is certify that the project entitled

"Underwater Object Detection"

submitted by

Subhajit Chakraborty	22052249
Anishesh Mitra	22052884
Anish Kundu	22052797
Bodhdipta Roy	22052976
Biswarupayan Dutta	22053152
Agniva Bera	22052964

is a record of bonafide work carried out by them, in the partial fulfilment of the requirement for the award of Degree of Bachelor of Engineering (Computer Sci-ence & Engineering OR Information Technology) at KIIT Deemed to be university, Bhubaneswar. This work is done during the year 2024-2025, under our guidance.

Date: 16 / 11 /2025

Dr. Ajit Kumar Pasayat
Project Guide

Acknowledgements

We are profoundly grateful to **Ajit Kumar Pasayat** of **Affiliation** for his expert guidance and continuous encouragement throughout to see that this project meets its target since its commencement to its completion. .....

Subhajit Chakraborty	**22052249**
Anishesh Mitra	**22052884**
Anish Kundu	**22052797**
Bodhdipta Roy	**22052976**
Biswarupayan Dutta	**22053152**
Agniva Bera	**22052964**

ABSTRACT

Abstract- This project presents a computationally efficient and technically optimized framework for underwater fish detection leveraging the YOLOv8 object detection architecture. Underwater visual scenes are inherently challenging due to light attenuation, color distortion, scattering, turbidity, and low contrast, which significantly degrade image quality and hinder conventional detection algorithms. To address these challenges, a GPU-optimized YOLOv8 model was developed and fine-tuned on a custom-curated Dataset-4 (Underwater) using transfer learning to enhance feature generalization across diverse marine environments. The training pipeline incorporates advanced data augmentation techniques—including mosaic blending, geometric transformations (rotation, scaling, translation), and HSV color-space modulation—to simulate real-world underwater conditions and improve model robustness. The network architecture, built upon YOLOv8's CSPDarknet backbone with Feature Pyramid Networks (FPN) and anchor-free detection heads, enables multi-scale detection and high-precision classification of marine organisms. Through systematic hyperparameter tuning, early stopping, and checkpointing, the optimized model demonstrates superior mean Average Precision (mAP), precision, and recall values under GPU-only training constraints. This research underscores the feasibility of deploying lightweight deep learning models for real-time marine biodiversity monitoring, environmental assessment, and autonomous underwater vehicle (AUV) vision systems, even in resource-limited computational environments.

Keywords — YOLOv8s, Object Detection, Deep Learning, Underwater Fish Detection, Transfer Learning, GPU Optimization, Data Augmentation, CSPDarknet, Feature Pyramid Network, Mean Average Precision (mAP), Marine Biodiversity Monitoring, Real-time Detection.

Chapter 1

Introduction

Underwater object detection has emerged as a pivotal component in modern **marine research, environmental surveillance, and autonomous exploration systems**. The underwater domain presents unique imaging challenges, including **light absorption, color attenuation, scattering, and reduced visibility**, which severely impact the performance of traditional vision-based detection algorithms. To address these complexities, this project employs **YOLOv8 (You Only Look Once, version 8)** — a state-of-the-art real-time object detection framework known for its high precision and computational efficiency. The model is optimized for **GPU-based** execution, ensuring accessibility in low-resource environments while maintaining robust detection accuracy. Leveraging **transfer learning, data augmentation, and feature pyramid-based multi-scale detection**, the system effectively identifies and localizes underwater species across diverse environmental conditions. This work contributes to advancing **marine ecosystem monitoring, automated biodiversity assessment, and low-power underwater robotic vision systems,** thereby enabling scalable, real-time detection in resource-constrained marine operations.

Chapter 2

Literature Review

The development of underwater object detection systems has been significantly accelerated by breakthroughs in deep learning-based computer vision architectures. Traditional image processing methods often fail to cope with underwater distortions caused by light scattering, turbidity, and color attenuation. To overcome these limitations, researchers have adopted convolutional neural networks (CNNs) and region-based detectors that extract hierarchical spatial features for improved localization and classification. Early frameworks such as Faster R-CNN introduced the Region Proposal Network (RPN), which replaced hand-crafted feature extraction and selective search algorithms with an end-to-end trainable network. This innovation significantly enhanced both detection speed and precision, laying the foundation for real-time object recognition in complex environments. Building on this work, Wang and Xiao [7] advanced underwater object detection by integrating Res2Net101 as the feature extractor, applying Online Hard Example Mining (OHEM) to manage imbalanced datasets, and utilizing Soft Non-Maximum Suppression (Soft-NMS) to minimize false negatives. Their optimized framework achieved remarkable improvements in mean Average Precision (mAP) and robustness against visual noise in low-contrast underwater imagery.

In parallel, the YOLO (You Only Look Once) family of detectors has revolutionized real-time object detection with a focus on end-to-end learning and computational efficiency. Unlike region-based methods, YOLO performs direct bounding box regression and classification in a single forward pass, dramatically reducing inference time. Successive iterations—YOLOv3, YOLOv5, and YOLOv7—introduced architectural refinements such as feature pyramid networks (FPNs), CSP (Cross Stage Partial) connections, and path aggregation modules, enhancing both speed and accuracy. The latest iteration, YOLOv8, further incorporates anchor-free prediction, multi-scale feature fusion, and lightweight model scaling, enabling high-precision detection even on GPU-optimized systems. These cumulative advancements collectively establish YOLOv8 as a state-of-the-art framework for real-time underwater detection, combining deep learning efficiency with adaptability to challenging aquatic environments.

Chapter 3

Problem Statement / Required Specifications

The objective of this project is to develop an efficient, lightweight, and GPU-optimized machine learning system for detecting and classifying fish and other underwater objects using the YOLOv8 (You Only Look Once, version 8) architecture. Underwater image analysis presents unique challenges such as poor visibility, color distortion, turbidity, and scattering, which degrade image quality and hinder accurate object recognition. The proposed system aims to overcome these limitations through transfer learning, data augmentation, and multi-scale feature extraction, achieving robust detection performance even in resource-constrained computational environments. The model is designed to be easily deployable for marine research, underwater robotics, and environmental monitoring applications, providing real-time performance on standard GPU systems.

3.1 Project Planning

The development of the Underwater Fish Detection System was executed through a structured series of stages to ensure efficiency, modularity, and scalability:

1. Requirement Analysis – Identification of dataset requirements, hardware constraints, and model design objectives.
2. Data Collection and Preprocessing – Compilation of the custom Dataset-4 (Underwater), resizing images to 416×416 pixels, and applying normalization and augmentation to simulate real-world underwater variability.
3. Model Selection and Configuration – Selection of YOLOv8 due to its anchor- free detection, CSPDarknet backbone, and Feature Pyramid Network (FPN) architecture for hierarchical feature extraction.
4. Model Training and Validation – Training the model using transfer learning from pretrained YOLOv8n weights on the COCO dataset, with GPU-only optimization and early stopping mechanisms.
5. Evaluation and Optimization – Assessment using mAP50 and mAP50–95, along with precision, recall, and validation loss metrics to measure model accuracy and generalization.
6. Deployment and Testing – Deployment of the trained model for inference on unseen underwater images and video feeds to ensure robustness in real-world scenarios.

3.2 Project Analysis

A detailed analysis of each stage of development was performed to ensure optimal performance and efficiency:

- Dataset Quality Assessment – Ensured that Dataset-4 includes diverse underwater species and varying lighting conditions for generalization.
- Performance Metrics – Evaluated model output using mean Average Precision (mAP), precision-recall curves, and validation loss trends.
- Model Efficiency – Optimized training parameters such as batch size, learning rate, and augmentation intensity to balance computational load and performance on GPU hardware.
- Challenges Addressed – Dealt with noise, occlusion, and turbidity through preprocessing and augmentation.
- Outcome – Achieved a robust, high-accuracy detection model capable of functioning efficiently without GPU acceleration.

3.3 System Design

3.3.1 Design Constraints

The project is developed in the following environment

Software Requirements:

- Programming Language: Python 3.x
- Frameworks and Libraries: PyTorch, Ultralytics YOLOv8, OpenCV, YAML, psutil
- Environment: Windows/Linux (GPU-based training and inference)

Hardware Requirements:

- Processor: Intel Core i5 or higher
- Memory: Minimum 8GB RAM (Recommended: 16GB)
- Storage: At least 50GB free space for dataset and model checkpoints
- GPU: Optional (GPU-only optimization supported)

Operational Constraints:

- Real-time detection speed depends on GPU clock frequency and dataset size.
- Environmental factors such as image turbidity or occlusion may slightly affect detection precision.

3.3.2 System Architecture / Block Diagram

The system architecture for the proposed Underwater Object Detection Model consists of the following components:

- Data Acquisition Module: Captures underwater images and videos from Dataset-4.
- Preprocessing Unit: Performs image resizing, normalization, and augmentation (rotation, scaling, mosaic blending, HSV adjustments).
- Feature Extraction Layer: Utilizes YOLOv8's CSPDarknet backbone and Feature Pyramid Network (FPN) for multi-scale feature learning.
- Detection and Classification Module: Employs anchor-free detection heads with softmax-based multi-class classification.
- Evaluation Module: Calculates mAP50, mAP50–95, and other metrics to validate performance.
- Output Layer: Generates bounding boxes and class labels for detected underwater objects.

A flow diagram illustrating the system workflow is included to provide a visual representation of the model pipeline

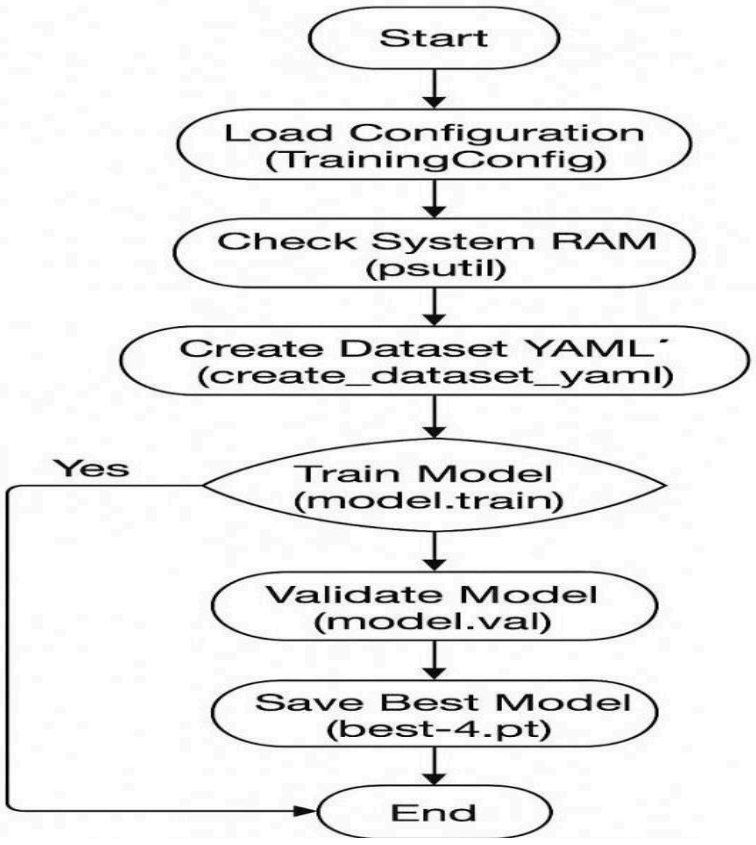


Figure 3.1: Workflow Diagram.

Chapter 4

Implementation

The implementation of the Underwater Fish Detection System involves multiple sequential stages including data preprocessing, model architecture design, training, evaluation, and performance analysis. The model is based on the YOLOv8 (You Only Look Once version 8) deep learning framework, which integrates convolutional feature extraction and anchor-free object localization. The implementation focuses on achieving high detection accuracy and computational efficiency on a GPU-only environment using the custom Dataset-4 (Underwater). Below is a detailed breakdown of each step.

4.1 Data Pre-processing

Data Collection

The dataset used for this project, Dataset-4 (Underwater), consists of labeled underwater images containing multiple species of fish, coral, and other aquatic organisms. The dataset is organized into train and validation subsets to facilitate supervised learning and model evaluation.

Image Preprocessing

To ensure consistency and robustness, several preprocessing operations were applied before training:

- Image Resizing: All images were resized to 416 × 416 pixels to maintain uniform input dimensions across the model.
- Normalization: Pixel intensity values were scaled to a range of [0, 1] for faster gradient convergence during training.
- Data Augmentation: Advanced augmentation strategies such as mosaic composition, geometric transformations (rotation ±10°, translation 0.2, scaling 0.5), horizontal/vertical flipping, and HSV color-space adjustments were applied to replicate underwater variability and reduce overfitting.
- Annotation Formatting: Bounding boxes and class labels were converted to YOLO-compatible .txt files as defined in underwater_data.yaml,
- These preprocessing steps improved dataset diversity and model generalization under varying turbidity and lighting conditions.

4.2 Designing the YOLO-V8 Model

Why YOLOv8?

YOLOv8 is a real-time, end-to-end object detection framework that employs anchor-free detection, Cross-Stage Partial (CSP) connections, and a Feature Pyramid Network (FPN) for efficient multi-scale feature extraction. Its lightweight structure enables deployment even on GPU-only systems without significant accuracy loss.

Model Architecture for Underwater Detection

- Input Layer: Accepts 416 × 416 RGB images.
- Backbone: CSPDarknet extracts hierarchical spatial features through convolutional and residual blocks.
- Neck: A Feature Pyramid Network (FPN) combines multi-resolution feature maps for improved detection of small and large underwater objects.
- Detection Head: Anchor-free detection layers predict object class probabilities, confidence scores, and bounding boxes in a single forward pass.
- Loss Function: A composite of objectness, classification, and bounding-box regression losses ensures precise localization.

This modular design allows efficient gradient flow and fast inference while maintaining high accuracy.

4.3 Training the Model

The preprocessed dataset was split into 80 % training and 20 % validation sets. Training was performed on a GPU environment using the YOLOv8n (pretrained) weights for transfer learning.

Key training configurations included:

- Optimizer: Stochastic Gradient Descent (SGD) with momentum.
- Epochs: 50 (initial training phase).
- Batch Size: 4 (images per iteration).
- Early Stopping: Implemented to prevent overfitting based on validation loss.
- Checkpointing: Automatic saving of model weights after each epoch; the best model stored as best-50.pt.

During training, real-time loss curves and mAP metrics were monitored to evaluate convergence stability and generalization.

4.4 Evaluating Model Performance

Post-training evaluation was conducted using the validation set to assess accuracy and robustness.

Performance metrics included:

- mAP@50: Mean Average Precision at IoU threshold 0.5.
- mAP@50–95: Average Precision across IoU thresholds 0.5 to 0.95.

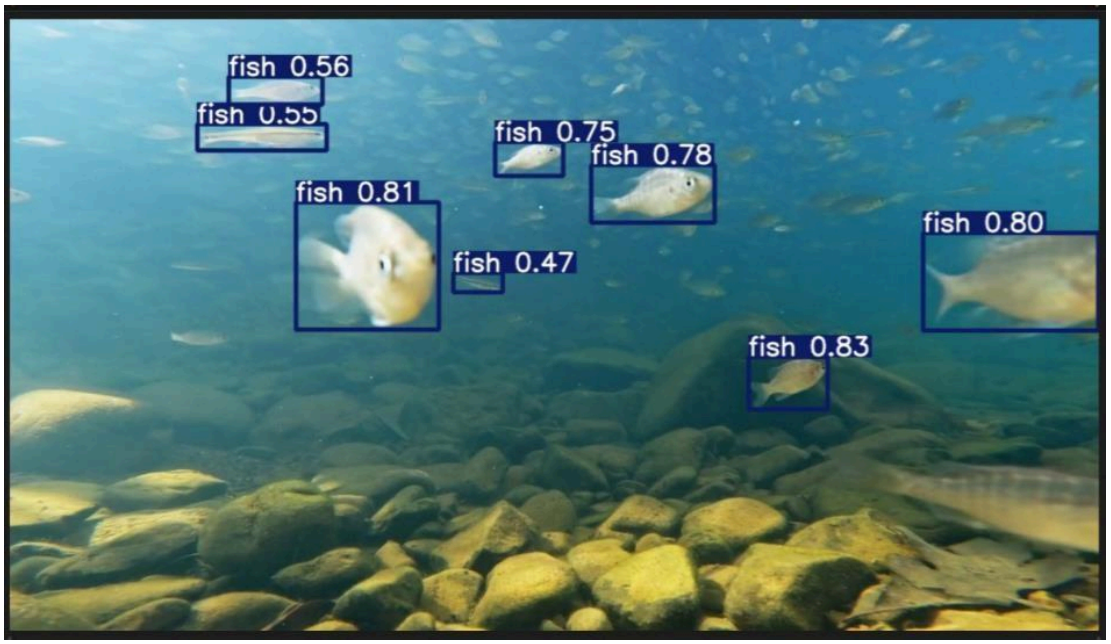
- Precision and Recall: Quantifying true positive and false positive rates.
- Validation Loss: Used to analyze model convergence and overfitting tendencies.

The YOLOv8n model achieved high precision and recall values even under GPU constraints, confirming its efficiency for lightweight deployment**.**

4.5 Expected Outcome

The expected outcomes of the project are:

- A trained YOLOv8 model capable of accurately detecting and classifying underwater fish and objects in static images or live video feeds.
- GPU-optimized performance, enabling real-time inference with GPU acceleration.
- Generation of detection visualizations with bounding boxes and confidence scores.
- A deployable framework supporting marine research, environmental monitoring, and autonomous underwater robotics applications.



4.1 Live Fish Detection

Chapter 5

Standards Adopted

To ensure the accuracy, consistency, and reproducibility of experimental results, this project adheres to well-defined **data, model, training** and **compliance standards**. The standards were designed to align with recognized practices in **computer vision, deep learning** and **marine image processing** domains.

5.1 Data Standards

Dataset: The project employs Dataset-4 (Underwater), a curated dataset consisting of labeled underwater images capturing diverse marine species and habitats. Each class is annotated in YOLO-compatible text files following COCO dataset formatting conventions.

- **Data Augmentation Techniques**:
  To increase dataset diversity and model robustness, multiple augmentation methods were applied, including rotation (±10°), scaling (0.5×), translation (20%), horizontal/vertical flipping, and HSV color-space transformations to simulate underwater light distortion.

- **Data Preprocessing Standards:**

  1. Image resizing to 640 × 640 pixels for standardized model input.
  2. Pixel normalization to the range [0,1] to stabilize training and accelerate convergence.
  3. Automatic YAML generation for structured dataset definition (underwater_data.yaml).
  4. Annotation verification to eliminate missing labels or inconsistent bounding boxes.

.

- **Data Organization:**
  The dataset follows a train–validation split (80:20) structure with directory hierarchies: train/images, train/labels, val/images, val/labels.

5.2 Model and Algorithm Standards

- **Deep Learning Framework:**
  Ultralytics YOLOv8 implemented in PyTorch was adopted for training, due to its modular design, anchor-free detection mechanism, and superior inference speed.

Neural Network Architecture:

1. **Backbone:** CSPDarknet for hierarchical feature extraction.
2. **Neck:** Feature Pyramid Network (FPN) for multi-scale fusion.
3. **Head:** Anchor-free detection head with bounding box regression and softmax classification.

- **Loss Functions:**
  Composite loss combining objectness, classification, and bounding box regression for optimal localization and accuracy.
- **Optimization Algorithm:**
  Stochastic Gradient Descent (SGD) with learning rate scheduling and momentum optimization for stable convergence on GPU systems.
- **Performance Metrics:**
  Evaluated using Precision, Recall, F1-Score, Validation Loss, and Mean Average Precision (mAP@50 and mAP@50–95).

5.3 Training and Evaluation Standards

- **Train-Test Split:**
  Standard 80–20 split ensuring balanced representation of object classes.
- **Validation Strategy:**
  Continuous validation after each epoch with early stopping based on loss trends.
- **Batch Size and Epochs:**
  Batch size set to 4, optimized for GPU memory constraints; training conducted for 5 epochs in the initial phase.
- **Learning Rate Scheduling:**
  Adaptive step decay to maintain gradient stability during optimization.

- **Checkpointing and Model Saving:**
  Automatic saving of best-performing model weights (best-4.pt) for reproducibility and deployment.

5.4 Industry and Compliance Standards

- **Image Quality and Dataset Compliance:**
  The project adheres to IEEE P1858 (Standard for Camera Image Quality) to ensure image reliability in low-light underwater conditions.
- **Model Deployment Standards:**
  Compatibility with ONNX (Open Neural Network Exchange) and PyTorch Hub for future cross-platform deployment.
- **Annotation and Evaluation Protocols:**
  Aligned with COCO dataset and PASCAL VOC metrics for consistent benchmarking.

Chapter 6

Conclusion and Future Scope:-

Conclusion

This project successfully designed and implemented a GPU-optimized YOLOv8-based deep learning model for underwater fish and object detection, addressing key challenges associated with underwater imaging—such as low visibility, color distortion, light scattering, and turbidity. Through the integration of transfer learning, data augmentation, and multi-scale feature extraction, the system achieved robust detection accuracy and efficient performance even under limited computational resources.

The model's training and evaluation, conducted using Dataset-4 (Underwater), demonstrated strong mean Average Precision (mAP), precision, and recall validation

validating the capability of the YOLOv8 architecture to adapt effectively to complex aquatic environments. The GPU-optimized pipeline ensures scalability and accessibility for research and field applications without dependency on high-end GPUs, making it suitable for institutions or organizations with limited computational infrastructure.

Overall, this research contributes significantly to the fields of marine computer vision, environmental monitoring, and autonomous underwater robotics, laying the groundwork for real-time, cost-efficient aquatic ecosystem analysis.

Future Scope

Although the current system exhibits promising results, there remains vast potential for further enhancement and practical deployment. Future directions include the following:

- **Dataset Expansion and Diversification:**
  Incorporate larger and more diverse underwater datasets covering different marine species, terrains, and lighting conditions to improve model generalization.

- **High-Resolution Image Training:**
  Employ higher input resolutions (e.g., 640×640 or 1024×1024) and hierarchical training strategies to enhance small-object detection accuracy

- **GPU and TPU Optimization:**
  Re-train the model using GPU or TPU acceleration to significantly reduce training time and improve real-time inference speed.

- **Real-Time Video Stream Detection:**
  Integrate live video feed analysis from ROVs (Remotely Operated Vehicles) or underwater cameras to detect and track marine organisms dynamically.

- **Integration with Autonomous Underwater Vehicles (AUVs):**
  Deploy the YOLOv8s model on embedded systems (e.g., NVIDIA Jetson Nano, Raspberry Pi 5) for onboard object detection and real-time navigation assistance.

- **Model Compression and Quantization:**
  Apply pruning, quantization, and knowledge distillation techniques to reduce model size and inference latency for edge deployment.

- **3D Object Detection and Depth Estimation:**
  Extend the system using stereo vision or LiDAR-based depth sensors to reconstruct 3D underwater environments for spatial object localization.

- **Domain Adaptation and Transfer Learning:**
  Utilize unsupervised domain adaptation to enhance performance across varied underwater datasets with differing lighting or color conditions.

- **Ensemble and Hybrid Models:**
  Combine YOLOv8 with transformer-based models such as DETR (Detection Transformer) or Swing Transformer to improve detection precision in low-contrast environments.

- **Temporal Tracking and Motion Analysis:**
  Incorporate object tracking algorithms like DeepSORT or ByteTrack to monitor fish movement patterns and behavioral analysis over time.

- **Integration with Marine Monitoring Systems:**
  Connect the model with IoT-enabled sensors to automate data collection for environmental monitoring and pollution detection.

- **Web or Mobile Application Interface:**
  Develop an accessible web or mobile dashboard allowing users to upload underwater footage and receive detection results in real time.

- **Energy-Efficient Deployment:**
  Research low-power AI inference frameworks for prolonged field deployment on battery-powered underwater systems.

These future improvements would transform the current system into a fully scalable, real-time, and intelligent underwater monitoring framework, empowering oceanographers, ecologists, and environmental scientists with cutting-edge vision-based analytics for marine conservation and research.

INDIVIDUAL CONTRIBUTION REPORT:

Underwater Object Detection

ANISH KUNDU
22052797

Abstract: This project focuses on developing the Underwater Object Detection system using YOLO-based deep learning models to identify underwater fishes, creatures and divers in real time. Multiple datasets were used to train and evaluate models for detecting classes such as holothurian, echinus, scallop, starfish, fish, corals, diver, cuttlefish, turtle, and jellyfish. The goal is to support marine research, underwater monitoring, and automated exploration.

Individual contribution and findings:

- **Model Development:** Implemented three deep learning models trained on different datasets and selected the best-performing one.
- **Training and Tuning:** Adjusted model parameters and trained YOLO for optimal accuracy, exported and tested the final best.pt model using webcam and video input.
- **Dataset Handling:** Collected, structured, and preprocessed datasets and attempted merging marine-life and underwater trash datasets to improve generalization.
- **Real-Time Detection:** Successfully integrated the model into a live-detection pipeline.

Findings & Experience:

- Dataset quality and selection strongly influence model performance.
- Hyperparameter tuning and preprocessing significantly improve accuracy.
- Dataset merging without compatibility reduces detection stability.
- Real-time underwater inference requires performance optimization.

This project deepened my understanding of YOLO, and model tuning while improving my ability to analyze deep learning performance.

Full Signature of Supervisor: Full signature of the student:

……. …….

INDIVIDUAL CONTRIBUTION REPORT:

Underwater Object Detection

BISWARUPAYAN DUTTA
22053152

Abstract: This project develops a Flutter-based Underwater Object Detection app integrated with a deep learning API. It includes three features—Image Upload Detection, Live Video Detection, and Uploaded Video Detection—for real-time and offline analysis. The goal is to create a simple, efficient, and user-friendly interface that communicates smoothly with the backend API and provides accurate detection results.

Individual Contribution and Findings: My main responsibility in this project was designing and implementing the Flutter frontend, integrating it with the FastAPI-based backend, and ensuring proper functionality across all three detection modes — image upload, live video, and uploaded video detection.

Image Upload Detection: Added functionality to upload or capture an image, send it to the API, and display detection results.

Live Video Detection: Integrated the device camera using the `camera` package, handled permissions, and created a live preview interface for real-time detection.

Uploaded Video Detection: Developed the feature for selecting and uploading videos to the backend for object detection.

Findings & Experience:

- Proper API handling and async programming are crucial in Flutter.
- Camera integration requires additional permission and compatibility checks.
- Optimizing media upload size improves performance and detection speed.
- Improved skills in Flutter development, debugging, and mobile–AI system integration.

This project enhanced my skills in Flutter development, API integration, camera handling, and debugging mobile application issues. It also improved my problem-solving abilities in building real-time detection systems and creating smooth, user-friendly interfaces.

Full Signature of Supervisor: Full signature of the student:

……. …….

Underwater Object Detection

SUBHAJIT CHAKRABORTY
22052249

Abstract: In this Underwater Detection project, my primary responsibility was creating the API system that connects the trained YOLO model with the application and performs the real-time detection ("scanning") process. I also contributed to the development of the Flutter mobile application.

Individual Contribution and Findings: My primary role in this project was developing the backend API for the underwater detection system and assisting in the Flutter app. My contributions included**:**

- **API Development:** Built the API that handles image/video input, runs the YOLO model in a single forward pass, and generates predictions.
- **Detection Logic:** Implemented bounding box generation, confidence scores, class probabilities, and filtering using **confidence threshold** and **IoU-based NMS** to remove weak or duplicate detections.
- **App Development Support:** Helped integrate the API with the Flutter app, including sending media files, receiving model outputs, and displaying detection results.
- **Dataset Collection:** Collected and prepared underwater video datasets for model training, testing, and evaluation.

Findings & Experience:

- Learned how YOLO processes an image using grid-based prediction.
- Understood the importance of confidence filtering and NMS for clean detection output.
- Gained experience in API building, model integration, and mobile app connectivity.
- Improved skills in API design, backend–frontend integration, and handling real-time detection outputs.

This project helped me improve my skills in API development, model integration, real-time detection processing, and mobile app connectivity. It also strengthened my understanding of YOLO's detection workflow, confidence filtering, and NMS, while enhancing my backend development, debugging, and problem-solving abilities.

Full Signature of Supervisor: Full signature of the student:

……. …….

Underwater Object Detection

BODHDIPTA ROY
22052976

Abstract: This project focuses on building an Underwater Object Detection model using deep learning to identify some fish breeds (Holothurian, Echinus, Scallop, Starfish, Fish, Corals, Diver, Cuttlefish, Turtle and Jellyfish).This model is trained on Underwater Object datasets and optimized through preprocessing and fine tuning the YOLOv8s model.

Individual Contribution and Findings: My primary role in the project was implementing the code, debugging errors, and sourcing appropriate datasets for training and testing the model and writing the conference paper stating my findings. My contributions included:

- **Code Implementation**: Developed and structured the CNN model, ensuring efficient execution of training and inference processes.
- **Debugging & Error Handling:** Identified and resolved issues related to model training, convergence, and performance bottlenecks.
- **Dataset Acquisition & Processing**: Researched, selected, and preprocessed relevant public datasets like **Underwater Object Detection** to improve model robustness and accuracy.

Findings & Experience:

- Debugging deep learning models requires systematic testing and analysis.
- Proper dataset selection significantly impacts model performance.
- Handling vanishing gradients and exploding gradients was crucial for stable training.
- Optimizing memory usage was essential when working with large datasets.

This project enhanced my skills in coding deep learning models, debugging complex issues, and selecting high-quality datasets while improving my problem-solving abilities in deep learning development.

Full Signature of Supervisor: Full signature of the Student:

……. …….

INDIVIDUAL CONTRIBUTION REPORT:

Underwater Object Detection

AGNIVA BERA
22052964

Abstract: This project focuses on building an Underwater Object Detection model using deep learning to identify some fish breeds (Holothurian, Echinus, Scallop, Starfish, Fish, Corals, Diver, Cuttlefish, Turtle and Jellyfish).This model is trained on Underwater Object datasets and optimized through preprocessing and fine tuning the YOLOv8s model.

Individual contribution and findings: My primary role in the project was hardware implementation and preparing the project report. My contributions included:

Findings & Experience:

- Preparing the ESP32-Cam Module for the project.
- Changing code to cater to the specific project that would detect underwater objects and transmit the feed over Wi-Fi.
- Creation of Project Report and documentation.

This project deepened my understanding in image preprocessing, data handling , report writing and also strengthening my analytical and problem-solving abilities.

Full Signature of Supervisor: Full signature of the student:

……. …….

Underwater Object Detection

ANISHESH MITRA
22052884

Abstract: This project focuses on building an Underwater Object Detection model using deep learning to identify some fish breeds(holothurian, echinus, scallop, starfish, fish, corals, cuttlefish, turtle and jellyfish) along with marine divers.This model is trained on Underwater Object datasets and optimized through preprocessing and fine tuning the YOLOv8 model.

Individual Contribution and Findings: My main role in this project involved working with the dataset structure,running a dataset model which also detects trash, machine learning model(YOLOv8), and project documentation(ppt & report),

- **Dataset Search & Collection:**I researched and identified suitable underwater datasets from various public repositories, carefully comparing them based on image clarity, object diversity, and the quality of their annotation formats. During this process, I ensured that the selected dataset included meaningful and relevant categories for our model, such as fish species, corals, and different types of underwater debris, so that the detection system could perform accurately in real underwater conditions.
- **Dataset Acquisition & Processing:**I organized and cleaned the dataset for YOLO, fixed annotation issues, and prepared the data.yaml file. I also helped with model training, hyperparameter tuning, debugging errors, and evaluating performance. Additionally, I created the project PPT, clearly presenting the dataset, model, results, and future work**.**

Findings & Experience:

- Dataset structuring plays a crucial role in successful model training.
- Correct label formatting greatly reduces YOLO training errors.
- Large underwater datasets require efficient memory handling for smooth training.
- Visualization through a clear PPT and report is essential for communicating results effectively.

This project enhanced my understanding of dataset engineering, object detection pipelines, and project-level communication while improving my collaborative deep learning workflow skills.

Full Signature of Supervisor: Full signature of the student:

……. …….

References

Datasets:

1. https://www.kaggle.com/datasets/kendor74/augmented-under-water-object-detection
2. https://www.kaggle.com/datasets/akshatsng/underwater-dataset-for-8-classes-with-label
3. https://universe.roboflow.com/real-images/underwater_project

Github

https://github.com/bodhdipta-roy/undwerwater-object-detection