Oyster Detection Using YOLOv8

AUTHORS

Hamid El Karim
Logan Kelsch
Dominick Rodriguez
Mentors: Dr. Yuanwei Jin and Dr. Enyue Lu
Salisbury University — REU 2025

AFFILIATIONS

Salisbury University
NSF REU
USDA NIFA

INTRODUCTION

Oyster farming is critical for aquaculture and water quality, but monitoring oyster health is still mostly manual. This project builds a browser-based tool that uses AI to detect and classify oysters from images, videos, or live webcam — no server needed.

OBJECTIVE

Build a real-time oyster detection tool that runs entirely in the browser, works on any device, and makes oyster monitoring faster, cheaper, and more accessible.

RELATED WORK

Builds on prior REU work using YOLOv5 and YOLOv10. We improved performance, accuracy, and deployment flexibility by shifting to YOLOv8 and client-side processing using ONNX.js and TensorFlow.js.

METHODOLOGY

- Trained YOLOv8s with FP64 precision to reduce noise sensitivity
- Used batch size 8 for better weight updates and generalization
- Included oysterless images to teach better background separation
- Final model achieved 0.860 mAP@0.5 across 3 oyster types

Deployment:

- Exported to ONNX and deployed with ONNX.js + TensorFlow.js
- Fully client-side: only the model is served; processing happens in-browser
- Quantized to int8 for faster inference on low-power devices

RESULTS/FINDINGS

- The model reached 86% accuracy (measured using a metric called mAP@0.5, which evaluates how well the model draws boxes around oysters)
- It performed well across all three categories: Open, Closed, and Indeterminate oysters
- It runs at 100+ frames per second on a GPU and still works smoothly on regular laptops and phones
- The system chose a detection threshold of 59% confidence, which gave the best balance between catching real oysters and avoiding false positives
- Users can upload a photo, video, or use a live webcam, and the system detects oysters in real time
- The tool runs entirely in the browser, meaning no internet, server, or software installation is needed

CONCLUSION

We built a real-time oyster detection tool that runs fully in the browser — no server, no setup, just open and use. It works on phones, laptops, and low-power devices.

Future improvements could include:

- Better, more balanced training data
- Adding oyster counting
- Public hosting for wider use

This project lays the foundation for fast, accessible oyster monitoring anytime.

Figure 4.
Project Files & Source Code

ANALYSIS

- We measured accuracy using mAP@0.5 (mean average precision), which checks how well the model draws boxes around oysters
- The final model achieved 86% mAP, meaning it detected oysters correctly most of the time with high confidence
- We chose a confidence threshold of 59%, based on the point where the model balanced precision (avoiding false positives) and recall (catching real oysters)
- A confusion matrix helped visualize how often each oyster type was correctly classified vs. confused with another
- We also looked at precision-recall curves to see how confident and consistent the model was across different detection thresholds

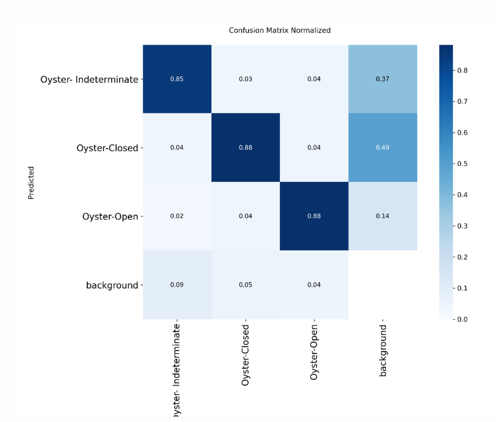


Figure 1.
Shows classification accuracy for each oyster type

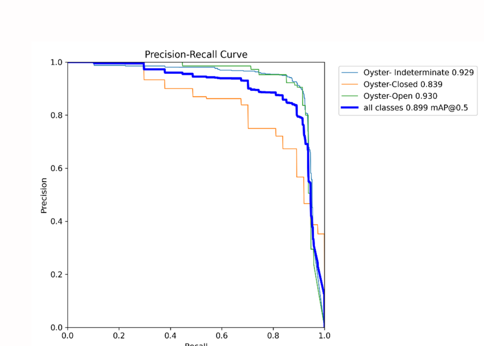
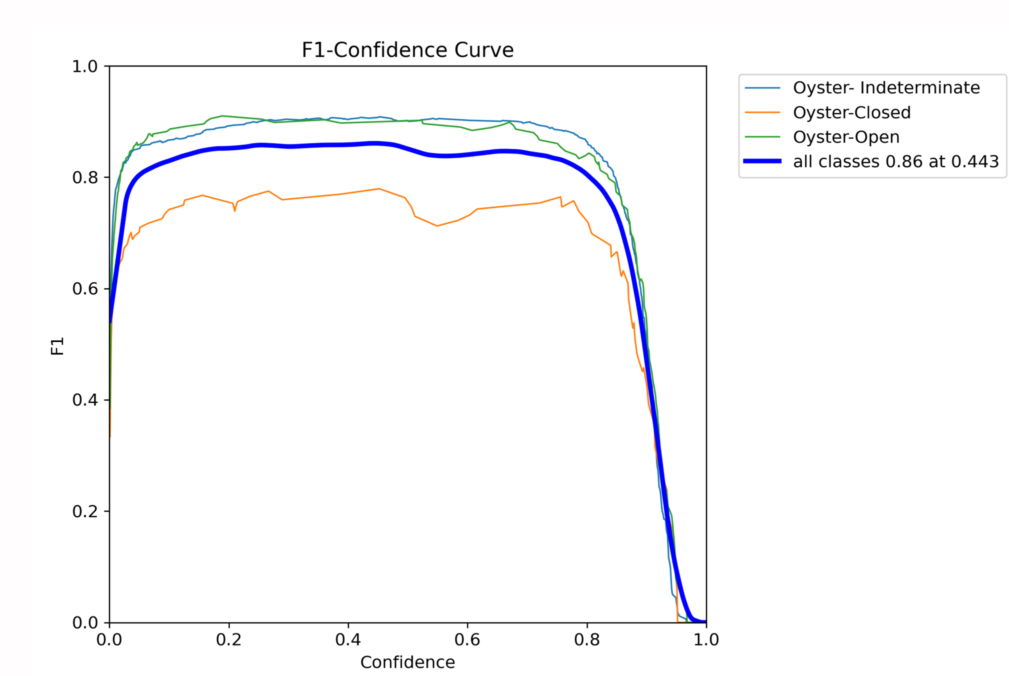


Figure 2.
Visualizes how precise and consistent the model is across all classes

Figure 3.
Supports why we used 59% as our threshold — sweet spot between catching oysters and avoiding false alarms



Acknowledgments

We would like to thank Dr. Yuanwei Jin and Dr. Enyue Lu for their mentorship and guidance throughout this project. We also thank Salisbury University, the USDA and National Science Foundation (NSF) for supporting this research experience.