International Journal of Emerging Technologies and Advanced Applications

Development of UAV Swarm Ad-Hoc Network Communication Technology for Emergency Scenarios: A Review

Catleen Glo M. Feliciano[1, 2], Maria Amelia E. Damian[3]
[1]University of the East, Manila, Philippines
[2] Isabela State University, Isabela, Philippines
[3]President Ramon Magsaysay State University, Philippines
madayag.catleenglo@ue.edu.ph

***Abstract*—Accurate fish counting is important in aquaculture hatcheries because it supports production planning, inventory monitoring, and fish dispersal management. Eels' small size, elongated body structure, transparency during the glass eel stage, and overlapping movement make it difficult to count them using image-based methods because, in contrast to other fish species, eels are prone to occlusion. Counting mistakes, duplicate detections, and missing detections are frequently caused by these traits. This study used adaptive confidence and Intersection over Union (IoU) threshold optimization to develop and evaluate an improved YOLO-ONNX-based eel counting framework. 1,541 eel images made up of glass eel and elver samples were used to train a YOLOv8n model, which was then exported to the Open Neural Network Exchange (ONNX) format for use. For threshold adjustment, a different set of 333 eel counting images arranged according to manual ground-truth counts was used. A confidence threshold of 0.25 and an Intersection over Union (IoU) threshold of 0.70 were utilized in the baseline YOLO-ONNX configuration. To find the optimal counting configuration, the adaptive optimization method tried several combinations of confidence and Intersection over Union. After selecting the optimized threshold configuration, a separate 200-image evaluation set was used to compare the baseline and optimized YOLO-ONNX configurations. Based on the results of the 200-image evaluation set, the optimized setup increased the mean counting accuracy by 11.29 percentage points, from 80.95% to 92.24%, using a confidence threshold of 0.15 and an IoU threshold of 0.40. The Mean Absolute Error decreased from 2.58 to 0.91, while the Root Mean Square Error decreased from 3.88 to 1.42. Based on the results, adaptive threshold optimization enhances detection-based eel counting by lowering overcounting and undercounting brought on by eel body overlap and occlusion. The proposed YOLO-ONNX framework provides a practical approach for improving eel counting accuracy through post-processing threshold ptimization, without modifying the internal structure of the YOLO model.**

***Index Terms*—Adaptive Threshold Optimization; Aquaculture; Eel Counting; Fish Counting; IoU; Object Detection; ONNX; YOLOv8n**

I. Introduction

Accurate fish counting plays an important role in aquaculture management [1] by supporting feeding control [2], fish dispersal [3], and inventory monitoring [4]. Traditional counting methods rely on manual counting and estimation [5], which is time-consuming and also prone to human error, especially in large-scale hatchery environments [6]. At present, the Bureau of Fisheries and Aquatic Resources (BFAR) Region 02 hatcheries still rely on traditional counting of fish during fish dispersal. Therefore, deep learning and object detection models offer a practical solution, as these technologies have been widely applied to automate counting tasks in images and videos [7], [8].

Among these models in deep learning and object detection, YOLO has gained popularity due to its real-time detection capability and accuracy [9]. However, applying YOLO to eel counting presents unique challenges. Eels exhibit elongated and flexible body structures that often overlap or curve in confined environments [10]. These characteristics can reduce counting accuracy due to multiple detections for a single eel or missed detections. Most existing fish detection-based counting approaches rely on raw model outputs [11] without addressing the issue regarding the characteristics of eels, thus limiting their effectiveness in real-world conditions.

Thus, this study was conducted to address the limitations of eel counting under occlusion and overlapping conditions by introducing an optimized YOLO-ONNX-based eel counting framework. The proposed framework uses ONNX-based deployment to support real-time inference and adaptive threshold optimization to improve counting accuracy. Specifically, this study aims to: (1) develop an optimized YOLO-ONNX counting pipeline for real-time eel detection and counting; (2) apply a data-driven selection strategy for confidence and Intersection over Union (IoU) thresholds; and (3) evaluate

ISSN:3006-2985(Print)
ISSN:3006-9297(Online)

Received: May. 3, 2026
Revised: Jul. 20, 2026
Accepted: Jul. 28, 2026

the counting performance of the optimized model using mean absolute error (MAE), root mean square error (RMSE), and counting accuracy.

II. Related Work

A. Deep Learning-Based Fish Detection

Recent studies have already explored deep learning-based methods for fish detection and counting in aquaculture environments [8]. Object detection models such as YOLO have been widely adopted due to their efficiency and real-time capability [12], [13]. Several studies have also demonstrated the effectiveness of YOLO in detecting fish species in controlled environments, achieving high detection accuracy under ideal conditions [14]. Recent studies also support YOLOv8 (especially improved variants) as a very strong and commonly preferred base for fish detection and counting, often outperforming other versions of YOLO and non-YOLO methods on tested datasets [12], [15].

B. Limitations in Real Hatchery Environments

Many existing study approaches still rely on datasets with minimal occlusion and uniform backgrounds [16]. But in real hatchery environments, fish often overlap, leading to multiple detections for a single instance [17]. These conditions reduce the reliability of detection-based counting systems when applied in real-world scenarios.

C. Post-Processing Approaches for Error Reduction

Some studies also attempted to address multiple detections in a single instance using post-processing techniques such as non-maximum suppression tuning or heuristic filtering. While these approaches can reduce duplicate detections, they often lack systematic parameter optimization and may not generalize across different datasets [18]. This limitation highlights the need for more adaptive and data-driven optimization methods.

D. Segmentation-Based Methods

Segmentation-based methods, such as Mask R-CNN and U-Net, have been explored to improve instance separation. Although these methods provide precise object boundaries, they require higher computational resources and are less suitable for real-time deployment [19], [20].

E. ONNX and Efficient Deployment

Open Neural Network Exchange (ONNX) enables efficient inference across different platforms [21]–[23]. It allows models to be deployed in optimized environments, improving performance without modifying the core architecture of the neural network.

F. Research Gap and Contribution

Post-processing optimization can improve counting performance and deployment efficiency [24] without modifying the internal structure of the neural network [25], [26].

This study is conducted to improve detection-based counting through adaptive parameter optimization while maintaining real-time performance using Open Neural Network Exchange (ONNX) deployment.

III. Research Design and Methodology

A. Research Design

This study used a quantitative experimental research design. A quantitative experimental design is a research approach that uses numerical data to test the effect of a specific intervention, treatment, or condition on a measured outcome [27]. In this study, the intervention was the adaptive selection of confidence and Intersection over Union (IoU) thresholds, while the measured outcomes were counting accuracy, Mean Absolute Error (MAE), and Root Mean Square Error (RMSE).

The study was conducted in two main phases. The first phase was model development. In this phase, the YOLOv8n detection model was trained using eel images and was exported to Open Neural Network Exchange (ONNX) format for deployment. The second phase was counting optimization and evaluation. In this phase, different confidence and IoU threshold combinations were tested using a separate eel counting dataset.

The best threshold configuration was selected based on counting accuracy, mean absolute error (MAE) and root mean square error (RMSE). This design allowed the researcher to determine whether adaptive confidence and Intersection over Union (IoU) threshold optimization could improve the counting performance of the YOLO-ONNX model without changing the internal structure of the trained neural network.

B. Model Development and Counting Optimization Workflow

Fig. 1 shows the processing pipeline of the YOLO-ONNX eel counting framework with adaptive confidence and IoU optimization. The workflow is divided into two main parts: the model development branch and the counting optimization and evaluation branch. This split is essential because the eel detection dataset was used to train the YOLOv8n detection model, but an additional eel counting image dataset with folder-based manual counts was used for threshold adjustment and counting evaluation.

In the model development branch, the prepared eel image dataset was used to train the YOLOv8n detection model. By recognizing bounding boxes surrounding visible eel bodies, the model was able to recognize instances of eels. Following training, the optimal YOLOv8n model was stored as a `.pt` model or PyTorch model weight file. After that, the trained model was exported as `best.onnx` in ONNX format.

After model training and ONNX export, a separate threshold optimization was conducted using 333 eel counting images. These images were arranged into manual-count folders ranging from 1 to 20, with each folder name representing the manually counted number of eels in the images. The model's counting performance was assessed using this folder-based count as the benchmark. The YOLO-ONNX model's predicted count and the ground truth number were compared by the testing script, which automatically read the folder names and assigned the relevant manual count to each image.

During the baseline detection stage, the YOLO-ONNX model produced raw detection outputs using a confidence

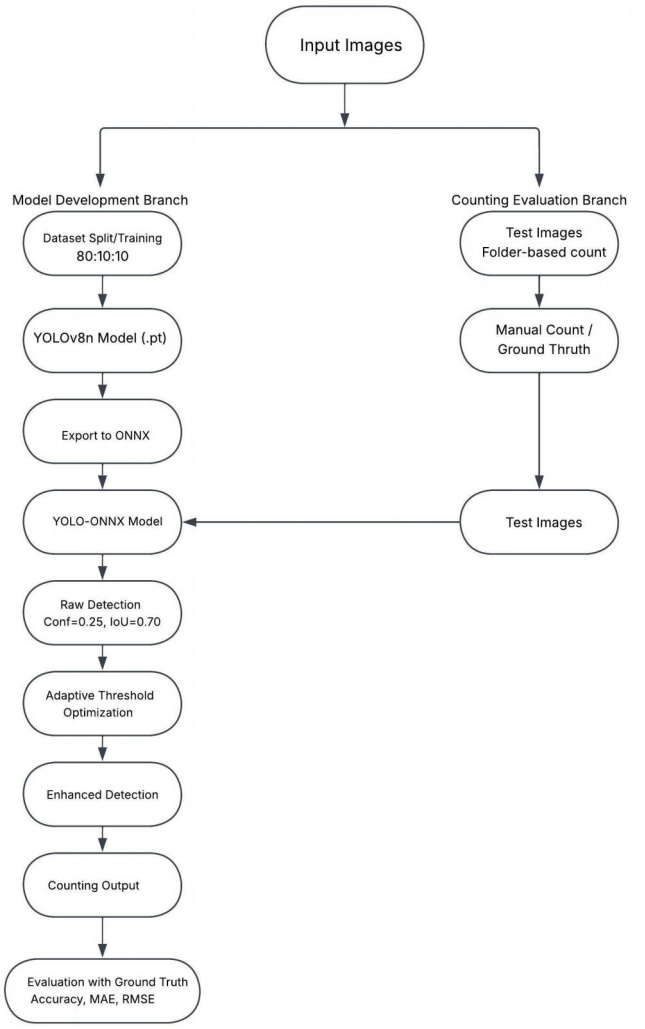

Fig. 1: Processing pipeline of the YOLO-ONNX eel counting system with adaptive threshold optimization.

threshold of 0.25 and an IoU threshold of 0.70. However, because eels are long and often overlap with one another, this baseline setting produced some duplicate detections and missed detections.

To address this issue, adaptive confidence and IoU optimization were applied during the post-processing stage. This process was performed outside the neural network; therefore, it did not change the internal structure of the YOLO model. Instead, it refined the detection results after inference by adjusting the confidence and IoU thresholds. This made the pipeline practical and lightweight for eel counting.

The optimization process tested confidence values of 0.10, 0.15, 0.20, 0.25, 0.30, 0.40, and 0.50, together with IoU values of 0.30, 0.40, 0.45, 0.50, 0.60, and 0.70. The best configuration was selected based on the highest mean counting accuracy, lowest MAE, lowest RMSE, and lowest mean inference time. After the optimization process, the enhanced detection output was used to produce the final eel count by adding all valid detected bounding boxes.

Fig. 2: Actual data-capture setup used for eel image acquisition.

The counting output from the 333-image dataset was compared with the folder-based ground truth count to determine the best confidence and IoU threshold configuration. The counting performance during threshold optimization was evaluated using mean counting accuracy, standard deviation of accuracy, MAE, RMSE, mean absolute difference, and mean inference time. After the optimized threshold configuration was selected, it was evaluated using an independent set of 200 images captured during separate sessions and containing different individual eels. These images were not used for model training or threshold selection. The final outputs included image-level results, summary results, results grouped by manual-count folder, threshold optimization results, and 300 dpi graphs for accuracy comparison, error metrics, and confidence–IoU performance visualization.

C. Experimental Setup and Implementation Details

The experimental setup was designed to evaluate the optimized YOLO-ONNX eel counting pipeline under a controlled image-capture environment. The model training and testing were conducted using a workstation with an Intel Core i7-10700F CPU at 2.90 GHz, 24 GB RAM, and an NVIDIA GeForce GTX 1660 SUPER GPU with 6 GB of dedicated memory. The computer used a 64-bit Windows 11 Home Single Language operating system, Version 25H2.

Fig. 2 shows the actual data-capture setup used for eel image acquisition. To capture and store images, an EMEET 4K webcam was mounted on a camera stand and connected to a laptop. During image collection, a white Styrofoam box was used as the container and the background to improve contrast between the eel samples and the surrounding region. The webcam was positioned above the box to capture the top-view images of the eels. This controlled setup reduces background noise and enables clearer images for annotation,

(a) Elver samples (b) Glass eel samples

Fig. 3: Sample eel images captured using the controlled data-capture setup.

model training, and counting evaluation. The dataset's images were captured at a resolution of 4K.

Fig. 3 shows sample eel images captured using the controlled data-capture setup. The eel samples, measuring approximately 3 to 7 inches in length, represented the glass eel and elver stages. The entire collection consisted solely of images of eels since eels are the primary focus of the research. Both the YOLOv8n model and the YOLO-ONNX counting framework were trained using the images. Elver samples with longer, darker bodies that contrast with the white Styrofoam background are shown in Fig. 3(a). Glass eel samples, which are more difficult to identify due to their small size, transparent bodies, and poor contrast against the backdrop, are shown in Fig. 3(b). Because of their elongated bodies, eels can overlap, curve, and congregate together within the container, eels are extremely susceptible to occlusion, as demonstrated by these example photographs, which also illustrate the visual variations between eel growth phases. This circumstance explains why it can be challenging to detect and count eels using images.

After image acquisition, the YOLOv8n detection model was trained using 1,541 eel images composed of 770 glass eel images and 771 elver images. Originally, approximately 2,000 images were captured. However, blurred images and images that did not capture the full range of the white Styrofoam box were removed during dataset cleaning. This screening process ensured that only clear and usable images were included in the final dataset for model training, validation, and testing. The dataset was divided into 1,233 training images, 154 validation images, and 154 testing images using an 80:10:10 split. Training was conducted for 100 epochs with an image size of 640 pixels, a batch size of 8, a patience value of 100, and 8 workers. The model was trained with the default Ultralytics automatic optimizer, GPU device 0, and automatic mixed precision enabled. The initial learning rate was 0.01 with a momentum of 0.937 and a weight decay of 0.0005. Data augmentation included hue, saturation, and value adjustment; translation; scaling; horizontal flipping; mosaic augmentation; random augmentation; and random erasing.

The software environment was implemented using Python 3.11.9. The object detection model was trained using the Ultralytics YOLOv8 framework, version 8.4.60, with Torch 2.5.1+cu121. The trained YOLOv8n model was exported to Open Neural Network Exchange (ONNX) format and deployed as `best.onnx`. The exported YOLO-ONNX model was loaded and executed using the Ultralytics YOLO framework, while ONNX Runtime version 1.26.0 was installed in the software environment to support ONNX-based inference. The threshold optimization and evaluation script used the exported YOLO-ONNX model to perform detection under different confidence and IoU threshold settings. NumPy was used for numerical computation of counting accuracy, Mean Absolute Error (MAE), Root Mean Square Error (RMSE), mean, and standard deviation. Pandas was used for organizing image-level results, summary results, and threshold optimization outputs into Excel and CSV files. Matplotlib was used to generate the accuracy comparison graph, error metric graph, and confidence–IoU heatmap. The implementation also used built-in Python libraries such as `os` for file and directory handling and `shutil` for compressing the generated result files.

The 1,541-image dataset was used for YOLOv8n model training, validation, and testing, while the 333-image dataset was used separately for threshold optimization, and a separate 200-image dataset was used for counting evaluation. Therefore, the 333-image and 200-image datasets were not used to train the model but were used to optimize and evaluate the counting performance of the exported YOLO-ONNX model.

The dataset used in this study was collected for research purposes. Due to institutional and operational considerations, the dataset is not publicly available at this time. However, selected sample images and experimental results may be made available upon reasonable request, subject to the approval of the concerned institution. The source code and configuration files of the optimized YOLO-ONNX eel counting model are publicly available at: https://github.com/catleenglormadayag-del/adaptive_yolo_onnx_eel_optimization.

D. Evaluation Metrics

The performance of the proposed eel counting system is evaluated using counting accuracy, mean absolute error (MAE), and root mean square error (RMSE).

$$C_{\text{predicted}} = N_{\text{boxes}} \tag{1}$$

The predicted eel count is computed as the total number of detected bounding boxes in each image. This predicted value is then compared with the ground truth count to evaluate system performance using accuracy, mean absolute error (MAE), and root mean square error (RMSE). The formulas used for counting evaluation are presented below.

$$\text{MAE} = \frac{1}{N}\sum_{i=1}^{N} |C_{\text{manual},i} - C_{\text{predicted},i}| \tag{2}$$

Mean Absolute Error (MAE) measures the average difference between the predicted and actual counts, without considering the direction of the error.

TABLE I: Top Confidence and IoU Threshold Combinations

Rank	Conf.	IoU	Accuracy (%)	SD Acc.	MAE	RMSE	Time (ms)
1	0.15	0.40	94.93	8.19	0.65	1.21	10.57
2	0.20	0.40	94.92	8.42	0.65	1.22	10.27
3	0.25	0.45	94.90	7.89	0.64	1.11	10.00
4	0.30	0.45	94.76	7.73	0.67	1.14	8.94
5	0.40	0.45	94.60	8.83	0.68	1.28	9.57

$$\mathrm{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(C_{\mathrm{manual},i} - C_{\mathrm{predicted},i}\right)^2} \tag{3}$$

Root Mean Square Error (RMSE) measures the average size of counting errors and gives more weight to larger errors by using a squared difference between predicted and actual counts.

$$\mathrm{Accuracy} = \left(1 - \frac{|C_{\mathrm{manual}} - C_{\mathrm{predicted}}|}{C_{\mathrm{manual}}}\right) \times 100 \tag{4}$$

Counting Accuracy measures how close the predicted count is to the actual count, expressed as a percentage of correct estimation.

IV. Results and Discussion

A. Data-Driven Selection Strategy for Confidence and IoU Thresholds

The first objective of the study was to apply a data-driven selection strategy for confidence and Intersection over Union (IoU) thresholds. This was done by testing different confidence and IoU threshold combinations and evaluating their effect on the mean counting accuracy of the YOLO-ONNX eel counting model using the 333-image dataset.

Table I presents the top five performing confidence and IoU threshold combinations as a result of the adaptive threshold optimization process. The full search tested was composed of forty-two (42) combinations based on seven (7) confidence thresholds and six (6) Intersection over Union (IoU) thresholds. The highest mean counting accuracy was obtained at confidence = 0.15 and IoU = 0.40, with an accuracy of 94.93%, MAE of 0.65, and RMSE of 1.21. This setting was selected as the optimized configuration as it achieved the highest counting accuracy while maintaining low error values. The table provides the numerical support for the heatmap shown in Fig. 4.

Fig. 4 shows the heatmap of mean counting accuracy across different confidence and IoU threshold combinations. The confidence thresholds tested were 0.10, 0.15, 0.20, 0.25, 0.30, 0.40, and 0.50, while the IoU thresholds tested were 0.30, 0.40, 0.45, 0.50, 0.60, and 0.70. The heatmap shows that counting accuracy changed depending on the threshold values. This confirms that the choice of confidence and IoU thresholds affects the counting performance of the model.

The highest mean counting accuracy was obtained at confidence = 0.15 and IoU = 0.40, with a mean counting accuracy of 94.9%. This setting was identified as the optimized threshold configuration. In comparison, the baseline setting of confidence = 0.25 and IoU = 0.70 obtained a lower mean counting

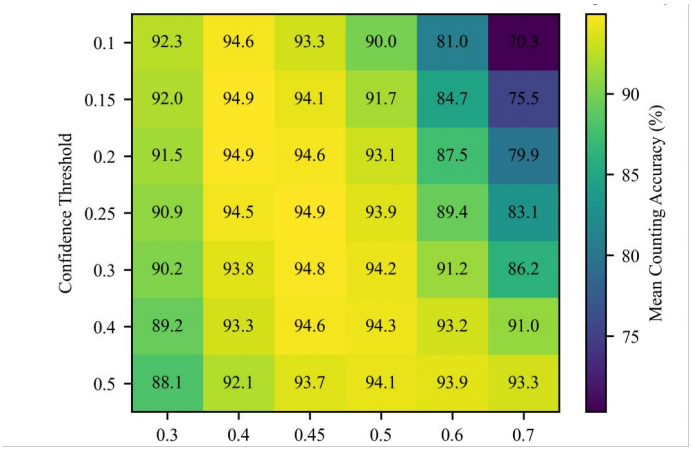

Fig. 4: Heatmap of mean counting accuracy across different confidence and Intersection over Union (IoU) threshold combinations.

TABLE II: Overall Performance of Baseline and Optimized YOLO-ONNX Configurations

Method	Conf.	IoU	Accuracy (%)	MAE	RMSE	Time (ms)
Baseline YOLO-ONNX	0.25	0.70	80.95	2.58	3.88	8.48
Optimized YOLO-ONNX	0.15	0.40	92.24	0.91	1.42	8.54

accuracy of 83.1%. The result also shows that the baseline threshold configuration was not the most suitable setting for eel counting under overlapping and occluded conditions.

The result indicates that lowering the confidence threshold allowed the model to retain more valid eel detections, while adjusting the IoU threshold helped reduce duplicate detections during post-processing. Therefore, the heatmap supports the use of a data-driven threshold selection strategy instead of relying only on default or manually selected threshold values.

B. Counting Performance of the Optimized Model Using MAE, RMSE and Counting Accuracy

Table II presents the overall counting performance of the baseline and optimized YOLO-ONNX configurations. The evaluation was conducted using 200 eel images organized in manual-count folders from 1 to 20. The optimized configuration improved the mean counting accuracy from 80.95% to 92.24%, while the MAE decreased from 2.58 to 0.91 and the RMSE decreased from 3.88 to 1.42. The mean inference time slightly increased from 8.48 ms to 8.54 ms; however, the difference was minimal and still indicates efficient real-time inference performance. The overall result of the table indicates that the optimized configuration improved counting accuracy while maintaining efficient inference performance.

Note: The inference times in Tables I and II were obtained from different evaluation sets and separate timing runs; therefore, minor differences are expected.

Fig. 5 shows the mean absolute counting error of the baseline and optimized YOLO-ONNX eel counting model across manual eel count groups from 1 to 20. The baseline model showed increasing counting error as the number of eels increased, particularly from 10 to 20 eels. The result indicates

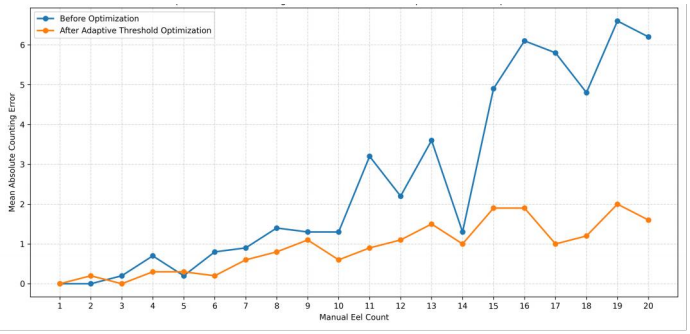

Fig. 5: Counting error comparison before and after adaptive threshold optimization.

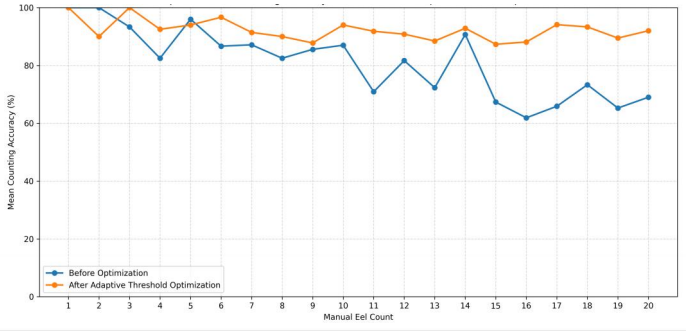

Fig. 7: Counting accuracy comparison before and after adaptive threshold optimization.

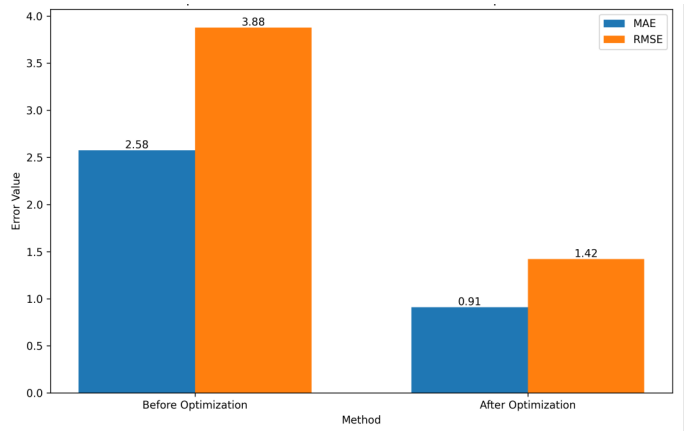

Fig. 6: Error metrics of baseline and optimized YOLO-ONNX eel counting model.

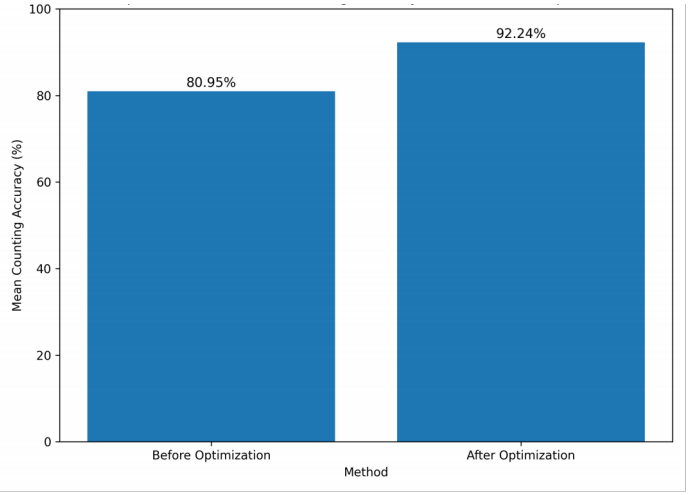

Fig. 8: Overall counting accuracy before and after adaptive threshold optimization.

that the baseline threshold setting had difficulty in handling images with higher eel density, overlapping bodies, and occlusion. After adaptive threshold optimization, the optimized model maintained lower counting errors in most eel count groups. This result reveals that in visually complicated aquaculture conditions, the chosen confidence and Intersection over Union (IoU) parameters reduced missed and false detections, leading to more reliable eel counting performance.

Fig. 6 presents the error metrics of the baseline and optimized YOLO-ONNX eel counting model using mean absolute error (MAE) and root mean square error (RMSE). The optimized model has a lower MAE of 0.91 and RMSE of 1.42 compared to the baseline model's 2.58 and 3.88, respectively. Adaptive threshold optimization enhanced the agreement between the manual and predicted eel counts, as seen by the decrease in both error measures. Larger counting mistakes, which are frequently seen in figures with higher eel density, overlap, and occlusion, were also decreased by the adaptive threshold optimization, as evidenced by the lower RMSE.

Fig. 7 shows the counting accuracy comparison before and after adaptive threshold optimization across manual eel count groups from 1 to 20. In lower-count images, the baseline YOLO-ONNX model demonstrated great accuracy; however, as the number of eels increased, its performance declined. In higher-density images, when eel bodies were more likely to overlap and lead to missing and false detections, this decrease was more evident. The optimized YOLO-ONNX model maintained higher and more consistent counting accuracy throughout the majority of eel count groups during adaptive threshold tuning. This implies that the model's capacity to count eels in challenging and occluded settings was enhanced by choosing the optimal confidence and Intersection over Union (IoU) thresholds.

Fig. 8 shows that the baseline YOLO-ONNX model obtained a mean counting accuracy of 80.95% while the optimized YOLO-ONNX model obtained a mean counting accuracy of 92.24% after adaptive threshold optimization. This shows an improvement of 11.29 percentage points, indicating that the eel detection model's counting performance was enhanced via adaptive threshold optimization.

The result reveals that, when compared to the baseline parameter, the optimized threshold configuration enhanced the model's counting performance. Due to missing and false detections in obstructed situations, the baseline model generated more counting errors since eels have elongated bodies and frequently overlap. The modified confidence and Intersection over Union (IoU) thresholds enhanced eel detection under overlapping and visually complex conditions, as evidenced by

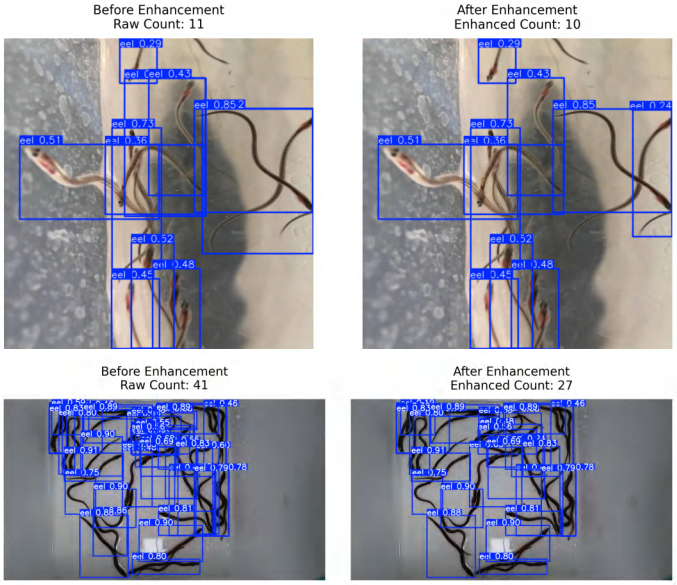

Fig. 9: Sample bounding box detection comparison of the YOLO-ONNX eel counting model before and after adaptive threshold optimization.

the model's increased mean counting accuracy and decreased error rate following adaptive threshold adjustment.

The test images were chosen based on eel count variation rather than model output quality to reduce selection bias. Images with varying eel counts, lighting settings, and eel growth stages—specifically, glass eel and elver—were included in the examination.

Fig. 9 presents sample bounding box detection results before and after adaptive threshold optimization of the YOLO-ONNX eel counting model. To further test the model's counting accuracy, the sample images used in this figure were not included in the training dataset or in the threshold optimization dataset. The comparison shows the difference between the baseline detection output and the optimized detection output using the selected confidence and Intersection over Union (IoU) thresholds.

In the first sample image, the manual count was 14. The baseline model produced a raw count of 16, resulting in an error of 2 eels. After adaptive threshold optimization, the enhanced model produced a count of 13, resulting in a lower error of 1 eel. This shows that the optimized threshold configuration reduced the counting difference and produced a prediction closer to the manual count.

In the second sample image, the manual count was 30. With a raw count of 41, the baseline model generated an error of 11 eels. The enhanced model reduced the error to 3 eels after adaptive threshold optimization, yielding a count of 27. Because the eels are heavily overlapped and concentrated in a particular area, this sample shows a more challenging counting condition. The model produced more bounding boxes under the baseline setup, showing over-detection brought on by occlusion, crossing body shapes, and dense eel arrangement. The number of identified boxes decreased and approached the actual count during adaptive threshold optimization.

Because eels have elongated bodies and frequently overlap, the results indicate that threshold optimization is essential while counting eels. If the threshold is not adjusted, the model might generate duplicate bounding boxes or false detections, which would lead to overcounting. The optimized model produced more consistent counting and decreased detection errors by choosing the best confidence and Intersection over Union (IoU) thresholds.

C. Development of the Optimized YOLO-ONNX Counting Pipeline for Real-Time Eel Detection and Counting

The third objective of the study was to develop a YOLO-ONNX counting pipeline for real-time eel detection and counting. The objective was achieved through YOLOv8n detection model training using the eel image dataset and exporting the best model to ONNX format for deployment purposes. The exported YOLO-ONNX model was used to detect eel instances from input images and live camera frames.

The developed pipeline generates bounding boxes around detected eel bodies. Each bounding box represents one detected eel instance, and the final predicted count is computed by summing all valid detections after post-processing. The pipeline was designed to use IoU optimization and adaptive confidence after inference. In other words, with no modification to the internal structure of the trained YOLOv8n model, the optimization technique improves the detection output.

Algorithm 1 presents the main steps of the optimized YOLO-ONNX eel counting pipeline. The algorithm shows how an input image or live camera frame is processed by the trained YOLO-ONNX model, which then employs confidence and IoU-based post-processing to provide the final projected eel count. The model first finds eel instances, filters the de-

Algorithm 1 Optimized YOLO-ONNX Eel Counting Pipeline

Require: Eel image or live camera frame; trained YOLO-ONNX model; optimized confidence threshold; optimized IoU threshold

Ensure: Predicted eel count

1: Load the YOLO-ONNX model.
2: Read the input eel image or live camera frame.
3: Run YOLO-ONNX inference on the input image.
4: Generate bounding box detections for visible eel instances.
5: Filter the detections using the optimized confidence threshold.
6: Apply Intersection over Union (IoU)-based post-processing to reduce duplicate detections.
7: Retain the remaining valid bounding boxes.
8: Count the number of valid bounding boxes as the predicted eel count.
9: Display the bounding boxes and predicted eel count.
10: Save or return the final predicted count.

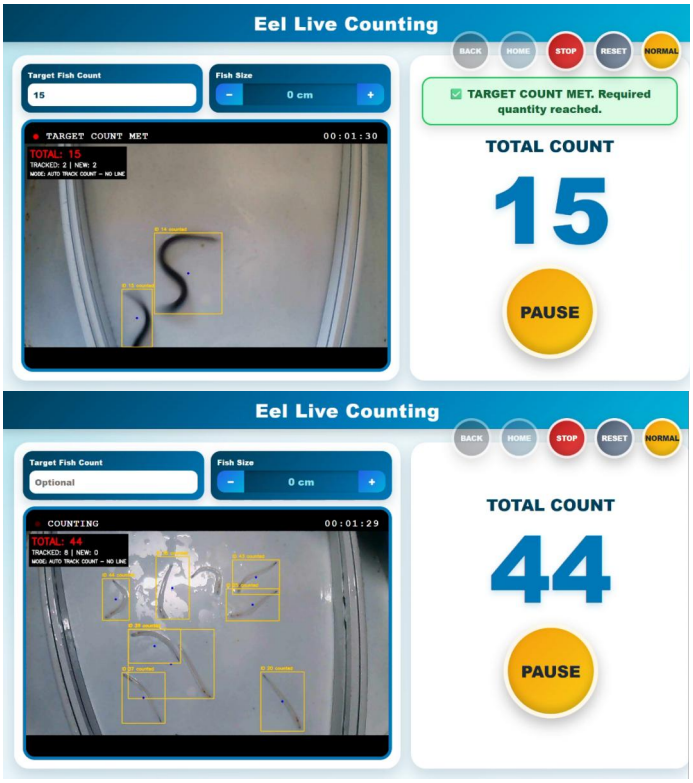

Fig. 10: Sample fish counting interface using the optimized YOLO-ONNX eel counting model.

tections using the best confidence threshold, reduces duplicate detections using IoU-based post-processing, and finally counts the remaining valid bounding boxes as the final predicted eel count.

Fig. 10 shows a sample interface output of the optimized YOLO-ONNX eel counting model using live video frames for demonstration purposes. The interface presents the software-side counting display, including the detected eel bounding boxes, target total, counting status, and predicted count. The quantitative performance evaluation reported in Table II was conducted using a separate set of 200 saved eel images, while the live video interface was included to demonstrate the possible real-time deployment of the optimized YOLO-ONNX counting pipeline in a controlled setup.

V. Conclusions

This research enhanced a YOLO-ONNX counting pipeline for real-time eel identification through the implementation of adaptive confidence and Intersection over Union (IoU) threshold optimization. The findings indicated that the suggested method enhanced counting performance relative to the baseline YOLO-ONNX setup.

Based on the 200-image evaluation set, the mean counting accuracy increased from 80.95% to 92.24%. The mean absolute error (MAE) decreased from 2.58 to 0.91, while the root mean square error (RMSE) decreased from 3.88 to 1.42. The findings indicate that adaptive threshold optimization can reduce the issues of overcounting and undercounting caused by overlapping eel bodies, a common challenge in image-based eel counting.

The findings also demonstrate that detection-based counting performance is enhanced by choosing suitable Intersection over Union (IoU) and confidence levels. While a balanced Intersection over Union (IoU) threshold reduced duplicate detections, a lower confidence threshold helped keep small or partially visible eel detections. Extreme occlusion may still have an impact on accuracy, though, indicating that segmentation-based methods may enhance accuracy even further.

The counting interface demonstrated the possible deployment of the optimized YOLO-ONNX framework using live video frames in a controlled environment. However, the quantitative performance evaluation reported in this study was conducted using a separate set of 200 saved eel images. To further confirm the model's effectiveness and generalizability, future studies should use larger independent test sets, continuous live-video evaluation, and a wider range of hatchery conditions.

Acknowledgements

The researchers acknowledge the Bureau of Fisheries and Aquatic Resources Region 02 for accommodating the study and for providing technical assistance on the conduct of the study. Researchers also acknowledge the support of hatchery operators and staff from the Cagayan Valley Freshwater Technology Outreach Station (CVFTOS) located at San Mateo, Isabela, Aparri Brackishwater Technology Outreach Station (ABTOS) at Punta, Aparri, Cagayan and Cauayan Aqua Ventures Corporation for the assistance in data validation and domain expertise.

Declaration on the Use of AI Tools

The author acknowledges the use of ChatGPT and Grammarly to organize ideas, polish phrasing, and improve the clarity of the paper. These tools were not used for data

generation, data analysis, or interpretation of results. The author is solely responsible for all interpretations, conclusions, and findings in this work. To ensure accuracy, originality, and adherence to ethical research guidelines, the manuscript was thoroughly reviewed.

REFERENCES

[1] G. Wang, J. Yu, W. Xu, A. Muhammad, and D. Li, "Automated fish counting system based on instance segmentation in aquaculture," *Expert Systems with Applications*, vol. 259, Art. no. 125318, 2025. DOI: 10.1016/j.eswa.2024.125318.

[2] X. Yu, Y. Wang, D. An, and Y. Wei, "Counting method for cultured fishes based on multi-modules and attention mechanism," *Aquacultural Engineering*, vol. 96, Art. no. 102215, 2022. DOI: 10.1016/j.aquaeng.2021.102215.

[3] S. M. D. Lainez and D. B. Gonzales, "Automated fingerlings counting using convolutional neural network," in *Proc. 2019 IEEE 4th Int. Conf. Computer and Communication Systems (ICCCS)*, 2019, pp. 67–72. DOI: 10.1109/CCOMS.2019.8821746.

[4] E. S. Yap et al., "Initial development of tilapia fingerlings counter machine with smart camera," in *Proc. 2024 IEEE 16th Int. Conf. Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment, and Management (HNICEM)*, 2024, pp. 1–6. DOI: 10.1109/HNICEM64917.2024.11258850.

[5] G. Farjon, L. Huijun, and Y. Edan, "Deep-learning-based counting methods, datasets, and applications in agriculture: A review," *Precision Agriculture*, vol. 24, no. 5, pp. 1683–1711, 2023. DOI: 10.1007/s11119-023-10034-8.

[6] G. D. Espeña, A. I. F. D. Casarungan, O. S. M. Villaverde, and J. R. D. Riñon, "Using YOLOv8 for milkfish (*Chanos chanos*) fry counting in Philippine hatcheries," in *Proc. 2025 IEEE Int. Conf. Internet of Things and Intelligence Systems (IoTaIS)*, 2025, pp. 41–46. DOI: 10.1109/IoTaIS67227.2025.11282093.

[7] H. Mokayed, T. Z. Quan, L. Alkhaled, and V. Sivakumar, "Real-time human detection and counting system using deep learning computer vision techniques," *Artificial Intelligence and Applications*, vol. 1, no. 4, pp. 221–229, 2023. DOI: 10.47852/bonviewAIA2202391.

[8] H. Liu, X. Ma, Y. Yu, L. Wang, and L. Hao, "Application of deep learning-based object detection techniques in fish aquaculture: A review," *Journal of Marine Science and Engineering*, vol. 11, no. 4, Art. no. 867, 2023. DOI: 10.3390/jmse11040867.

[9] M. L. Ali and Z. Zhang, "The YOLO framework: A comprehensive review of evolution, applications, and benchmarks in object detection," *Computers*, vol. 13, no. 12, Art. no. 336, 2024. DOI: 10.3390/computers13120336.

[10] K. Zhang, Y. Lv, Y. Zhang, C. Bian, J.-H. Wu, Q. Shi, et al., "Genomics comparisons provide new insights into the evolution of karyotype and body patterns in Anguilliformes species," *International Journal of Biological Macromolecules*, vol. 308, Part 1, Art. no. 142504, 2025. DOI: 10.1016/j.ijbiomac.2025.142504.

[11] S. Abba, A. M. Bizi, J.-A. Lee, S. Bakouri, and M. L. Crespo, "Real-time object detection, tracking, and monitoring framework for security surveillance systems," *Heliyon*, vol. 10, no. 15, Art. no. e34922, 2024. DOI: 10.1016/j.heliyon.2024.e34922.

[12] V. Mohankumar and S. Anbalagan, "A benchmark dataset and ensemble YOLO method for enhanced underwater fish detection," *ETRI Journal*, 2025. DOI: 10.4218/etrij.2024-0383.

[13] U. Sirisha, S. P. Praveen, P. N. Srinivasu, P. Barsocchi, and A. K. Bhoi, "Statistical analysis of design aspects of various YOLO-based deep learning models for object detection," *International Journal of Computational Intelligence Systems*, vol. 16, Art. no. 126, 2023. DOI: 10.1007/s44196-023-00302-w.

[14] M. Işık and M. Yalçinkaya, "Enhancing aquarium fish classification through YOLO: A deep learning approach," in *Proc. 2024 8th Int. Artificial Intelligence and Data Processing Symp. (IDAP)*, 2024, pp. 1–4. DOI: 10.1109/IDAP64064.2024.10710813.

[15] Z. Zhang et al., "An improved YOLOv8n used for fish detection in natural water environments," *Animals*, vol. 14, no. 14, Art. no. 2022, 2024. DOI: 10.3390/ani14142022.

[16] M. Vijayalakshmi and A. Sasithradevi, "A comprehensive annotated image dataset for real-time fish detection in pond settings," *Data in Brief*, vol. 57, Art. no. 111007, 2024. DOI: 10.1016/j.dib.2024.111007.

[17] M. Cui et al., "Fish tracking, counting, and behaviour analysis in digital aquaculture: A comprehensive survey," *Reviews in Aquaculture*, 2024. DOI: 10.1111/raq.13001.

[18] K. Noh, S. Hong, S. Makonin, and Y. Lee, "Enhancing object detection in dense images: Adjustable non-maximum suppression for single-class detection," *IEEE Access*, vol. 12, pp. 130253–130263, 2024. DOI: 10.1109/ACCESS.2024.3459629.

[19] S. Yadav et al., "An improved deep learning-based optimal object detection system from images," *Multimedia Tools and Applications*, vol. 83, pp. 30045–30072, 2023. DOI: 10.1007/s11042-023-16736-5.

[20] C. Mohan et al., "A comprehensive review of image segmentation architectures," in *Proc. 2024 11th Int. Conf. Electrical Engineering, Computer Science and Informatics (EECSI)*, 2024, pp. 223–229. DOI: 10.1109/EECSI63442.2024.10776322.

[21] A. Chinnaraju, "Benchmarking cross-platform AI: WebAssembly, ONNX Runtime and TVM for real-time web, mobile, and IoT deployment," *World Journal of Advanced Research and Reviews*, vol. 26, no. 2, pp. 1937–1963, 2025. DOI: 10.30574/wjarr.2025.26.2.1832.

[22] C. Mani, T. Paul, P. Archambault, and A. Marois, "Machine learning workflow for edge computed arrhythmia detection in exploration class missions," *NPJ Microgravity*, vol. 10, 2024. DOI: 10.1038/s41526-024-00409-0.

[23] R. Chang and T. Hoang, "Constructing an AI compiler for ARM Cortex-M devices," *Computer Systems Science and Engineering*, vol. 46, pp. 999–1019, 2023. DOI: 10.32604/csse.2023.034672.

[24] Z. Lv et al., "Efficient deployment of peanut leaf disease detection models on edge AI devices," *Agriculture*, vol. 15, no. 3, Art. no. 332, 2025. DOI: 10.3390/agriculture15030332.

[25] D. Annam, "Advancements in latency reduction for real-time data processing in the cloud," *World Journal of Advanced Engineering Technology and Sciences*, 2025. DOI: 10.30574/wjaets.2025.15.2.0673.

[26] H. Wen et al., "AdaptiveNet: Post-deployment neural architecture adaptation for diverse edge environments," in *Proc. 29th Annual Int. Conf. Mobile Computing and Networking*, 2023. DOI: 10.1145/3570361.3592529.

[27] P. Slater and F. Hasson, "Quantitative research designs, hierarchy of evidence and validity," *Journal of Psychiatric and Mental Health Nursing*, vol. 32, no. 3, pp. 656–660, 2025. DOI: 10.1111/jpm.13135.