YOLOv8-WBF: Ensemble Learning for Reliable Detection of Endangered Medaka (Oryzias)

Armin Lawi[a,b], Muhammad Haerul[a], Iman Mustika Ismail[a], Rahmatullah R.[a], Irma Andriani[c], and Andi Iqbal Baharuddin[d]

[a]Information Systems Study Program, Hasanuddin University, Makassar, Indonesia
[b]B.J. Habibie Institute of Technology, Parepare, Indonesia
[c]Department of Biology, Hasanuddin University, Makassar, Indonesia
[d]Department of Fishery, Hasanuddin University, Makassar, Indonesia

September 5, 2025

Abstract

Reliable detection of Medaka (Oryzias) fish is crucial for ecological monitoring and conservation, particularly for assessing population trends of endangered species. In this study, we evaluate the performance of a state-of-the-art deep learning model (YOLOv8) and an ensemble approach using Weighted Box Fusion (WBF) on a manually annotated dataset of Medaka images collected from online sources. Models were trained and validated using 5-fold cross-validation, and performance was assessed using COCO metrics, including mean Average Precision (mAP), precision, recall, and bounding box regression error. The YOLOv8-WBF ensemble achieved a mAP@0.5:0.95 of 0.578, representing an 8.04% improvement over the best single model (0.535). It also reduced bounding box localization error (RMSE: 45.41 to 29.93) and improved classification loss, indicating more reliable detection of small and visually challenging fish instances. These accuracy gains came at the expense of computational efficiency, with inference requiring approximately five times more operations than a single YOLOv8 model. While less suited for real-time deployment, the ensemble approach provides more accurate population monitoring in offline ecological workflows, where reliability is often prioritized over speed. By reducing missed detections of small or occluded fish, our findings contribute to more robust biodiversity monitoring and establish a baseline for developing optimized ensemble and lightweight detection models in aquatic conservation domains.

1 Introduction

The advancement of artificial intelligence (AI) has significantly contributed to wildlife conservation efforts, particularly in automating species identification and monitoring. These technologies are invaluable in environments where manual observation is difficult, such as underwater ecosystems, where factors like turbidity and low light complicate visibility (LeCun et al., 2015; Liu et al., 2016). AI-based systems can process extensive volumes of visual data from images and videos to assist in detecting species presence, tracking their behavior, and supporting conservation strategies with minimal human intervention (Kalafi and Javanmard, 2018).

In this context, object detection models have gained traction for their speed and efficiency in real-time applications. The You Only Look Once (YOLO) architecture, particularly its latest version, YOLOv8, enhances prior iterations with improved accuracy and architectural efficiency, making it suitable for challenging tasks like aquatic species recognition. YOLOv8 effectively counters issues surrounding image degradation commonly faced underwater, such as turbidity, unfavorable lighting, and occlusion (Redmon and Farhadi, 2017).

Our study specifically targets the detection of two notable Medaka fish species—*Oryzias javanicus* and *Oryzias celebensis*. These species pose unique monitoring challenges due to their dwindling populations and the complexities of underwater imaging. To bolster this investigation, we constructed a custom dataset comprising both manually collected images using cameras and publicly available images from internet resources. This dataset contains a total of 2,016 images, segregated into 1,857 images (92%) for training and 159 images (8%) for testing. We utilized Roboflow for manual annotation of the dataset, alongside conducting essential preprocessing and augmentation (Dang et al., 2020).

The preprocessing steps involved automatic orientation adjustments, resizing images to a standard 640×640 pixels, and filtering out any empty annotations. To enhance the robustness of the model and improve generalization, augmentations such as horizontal and vertical flips, as well as rotations (90°, 180°, and 270°), were applied (Arbogast and Mehta, 2016). These methodological choices mitigate the risks of overfitting and aim to simulate various real-world conditions that the model may encounter.

For the detection task, we employed a YOLOv8-based approach featuring 5-fold cross-validation to train five individual models. Predictions from these distinct models were subsequently refined and combined using Weighted Box Fusion (WBF), an advanced ensemble method that enhances final bounding boxes by evaluating confidence scores and overlaps—ultimately bolstering the model's overall detection performance (Solovyev et al., 2021; Leow and Savariar, 2015). Our findings indicate that this ensemble approach achieves a substantial increase in mean Average Precision (mAP) relative to a single YOLOv8 model. Notably, while ensemble methods do increase inference time, they significantly enhance the model's robustness, especially when detecting small or less visible targets such as Medaka fish. Through this research, we establish a foundational reference for the development of lightweight ensemble methodologies applicable to underwater object detection, utilizing advanced real-time detection models (Salimi and Bai, 2016).

2 Materials and Method

2.1 Data Sources

The data used in building the computational model for detecting and classifying Oryzias fish species consists of a combination of primary and secondary data. These sources are used to enhance dataset diversity and generalization, which is crucial given the limitations of real-world data.

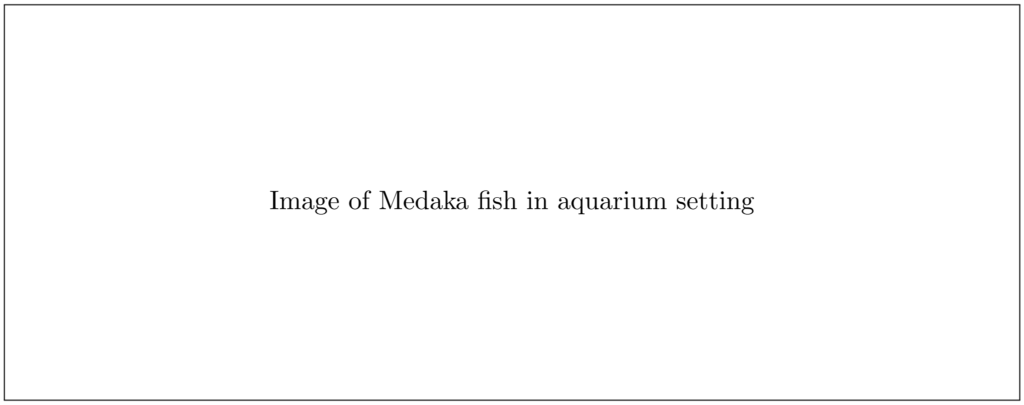


Figure 1: Example of primary data: Medaka fish in aquarium

- **Primary data** was obtained by directly capturing images of Medaka fish in an aquarium using a digital camera. This dataset includes images of two Medaka species: *Oryzias javanicus* and *Oryzias celebensis*.
- **Secondary data** was collected online via web scraping, including from research websites, aquatic forums, and public databases.

Screenshot of web-scraped fish images from aquatic forums and databases

Figure 2: Example of secondary data: web-scraped images

2.2 Research Methodology

The data mining concept is used as the foundation of this research approach, which aims to detect and classify two Medaka fish species: *Oryzias javanicus* and *Oryzias celebensis*. This study employs YOLOv8, a state-of-the-art and efficient object detection algorithm, to enable simultaneous detection and classification.

The research process is divided into three main stages: data preprocessing, data training/modeling, and post-processing/evaluation. The preprocessing stage is critical to ensure data quality and optimal model performance.

Flowchart illustrating research steps: data preprocessing → modeling → evaluation

Figure 3: Overview of the Research Pipeline

2.3 Data Preprocessing Stage

2.3.1 Problem Understanding

Medaka fish are endemic species with small and transparent bodies, making them difficult to detect. Images of Medaka fish are also very limited online. Therefore, a combination of primary and secondary data is used in a minimum ratio of 70% to 30%, followed by normality tests to ensure that the mixed data do not have significant deviation.

2.3.2 Mixture Data Sources

The dataset consists of a normalized combination of primary and secondary data to avoid significant differences. Orientation, size, and image parameters are standardized so that the dataset is suitable for use by the detection model. With variation in lighting and shooting angles, this dataset is designed to train the YOLOv8 model to perform well in general.

Example images showing lighting and angle variations in Medaka fish photos

Figure 4: Sample images illustrating variations in lighting and angles

2.3.3 Data Labeling

The labeling process was carried out using the Roboflow platform. Each image may contain more than one object with different classes. Steps included:

- 792 images placed in a single folder.
- Each image annotated with bounding boxes and class labels.
- Annotation results saved in .txt files named identically to their corresponding images.

Screenshot of image annotation with bounding boxes and labels

Figure 5: Example of image annotation process

2.3.4 Splitting Data

The data was split with a 4:1 ratio between training and testing data. Data splitting also considered the availability of validation data for the model tuning process.

2.3.5 Data Understanding

Data were collected under various lighting conditions and shooting angles to produce a diverse dataset. The dataset consists of Medaka fish images (X) and labels in the form of species classes and bounding box coordinates (Y). These labels are used to train the model to detect and classify objects accurately.

2.3.6 Data Preparation

This stage includes:

Pie chart showing training, validation, and test data split ratios

Figure 6: Data splitting into training, validation, and test sets

a. **Image Restoration:** Enhancing image quality such as sharpening blurred images, increasing contrast, and removing artifacts.

b. **Data Uniformity:** Standardizing aspects such as orientation, image size, contrast, saturation, and blur level.

c. **Data Augmentation:** Adding data variation through horizontal/vertical flipping and rotations of 90°, 180°, and 270°.

d. **Dataset Split:** Data divided into training, validation, and test sets with proportions of 80%-10%-10%. Validation data is used to monitor performance during training.

2.4 Data Training or Modeling

Three main experiments were conducted:

- **Fine-tuning YOLOv8:** Adapting the pretrained YOLOv8 model to the Medaka fish dataset.

Diagram or sample YOLOv8 detection output images

Figure 7: Example of YOLOv8 detection results

- **5-Fold Cross Validation:** Testing the model on 5 data subsets to evaluate performance consistency.
- **Ensemble Learning:** Combining multiple YOLOv8 models with different weights to improve classification accuracy.

Illustration of 5-fold cross-validation data splits

Figure 8: 5-Fold Cross Validation scheme

2.5 Data Post-processing or Evaluation

Model evaluation was conducted using loss functions and confusion matrix-based metrics (such as precision, recall, and mAP). The evaluation was performed on test data to measure YOLOv8's ability to generalize to unseen data.

Confusion matrix and performance metric charts

Figure 9: Model evaluation results: confusion matrix and metrics

3 Experiments

3.1 Experimental environment and parameter configuration

The experimental setup for this study utilized an NVIDIA GeForce RTX4080 GPU with 16GB of memory, Python 3.8.18, PyTorch 1.7.1 framework, and CUDA 12.0. The training is performed using a homemade wildlife dataset, and the experimental parameters are shown in Table 2.

3.2 Evaluation index

(1) Precision evaluation metrics: Precision (P), Recall (R), mean Average Precision (mAP), and F1 score. mAP@0.5 and mAP@0.5–0.95 denote the mAP value at an IoU threshold of 0.5 and the average mAP value as the IoU ranges from 0.5 to 0.95 with a step size of 0.05, respectively. The formulas for each index are shown in Eqs. 12–15, where TP represents the number of true positive predictions, FP denotes the number of false positive predictions, and FN signifies the number of false negative predictions. (2) Speed evaluation metrics: inference time (time from preprocessed image input into the model to model output result), post-processing time, floating-point operations(GFLOPs), model size, and parameter. P = TP TP + FP (12) R

= TP TP + FN

3.3 Experiment 1: Fine-Tuning YOLOv8

The first experiment was conducted by applying fine-tuning to the YOLOv8 model. YOLOv8 is one of the latest object detection models designed to detect and classify objects in real-time. In this experiment, the pre-trained YOLOv8 model was fine-tuned to the collected Medaka fish dataset. Fine-tuning involves adjusting the pre-trained weights using the Medaka fish data to allow the model to recognize the unique features of the species. *Oryzias javanicus* and *Oryzias celebensis.*

The fine-tuning process is performed using a dataset that has been prepared through preprocessing and augmentation stages, with optimal hyperparameter settings. The goal of this experiment is to improve the accuracy of fish species detection and classification by using the knowledge gained from large datasets and adapting it to more specific datasets.

Fine-tuning the YOLOv8 model is a crucial step to adapt the model's performance to the specific characteristics of the dataset used in this study. YOLOv8, as one of the most advanced object detection algorithms today, requires a fine-tuning process to optimize its ability to detect rare or endangered species. In this process, the pre-trained weights of the model were adjusted using training data from the mixture dataset we collected, which is a combination of primary and secondary data.

The fine-tuning process allows the model to learn unique features of our dataset, such as specific environmental variations, lighting conditions, as well as special traits present in the target species. By applying an approach where some network layers remain frozen while other layers are updated, we were able to utilize the general knowledge gained from the large-scale dataset while still tailoring the model to the unique characteristics of this dataset. This strategy is important for improving detection accuracy, reducing the risk of overfitting, and ensuring the model is reliable in various environmental conservation scenarios.

In this study, the YOLOv8 model is fine-tuned using a mixture dataset that has been divided into 80% for training data, 20% for testing data. The model was trained with optimally tuned hyperparameters, namely for 100 epochs, using a batch size of 16, with an automatic optimizer, and a learning rate of 0.01. The model training lasted for approximately 2 hours, utilizing GPU-based computing infrastructure to speed up the process.

3.4 Experiment 2: 5-Fold Cross Validation

In the second experiment, the model was evaluated using the 5-fold cross validation technique. Cross-validation is a method used to ensure that the model has stable performance and can be generalized to various subsets of data. In 5-fold cross-validation, the train data is divided into five subsets, where at each iteration, four subsets are used to train the model, and one subset is used for validation. This process is repeated five times so that each subset serves as a validation set once.

The results of each iteration are then averaged to obtain more accurate performance metrics, such as precision, recall, and mean average precision (mAP). By using cross-validation, the risk of overfitting the model can be minimized, allowing the model to perform optimally on unseen data.

Cross-validation is used to evaluate the reliability and generalizability of YOLOv8 model performance. In this study, we applied 5-fold cross-validation, where the dataset is divided into five subsets (folds). At each iteration, the model is trained using four subsets and tested on the remaining subset. This process is repeated five times, with each fold acting as a one-time validation set. Performance metrics, such as precision, recall, and mean average precision (mAP), are calculated at each iteration and then averaged to provide an accurate and reliable estimate of the model's overall capability.

Cross-validation is important to reduce the risk of overfitting, because by dividing the data into multiple subsets and testing the model iteratively, we can ensure that the model performs consistently even if the data is divided in different ways. It also helps in ensuring that the model is more resilient to variations in the dataset and reliable when applied to data outside the training sample.

In this research, the dataset is divided into 80% (508 images) for the training set and 20% (158 images) for the test set. The training set was further divided into five subsets (A, B, C, D, and E), each containing 127 images. The model was trained using four subsets as training set and one subset as validation set at each iteration. This training process was repeated five times, rotating among the five subsets. Illustration of experiment 2 or cross validation as shown in Figure 2.

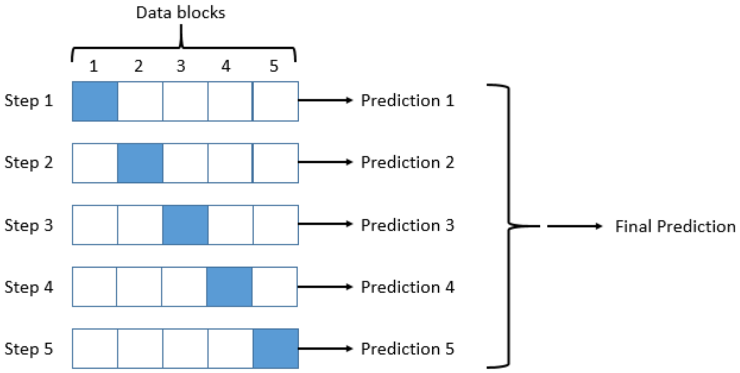


Figure 10: 5-Fold Cross Validation

The model was trained using optimally tuned hyperparameters for 100 epochs, with batch size 16, auto optimizer, and learning rate 0.01. Each training iteration took about 2 hours, so the total time required to complete five cross-validations was approximately 10 hours.

3.5 Experiment 3: Ensemble Method (ADABoost)

The third experiment involved applying ensemble methods to further improve detection and classification performance. The method used is AdaBoost, an ensemble algorithm that works by combining multiple models to improve prediction accuracy. In this approach, the YOLOv8 model is combined with several other models, or variations of the YOLOv8 model with different hyperparameter settings, to form an ensemble system.

The main motivation for using AdaBoost is to improve model performance by correcting the weaknesses of individual models that may fail to recognize certain patterns. The AdaBoost algorithm gives greater weight to the prediction errors of the previous model, so that the next iteration of the model focuses more on correcting those errors. In this way, the resulting model is more robust and accurate, and has a better ability to generalize predictions, even on complex or variable data.

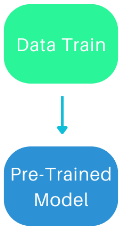
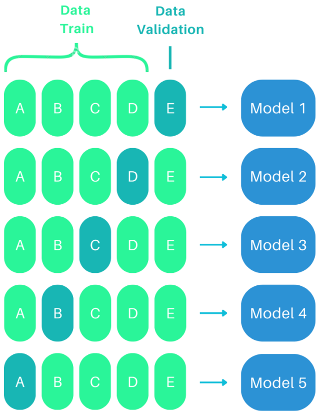
Ensemble learning is a technique used to improve model performance by combining predictions from multiple models. In this study, we apply an ensemble approach by combining multiple YOLOv8 models, where each model is trained with slightly different hyperparameters or trained on different subsets of data. The purpose of this ensemble is to reduce the variance and bias that may exist in each individual model, resulting in more accurate and reliable predictions. This ensemble approach can be done by averaging outputs (such as bounding box coordinates and confidence scores) or using a majority voting mechanism for classification. This technique helps improve overall robustness and predictive ability, especially when dealing with real-world data complexity and diversity.

Experiments using the ADABoost ensemble technique were conducted by utilizing 5 models from the cross validation results in experiment 2. The ADABoost algorithm is as follows

```
Algorithm: AdaBoost

1. Input: Dataset $D = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$,
          Learner $\Gamma$ and the number of learning iteration $T$
2. Initialize weight sample $w_i = \frac{1}{N}, \forall i = 1, 2, \ldots, N$
3. Iterate, for $t = 1$ to $T$ do
    (a) Train a weak learner $h_t$ from $D_t (\in D)$ to train sample $w_i$. $h_t = \Gamma(D, D_t)$
    (b) Compute error of $h_t$: $\varepsilon_t = \frac{\sum_{i=1}^{N} w_i \cdot I(h_t(x_i) \neq y_i)}{\sum_{i=1}^{N} w_i}$
    (c) Compute the weight of $h_t$: $\alpha_t = \frac{1}{2} \ln \left( \frac{1-\varepsilon_t}{\varepsilon_t} \right)$
    (d) Assign: $w_i \leftarrow w_i \cdot e^{(\alpha_t \cdot I(h_t(x_i) \neq y_i))}$
4. Output: $H(x) = \text{sign} \sum_{t=1}^{T} (\alpha_t \cdot h_t(x))$
```

Table 1: table

Fine-Tuning	Cross Validation	Ensemble (AdaBoost)
	 Data Train Data Validation A B C D E Model 1 Model 2 Model 3 Model 4 Model 5	gambar

4 Results

4.1 Mixture Data Preparation

This subsection presents the results of the mixed data preparation process with the stages described in the subsection **??**.

Data preparation sequence:

1. download digital image file
2. Restoration of all digital objects in the file by removing the noise in the file. For video files or moving images, restoration is performed on each frame of the captured image per-second.
3. The file restoration process is performed by removing noise applying a Gaussian kernel function.
4. All image files are then normalized to 640x640 pixels.
5. Normalized files containing multiple objects are annotated according to their respective species names.

4.1.1 Data Acquisition and Labeling

The data preparation stage was initially performed by collecting primary data from captured still images and moving videos into a main folder[1]. Next, all digital objects in a file were provided with bounding boxes and annotated with labels corresponding to their species. (The annotation results of the digital objects were stored as metadata along with other information such as location, time, authorship, etc., in a file). A summary of the dataset obtained using the methods described in subsection **??** can be seen in Table 2

Table 2: Recapitulation of Primary and Secondary Data

No.	Fish Class	Data Primer	Data Sekunder	Jumlah
1.	*O. Javanicus*	257	178	435
2.	*O. Celebensis*	287	70	357
	Total	**544**	**248**	**792**

The mixture data consisted of 792 total images, i.e., 435 (55%) images of O. Javanicus and 357 (45%) O. celebensis fish images which after preprocessing were divided into subsets for each experiment. Primary data yielded 544 (69%) fish and secondary data 248 (31%) fish.

4.1.2 Image Restoration

Remove noise by applying brightness, contrast and gaussian kernel function settings. Size

4.1.3 Normality Test of Data

A normality test is performed to show that the primary and secondary data in the dataset are equivalent.

4.1.4 Data Splitting

Experiment 1: Data Train 634 (80%), Data Test 158 (20%) - Model Experiment 1

Experiment 2: Train data is divided into 5 groups with an arrangement of 127 (16%) each. Each group is named dataset A, B, C, D, and E. Then the model is cross validated with a combination of train data and validation data ratio of 4:1 or 80% training data and 20% validation data from 634 train data in experiment 1. Overall, train data 508 (64%), Validation Data 127 (16%), and Test Data 158 (20%). The simulated cross validation experiment design can be organized as follows.
Model 1: ABCD as Train, E as Validation (80,20)
Model 2: ABCE as Train, D as Validation
Model 3: ABDE as Train, C as Validation

[1] Digital still image and moving video of an object hereafter are called digital object

Model 4: ACDE as Train, B as Validation
Model 5: BCDE as Train, A as Validation
Selection of the best model is obtained by selecting the model with Accuracy and Best Fitting.

Experiment 3: AdaBoost ensemble with weak learner model taken from experiment 2.

4.2 The Model Results

4.2.1 Experiment 1: Single Model (YOLOv8 Fine-Tuning)

In the first experiment, the YOLOv8 model was trained with data divided in the ratio of 70:20:10 for training, validation, and testing. The dataset consists of: Train data: 556 images (70%) Validation Data: 160 images (20%) Test Data: 77 images (10%)

The model was trained for 100 epochs with a batch size of 16. Figure 1 shows a consistent downward trend in train/box_loss, train/cls_loss, and train/dfl_loss, indicating the model was able to learn effectively. The loss on the validation data also decreased despite slight fluctuations at the beginning of training, which stabilized near the end of training.

Figure 2 displays the Precision and Recall metrics, both of which show significant improvement. The model's Precision stabilized around 0.8, while the Recall reached 0.9, demonstrating the model's ability to correctly classify both fish species.

The model was evaluated using Confusion Matrix (Figure 3), where the model successfully detected O. celebensis with a precision of 0.96 and recall of 0.95, and O. javanicus with a precision of 0.87 and recall of 0.81. Although there were some errors in the classification of O. javanicus, the overall results show that the model was quite effective in this fish species detection and classification task.

Based on the results of 5-fold cross-validation and visual analysis on the performance metrics of each model, Model 4 showed the best results with more consistent and higher precision, recall, and mAP values than the other models.

4.2.2 Experiment 3: Ensemble Method (AdaBoost)

In the third experiment, the ensemble approach was applied by combining five models from the cross-validation results using the AdaBoost algorithm. The aim was to improve the prediction accuracy by correcting the weaknesses of each individual model. Each model was given a different weight based on their performance, with the model that had more prediction errors gaining more weight in subsequent iterations.

After using the AdaBoost ensemble, mAP50 increased to 0.81, and mAP50-95 increased to 0.63. In addition, precision increased to 0.82, while recall increased to 0.86. Combining these models proved to be effective in correcting errors that may occur in individual models, especially in the case of images that are more difficult to identify due to differences in lighting or object position.

In the third experiment, the ensemble approach was applied by combining five models from the cross-validation results using the AdaBoost algorithm. The aim was to improve the prediction accuracy by correcting the weaknesses of each individual model. Each model was given a different weight based on their performance, with the model that had more prediction errors gaining more weight in subsequent iterations.

After using the AdaBoost ensemble, mAP50 increased to 0.81, and mAP50-95 increased to 0.63. In addition, precision increased to 0.82, while recall increased to 0.86. Combining these models proved to be effective in correcting errors that may occur in individual models, especially in the case of images that are more difficult to identify due to differences in lighting or object position.

4.3 Performance Evaluation

Performance evaluation of the ensemble models showed that combining the five models with the AdaBoost technique resulted in a significant improvement in the detection and classification of both fish species. With an mAP50 of 0.80 and mAP50-95 of 0.65 on the test dataset, this indicates that the ensemble method is highly effective in dealing with real-world image complexity.

The Confusion Matrix of the ensemble model shows an increase in better detection for both species. O. celebensis had a precision of 0.96 and recall of 0.95, while O. javanicus had a precision of 0.87 and recall of 0.81. These results show a steady improvement from each experiment, with the ensemble model giving the best performance.

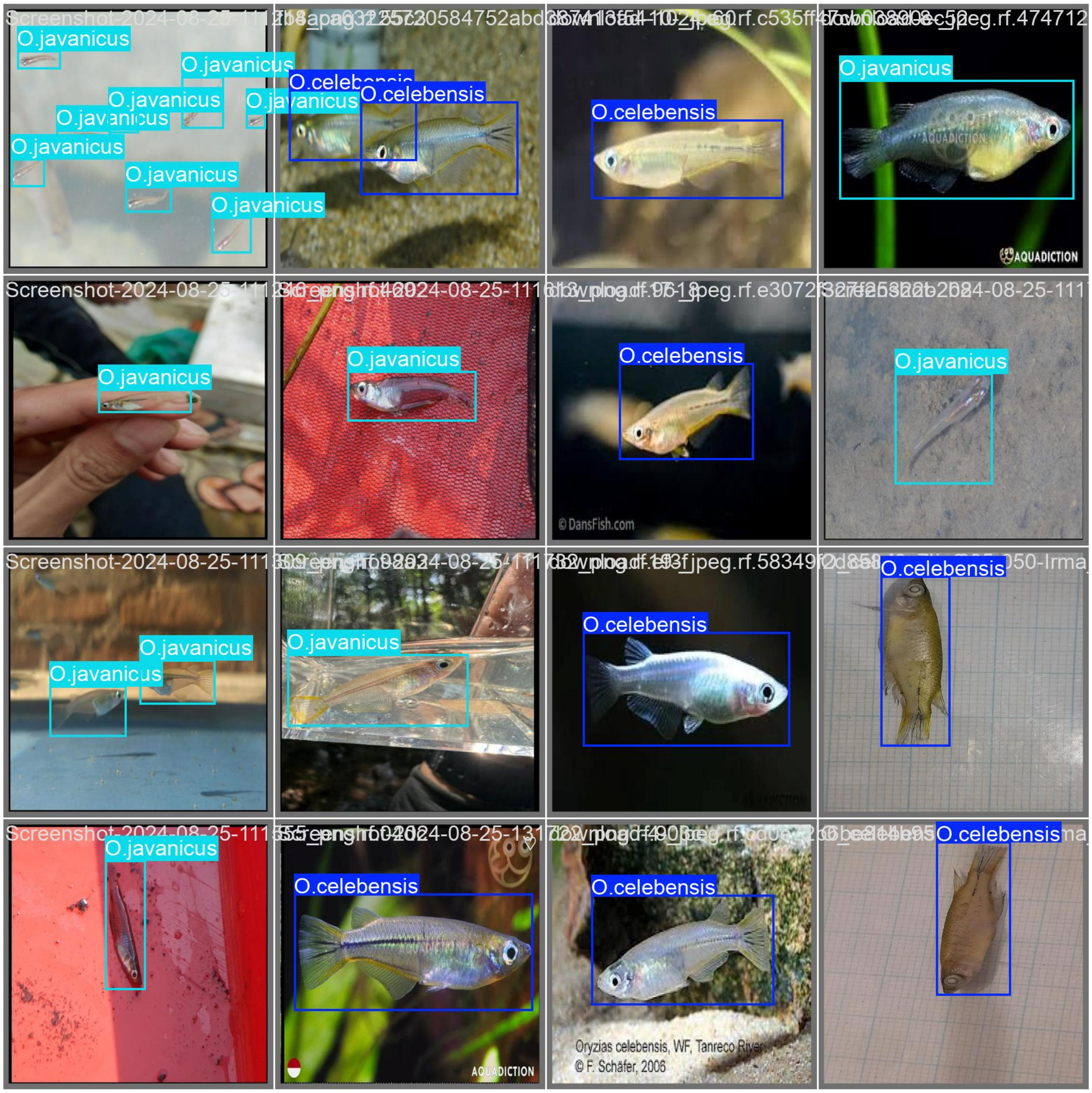


Figure 11: Resulted mixture images data with their annotation respectively

5 Discussion

5.1 AdaBoost Ensemble Method

One of the main contributions of this research is the application of the AdaBoost ensemble method to the YOLOv8 model. The AdaBoost algorithm helps mitigate the limitations of individual models by focusing on misclassified samples. In each iteration, greater weight is assigned to samples that were incorrectly predicted in the previous iteration, allowing the ensemble to improve detection accuracy for more challenging images, such as those involving small or partially obscured Medaka fish.

By integrating AdaBoost with YOLOv8, the model becomes more robust, especially in scenarios where the data is imbalanced or complex. This ensemble approach also contributes to reducing bias, enhancing the model's ability to generalize beyond the training data. The improved mAP values, particularly in difficult image scenarios, demonstrate the ensemble's effectiveness in boosting prediction accuracy.

5.2 5-Fold Cross-Validation for Model Generalization

Another major contribution of this paper is the implementation of 5-fold cross-validation, which played a critical role in preventing overfitting and ensuring that the model performed well on unseen data. By splitting the dataset into five subsets and training the model iteratively, the process provided more reliable and stable performance metrics. Each subset was used once as the validation set while the other four subsets served as training data, resulting in averaged performance metrics that better reflect the model's true capabilities.

This method enabled a thorough evaluation of YOLOv8's performance under different data conditions, showing that the model was capable of detecting and classifying Medaka fish species consistently, even in challenging real-world images. The consistent high precision and recall values across the folds demonstrate the model's strong generalization ability.

5.3 Key Contributions to Real-World Application

The integration of these techniques into a YOLOv8-based system for endangered species detection contributes significantly to the field of conservation technology. The ability to detect small, rare species with high accuracy has immediate applications in monitoring populations, tracking habitat changes, and assisting conservation efforts in real-time. AdaBoost and 5-fold cross-validation ensure the model's robustness and reliability in varying environments, offering a practical solution for real-world applications where data might be limited and imbalanced.

6 Conclusion

In this research, we applied AdaBoost and 5-fold cross-validation techniques to enhance the YOLOv8 model for the detection and classification of rare Medaka fish species. These contributions are particularly significant in the context of species conservation, where the accurate identification of endangered animals is critical. Our experiments demonstrated that combining AdaBoost with YOLOv8 improved overall precision and recall, making the model more robust in complex environments, such as underwater habitats where visual challenges are frequent.

Additionally, the use of 5-fold cross-validation allowed for a more thorough evaluation of the model's performance, minimizing the risk of overfitting and enhancing the generalization capability. These techniques resulted in more accurate predictions, particularly for small objects, addressing a notable gap in the application of object detection algorithms to conservation challenges.

While boosting algorithms like AdaBoost provide substantial performance improvements, their computational cost should be considered, especially for real-time applications. Future work can explore other boosting techniques, such as XGBoost or Gradient Boosting, to further optimize the model's performance without excessively increasing computational demands. Hybrid approaches that combine ensemble learning with architectural innovations in deep learning could yield even more efficient models for endangered species detection and other complex image classification tasks.

References

Arbogast, J. W. and Mehta, S. (2016). Data augmentation for deep learning based accelerated mri reconstruction. *arXiv preprint arXiv:1609.05148*.

Dang, T., Le, D.-H., and Nguyen, H. T. (2020). A survey on deep learning for object detection. *Journal of King Saud University - Computer and Information Sciences*.

Kalafi, E. and Javanmard, M. (2018). A review on deep learning approaches in underwater image processing. *International Journal of Computer Vision and Image Processing*, 8(1):1–15.

LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *Nature*, 521(7553):436–444.

Leow, W. K. and Savariar, B. (2015). Challenges and techniques in underwater imaging. In *2015 International Conference on Underwater Systems Technology: Theory and Applications (USYS)*, pages 1–5.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., and Berg, A. C. (2016). Ssd: Single shot multibox detector. *Computer Vision and Pattern Recognition (CVPR)*, pages 21–37.

Redmon, J. and Farhadi, A. (2017). Yolo9000: Better, faster, stronger. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7263–7271.

Salimi, M. and Bai, Y. (2016). Real-time fish detection and tracking in underwater videos based on deep learning. *Neurocomputing*, 275:1–12.

Solovyev, R., Wang, W., and Gabruseva, T. (2021). Weighted boxes fusion: Ensembling boxes from different object detection models. *Image and Vision Computing*, 117:104–127.