Electronic Letters on Computer Vision and Image Analysis 24(2):312-337, 2025

PFM-TurtleNet: Sea Turtle Species Identification Using a Parallel Fusion Module

Hebron Prasetya[1], Duy-Linh Nguyen[2], Shara Oktavia Pantouw[3]
Imanuel Kutika[4], Feisy Diane Kambey[5], and Muhamad Dwisnanto Putro[6]

[1,4,6]*Master Program of Informatics, Postgraduate Program, Sam Ratulangi University, Manado, Indonesia*
[3,5] *Department of Electrical Engineering, Sam Ratulangi University, Manado, Indonesia*
[2] *Department of Electrical, Electronic and Computer Engineering, University of Ulsan, Ulsan, Korea*

Received 28th of June, 2025; accepted 2nd of December, 2025

Abstract

Sea turtle species identification is vital for marine biodiversity conservation, as sea turtles impact marine ecosystem balance by consuming dead seagrass and maintaining coral reefs. They help preserve the health of seagrass beds and coral reefs that benefit commercially valuable species. Therefore, to sustain sea turtle populations, detection systems that facilitate conservation efforts are essential. In developing underwater detection models, researchers must address several challenges specific to the underwater environment, including low illumination conditions, complex backgrounds, and underwater blur effects. In addition, YOLOv10-nano has emerged as the most efficient object detector in its family, though improving its performance remains a challenge. To overcome this issue, we propose an advanced deep learning approach using modified YOLOv10-nano with a new Parallel Fusion Module (PFM) integrated into the backbone alongside self-attention to enhance detection performance, named TurtleNet. The Parallel Fusion Module enhances detection performance by capturing channel-wise representational features. It emphasizes channels with relevant information through a dual-scaling process, improving feature quality. PFM is integrated into the untouched branch of the Partial Self-Attention mechanism to enrich the split half of the feature channels. Our model uses 48,302 images from Bunaken National Marine Park containing Green, Hawksbill, and Olive Ridley turtles with data augmentation applied. The method leverages YOLOv10-nano's real-time detection capabilities while the PFM optimizes feature fusion and localization accuracy. Experimental results show our model achieves an mAP50 score of 0.856 and runs at 28 FPS on CPU devices, outperforming existing approaches in precision, recall, and efficiency. This research combines computer vision with marine biology, creating an automated system that helps researchers and conservationists monitor endangered turtles.

Key Words: Underwater Detection, Turtle Species, Conservation, YOLOv10-nano, Parallel Fusion Module.

Correspondence to: dwisnantoputro@unsrat.ac.id

Recommended for acceptance by Angel D. Sappa
https://doi.org/10.5565/rev/elcvia.2297
ELCVIA ISSN:15108-5097
Published by Computer Vision Center / Universitat Autònoma de Barcelona, Barcelona, Spain

1 Introduction

Sea turtles are endemic animals that play a crucial role in the environmental state of their habitat. Recent estimates from the Olive Ridley Project indicate that approximately 6.5 million sea turtles remain in the wild [1], with population numbers varying significantly between species. For instance, the critically endangered hawksbill turtle is estimated to have only 57,000 to 83,000 individuals left worldwide. Recognized for their crucial ecological roles, these marine animals contribute to ecosystem structure and dynamics as prey, consumers, competitors, and hosts. They aid in nutrient and energy transfer between ecosystems and can significantly impact the physical structure of marine habitats. The National Bunaken Marine Park ecosystem is inhabited by three sea turtle species: Chelonia mydas, Lepidochelys olivacea, and Eretmochelys imbricata [2]. Each possesses distinctive characteristics and ecological functions as keystone organisms that regulate marine ecosystem dynamics. Chelonia mydas, commonly known as green turtles, derive their nomenclature from the distinctive green-colored adipose tissue resulting from their seagrass-based diet, which enables them to function as ecological regulators controlling seagrass proliferation within marine ecosystems. Eretmochelys imbricata, characterized by its distinctive morphological features including a pointed beak and scute-forming shell margins that inspire its common name ”hawksbill turtle”, contributes to coral reef ecosystem maintenance through selective predation of sponges, facilitated by its elongated rostrum that enables access to sponges situated within coral interstices. Lepidochelys olivacea, exhibiting the lowest population density among the three sea turtle species in the National Bunaken Marine Park, is morphologically distinguished by its comparatively larger body mass, more prominent carapacial dome, and characteristically darker pigmentation; this species contributes to coastal geomorphological enrichment through its nesting behaviors along shoreline habitats. But sea turtle facing significant anthropogenic challenges like habitat loss and climate change, these marine reptiles require urgent conservation to maintain ecosystem resilience[3, 4].

Current conservation activities are typically performed manually, which involves diving into the ocean to directly conserve sea turtle habitats. This approach can be considered inefficient because it requires substantial resources and effort. To enhance conservation efforts, a detection system that can distinguish between turtle species based on their physical characteristics would be beneficial[5]. Similar approaches have been explored in biometric recognition systems using multichannel moment-based feature extraction for accurate identity classification[39]. The development of detection systems in recent years has significantly advanced detection algorithms, improving both performance and computational efficiency[6, 20]. In the specialized domain of marine biology, developing detection systems capable of recognizing sea turtle species in National Bunaken Marine Park requires overcoming substantial underwater obstacles. These challenges include color variations induced by water depth, extremely low and inconsistent light sources, and optical distortions created by the marine environment[30]. Such environmental complexities substantially complicate the accurate identification of different sea turtle species through advanced imaging and detection technologies. The system must have algorithms that can adapt to these dynamic underwater conditions, ensuring reliable species recognition despite visual interference[7, 21]. Beyond underwater challenges, an effective detection system must also integrate seamlessly with the existing technological ecosystem, necessitating an approach that not only performs exceptional detection tasks but also maintains reasonable computational requirements[8, 22]. The ideal system would balance high-precision species identification with efficient resource utilization, making it practical for marine researchers and conservationists. The development of such advanced detection systems represents a critical intersection of marine biology, computer vision, and artificial intelligence, offering potential breakthroughs in understanding and protecting marine biodiversity[9, 23].

Convolutional neural network (CNN) has been rapidly developed as excellent extractor in object detection and suitable to address this issue[33, 34]. Recent works have also explored advanced hybrid feature extraction techniques, such as fractional-order Legendre moments combined with deep neural networks, to improve image representation and recognition performance in complex visual environments[38]. CNN using updated weights to discriminate relevant features from irrelevant features regarding the physical attributes of the turtle by input image. Among the (CNN) architectures, YOLO (You Only Look Once) stands out as a high-performance object

detection framework characterized by its computational efficiency and real-time processing capabilities[10, 24]. This architecture leverages advanced deep-learning techniques to achieve rapid and accurate object detection across diverse visual environments. The YOLO (You Only Look Once) family of object detection algorithms has evolved through multiple iterations, each bringing improvements to the field of computer vision. Among these advancements, YOLOv10 presents the trade-off between detection performance and computational speed. Notably, YOLOv10 offers lower latency than its predecessors YOLOv8 and YOLOv9[11], while maintaining comparable or superior detection accuracy. This makes YOLOv10 particularly advantageous for resource-constrained environments such as embedded systems in real-time underwater monitoring. The architecture of YOLOv10 is designed with a focus on lightweight computation, making it highly suitable for scenarios where quick decision-making is essential. In recent research, using the YOLO architecture focuses on computational efficiency for underwater object detection, but in this work, the YOLOv10 algorithm needs to improve detection performance to overcome underwater challenges, so that it can recognize turtle characteristics in more detail[25]. In certain studies, attention modules are pivotal in improving model performance by enabling the model to focus more effectively on critical information across spatial and channel dimensions. The Partial Self-Attention (PSA) mechanism in YOLOv10 is designed to focus on spatial information, but it has a limitation. It only processes half of the feature maps at a time, leaving the other half unused. This means some important information may not be considered. To solve this, a channel attention mechanism is needed to help the model capture important features across both spatial and channel dimensions. To address this issue, we introduce the Parallel Fusion Module (PFM). The PFM works by ensuring that all feature maps are utilized. In our channel attention algorithm, Global Average Pooling (GAP) and Global Max Pooling (GMP) reduce each feature map to a single scalar value. GMP selects the maximum value, while GAP computes the average. These values are then passed through a fully connected layer to identify which feature maps contain the most important information. This approach ensures that no feature map is ignored, improving the model's ability to capture essential features. Subsequently, a probabilistic activation function is applied to highlight the feature maps with the highest likelihood of containing relevant information[12].

In this study, we propose an advanced deep-learning framework for identifying sea turtle species within the Bunaken Marine Park ecosystem. By implementing a YOLOv10-nano model enhanced with a parallel fusion mechanism incorporating a refined channel attention algorithm, we aim to improve the accuracy of marine species detection. The proposed approach dynamically recalibrates feature representations by replacing the fully connected layer with a 1×1 convolution, enabling a more precise capture of the unique physical characteristics of sea turtles. Additionally, the parallel fusion mechanism enhances feature maps by leveraging identity values, facilitating the accurate discrimination of turtle features within complex underwater environments. The key contributions of this study can be outlined as follows:

- We propose a novel vision-based sea turtle species detection framework designed to recognize the species of sea turtles found in Bunaken Marine Park, utilizing an advanced and efficient architecture.
- A new lightweight model is introduced to extract distinctive features that effectively differentiate between sea turtle species. The model employs a minimal number of convolutional layers while focusing on enhancing precision in contextual localization.
- We introduce a novel Parallel Fusion Module (PFM) designed to enhance feature extraction by improving channel attention integration. The proposed module refines the Partial Self-Attention (PSA) mechanism and is modified to dynamically optimize feature representation by selectively emphasizing both spatial and channel-wise critical information.
- A comprehensive comparative evaluation is conducted to assess the performance of the proposed detector against existing YOLO architectures. The results demonstrate the superior detection capability of our improved YOLOv10-nano model, enhanced by the proposed attention mechanism. The study also compares performance with several popular attention mechanisms using the Bunaken Marine Park sea

turtle dataset. In addition, it evaluates the model's efficiency and runtime speed on low-cost devices such as CPUs, confirming its practical deployment potential.

2 Related Works

This section introduces the context of sea turtle conservation in Bunaken National Sea Park and explains the development of an automated sea turtle species detection system. Finally, it describes the implementation of YOLOv10, a state-of-the-art object detection algorithm, for the sea turtle species identification task.

2.1 Application of YOLO

YOLO is one of the Convolutional Neural Network architectures widely used by researchers for object detection tasks due to its ability to perform real-time detection[29, 15]. YOLOv10 is an iteration of the previous YOLO versions, where each iteration has improved its performance to achieve more precise detection results. One notable application of object detection was demonstrated in the work of[13], which focused on underwater object detection. The research modified the YOLOv10 architecture to enhance the model's ability to detect underwater objects by utilizing two key modules, the dual partial attention mechanism (DPAM) and the dual adaptive label assignment module with the sun glint removal module (DALSM). Adding these modules successfully improved YOLOv10's performance, achieving mAP50 scores of 0.893 and mAP50:95 scores of 0.747 on the Underwater Object Tracking (UOT32) and Deep Fish Object Detection Datasets, respectively.

2.2 Object Underwater Detection

Developing deep learning methods has led to breakthroughs in new algorithms that offer improved performance in underwater object detection. The[14] research analyzes the performance of several YOLO algorithms, namely YOLOv3, YOLOv5, and YOLOR. These three YOLO architectures were trained and tested on underwater datasets with two class categories Fish and Turtle as objects to be detected to determine the performance of these three architectures. The results showed that YOLOv3 had a higher mAP score compared to the YOLOv5 architecture. YOLOv5 achieved mAP50-95 scores of 0.464 for the fish category and 0.771 for the Turtle category, while YOLOR-P6 and YOLOR-W6 achieved mAP50-95 scores for the Turtle category of 0.992 and 0.902 respectively. However, YOLOv3 had better detection accuracy performance than both architectures, achieving a mAP50-95 score of 0.902 for Turtle.

This work[15] uses YOLOv7 as the primary framework for an underwater object detection system, referred to as the YOLOv7–CHS network. The proposed model replaces the Efficient Layer Aggregation Network (ELAN) backbone with High-Order Spatial Interaction (HOSI) to enhance feature extraction. Additionally, a Contextual Transformer (CT) is integrated, facilitating the combination of both dynamic and static contextual representations. To further improve detection capabilities, the model incorporates the Simplified Parameter-Free Attention (SPFA) module, which enables simultaneous channel and spatial attention, enhancing the extraction of relevant features. These modifications result in a system with robust underwater object detection performance, achieving a mean Average Precision (mAP) of 52.8% at 50% Intersection over Union (IoU), with a computational complexity of 40.3 GFLOPS and an inference speed of approximately 32 Frames Per Second (FPS).

This work[16] utilizes a YOLOv10-based architecture, enhanced with a Dual Partial Attention Mechanism (DPAM) to improve feature extraction by focusing on key scene highlights, such as temporal context and critical elements necessary for interpreting the dynamic underwater environment. Additionally, a Dual Adaptive Label Assignment with Sun Glint Removal Module (DALSM) is incorporated to refine detection accuracy. The proposed system also integrates Marine Fusion Loss (MFL), which combines Binary Cross-Entropy (BCE) loss and Complete Intersection over Union (CIoU) loss to enhance bounding box localization while leveraging spatial context to capture essential underwater features. These enhancements result in a high-performance

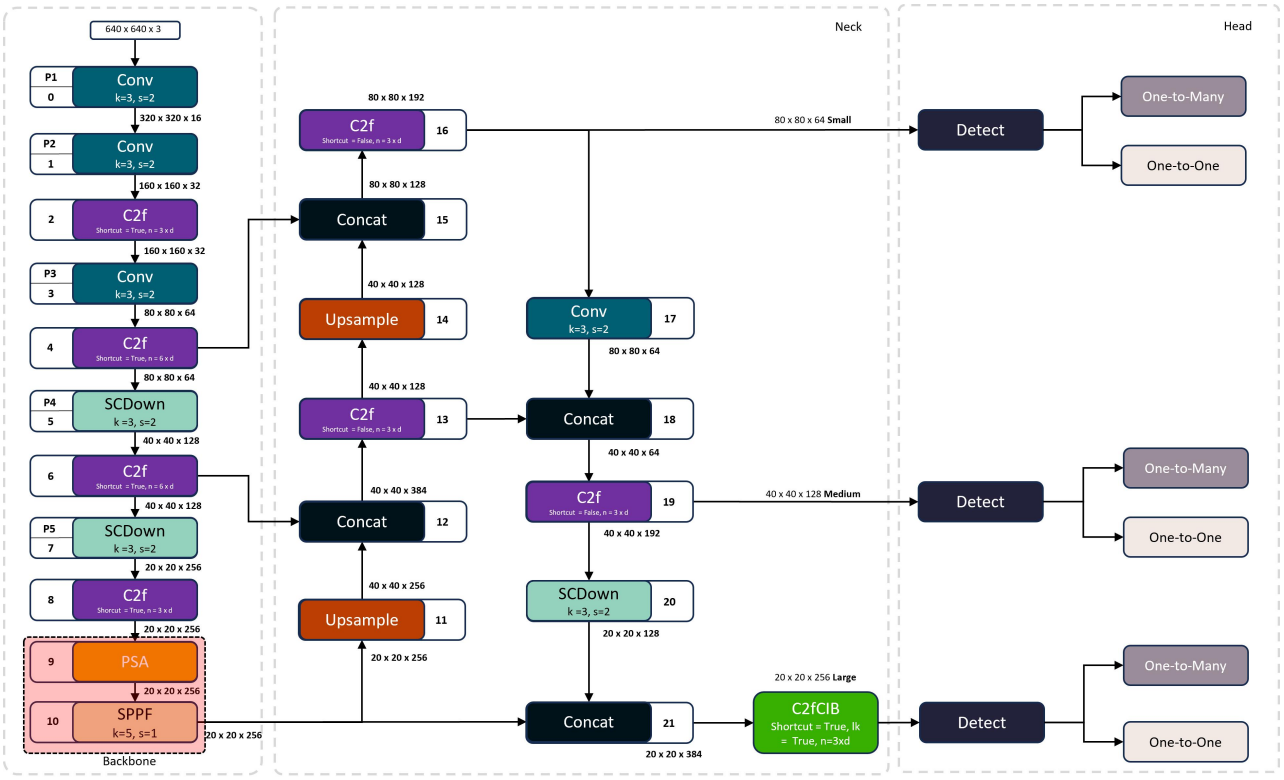


Figure 1: A visualization depicting the TurtleNet architecture with PFM module integration at the PSA component.

underwater object detection system, achieving a mean Average Precision (mAP) of 90.21% at 50% Intersection over Union (IoU) and 75.04% across mAP 50–95.

2.3 Sea Turtle Detection

In the previous years, there is a research that attempted to create a turtle detection system by utilizing a combination of Unoccupied Aircraft Systems, commonly known as drones, and CNN algorithms as a method for detecting turtles. This system would detect turtles from the sea surface or the air. This approach using the system certainly requires algorithms that need efficient computation, and this approach has not yet identified turtle species because images taken from above the sea are difficult to recognize due to the challenges of light reflection from the sea and detection errors on other objects in the sea. This certainly occurs due to the limitations of an overly simple model, and the drone approach does not represent the actual marine environment. The research [17] achieved 99.83% accuracy, but the recall and precision scores were only 76.5% and 16.3%, respectively, indicating the need for a more appropriate approach to identify sea turtles.

Lately, some studies focused on turtle detection algorithms, trying to improve their performance. Recognizing turtles requires a robust module to distinguish information between the turtle and the background. This is because underwater environments have many visual challenges, such as unstable lighting, visual disturbances such as blur and noise, and the resemblance of the turtle to the Background, therefore research[18] improves the performance of the YOLOv8 architecture by adding an Excited Group Attention (EGA) module placed on the YOLOv8 backbone, It functions to capture specific information from turtles using low computational costs with 2.12 million parameters, by utilizing grouped convolution, in addition to the shuffling mechanism applied in this study to spread features to be more evenly distributed, then grouped attention is also used to give more attention to which feature maps have potential turtle information from each group. The result of this research

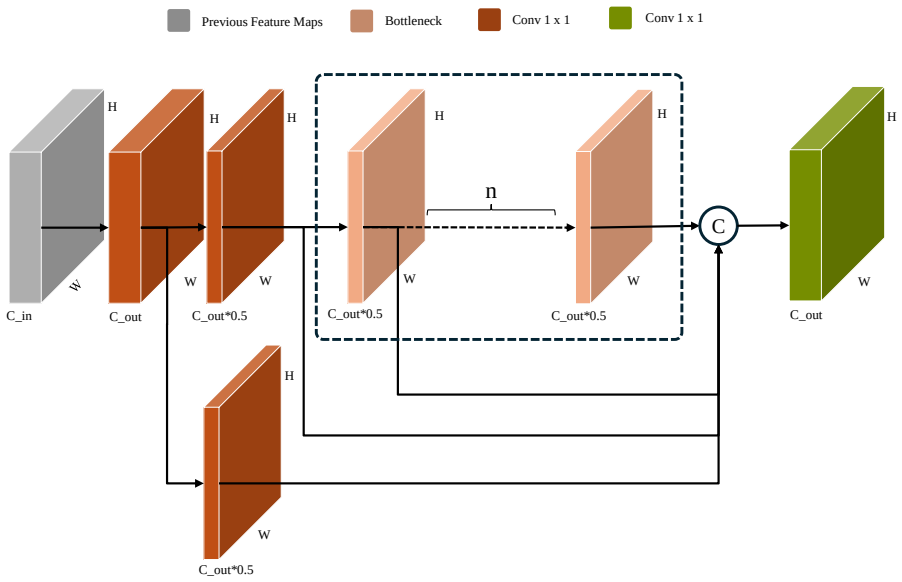


Figure 2: Structure of the C2f block in YOLOv10.

successfully obtained a mAP50 score of 0.909 with a mAP50:95 of 0.616, this result proves that the proposed EGA module has succeeded in improving the performance of the YOLOv8 architecture.

The work[19] with a similar approach still uses the YOLOv8 architecture as the main framework for an efficient sea turtle detection system but with the addition of two modules that are intended to help give more spatial attention by using dual-spatial enhancement that utilize two representations of Global Average Pooling, and Global Max Pooling to highlight the local context. In addition, there is also help from the Gradual Receptive Enhancement (GRE) module that utilizes the Grouping convolution method to improve the extraction and discriminative performance of features between sea turtles and the background, and there is also help from the Shuffle channel method that serves to spread the features to be more well distributed. All these approaches resulted in a system that is efficient in detecting sea turtles with good performance, achieving a score of mAP: 50 of 0.944 and mAP50: 95 of 0.604 with only 7.2 Glops of computation.

3 Proposed Method

This section presents the proposed approach for detecting and classifying sea turtle species in Bunaken National Sea Park. The research utilizes YOLOv10, an object detection architecture featuring a Cross Stage Partial Network backbone and PANet feature fusion network. It uses unique dataset that contains underwater images of three sea turtle species captured in Bunaken National Sea Park, with appropriate annotations for training the detection model.

The development of CNN methods in various research has been progressing rapidly, and one of the architectures implementing the CNN method is the YOLO (You Only Look Once) architecture. The TurtleNet employs a modified YOLOv10-nano architecture, comprising several CNN modules that extract information from detected objects. YOLO has evolved to YOLOv10, similar to other YOLO versions, which consists of three main components: Backbone, Neck, and Head.

3.0.1 Backbone

As show in Figure 1, the backbone in an object detection network is the core component responsible for extracting and refining features from input data by preserving relevant information while minimizing noise prediction.

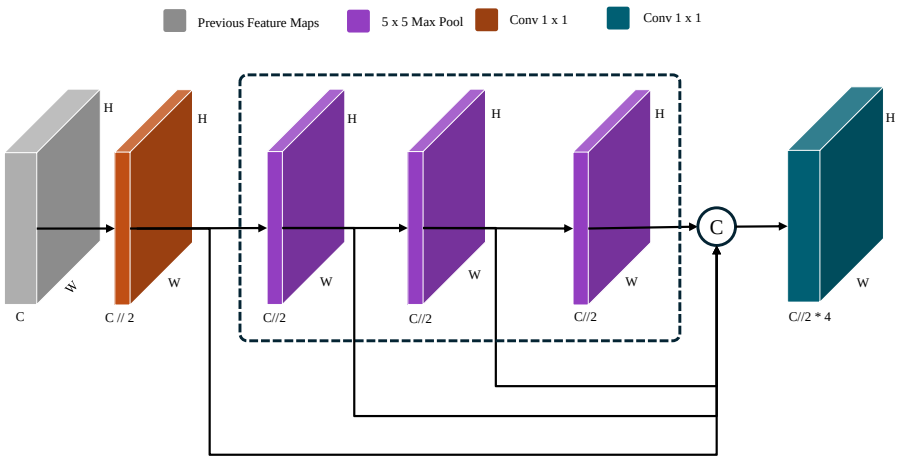


Figure 3: Structure of the Spatial Pyramid Pooling-Faster (SPPF) block in YOLOv10.

This process relies on convolutional layers that efficiently capture essential patterns and enhance feature representation. By leveraging multi-kernel weighting, the backbone selectively emphasizes critical features, while its filter weights are iteratively updated during training to improve predictive performance. Additionally, the hierarchical organization of extracted features spanning low, medium, and high levels, facilitates robust multi-scale object detection. Low-level features retain fine-grained details crucial for identifying small objects, whereas high-level features provide broader contextual information necessary for recognizing larger and more complex objects. This structured approach ensures that the backbone enhances both the accuracy and computational efficiency of the object detection model, making it well-suited for real-world applications.

3.0.2 C2f

The C2F (CSP Bottleneck with 2 Convolutions) module is integral to optimizing feature extraction in object detection models. By partitioning the input data into multiple distinct channels, the module facilitates parallel processing of diverse feature sets. This allows the model to simultaneously focus on different aspects of the data, thereby enhancing the granularity of feature extraction, as shown in Figure 2. The outputs from these separate channels are processed independently before being recombined, resulting in a more comprehensive and detailed feature representation. This approach reduces redundancy, helping the model focus on important features, which lowers the risk of overfitting and improves generalization.

3.0.3 SPPF

Spatial Pyramid Pooling Fast (SPPF) is a technique developed to address the challenges of scaling and computational efficiency in object detection, particularly when dealing with objects of varying sizes within a single image. This approach is specifically designed to enhance the model's capability to efficiently extract features from images, optimizing both speed and accuracy in real-time processing. The SPPF module begins with a 1×1 convolution operation, which serves to reduce the dimensionality of the input data, thereby preparing it for subsequent processing. After the 1×1 convolution, three 2D max pooling layers with 5×5 kernels are applied, as shown in Figure 3. These pooling layers operate concurrently, effectively capturing features with a broader receptive field. This enables the model to extract both local and global information simultaneously, allowing for a more comprehensive representation of the image. The choice of utilizing three max pooling layers with identical 5×5 kernel sizes is crucial in expanding the receptive field across network layers while minimizing computational overhead. This design choice ensures that the model remains computationally efficient, even when processing images containing objects of diverse sizes.

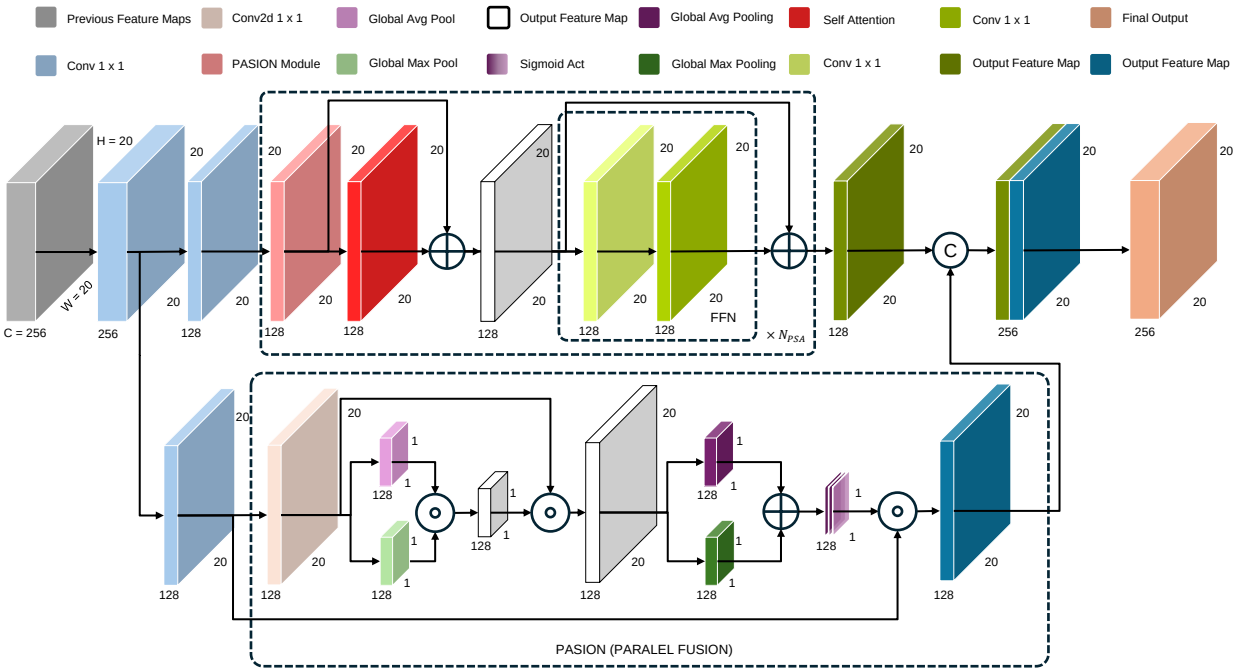


Figure 4: The Proposed Model Parallel Fusion Module with PSA Block.

3.0.4 Partial Self Attention (PSA)

Partial Self Attention (PSA) is a new module implemented in YOLOv10. It uses self-attention to extract features by focusing on its own values during the process. Self-attention uses the feature map values to produce weights that will be multiplied using matrix multiplication operations. It generates fewer parameter, but the amount of computation required will be higher. Therefore, Partial Self Attention only uses a portion of the total channels to avoid excessive computation cost. Figure 4 illustrates Partial Self Attention, beginning with a 1×1 convolution to regulate the number of channels needed. In addition, it is divided into two parts with the number of channels multiplied by a 0.5 ratio. One of the parts involves the Self-Attention and Feed-Forward Network (FFN), which then combines with the original feature map through a residual connection, enhancing the representational capacity of the input feature map. The PSA module comprises the Self-Attention module and includes a Feed-Forward Network (FFN) module consisting of two 1×1 convolution blocks with channel configurations similar to the Inverted Bottleneck, which increases the amount of feature map information. The output feature map from the FFN is further integrated with its input via a residual connection, facilitating better gradient flow and improved feature representation. Moreover, the output feature maps from the two parts are combined using a concat operation and then mixed using the 1×1 convolution.

- **Self Attention**
  As shown in Figure 5, the self-attention block focuses on global learning by applying multi-dimensional transformations that capture global information while keeping computational costs low. In the first stage of self-attention, denoted as X_f, the feature maps expand to generate the query (Q), key (K), and value (V) representations. The number of feature maps for V becomes twice the size of those for Q and K. This process follows the equation below.

$$X_f = C_{(1\times1)}(X_c^1), \tag{1}$$

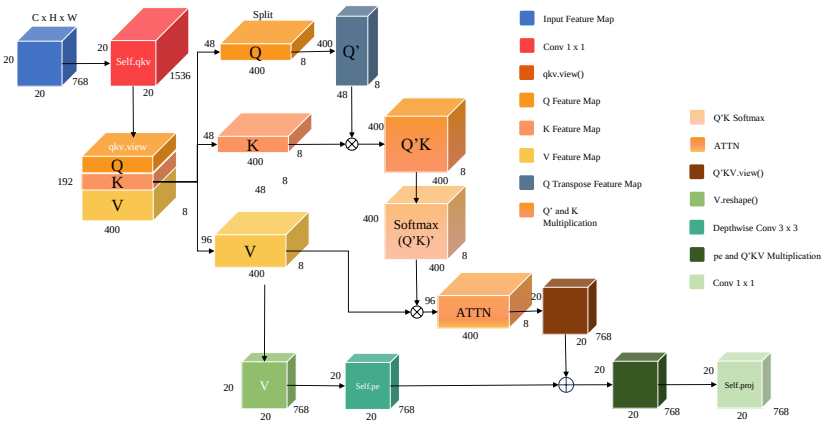


Figure 5: Structure of the Self-Attention block in YOLOv10.

where

$$X_f = (Q, K, V). \tag{2}$$

This expansion stage uses the $C_{(1\times1)}$ block which contains 1×1 convolution, batch normalization, and SiLU activation blocks. It produces maps with $X_c^1 \in \mathbb{R}^{(B,C,H,W)}$, which will be divided into Q, K, and V, where their respective dimensions are $Q \in \mathbb{R}^{(B,numhead,keydim,N)}$, $K \in \mathbb{R}^{(B,numhead,keydim,N)}$, and $V \in \mathbb{R}^{(B,numhead,headdim,N)}$. Subsequently, the module extracts the representations of Q, K, and V. The context modeling is then computed by performing matrix multiplication between the transpose of Q and K, normalized by a scaling factor (s). To further investigate the underlying mechanisms of this process, as described follow as:

$$CTX = (Q^T \times K) \times s. \tag{3}$$

Matrix multiplication is applied to capture the positional relationships between Q^T and K, enabling the model to independently extract global information without requiring a large number of parameters. To emphasize the most relevant features, the attention context is passed through a softmax function, producing attention weights $(attn)$ that assign relative importance to each element within the value representation. This process can be expressed as follows:

$$attn = Softmax(CTX). \tag{4}$$

Non-linear activation plays a crucial role in this stage, particularly the softmax function, which generates probability values concentrated on the most significant elements. The resulting attention weights are then used to compute the primary self-attention output, denoted as M_p, through matrix multiplication between the transposed attention $(attn^T)$ and the value representation V. This process is expressed as follows:

$$M_p = V \times attn^T. \tag{5}$$

In this process, the context modeling feature maps aim to identify positional combinations within the input values that contain relevant object information, thereby refining the input representations. To further enhance feature representation by incorporating local details, a depth-wise convolutional transformation

(C_{DW}) is applied to V, enabling the extraction of spatial information. This process can be explained as follows:

$$FM = V_c + M_p. \tag{6}$$

where

$$V_c = C_{DW}(V). \tag{7}$$

Depthwise convolution is applied to the Value (V) feature maps with a 3×3 kernel to enhance learning through weight operations. The refined feature representation (FM) results from aggregating the self-attention output (M_p) with the transformed feature representation (V_c) using element-wise summation. Afterward, a 1×1 convolution mixes the combined feature maps to integrate the information. The overall block appears as follows:

$$SA = C_{(1\times1)}(FM), \tag{8}$$

Channel relationships are crucial, but depthwise convolution lacks inter-channel connections. To address this, a 1×1 convolution is applied to reconstruct the information and establish relationships across all channels.

- **Parallel Fusion Module (PFM)**
The Partial Self-Attention (PSA) module consists of two branches. As explained before, only one branch applies self-attention to capture global dependencies and positional relationships between features, while the other acts as a bypass, forwarding features directly without processing. This process extracts only half of the feature maps, and the attention mechanism does not operate globally. To improve feature extraction in the bypass branches, the Parallel Fusion Module (PFM) is introduced. The PFM complements the PSA by capturing global information, where the PSA focuses on spatial-wise attention, and the PFM captures channel-wise information. The PSA module starts by splitting the input feature maps X_c into two branches. The split ratio is set to 0.5, so that each branch contains half of the original channels X_c^1 and X_c^2, respectively. Moreover, the feature map is processed by a 2D convolution with a 1×1 kernel to adjust the number of channels, followed by batch normalization to normalize data across the batch, and SiLU activation for non-linearity, which together form the block called $C_{(1\times1)}$ and can be expressed as:

$$[X_c^1, X_c^2] = X_c, \tag{9}$$

where

$$X_c = C_{(1\times1)}(x). \tag{10}$$

Maintaining relationships between feature maps is essential for preserving information across channels. An efficient way to achieve this is through pointwise convolution with a 1×1 kernel, which ensures balanced information flow between the two branches. Subsequently, the feature maps are divided into two branches using a straightforward split operation, becoming X_c^1 and X_c^2, respectively. The X_c^1 branch serves as a bypass, providing input information to the proposed module P to identify channels that contain important feature information. The Convolutional Block Attention Module (CBAM) [12] serves as the primary inspiration in designing this proposed module, where the transformation of tensor dimensions into vectors constitutes a fundamental component in the process of implementing channel-wise attention. This dimensional transformation summarizes spatial information into a 1×1 size using global pooling. To prevent information loss during summarization, both global average pooling and global max pooling

are employed to capture mean and maximum values, respectively, thereby complementing each other's informational content. This process can be expressed as follows:

$$Z = Conv_{(1\times1)}(X_c^2). \tag{11}$$

As seen in Figure 4, the bypass channel from PSA uses a channel attention approach but with different algorithms for weighting. the proposed module utilizes 1×1 convolution at the beginning of the process, which evaluates feature maps at each pixel without significantly adding weight usage. After that, the maximum value and average value of the convolution results are taken using Global Max Pooling and Global Average Pooling as two different representative values for each feature map. This process can be described as follows:

$$ZA_1 = Z \circ ((GAP(Z)) \circ (GMP(Z))). \tag{12}$$

In order to capture global spatial information, Global Average Pooling and Global Max Pooling are applied to represent mean and high-value features, respectively. Their outputs are combined using the Hadamard product (element-wise multiplication), producing a single tensor with spatial dimensions of 1×1. This tensor is then multiplied element-wise with the input feature map, enhancing feature representation through pooling-based scaling factors and generating a new tensor $(ZA)_1$. The operation is performed twice with a different fusion process, and the second process concludes with a sigmoid activation. The second process can be defined as follows.

$$ZA_2 = \sigma((GAP(ZA_1)) + (GMP(ZA_1))). \tag{13}$$

Additionally, the second process of pooling is applied again to the feature map from the previous process, which are then combined using additional operations. The result of this combination is activated using a σ sigmoid activation function to create probability values between channels, indicating object locations based on channel positions. This activation result is then operated through element-wise multiplication on the feature map before the convolution at the beginning of the proposed module process. This process can be described as follows:

$$P_2 = ZA_2 \circ X_c^2. \tag{14}$$

The PFM module P_2 is placed in the bypass branch to enhance the untouched tensor. In parallel, another PFM module P_1 is applied to the second branch before the self-attention process, where it operates on the input feature maps X_c^1. The use of P is necessary to strengthen feature selection before context modeling with self-attention. This overall operation is outlined in the following process.

$$P_1 = P(X_c^1). \tag{15}$$

P_1 denotes the application of the proposed module before the self attention (SA) process, where the placement of the proposed module before SA is necessary for channel modeling before being processed by SA. This arrangement considers that self-attention relies on matrix multiplication, making it essential first to prioritize attention to channels with object relevance, thereby enhancing SA performance in Global Context modeling. Moreover, a residual operation is performed on the feature map generated by self-attention to preserve the input information from P_1 process. This process can be expressed as follows:

$$Att = SA(P_1) + P_1. \tag{16}$$

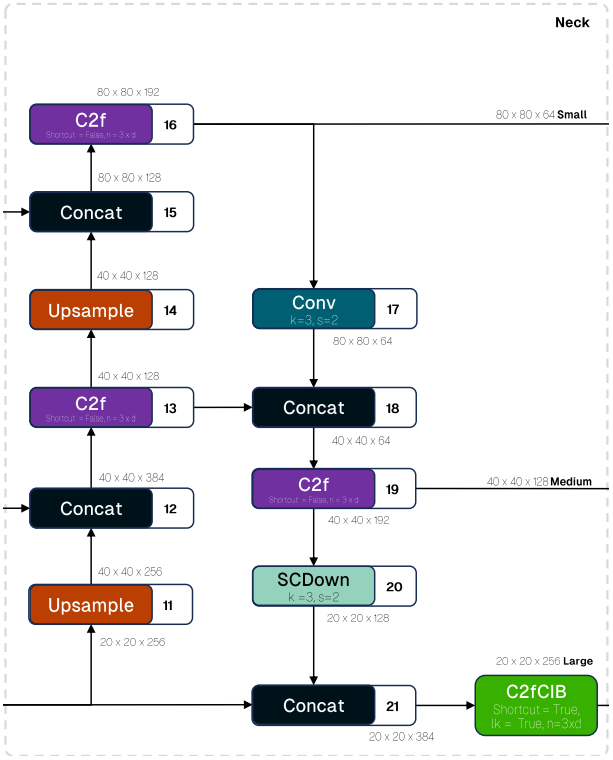


Figure 6: Structure of Path Aggregation Network as YOLOv10 Neck.

The feature map from attention process (Att) passes through the Feed Forward Network (FFN) block for further weighting using a convolutional block with a kernel size of 1×1, applied twice, followed by a residual operation with the feature map resulting from the self-attention. This process can be described as follows:

$$FFN = C_{(1\times1)}(C_{(1\times1)}(Att)) + Att. \tag{17}$$

Furthermore, the feature maps generated by P_2, which originate from the input X_c^2, are concatenated with the results of the Feed-Forward Network (FFN) and then processed through a convolutional block with a 1×1 kernel to mix information across channels between the outputs of the FFN and P_2. This process can be expressed as follows:

$$PSA = C_{(1\times1)}(Concat(FFN, P_2)). \tag{18}$$

In the Figure 4 illustrates the dimensions and process flow of the modified partial self-attention mechanism, wherein the pathway for processing X_c^1 from the original design has been augmented with a proposed block before self-attention processing. Furthermore, the X_c^2 pathway, which initially functioned solely as a bypass channel, has been transformed by incorporating the same proposed block in the self-attention pathway. In summary, the proposed PFM starts with a 1×1 convolution to adjust the input channels, followed by global average pooling and global max pooling operations. The outputs are combined through element-wise multiplication, and the resulting features pass through an additional pooling step before being processed by a sigmoid activation to generate channel-wise attention weights. The convolutional transformations use the SiLU activation function, and this module does not apply dropout.

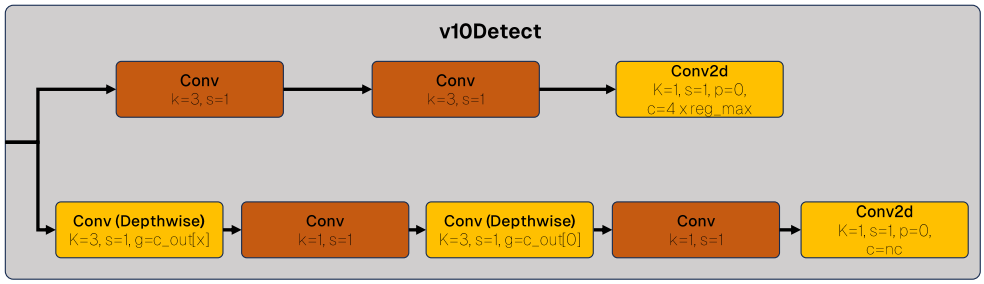


Figure 7: Structure of the Head block in YOLOv10.

3.0.5 Neck

In the neck of YOLOv10, Path Aggregation Network (PAN) plays a vital role in improving multi-scale feature fusion, which is essential for detecting objects of various sizes. The neck acts as a bridge, connecting the backbone responsible for feature extraction with the head, where object detection takes place menggunakan beberapa modul convolusi yaitu dengan memanfaatkan modul C2f, Conv, dan SCDown. Furthermore, PAN optimizes the flow of information between different levels of the feature pyramid, effectively merging data from multiple resolutions[11].

This allows the model to handle objects of varying sizes and scales with greater efficiency, thereby boosting detection accuracy. By reducing the loss of valuable information that can occur when relying on a single feature scale, as shown in Figure 6, PAN enhances the model's precision. Moreover, PAN introduces a more efficient mechanism for fusing both low and high-level features, which improves the model's ability to dynamically refine object representations. As a result, by strengthening the connections between layers in the feature pyramid, PAN enables the model to be more sensitive to crucial features, even those that might be hidden at either smaller or larger scales. This dynamic interaction between feature layers ensures that YOLOv10 can detect objects with greater accuracy, regardless of their size or complexity.

3.0.6 Detection Layer

The YOLOv10 Detection Layer (v10Detect) as shown in Figure 7 enhances object detection capabilities through the integration of depthwise separable convolutions with standard convolutional operations. This architecture implements a parallel pathway approach to process both spatial and contextual information, thereby improving object detection precision. The layer architecture initiates with dual Depthwise convolution operations, each utilizing a 3×3 kernel with a stride of 1 and appropriate padding to maintain spatial dimensions. These depthwise convolutions facilitate efficient feature extraction while minimizing computational complexity. Through the application of single filters per input channel, the operations preserve spatial details essential for object localization. Concurrently, standard convolution layers employing 3×3 kernels with a stride of 1 capture complex patterns and inter-channel relationships that may not be fully extracted through depthwise convolutions alone. The feature extraction phase culminates in the convergence of both pathways into convolution layers with 1×1 kernels. This convergence reduces feature map dimensionality and synthesizes the extracted information into a more compact and informative representation as shown in Figure 7.

A critical component of the final layers implements $C \times (4 + 1 + n_cls)$ filters, where C denotes the number of anchors, 4 represents the bounding box coordinates, 1 indicates the objectness score, and n_cls specifies the number of object classes, particularly applicable to sea turtle species classification. The v10Detect layer output undergoes subsequent post-processing procedures, including non-maximum suppression (NMS) and confidence thresholding, to optimize detected bounding boxes and eliminate redundant or false-positive detections. This methodical integration of depthwise and standard convolutions establishes an equilibrium between detection accuracy and computational efficiency, rendering it particularly suitable for sea turtle species detec-



Figure 8: Visualization of Sea Turtle Dataset in Various Angles.

tion in complex environmental contexts. The YOLOv10 loss function incorporates multiple specialized loss components, each optimizing distinct aspects of object detection. The composite loss integrates Distribution Focal Loss (DFL) for bounding box regression precision, Binary Cross-Entropy (BCE) Loss for objectness and classification accuracy, Complete Intersection over Union (CIOU) Loss for spatial alignment optimization, and Matching Metric Loss for prediction-ground truth assignment optimization. This integrated approach facilitates balanced optimization, enhancing detection performance in complex scenarios such as marine species identification. DFL enhances bounding box regression precision by modeling coordinate prediction as a probability distribution across discrete bins rather than continuous values. Instead of directly predicting continuous bounding box coordinates (x, y, w, h), the model estimates probabilities over adjacent bin locations. Given two consecutive bins Si and Si+1, the final coordinate is interpolated based on the predicted probability distribution. This process can be described as follows:

$$DFL(Si, Si+1) = -((yi+1-y)log(Si) + (y-yi)log(Si+1)). \tag{19}$$

This approach refines coordinate estimation by interpolating between adjacent bin probabilities, allowing for sub-bin level precision. Such methodology is particularly advantageous for detecting partially occluded specimens or sea turtles in diverse orientations, where accurate localization is crucial. Binary Cross-Entropy (BCE) Loss functions as the metric for objectness prediction and class probability estimation, quantifying the divergence between predicted probabilities and ground truth labels. where n represents the sample quantity, yi denotes the ground truth label (1 indicating object presence, 0 indicating absence), and pi represents the predicted probability. This component ensures accurate object detection and species classification within the marine environment. This process can be expressed as follows:

$$BCE = -\frac{1}{n}\sum_{i=1}^{n}\left(y_i \cdot \log(\hat{y}_i) + (1-y_i)\cdot \log(1-\hat{y}_i)\right). \tag{20}$$

Complete Intersection over Union (CIOU) Loss optimizes bounding box regression by incorporating three critical parameters such as, overlap area, center point distance, and aspect ratio consistency between predicted and ground truth boxes. where IoU represents the Intersection over Union between predicted box b and ground truth bgt, ρ^2 (b,bgt) denotes the squared Euclidean distance between central points, c represents the diagonal length of the minimal enclosing box, and v quantifies aspect ratio consistency, with α as the trade-off parameter. This formulation optimizes spatial alignment and dimensional consistency across varied specimen orientations. This process can be described as follows:

$$TotalLoss = \lambda_{box} L_{box} + \lambda_{dfl} L_{dfl} + \lambda_{cls} L_{cls}. \tag{21}$$

Parameters	Setup
Training	30.934
Testing	7.726
Validation	9.612

Table 1: Dataset configuration showing the number of images used for training, testing, and validation.

$$L_{CIoU} = 1 - IoU + \frac{\rho^2(b, b^{gt})}{c^2} + \frac{v^2}{(1 - IoU) + v}. \tag{22}$$

The Matching Metric Loss governs the assignment between predictions and ground truth annotations during the training phase by optimizing the selection of anchor-prediction pairs. The assignment cost A is calculated by combining three key components. First, the direct overlap term (to2o,i) measures the one-to-one matching cost between prediction i and the ground truth, ensuring spatial alignment. Second, the masking penalty (II (i $\in \Omega$) to2m, i) applies a penalty if prediction i is part of the matched set Ω, with II serving as an indicator function that activates this penalty only when the condition is met. Lastly, the cumulative mismatch penalty ($\sum$ k $\in \Omega$ n i to2m, k) aggregates the costs from other matched predictions, excluding prediction i, to account for potential mismatches within the matched set. This formulation ensures that the overall assignment cost is minimized by balancing spatial accuracy and confidence during training. Here, to2o,i, and to2m,i capture spatial and confidence-related costs, while the indicator function enforces correct assignment constraints and the summation term penalizes inconsistencies across the matched predictions[11]. The comprehensive YOLOv10 loss function represents a weighted summation of these components where λ coefficients modulate individual component contributions. This multifaceted approach maintains equilibrium between localization precision, classification confidence, and object matching efficiency, facilitating robust marine species detection and classification across diverse underwater environments. This process can be expressed as follows:

4 Datasets

The curated dataset employed in this investigation derives from previously published research[19], focusing on sea turtle species detection within the Bunaken National Marine Park ecosystem. The dataset encompasses three distinct chelonian species, including Chelonia mydas (Green Turtle), Lepidochelys olivacea (Olive Ridley), and Eretmochelys imbricata (Hawksbill Turtle). The comprehensive dataset consists of 48,272 high-resolution images, comprising 28,053 specimens of Chelonia mydas, 11,382 specimens of Eretmochelys imbricata, and 8,837 specimens of Lepidochelys olivacea. Regarding class imbalance, the dataset exhibits an unequal distribution of specimens across species, with Chelonia mydas being the most represented class. To address this issue, no artificial data augmentation is applied to balance the class distribution, ensuring that the dataset remains representative of real-world conditions.

Based on Table 1, this dataset configuration prevents any overlap between the training set and the validation or testing sets, while maintaining a controlled relationship between the validation and testing sets to mitigate overfitting. As shown in Figure 8, image acquisition is conducted exclusively within the marine ecosystem of Bunaken National Marine Park, capturing specimens under diverse environmental conditions and authentic ecological contexts. This approach enables robust model training and evaluation under real-world scenarios. Each image includes precise bounding box annotations indicating specimen locations, following the standardized YOLO annotation protocol to ensure compatibility with the YOLOv10 architecture.

Parameters	Setup
CPU	Intel i5-2320
GPU	GTX 1080 Ti
RAM	16 GB
GPU Memory	12 GB
Operating System	Ubuntu Linux

Table 2: Hardware specification for training.

5 Results and Discussion

This section examines the performance of the enhanced YOLOv10n model for sea turtle species detection in underwater environments. The analysis presents three comprehensive evaluations. First, the Component Analysis reveals how each architectural modification contributes to improved detection capabilities across different turtle species. Second, the underwater dataset testing demonstrates competitive recognition accuracy even in challenging visibility conditions, varying lighting, and complex underwater backgrounds. Finally, the processing speed assessment confirms the model's capability to operate efficiently on standard marine research equipment, making it practical for field deployment. Together, these findings validate how the enhancements to the YOLOv10n architecture significantly advance the state of underwater sea turtle identification and monitoring.

Parameters	Setup
Input Image	640×640
Epoch	100
Batch Size	16
Optimizer	Stochastic Gradient Descent (SGD)
Learning Rate	0.01

Table 3: Training configuration detailing the hyperparameter settings used during model optimization.

5.1 Experimental Setup

The proposed module and YOLOv10 architecture are implemented in Python using the PyTorch framework. As shown in Table 2, the training process leverages an NVIDIA GeForce GTX 1080 Ti graphics card (12 GB VRAM) as the primary accelerator. The system is supported by an Intel Core i5-2320 processor (3.0 GHz) and 16 GB of RAM, which manage auxiliary tasks and maintain overall workflow. The implementation is conducted on Ubuntu Linux as the operating system. The network was trained using input images normalized to 640×640 pixels, with training conducted over 100 epochs using a batch size of 16. The optimization procedure employed Stochastic Gradient Descent (SGD) with an initial learning rate of 0.01. To enhance model generalization and prevent overfitting, various data augmentation techniques were implemented, including mosaic augmentation, random translation, scaling transformations, and horizontal flipping. Model performance during the inference phase was quantitatively evaluated using the mean Average Precision at 50% Intersection over Union (mAP50) metric.

5.2 Ablation Study

Conducting an ablation study is essential for evaluating the significance of each component in a model. By isolating and testing specific modifications, researchers can identify which elements contribute to performance gains. This approach ensures that the improvements are deliberate and not merely incidental. Table 4 presents an ablation study comparing the baseline YOLOv10n, YOLOv10n-PSA, and the proposed model, highlighting

Model	Params	GFLOPs	mAP50
YOLOv10n	2,708,210	8.4	0.830
YOLOv10n - PSA	2,458,482	8.2	0.842
Proposed	**2,749,554**	**8.4**	**0.856**

Table 4: Ablation study showing the effect of the proposed PFM module compared to YOLOv10-nano and YOLOv10n-PSA in terms of parameters, computation, and mAP50.

their computational efficiency and detection performance. The evaluation metrics include the number of parameters (indicating model complexity), GFLOPS (measuring computational cost), and mAP50 (mean average precision at 50% Intersection over Union), which collectively illustrate the trade-off between model complexity and detection accuracy. The results in Table 4 demonstrate that the proposed model achieves superior detection accuracy, with an mAP50 of 85.6%, outperforming YOLOv10n-PSA (84.2%) and the baseline YOLOv10n (83.0%). Although the proposed model exhibits a slightly increased parameter count (2,749,554) compared to YOLOv10n (2,708,210), it maintains an equivalent computational cost of 8.4 GFLOPS. These findings indicate that the architectural enhancements introduced in the proposed model effectively improve detection performance while preserving computational efficiency. The evaluation considers key performance metrics, including the number of parameters, computational cost (GFLOPS), and detection accuracy measured by mAP50. These metrics provide insights into the trade-off between model complexity and detection effectiveness.

Model	Params	GFLOPs	mAP50
YOLOv10n	2,707,430	8.4	0.994
YOLOv10n - PSA	2,457,702	8.2	0.995
Proposed	**2,732,262**	**8.4**	**0.995**

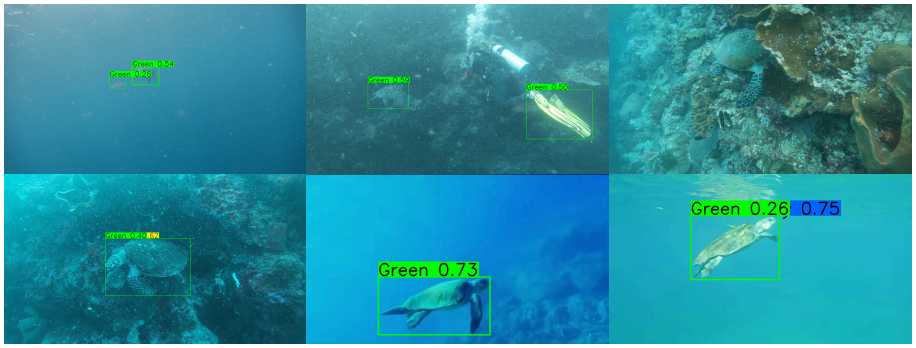
Table 5: Ablation study on SeaTurtleID2022 to evaluate cross-validation between two datasets, comparing YOLOv10-nano, YOLOv10n-PSA, and the proposed model.

The performance improvement can be attributed to the integration of the Parallel Feature Module (PFM), which enhances the model's capability to focus on the most informative feature channels. The PFM selectively emphasizes feature maps that contain discriminative patterns, such as edges, textures, and structural cues, which are crucial for both detection and classification tasks. This channel-wise recalibration enables the network to better distinguish subtle differences among classes, leading to more accurate predictions. As illustrated in Figure 9, the proposed model demonstrates clearer class separation and improved recognition stability under varying underwater conditions. Consequently, the inclusion of PFM not only increases detection precision but also enhances the model's overall classification robustness.

Model	Params	GFLOPs	mAP50 μ
YOLOv10n	2,708,210	8.4	0.823($\pm$ 0.006)
Proposed	**2,749,554**	**8.4**	**0.846($\pm$ 0.007)**

Table 6: Performance comparison of the proposed model based on three training runs.

Subsequently, to evaluate the cross-validation performance of the proposed model, the SeaTurtleID2022 dataset was utilized in a closed-set configuration as described in[40]. The dataset consists of 4,679 training images, 1,418 validation images, and 2,632 testing images. The results, summarized in Table 5, show that the proposed model achieves comparable computational efficiency (8.4 GFLOPs) to YOLOv10-nano while maintaining the highest detection accuracy (mAP50 = 0.995). This improvement is attributed to the integration of the PFM module, which enhances the feature refinement capability of the Parallel Spatial Attention (PSA) mechanism, resulting in more robust feature extraction and improved detection performance across cross-dataset



(a)



(b)

Figure 9: Visualization of evaluation results on the datasets using the PFM module, (a) YOLOv10-nano and (b) the proposed model.

evaluations.

5.3 Evaluation on Dataset

Model stability is essential to ensure the reliability of the evaluation. To assess this stability, the training process was repeated three times, and the reported results represent the average performance across these runs. As shown in Table 6, the proposed model achieved an average mAP50 of 0.846 (±0.007) compared to 0.823 (±0.006) for the original YOLOv10n, indicating an improvement of 0.023 in detection performance. In this context, the symbol μ denotes the mean (average) value, while $\pm$ represents the standard deviation across the three training runs. Furthermore, the best-performing model, obtained from the first training iteration, achieved an mAP50 of 0.856, which was used for further comparison and analysis.

Based on Table 7, several widely recognized attention mechanisms were evaluated alongside the proposed module. To ensure a fair and consistent comparison, each attention mechanism was individually integrated in place of the PFM module and tested under identical experimental conditions. the results indicate that the proposed model achieves the highest detection accuracy, with a mAP50 of 85.6%, surpassing other variants such as YOLOv10n + SE (85.4%) and YOLOv10m (86.0%). While the YOLOv10m model records the highest accuracy, it comes at a significantly higher computational cost (63.6 GFLOPS) and parameter count (16,661,794), making it less efficient for resource-constrained applications. In contrast, the proposed model maintains an optimal balance, achieving superior accuracy with only 2,749,554 parameters and 8.4 GFLOPS. Compared to other lightweight models, the proposed approach outperforms YOLOv5n (84.8%) and YOLOv8n (83.7%) while utilizing a similar computational budget. Furthermore, when compared to attention-enhanced variants such as YOLOv10n + CBAM (83.5%) and YOLOv10n + ECA (84.4%), the proposed model demonstrates superior accuracy, confirming the effectiveness of the introduced architectural modifications. In addition, transformer-based models such as YOLOv10n + ViT [36] (80.3%), YOLOv10n + Swin Transformer [37] (81.0%), and

YOLOv10n + DenseNet [35] (82.9%) also show lower detection accuracy than the proposed model, despite having comparable or higher computational costs (9.7, 15.3, and 8.4 GFLOPs, respectively). Compared with the related lightweight detector YOLOv8n [19], which achieved an mAP50 of 83.2% at a similar computational cost of 8.0 GFLOPS, the proposed PFM-TurtleNet attains a notably higher mAP50 of 85.6%. These results further highlight that the proposed PFM-TurtleNet effectively leverages lightweight design and refined feature extraction to achieve state-of-the-art performance in detecting sunken litter within complex underwater environments while maintaining computational efficiency.

Model	**Params**	**GFLOPs**	**mAP50**
YOLOv3-tiny	12,133,670	19	0.834
YOLOv5n	2,509,033	7.2	0.848
YOLOv8n	3,011,433	8.2	0.837
YOLOv8n [19]	2,500,900	8.0	0.832
YOLOv10n + CBAM	2,766,164	8.4	0.835
YOLOv10n + SE	2,768,348	8.4	0.854
YOLOv10n + ECA	2,749,541	8.4	0.844
YOLOv10n + ViT	3,521,266	9.7	0.803
YOLOv10n + DenseNet	2,733,042	8.4	0.829
YOLOv10n + Swin Transformer	2,911,478	15.3	0.81
YOLOv10m	16,661,794	63.6	0.860
YOLOv10n	2,708,210	8.4	0.830
Proposed	**2,749,554**	**8.4**	**0.856**

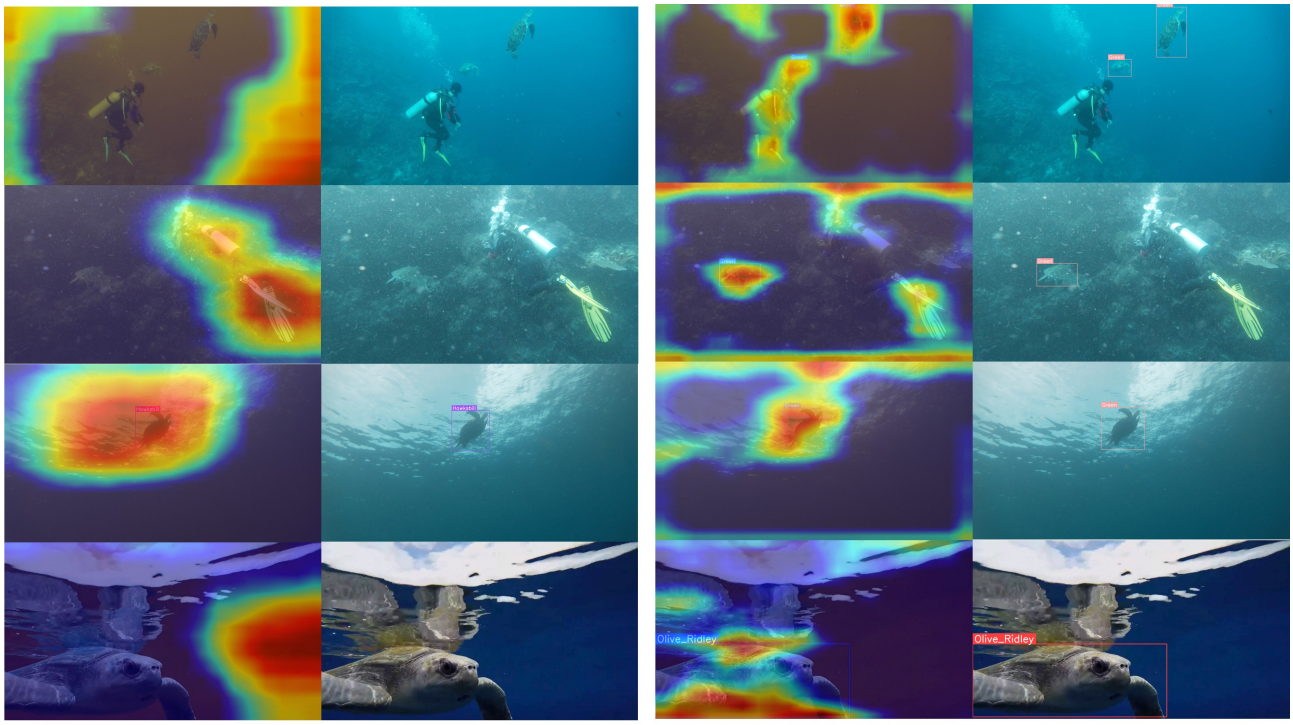
Table 7: Comparative analysis of the proposed model with other modules.

Since the dataset contains an imbalanced distribution of species, Table 8 presents the performance achieved across different classes. The Green turtle class, which has the largest representation, achieves the highest performance with a 0.961 mAP50 score. In contrast, the Olive Ridley class, with the smallest distribution, achieves the lowest performance at 0.746 mAP50, which is lower than the Hawksbill class, which reaches 0.860 mAP50. In figure 9 presents a comparative analysis of object detection performance between the original YOLOv10n model and the proposed model on the testing dataset. The evaluation focuses on detection accuracy, particularly in identifying marine debris and sea turtles within complex underwater environments. In Figure 9 (a), the original YOLOv10n model struggles to distinguish turtles from other objects. Moreover, image noise blurs their shapes, which leads to misclassifications and lower detection accuracy. As a result, the model struggles to separate turtles from surrounding objects, which limits its performance in real-world underwater environments. Conversely, Figure 9 (b), illustrates the enhanced performance of the proposed model.

Class	**mAP50**
Hawksbill	0.860
Olive Ridley	0.746
Green	0.961

Table 8: Comparative analysis of the proposed model against three classes.

It consistently demonstrates superior detection accuracy by providing more precise bounding boxes, particularly for small and partially visible objects. The proposed model exhibits improved resilience against background noise and varying object scales, resulting in fewer missed detections and more reliable object localization. These findings highlight the effectiveness of the proposed modifications in improving detection precision and robustness in underwater environments. These findings highlight the capability of the proposed model in accurately detecting turtle species within Bunaken National Park, particularly in challenging underwater envi-



(a) (b)

Figure 10: Comparison of heatmap visualizations: (a) YOLOv10-nano baseline and (b) proposed YOLOv10n with PFM module.

ronments with varying visibility and complex backgrounds. This improvement enhances its applicability for real-time marine biodiversity monitoring and conservation initiatives.

The heatmap visualizations presented in Figure 10 provide a comparative analysis of the attention distributions generated by the YOLOv10-nano model (a) and the proposed model (b) using the Eigen-CAM technique [26]. Eigen-CAM is a gradient-based class activation mapping approach that visualizes the internal representation of deep neural networks by highlighting the spatial regions that contribute most significantly to the model's decision-making process. Through these heatmaps, it becomes possible to interpret and qualitatively assess the discriminative focus of each model during underwater object detection. From the visual comparison, it is evident that the YOLOv10-nano model produces broader and more dispersed activation patterns across the entire image, including irrelevant background regions such as water texture and lighting variations. This behavior suggests that the YOLOv10-nano model struggles to effectively differentiate between foreground and background features, which may lead to reduced detection precision, particularly in complex underwater scenes characterized by turbidity, illumination distortion, and occlusion.

Model	**Params**	**GFLOPs**	**mAP50**
YOLOv10n	2,708,210	8.4	0.796
Proposed	**2,749,554**	**8.4**	**0.815**

Table 9: Comparative analysis of the proposed model with augmentation dataset.

In contrast, the proposed model demonstrates a more concentrated and semantically coherent attention distribution, with high-intensity activations localized primarily around the sea turtle's distinctive anatomical structures, including the shell, flippers, and head region. This focused response indicates that the proposed model has learned to capture the most informative and discriminative cues necessary for reliable object identification.



Figure 11: Augmentation visualization showing brightness and contrast adjustments used to enhance the model's adaptability to underwater conditions.

Furthermore, in scenarios with low visibility or partially occluded turtles, the heatmaps reveal that the proposed model maintains consistent attention around structural edges and texture variations, enabling it to distinguish the target object even when visual clarity is compromised.

In order to evaluate the model's robustness, augmentation with varying illumination conditions was applied to simulate realistic underwater environments. Specifically, pixel intensity values were adjusted using randomized brightness scaling factors ($\alpha \in [0.3, 1.2]$) and contrast shifting factors ($\beta \in [-15, 22]$). As shown in Figure 11, the augmentations introduced visual diversity in lighting and orientation, allowing the model to adapt more effectively to dynamic underwater conditions. As shown in Table 9, the proposed model achieved an mAP50 of 0.815 compared to 0.796 for the original YOLOv10n. This improvement indicates that the proposed architecture maintains detection accuracy even under degraded visual conditions, confirming its enhanced robustness to noise and distortion.

Parameters	**Setup**
CPU	AMD Ryzen 5600H
RAM	16 GB
Operating System	Ubuntu Linux

Table 10: Hardware specification on inferences.

5.4 Runtime Efficiency

The inference stage for all detection models, including the proposed module and the baseline YOLOv10 architecture, is conducted under an identical hardware environment to ensure fair comparison. All runtime performance evaluations utilize an AMD Ryzen 5600H CPU without GPU acceleration. The detailed hardware specifications for inference are included in Table 10, which summarizes the system components employed during the experimental evaluation phase. Runtime efficiency plays a crucial role in assessing the practicality of object detection models for real-time marine wildlife monitoring, particularly for applications such as autonomous underwater vehicles (AUVs) and edge computing-based conservation systems. This metric evaluates the balance between computational complexity and inference speed, commonly measured in latency and frames per second (FPS). Table 11 presents a comparative analysis of the runtime performance of various YOLO models, including the proposed model, in detecting sea turtle species in underwater environments.

Model	Gflops	Latency (ms)	FPS
YOLOv3-tiny	19.0	56.0	18.61
YOLOv5n	7.2	36.0	29.72
YOLOv8n	8.2	37.9	28.23
YOLOv8n [19]	8.0	39.6	26.23
YOLOv10n + CBAM	8.4	39.0	27.23
YOLOv10n + SE	8.4	37.9	28.13
YOLOv10n + ECA	8.4	39.2	27.35
YOLOv10n + ViT	9.7	45.2	13.33
YOLOv10n + DenseNet	8.4	37.5	17.10
YOLOv10n + Swin Transformer	15.3	40.5	15.60
YOLOv10m	63.6	151.5	6.73
YOLOv10n	8.4	37.6	28.33
Proposed	**8.4**	**38.0**	**28.00**

Table 11: Runtime efficiency comparison between the proposed model and benchmark networks.

Based on the results in Table 11, the proposed model maintains an optimal balance between computational efficiency and detection speed. With a computational cost of 8.4 GFLOPS, it achieves an inference latency of 38 ms and a frame rate of 28 FPS. While it performs slightly slower than YOLOv5n (36 ms latency, 29.72 FPS), it offers comparable performance to YOLOv10n (37.6 ms latency, 28.33 FPS) while integrating architectural enhancements to improve detection accuracy. The ViT-integrated variant shows higher computational cost (9.7 GFLOPS) and latency (45.2 ms) due to the self-attention operations in transformer architectures. While ViT captures global context effectively, its patch-based attention increases inference time. The DenseNet-based model (8.4 GFLOPS, 37.5 ms) improves feature reuse and gradient flow, offering efficient computation. The Swin Transformer variant (15.3 GFLOPS, 40.5 ms) enhances local and global feature extraction through shifted-window attention but adds extra complexity, lowering the frame rate to 15.6 FPS. Similarly, the enhanced YOLOv8n [19] model records a latency of 39.6 ms and a frame rate of 26.23 FPS, indicating a slight reduction in speed compared to YOLOv8n despite having a comparable computational cost of 8.0 GFLOPS. Overall, the proposed model achieves a better balance between accuracy and speed than transformer-based variants, while maintaining efficiency similar to lightweight CNN models like YOLOv5n and YOLOv10n.

6 Conclusions

This study proposed an advanced deep-learning framework for automated sea turtle species detection in Bunaken National Park, addressing key challenges in underwater environments such as variable lighting, occlusions, and complex marine backgrounds. By integrating a parallel fusion mechanism with a refined channel attention module, the proposed model significantly enhanced feature extraction and spatial awareness, leading to improved detection performance. The experimental evaluation demonstrated that the model achieved a mean average precision (mAP50) of 85.6%, outperforming conventional YOLOv10n-based models while maintaining computational efficiency with a latency of 38 ms and a frame rate of 28 FPS. The superior performance of the proposed model underscores its suitability for real-time marine conservation applications, particularly in automated monitoring of sea turtle populations. By accurately distinguishing between different species and effectively handling environmental complexities, this system provides a robust solution for marine biodiversity assessment. For future work, further optimizations may focus on deploying the model on edge computing platforms to support real-time field applications. Additionally, integrating the system with autonomous underwater vehicles (AUVs) and remote sensing technologies could expand its operational scope, enabling large-scale monitoring efforts. These advancements would contribute to a more efficient and scalable approach to sea turtle

conservation, reinforcing the role of artificial intelligence in ecological research and wildlife preservation.

Acknowledgments

This research is funded by Applied Research of UNSRAT Excellence (RTUU) Sam Ratulangi University 2025 with contract number 638/UN12.27/LT/2025.

References

[1] Olive Ridley Project, "How many sea turtles are left?", 2025. Available: https://oliveridleyproject.org/ufaqs/how-many-sea-turtles-are-left/ (Accessed: May 23, 2025).

[2] B. N. Park, *Peta Sebaran Jenis Penyu di Perairan Bunaken dan Siladen*, Balai Taman Nasional Bunaken, 2025. Available: https://tn-bunaken.com/2021/07/15/peta-sebaran-jenis-penyu-di-perairan-bunaken-dan-siladen/ (Accessed: May 23, 2025).

[3] A. J. Paul, "The Need and Status of Sea Turtle Conservation and Survey of Associated Computer Vision Advances," in *Proc. IEEE 8th UPCON*, 2021, pp. 1–8, doi: https://doi.org/10.1109/UPCON52273.2021.9667626.

[4] J. Escorcia-Gutierrez, M. Gamarra, E. Leal, N. Madera, C. Soto, R. F. Mansour, M. Alharbi, A. Alkhayyat, and D. Gupta, "Sea turtle foraging algorithm with hybrid deep learning-based intrusion detection for the internet of drones environment," *Computers and Electrical Engineering*, vol. 108, 2023, Art. no. 108704, doi: https://doi.org/10.1016/j.compeleceng.2023.108704.

[5] W. Songpan, T. Chaianunporn, K. Lomthaisong, and S. Keithmaleesatti, "Automated classification in turtles genus Malayemys using ensemble multiview image based on improved YOLOv8 with CNN," *Scientific Reports*, vol. 14, no. 1, pp. 24993, 2024, doi: https://doi.org/10.1038/s41598-024-76431-9.

[6] H.-Y. Lin, S.-L. Tseng, and J.-Y. Li, "SUR-Net: A Deep Network for Fish Detection and Segmentation With Limited Training Data," *IEEE Sensors Journal*, vol. 22, no. 18, pp. 18035–18044, 2022, doi: https://doi.org/10.1109/JSEN.2022.3195365.

[7] F. Han, J. Yao, H. Zhu, and C. Wang, "Underwater Image Processing and Object Detection Based on Deep CNN Method," *Journal of Sensors*, vol. 2020, no. 1, pp. 6707328, 2020, doi: https://doi.org/10.1155/2020/6707328.

[8] C. Liu, J. Wen, J. Huang, W. Lin, B. Wu, N. Xie, and T. Zou, "Lightweight Underwater Object Detection Algorithm for Embedded Deployment Using Higher-Order Information and Image Enhancement," *Journal of Marine Science and Engineering*, vol. 12, no. 3, Art. no. 506, 2024, doi: https://doi.org/10.3390/jmse12030506.

[9] Y. Jin, X. Xiao, Y. Pan, X. Zhou, K. Hu, H. Wang, and X. Zou, "A Novel Method for the Object Detection and Weight Prediction of Chinese Softshell Turtles Based on Computer Vision and Deep Learning," *Animals*, vol. 14, no. 9, Art. no. 1368, 2024, doi: https://doi.org/10.3390/ani14091368.

[10] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," in *Proc. IEEE CVPR*, 2016, pp. 779–788, doi: https://doi.org/10.1109/CVPR.2016.91.

[11] A. Wang, H. Chen, L. Liu, K. Chen, Z. Lin, J. Han, and G. Ding, "YOLOv10: Real-Time End-to-End Object Detection," arXiv preprint, arXiv:2405.14458, 2024. [Online]. Available: https://arxiv.org/abs/2405.14458.

[12] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon, "CBAM: Convolutional Block Attention Module," arXiv preprint, arXiv:1807.06521, 2018. [Online]. Available: https://arxiv.org/abs/1807.06521.

[13] S. Sriram, A. P., A. K. T. P., N. Vijayaraj, and T. Murugan, "Enhanced YOLOv10 Framework Featuring DPAM and DALSM for Real-Time Underwater Object Detection," *IEEE Access*, vol. 13, pp. 8691–8708, 2025, doi: https://doi.org/10.1109/ACCESS.2025.3527315.

[14] J. Zhong, M. Li, J. Qin, Y. Cui, K. Yang, and H. Zhang, "Real-Time Marine Animal Detection Using YOLO-Based Deep Learning Networks in the Coral Reef Ecosystem," *Int. Arch. Photogramm. Remote Sens. Spatial Inf. Sci.*, vol. XLVI-3/W1-2022, pp. 301–306, 2022, doi: https://doi.org/10.5194/isprs-archives-XLVI-3-W1-2022-301-2022.

[15] L. Zhao, Q. Yun, F. Yuan, X. Ren, J. Jin, and X. Zhu, "YOLOv7-CHS: An Emerging Model for Underwater Object Detection," *Journal of Marine Science and Engineering*, vol. 11, no. 10, Art. no. 1949, 2023, doi: https://doi.org/10.3390/jmse11101949.

[16] M. Zhang, Z. Wang, W. Song, D. Zhao, and H. Zhao, "Efficient Small-Object Detection in Underwater Images Using the Enhanced YOLOv8 Network," *Applied Sciences*, vol. 14, no. 3, Art. no. 1095, 2024, doi: https://doi.org/10.3390/app14031095.

[17] P. C. Gray, A. B. Fleishman, D. J. Klein, M. W. McKown, V. S. Bézy, K. J. Lohmann, and D. W. Johnston, "A convolutional neural network for detecting sea turtles in drone imagery," *Methods in Ecology and Evolution*, vol. 10, no. 3, pp. 345–355, 2019, doi: https://doi.org/10.1111/2041-210X.13132.

[18] M. D. Putro, Y. Mose, A. C. Andaria, J. Litouw, V. C. Poekoel, and X. B. N. Najoan, "Streamlining Deep Learning Network for Real-time Sea Turtle Detection," *Jurnal Rekayasa Elektrika*, 2024, doi: https://doi.org/10.17529/jre.v20i3.35236.

[19] M. D. Putro, D. G. S. Ruindungan, R. Syahputra, T.-H. Oh, I. Y. Chun, and V. C. Poekoel, "An Efficient and Effective Sea Turtle Detection Using Positioning Enhancement Module," in *Proc. 2024 Int. Workshop on Intelligent Systems (IWIS)*, pp. 1–6, 2024, doi: https://doi.org/10.1109/IWIS62722.2024.10706029.

[20] M. Zhang, S. Xu, W. Song, Q. He, and Q. Wei, "Lightweight Underwater Object Detection Based on YOLO v4 and Multi-Scale Attentional Feature Fusion," *Remote Sensing*, vol. 13, no. 22, Art. no. 4706, 2021, doi: https://doi.org/10.3390/rs13224706.

[21] J.-W. Baek, J.-I. Kim, and C.-B. Kim, "Deep learning-based image classification of turtles imported into Korea," *Scientific Reports*, vol. 13, no. 1, p. 21677, 2023, doi: https://doi.org/10.1038/s41598-023-49022-3.

[22] S. Zhao, J. Zheng, S. Sun, and L. Zhang, "An Improved YOLO Algorithm for Fast and Accurate Underwater Object Detection," *Symmetry*, vol. 14, no. 8, Art. no. 1669, 2022, doi: https://doi.org/10.3390/sym14081669.

[23] L. Chen, G. Li, S. Zhang, W. Mao, and M. Zhang, "YOLO-SAG: An improved wildlife object detection algorithm based on YOLOv8n," *Ecological Informatics*, vol. 83, Art. no. 102791, 2024, doi: https://doi.org/10.1016/j.ecoinf.2024.102791.

[24] L. Bellinati, S. Pesaro, F. Marcer, P. Danesi, A. Natale, and L. Ceglie, "Detection of a Novel Chlamydia Species in Invasive Turtles," *Animals*, vol. 12, no. 6, Art. no. 784, 2022, doi: https://doi.org/10.3390/ani12060784.

[25] Z. Hu, L. Cheng, S. Yu, P. Xu, P. Zhang, R. Tian, and J. Han, "Underwater Target Detection with High Accuracy and Speed Based on YOLOv10," *Journal of Marine Science and Engineering*, vol. 13, no. 1, Art. no. 135, 2025, doi: https://doi.org/10.3390/jmse13010135.

[26] M. Bany Muhammad and M. Yeasin, "Eigen-CAM: Class Activation Map using Principal Components," in *Proc. 2020 Int. Joint Conf. Neural Networks (IJCNN)*, pp. 1–7, 2020, doi: https://doi.org/10.1109/IJCNN48605.2020.9206626.

[27] R. Khanam and M. Hussain, "YOLOv11: An Overview of the Key Architectural Enhancements," *arXiv preprint*, arXiv:2410.17725, 2024. Available: https://arxiv.org/abs/2410.17725.

[28] Y. Tian, Q. Ye, and D. Doermann, "YOLOv12: Attention-Centric Real-Time Object Detectors," *arXiv preprint*, arXiv:2502.12524, 2025. Available: https://arxiv.org/abs/2502.12524.

[29] Q. Li and H. Shi, "YOLO-GE: An Attention Fusion Enhanced Underwater Object Detection Algorithm," *Journal of Marine Science and Engineering*, vol. 12, no. 10, Art. no. 1885, 2024, doi: https://doi.org/10.3390/jmse12101885.

[30] Q. Li, M. J. Er, L. Li, J. Chen, and J. Wu, "Underwater Object Detection Based on Improved SSD with Convolutional Block Attention," in *Proc. 2022 5th Int. Conf. on Intelligent Autonomous Systems (ICoIAS)*, pp. 37–42, 2022, doi: https://doi.org/10.1109/ICoIAS56028.2022.9931319.

[31] Y. Sahmoudi, O. El Ogri, A. Hjouji, and J. El-Mekkaoui, "New Biometric Recognition System By Multichannel Orthogonal Radial Triangular Pulse Moments," in *Proc. 2024 3rd International Conference on Embedded Systems and Artificial Intelligence (ESAI)*, pp. 1–11, 2024, doi: https://doi.org/10.1109/ESAI62891.2024.10913855.

[32] S. Ye, Y. Adamovsky, H. Chen, R. Bohush, and S. Ablameyko, "Real-Time Smoke Detection in Video Based on Two-Step Selection of Regions of Interest and Directional Movement Analysis," *Pattern Recognition and Image Analysis*, vol. 34, no. 4, pp. 1323–1332, 2024, doi: https://doi.org/10.1134/S1054661824701396.

[33] O. El Ogri, J. El-Mekkaoui, M. Benslimane, and A. Hjouji, "Automatic Lip-Reading Classification Using Deep Learning Approaches and Optimized Quaternion Meixner Moments by GWO Algorithm," *Knowledge-Based Systems*, vol. 304, Art. no. 112430, 2024, doi: https://doi.org/10.1016/j.knosys.2024.112430.

[34] O. El Ogri, M. Sayyouri, and H. Qjidaa, "Novel Fractional-Order Jacobi Moments and Invariant Moments for Pattern Recognition Applications," *Neural Computing and Applications*, vol. 33, no. 20, pp. 13539–13565, 2021, doi: https://doi.org/10.1007/s00521-021-06007-y.

[35] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, "Densely Connected Convolutional Networks," *arXiv preprint arXiv:1608.06993*, 2018, doi: https://doi.org/10.48550/arXiv.1608.06993.

[36] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An Image is Worth 16×16 Words: Transformers for Image Recognition at Scale," *arXiv preprint arXiv:2010.11929*, 2021, doi: https://doi.org/10.48550/arXiv.2010.11929.

[37] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin Transformer: Hierarchical Vision Transformer Using Shifted Windows," *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 9992–10002, 2021, doi: https://doi.org/10.1109/ICCV48922.2021.00986.

[38] O. El Ogri, H. Karmouni, M. Sayyouri, and H. Qjidaa, "3D Image Recognition Using New Set of Fractional-Order Legendre Moments and Deep Neural Networks," *Signal Processing: Image Communication*, vol. 98, Art. no. 116410, 2021, doi: https://doi.org/10.1016/j.image.2021.116410.

[39] Y. Zhao, X. Chen, and H. Zhang, "New Biometric Recognition System by Multichannel Orthogonal Radial Triangular Pulse Moments," *Proceedings of the 2024 3rd International Conference on Embedded Systems and Artificial Intelligence (ESAI)*, IEEE, pp. 1–6, 2024, doi: https://doi.org/10.1109/ESAI61559.2024.10523456.

[40] L. Adam, V. Čermák, K. Papafitsoros, and L. Picek, "SeaTurtleID2022: A Long-Span Dataset for Reliable Sea Turtle Re-Identification," *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 7131–7141, 2024, doi: https://doi.org/10.1109/WACV57701.2024.00699.