

A Fast and Efficient Ship Detector Via HDR-CoT Module

Jane Litouw ^{1*}, Marsel Marhaen Wungow ^{2*}, Dayen Manoppo ^{3*}, Ni Made S. Mustikayani ^{4*},
Hoang Van Thanh ^{5**}, Muhamad Dwisananto Putro ^{6*}

* Department of Electrical Engineering, Faculty of Engineering, Sam Ratulangi University Kampus Bahu, Manado 95112, Indonesia

** Faculty of Information Technology, Quang Binh University, Vietnam

jane.litouw@unsrat.ac.id ¹, marselwungow026@student.unsrat.ac.id ², dayenmanoppo026@student.unsrat.ac.id ³,
nimustikayani026@student.unsrat.ac.id ⁴, thanhhv@qbu.edu.vn ⁵, dwisanantoputro@unsrat.ac.id ⁶

Article Info

Article history:

Received 2026-08-18

Revised 2026-09-07

Accepted 2026-10-02

Keyword:

*Ship Detection,
Modified YOLO-FAST,
Efficient Model,
Deep Learning,
Contextual Transformer.*

ABSTRACT

Reliable vessel surveillance requires robust computer vision for accurate ship detection. Although YOLO-FAST is computationally efficient, its lightweight architecture has limited capability to extract discriminative features from small objects. To address this limitation, this study proposes an enhanced ship detection framework by integrating a novel Hadamard-Dilated Residual Contextual Transformer (HDR-CoT) module into the YOLO-FAST architecture. The proposed HDR-CoT module integrates Hadamard multiplication with parallel dilated depthwise convolutions. In addition, residual connections enhance feature representation while maintaining low computational overhead. Experimental results demonstrate the effectiveness of the proposed model. It achieves an mAP50 of 0.952 and an mAP50:95 of 0.689 while requiring only 1.73 million parameters and 6.3 GFLOPs. The proposed network achieves a favorable balance between detection performance and efficiency, improving mAP50:95 by 2.99% while reducing the parameters by 24.93%. It also achieves competitive performance compared with other lightweight architectures. CPU-based evaluation further shows that the model operates at 20.54 FPS. This configuration achieves an optimal trade-off between near-real-time inference speed and detection reliability compared with YOLO-FAST for resource-constrained maritime applications.



This is an open access article under the [CC-BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.

I. INTRODUCTION

The increasing number of ships operating in ports and urban waterways has increased the risk of ship collisions [1]. This challenge is further intensified by the diversity of vessel types, heavy maritime traffic, and the growing adoption of autonomous ships, which require more reliable monitoring systems. Conventional monitoring systems, such as the Automatic Identification System (AIS) and radar, have limitations in detecting small vessels or ships without active transponders [2], [3]. Therefore, image-based detection has emerged as an effective alternative for improving navigation safety and port operations [4]. However, these visual systems still face various challenges in complex maritime environments, such as wide variations in ship sizes, visual similarities among targets, and extreme weather conditions,

which make it difficult to balance accuracy, speed, and model efficiency [5]-[8].

Vessel detection is a key challenge in water monitoring systems, especially in dense, complex maritime environments. Visual detection systems face considerable difficulties when dealing with occluded vessels or with vessels of different sizes in the same image [9], [10]. Under such conditions, detection models tend to prioritize larger vessels, often leaving smaller vessels undetected, resulting in missed detections and false positives [11]-[13]. This problem indicates that current ship detection systems are not yet capable of working optimally in various dynamic field conditions. Therefore, there is an urgent need to develop a lighter, more adaptable, and more accurate visual detection model that can operate efficiently even on devices with limited specifications [14]-[17].

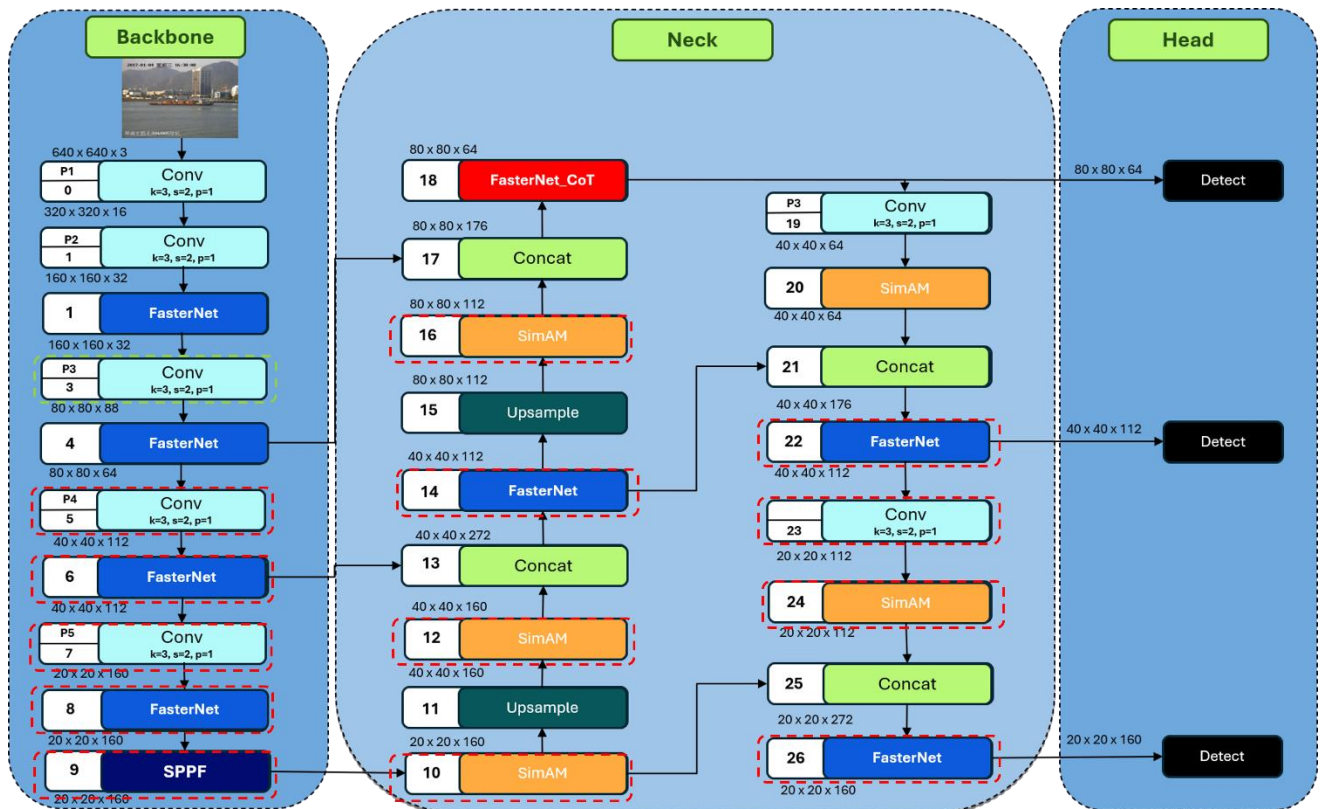


Figure 1. Overview of the proposed YOLO-FAST architecture. The red dashed boxes indicate layers where the number of channels is reduced, whereas the green dashed boxes indicate layers where the number of channels is increased.

To address the need for an adaptive, lightweight monitoring system, modern ship detection systems have shifted to deep learning. Convolutional Neural Networks (CNNs) enable automatic extraction of visual features from maritime images [18]. In computer vision, object detection methods are generally categorized into two-stage and one-stage detectors. Two-stage detectors, such as the widely known Faster R-CNN architecture, operate by first identifying potential object locations and then classifying them. This approach has proven capable of achieving very high accuracy, but unfortunately requires significant computational time.

To address this speed constraint, one-stage detectors introduce a major change by performing all stages simultaneously. Detection models such as YOLO (You Only Look Once) and SSD (Single Shot MultiBox Detector) can predict an object's location and class in a single computational step [19]. This characteristic enables single-stage detectors to operate very quickly and makes them the ideal choice for real-time monitoring system applications. In deep learning-based maritime monitoring, Convolutional Neural Networks (CNNs) serve as the primary foundation for automatically extracting visual features from images. Through hierarchical convolutional operations, CNNs can learn low-level patterns such as edges and textures, as well as higher-level semantic structures that are important for recognizing ships under complex visual conditions. This capability makes CNNs

highly suitable for ship detection tasks, where object appearance may vary significantly due to scale differences, occlusion, illumination changes, and background clutter in maritime scenes.

Building on the feature-extraction capabilities of CNNs, the YOLO (You Only Look Once) architecture was developed as a one-stage object detector that predicts object locations and class labels in a single forward pass. Unlike two-stage detectors, YOLO performs localization and classification simultaneously, resulting in significantly faster inference while maintaining competitive detection accuracy. Owing to this balance between speed and performance, YOLO has become one of the most widely adopted frameworks for real-time object detection applications [20], [21], [22]. In this study, the detection framework is based on YOLO-FAST [23], a lightweight one-stage detector that achieves an optimal balance between rapid inference speed (FPS) and low computational cost. By employing FasterNet [24] as its backbone, YOLO-FAST efficiently processes visual information with minimal parameter overhead. This efficiency makes the model highly suitable for maritime monitoring systems constrained by limited hardware resources, such as coastal surveillance devices and port platforms. Additionally, the SeaShips dataset is utilized during training to provide diverse ship categories and realistic maritime visual conditions. Although YOLO-FAST excels in fast inference and low computational demands, its extremely

lightweight architecture inherently limits its overall detection performance.

To address this performance limitation, integrating the Contextual Transformer (CoT) block [25] provides an effective solution. The CoT module fully leverages the contextual information among input keys to guide the learning of a dynamic attention matrix. This mechanism significantly enhances the capacity of visual representation, thereby improving detection accuracy, without substantially increasing the parameter and FLOP budgets. To enable effective integration into the YOLO-FAST architecture, the CoT mechanism requires structural adjustments to align with the detector's feature extraction characteristics. These modifications ensure efficient context mining while maintaining training stability and the model's fundamental computational efficiency.

Consequently, the proposed approach improves feature representation quality and enhances object detection accuracy across diverse marine environments. This study proposes a modified YOLO-FAST architecture enhanced through strategic channel tuning and the integration of a novel attention module. The proposed method improves small ship detection in complex maritime environments while maintaining high computational efficiency for real-time applications. The main contributions of this research are summarized as follows:

- 1) This study proposes a novel ship detection model based on the YOLO-FAST architecture. The proposed design separately optimizes computational efficiency and detection performance. A channel optimization strategy decreases the parameter count by 24.9% and reduces computational complexity for real-time operation. In addition, the proposed Hadamard-Dilated Residual Contextual Transformer (HDR-CoT) module is integrated into the C2f_faster block to improve feature representation and detection accuracy.
- 2) The proposed HDR-CoT module specifically enhances spatial representation and feature extraction capabilities. The HDR-CoT architecture improves upon the original Contextual Transformer design by incorporating a dilated depthwise convolution branch and residual connections. This architectural enhancement expands the receptive field and adaptively strengthens crucial spatial cues without adding significant computational overhead. Consequently, the module provides more reliable detection of small and distant vessels than the original architecture.
- 3) A comprehensive evaluation assesses the memory and computational efficiency of the proposed modifications. Extensive quantitative comparisons with several recent

lightweight YOLO architectures validate both detection accuracy and computational cost. These results confirm the model optimally balances rich feature extraction with lightweight processing, proving its feasibility for real-time deployment on low-cost hardware platforms.

II. METHODS

YOLO architectures lead real-time object detection by effectively balancing predictive accuracy and inference speed. Building on this foundation, You Only Look Once-FAST (YOLO-FAST) provides an optimized framework engineered for low-cost devices with limited computing capabilities. As depicted in Figure 1, the YOLO-FAST architecture comprises three synergistic components: the backbone, the neck, and the detection head.

Unlike conventional detectors that rely on computationally intensive networks, YOLO-FAST prioritizes operational efficiency by utilizing lightweight modules. This design allows the network to process spatial data in a single stage, seamlessly integrating region proposals and object classification into a unified, high-speed workflow. The architecture performs region proposal and object classification within a unified single-stage framework for efficient object detection. It employs FasterNet as the backbone to extract visual features while minimizing computational latency and maintaining strong feature representation [26]. Furthermore, the network incorporates the SimAM (Simple, Parameter-Free Attention Module) mechanism to enrich the extracted features. This module enables the system to highlight essential spatial details dynamically without incurring extra parameter overhead. Consequently, the architecture ensures optimal detection performance in resource-constrained environments.

In this study, channel optimization is introduced to reallocate the parameter budget and computational cost (GFLOPs), allowing the proposed enhancement modules to be incorporated without exceeding the computational complexity of the original YOLO-FAST architecture. As illustrated in Figure. 1, the red dashed boxes indicate layers where the number of channels is reduced, whereas the green dashed boxes indicate layers where the number of channels is increased. Specifically, the channel dimensions of the backbone layers at the middle and higher stages are reduced to decrease computational cost. Similarly, several layers in the neck also apply channel reduction to further improve computational efficiency. In contrast, the number of channels in the L3 feature map is increased to enrich feature representation at the middle stage, where important semantic information for object detection is preserved.

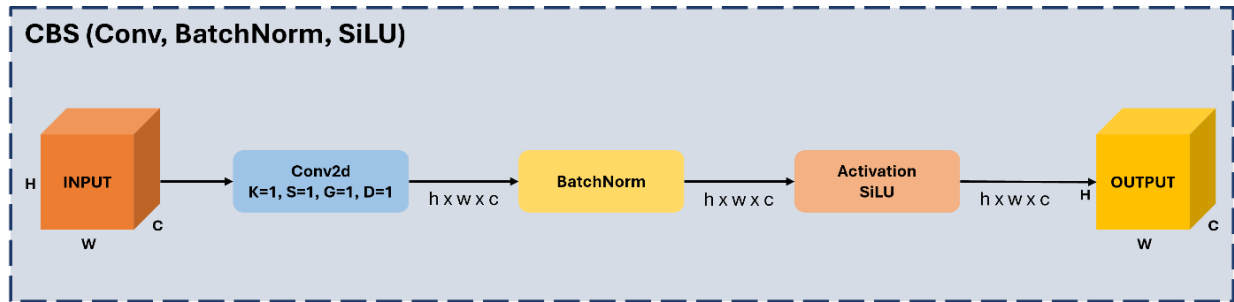


Figure 2. CBS (Convolution, Batch Normalization, Silu Activation).

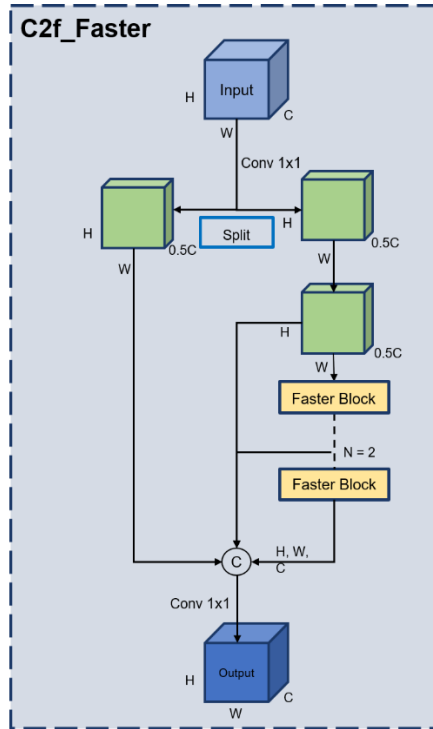


Figure 3. C2f_Faster.

This channel optimization strategy reduces feature redundancy in layers that mainly contain background information while allocating more channels to layers with richer object information. As a result, computational resources are utilized more efficiently while maintaining the detection performance of the original YOLO-FAST model without increasing its computational complexity. This allows the proposed enhancement modules to be incorporated.

A. Backbone

In the YOLO-FAST architecture, the backbone serves as the main feature-extraction network, transforming input images into hierarchical visual representations for object detection. Its primary role is to capture discriminative information from the image and progressively refine it from low-level visual patterns to higher-level semantic features that are more relevant for recognizing and localizing ships.

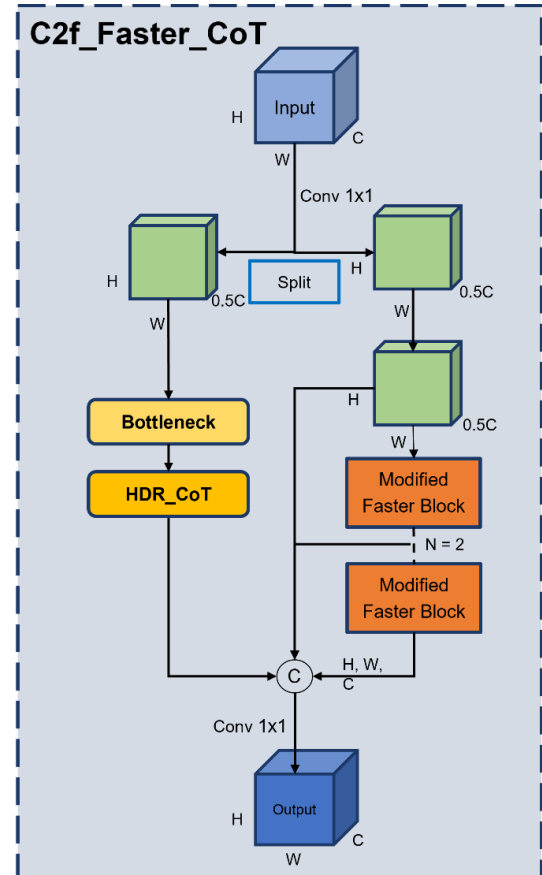


Figure 4. C2f_Faster_CoT.

In maritime scenes, where object scales vary widely and visual conditions are often complex, the backbone must preserve essential spatial information while maintaining efficient computation. Therefore, the backbone in this study is designed to gradually abstract features through a staged extraction process, enabling the network to generate informative feature maps that are robust to scale variation, background complexity, and dense object arrangements. This design is particularly important for ship detection, as the model must accurately identify both small and large vessels while remaining computationally efficient for real-time or resource-constrained deployment.

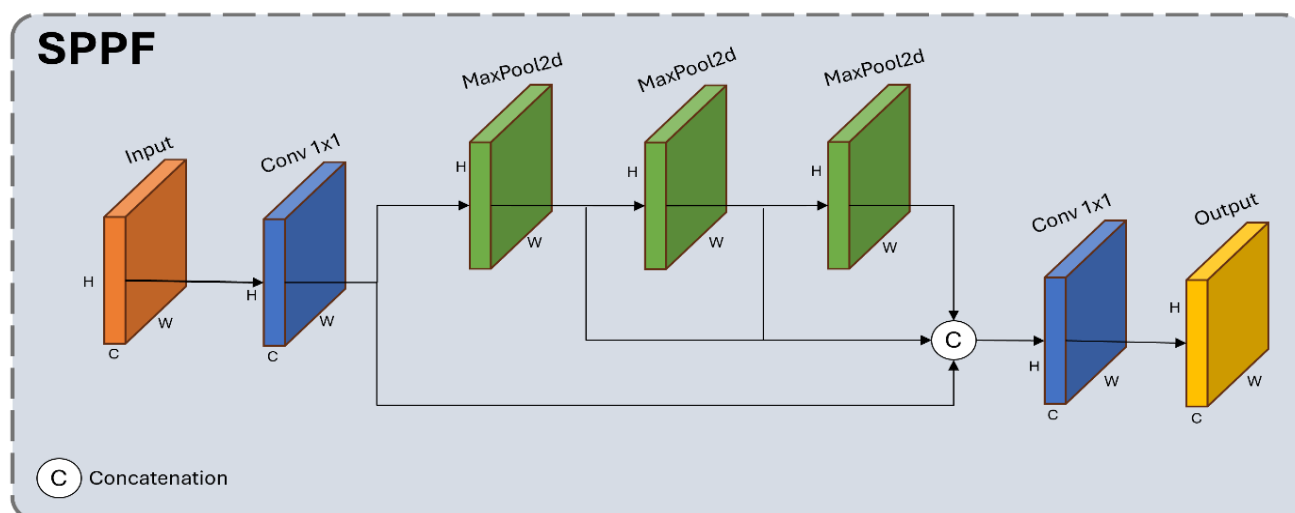


Figure 5. SPPF (Spatial Pyramid Pooling-Fast).

Overall, the backbone is intended to provide a balanced foundation for the detection framework by supporting strong feature representation, stable information flow, and efficient processing in challenging maritime environments. Based on Figure 2, the CBS module is a fundamental feature-extraction block that consists of three sequential operations: Convolution, Batch Normalization, and SiLU activation. In this block, the convolution layer captures local visual patterns from the input image, such as edges, textures, shapes, and early object structures. Batch normalization is then applied to stabilize the distribution of intermediate activations during training, thereby improving optimization stability, accelerating convergence, and reducing sensitivity to parameter initialization. Subsequently, the SiLU activation function introduces a smooth nonlinear transformation, enabling the network to learn richer and more discriminative feature representations.

Within the backbone, CBS serves as a basic feature-extraction unit that supports effective spatial information modeling while maintaining training stability and computational efficiency. Therefore, CBS plays an important role in constructing informative feature maps, particularly in object detection tasks that require a balance between representational quality and computational cost. In ship detection, this role becomes increasingly important because the model must preserve visual cues from objects with diverse scales, shapes, and background conditions.

C2f_Faster is a feature extraction module utilized within the backbone and neck architectures as illustrated in Figure 3. The module initially applies a standard convolution with a 1×1 kernel and a stride of 1 to the input. A split operation then divides these input channels into two halves. The first half is processed through a bottleneck block, while the second half is routed through a bypass branch, serving as an identity mapping. The network subsequently concatenates these two halves. Following concatenation, a final 1×1 standard

convolution with a stride of 1 projects the features to the desired channel depth.

Departing from the standard C2f module in YOLOv8, C2f_Faster incorporates a Faster block instead of the traditional Bottleneck block. This architectural modification minimizes both parameter count and computational complexity, directly addressing the high resource demands of the original design. The Faster block achieves this efficiency by utilizing a Partial Convolution, known as PConv. PConv applies standard convolution to only a subset of the channels while leaving the remaining channels unprocessed for the residual connection. Consequently, this updated architecture requires significantly fewer parameters and computational resources than the original Bottleneck configuration.

Furthermore, C2f_Faster_CoT is an advanced feature extraction module utilized within the backbone and neck architectures as shown in Figure 4. The module initially applies a standard convolution with a 1×1 kernel and a stride of 1 to the input. A split operation then divides these input channels into two halves. The first half passes through a primary Faster block. Unlike standard architectures, where the second half routes through a bypass branch as a simple identity mapping, this modified architecture routes the second half through a bottleneck block to filter essential information. Subsequently, these filtered features pass through a Contextual Transformer, known as CoT. This integration actively captures global contextual information and discriminates against irrelevant features before fusion. The network then concatenates the outputs from both branches. Following concatenation, a 1×1 standard convolution with a stride of 1 projects the combined features to the desired channel depth. Building upon the base C2f_Faster design, this architecture maintains computational efficiency through a Partial Convolution, or PConv, within the primary Faster block.

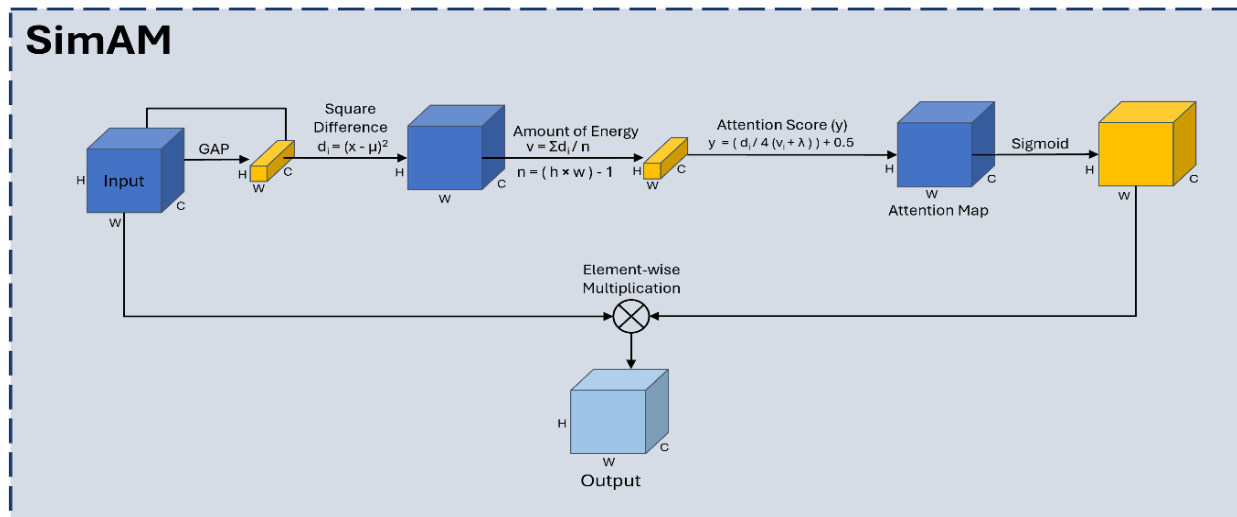


Figure 6. SimAM (Simple-Attention Module).

However, a primary distinction lies in the channel processing ratio. While the standard C2f_Faster module processes only 25% of the channels through PConv and leaves 75% for the residual connection, the C2f_Faster_CoT module significantly increases this proportion. Specifically, PConv within this advanced module applies standard convolution to 85% of the channels while leaving the remaining 15% unprocessed for the residual connection. By pairing this intensive feature extraction in the primary branch with the robust context-capturing capabilities of the CoT module in the secondary branch, the architecture achieves a superior balance between minimal resource demands and high feature extraction accuracy.

Subsequently, SPPF (Spatial Pyramid Pooling-Fast) [27] is a feature aggregation module that enlarges the receptive field and enriches multi-scale contextual information in feature maps as shown in Figure 5. Conceptually, it follows the spatial pyramid pooling approach, combining information across different spatial scales so the network can capture not only local details but also broader contextual cues. This mechanism is important in object detection because objects may appear at different scales and in complex backgrounds, requiring feature representations that preserve both local and global information. Unlike conventional SPP, which employs multiple parallel pooling operations with different kernel sizes, SPPF builds the multi-scale effect more efficiently through sequential pooling operations. This design produces an output equivalent to pyramid pooling with kernel sizes of 5, 9, and 13, while reducing computational overhead and improving processing speed. As a result, SPPF enhances efficiency without sacrificing the quality of feature representation required for object detection.

In this study, SPPF is employed to enhance feature representation at the final stage of feature extraction, thereby preserving important information from objects of varying scales before they are forwarded to subsequent fusion and prediction stages. In ship detection, this role is particularly

important because maritime images often contain both small and large vessels within the same frame, along with complex disturbances such as waves, light reflections, haze, and dense object arrangements. By efficiently capturing multi-scale context, SPPF helps the model generate more informative and adaptive feature maps for challenging maritime scenes.

B. Neck

The neck serves as an intermediate feature-fusion component that connects the backbone to the detection layer. Its primary role is to integrate feature maps across different spatial resolutions, preserving fine-grained local details alongside high-level semantic information. This stage is crucial in object detection because varying target scales render single-level features insufficient for effective representation. Consequently, the neck aggregates shallow and deep features through repeated interactions, enabling the detector to construct highly informative multi-scale representations before prediction. Within the proposed model, the neck executes multi-scale feature fusion via upsampling, feature concatenation, and refinement. This architecture seamlessly propagates high-level semantic features to higher-resolution representations while retaining crucial lower-level spatial data. Simultaneously, the fusion process injects deep contextual information into lower-resolution features, thereby enhancing the detection of medium and large targets. This bidirectional aggregation mechanism proves highly relevant for ship detection, as vessels frequently exhibit substantial scale variations against visually complex maritime backgrounds. To address the original model's limitations in detecting small objects, this research introduces a targeted architectural modification.

Specifically, the design replaces the standard C2f_Faster module at layer 18 with the advanced C2f_Faster_CoT module. Layer 18 plays a critical role in processing high-resolution feature maps essential for identifying minute targets. By integrating the Contextual Transformer at this

specific depth, the network gains a stronger ability to capture global context and separate essential target features from irrelevant background noise. Consequently, this strategic placement effectively overcomes the baseline model's weakness, significantly improving small ship detection while preserving overall computational efficiency.

SimAM is an advanced attention mechanism designed to enhance feature representation by emphasizing informative neurons and suppressing less relevant responses as illustrated in Figure 6. Unlike conventional attention modules [28] that introduce additional learnable parameters, SimAM evaluates neuron importance through an energy-based formulation. This unique approach generates precise attention weights without increasing model complexity. Consequently, SimAM is highly suitable for lightweight detection frameworks that require improved feature discrimination without incurring substantial computational overhead.

In the proposed architecture, SimAM refines feature maps during the fusion stage to help the network focus on salient visual cues. The module adaptively recalibrates neurons to emphasize key target features while filtering out background interference. This capability is vital for analyzing maritime images, where visual disturbances such as waves, reflections, haze, and cluttered environments frequently disrupt object recognition. Through this attention-guided refinement, SimAM significantly elevates the quality of fused features, ensuring robust ship detection across diverse visual conditions.

C. Detection Layer and Loss Function

The detection layer is responsible for transforming the multi-scale feature maps produced by the preceding network into final object predictions. In this study, the detection process is performed at three feature levels corresponding to small, medium, and large-scale targets, allowing the model to better handle vessels with substantial scale variation. The detection head follows a decoupled design, with bounding box regression and category classification learned via separate prediction branches. This design enables the model to optimize localization and semantic discrimination more effectively, while preserving the efficiency required for real-time detection.

In addition, the head uses a distribution-based bounding-box representation rather than directly regressing box coordinates in a single step. This strategy allows the model to produce more refined localization outputs before they are decoded into final bounding boxes. During inference, predictions across all detection scales are aggregated to produce final box coordinates and class probabilities, enabling unified multi-scale detection in complex maritime scenes.

The optimization objective in this study is a complete detection loss comprising box loss, classification loss, and Distribution Focal Loss (DFL). The classification term is computed using Binary Cross-Entropy with Logits Loss, which supervises the predicted class scores after positive sample assignment. For localization, the model uses a bounding box loss that combines an IoU-based regression term with DFL, enabling the detector to learn both accurate box overlaps and fine-grained boundary localization. In the regression process, DFL models each side of the bounding box as a discrete probability distribution, which improves localization precision compared with direct scalar regression. Furthermore, positive samples are assigned using a Task-Aligned Assigner that matches predictions and ground truths by jointly considering classification confidence and localization quality. The final training objective is obtained by weighting the three components according to the predefined box, classification, and DFL gain coefficients.

D. HDR-CoT Module

In order to optimize the standard Contextual Transformer (CoT) for complex maritime environments, this study proposes the Hadamard-Dilated Residual Contextual Transformer (HDR-CoT) module. The proposed block introduces explicit structural and mathematical modifications to boost feature extraction, specifically targeting the challenges of localizing small vessels and distinguishing visually similar ship categories. As illustrated in Figure 7, the complete process from input to output can be described sequentially through convolutional and attention operation. HDR-CoT introduces two complementary mechanisms. Hadamard multiplication enables direct interaction between the original and contextual features, selectively emphasizing consistent and discriminative responses while suppressing irrelevant activations. Meanwhile, the parallel dilated depthwise convolution expands the receptive field to capture broader spatial context with minimal computational overhead. By combining these mechanisms, HDR-CoT simultaneously improves feature selectivity and contextual representation while maintaining suitability for lightweight object detection.

To further clarify the HDR-CoT architecture, the module is applied at the 18th neck layer with an input of 32 channels. The input X generates Key K through a 3×3 convolution, while Query Q is obtained directly from X . Q and K are combined using a Hadamard product, followed by a 1×1 convolution to generate dynamic attention weights W . Value V is also generated using a 1×1 convolution and processed with W through Local Multiplication. The resulting feature is concatenated with K and refined by the SE block. In parallel, X passes through a depthwise convolution with dilation rate 2 and a 1×1 convolution. Finally, the two branches are fused through element-wise addition as a residual connection to produce Y out.

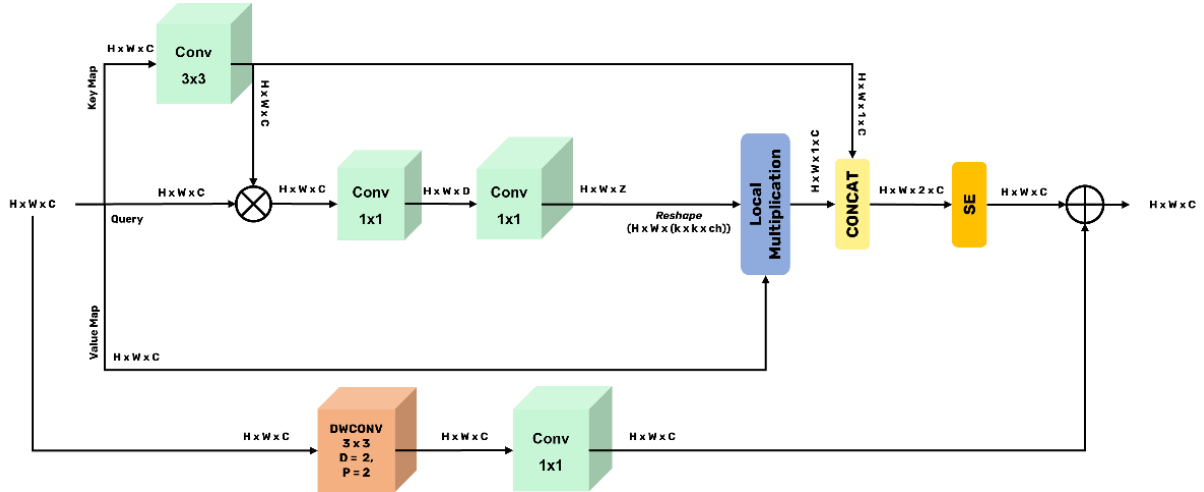


Figure 7. HDR-CoT Module.

The process begins by defining the input feature map as X . In the first stage, the Key representation (K) is extracted. The input X is passed through a 3×3 convolution operation equipped with Batch Normalization and an activation function, collectively denoted as F_{key} . Meanwhile, the Query representation (Q) implicitly remains as the input X . This extraction is formulated as:

$$K = F_{key}(X). \quad (1)$$

To address the difficulty in distinguishing similar vessels, the proposed module replaces the standard concatenation operation between the Query (X) and Key (K) with a Hadamard product. This explicit gating mechanism is visually denoted by $\otimes$ the operation:

$$M = X \otimes K. \quad (2)$$

By element-wise multiplying these features, the network adaptively highlights discriminative visual cues essential for class differentiation while suppressing irrelevant background noise. This process avoids the computational overhead and parameter inflation typically caused by standard concatenation. The resulting gated representation M is then passed through an embedding function (F_{embed}), consisting of sequential 1×1 convolution, to generate dynamic attention weights (W):

$$W = F_{embed}(M). \quad (3)$$

Concurrently, a Value representation (V) is generated by a 1×1 convolution (F_{val}) to the input X . The embedding function reshapes the generated attention weights (W) into a set of spatially varying local kernels with dimensions $H \times W \times (k \times k \times ch)$, enabling both information to participate in the subsequent local multiplication (LM) operation. During this operation, the dynamic local kernels are applied to the

Value representation through spatial convolution to produce the contextualized local feature:

$$Y_{local} = LM(V, W) \quad (4)$$

where,

$$V = F_{val}(X). \quad (5)$$

To enrich this representation, the architecture fuses Y_{local} and the initial Key K . Following the visual flow, these two representations are first combined using a concatenation operation (*Concat*) along the channel dimension. The concatenated feature map is then processed by a Squeeze-and-Excitation (SE) block (F_{SE}) to compute global channel-wise attention weights and perform a weighted sum:

$$Y_{concat} = Concat(Y_{local}, K), \quad (6)$$

$$Y_{SE} = F_{SE}(Y_{concat}). \quad (7)$$

This formulation ensures a focused and robust channel-wise feature representation. To improve small object detection, the architecture introduces a parallel spatial context branch. This branch extracts features directly from the input X through a sequential process:

$$Y_{context} = F_{1 \times 1}(DWConv_{d=2}(X)). \quad (8)$$

Following the mathematical order of operations, the input X is first processed by a depthwise convolution with a dilation rate of 2 ($DWConv_{d=2}$). The expanded spatial features are subsequently passed into a 1×1 convolution ($F_{1 \times 1}$) for precise channel alignment.

TABLE I
DISTRIBUTION OF INSTANCES ACROSS VESSEL CATEGORIES

Subset	Ore Carrier	Passenger Ship	Container Ship	Bulk Cargo Carrier	General Cargo Ship	Fishing Boat
Train	542	100	245	489	393	510
Val	586	126	219	468	368	492
Test	1071	248	437	995	744	1188

TABLE II
DISTRIBUTION OF OBJECT SIZES ACROSS DATASET SPLITS

Subset	Total Objects	Small Objects	Medium Objects	Large Objects
Train	2279	20	174	2085
Val	2259	22	187	2050
Test	4683	56	412	4215

This expansion enables the network to capture broader contextual cues, such as surrounding waves or ship wakes, which are critical for localizing exceedingly small, distant vessels. Finally, a terminal residual connection merges the spatial features from the dilated branch ($Y_{context}$) with the channel-wise attention output (Y_{SE}):

$$Y_{out} = Y_{SE} \oplus Y_{context}. \quad (9)$$

The element-wise addition denotes as $\oplus$. By fusing broad spatial context with focused channel attention, this residual connection preserves the fine-grained details of small targets and prevents vanishing gradients during deep network training. Consequently, the HDR-CoT module optimizes feature representation to deliver superior accuracy across diverse marine environments.

E. Dataset

This study utilizes the SeaShips dataset to train and evaluate the proposed model. The dataset consists of 7,000 high-resolution images with a resolution of 1920×1080 pixels and provides bounding box annotations for ship detection. It encompasses six vessel classes: ore carriers, bulk cargo carriers, general cargo ships, container ships, fishing boats, and passenger ships. The distribution of images across the six classes is presented in Table I. Among all vessel classes, Fishing Boat has the largest number of instances, with the highest count observed in the test subset (1,188 instances), whereas Passenger Ship has the smallest number of instances, with the lowest count observed in the training subset (100 instances).

To provide further insight into the detection characteristics, the object size distribution based on bounding box dimensions is presented in Table II. The table shows that the dataset predominantly contains large ship objects with 8,350 instances. In contrast, small objects are less frequent with 98 instances. It presents additional detection challenges, particularly when vessels occupy relatively small regions of the image.

TABLE III
TRAINING AND TESTING CONFIGURATION

Parameters	Setup
Device	Kaggle
GPU	P100
Image Size	640 x 640 pixels
Epoch	200
Batch Size	16
Optimizer	Stochastic Gradient Descent (SGD)
Learning Rate	0,01
Augmentation	Mosaic, Albumentations, MixUp, HSV, Affine Transformations, Flipping
Scheduler	Cosine Annealing Scheduler

TABLE IV
INFERENCE CONFIGURATION

Parameters	Setup
Operating System	Ubuntu
Compiler	Python 3.11.13
Network Construction Method	Pytorch 2.6.0
CPU	Intel Xeon 2.00 GHz 2-core

To facilitate the experiments, this research divides the dataset into three subsets: 25% for training (1,750 images), 25% for validation (1,750 images), and 50% for testing (3,500 images), following the data partitioning protocol proposed in [29].

F. Training and Testing Configuration

In this study, the proposed YOLO-FAST-based model was trained and evaluated using a fixed experimental configuration. Table III presents the training and testing configuration used in this study. The model was trained for 200 epochs with a batch size of 16 using SGD with an initial learning rate of 0.01 and a Cosine Annealing Scheduler. Data augmentation included Mosaic, Albumentations, MixUp, HSV, affine transformations, and flipping. Training was conducted on an NVIDIA Tesla P100 GPU through Kaggle with an input size of 640×640 pixels, while the inference environment used Ubuntu, Python 3.11.13, and PyTorch 2.6.0. Overall, this configuration was designed to support stable convergence and effective feature learning for ship detection under complex maritime conditions.

For inference, the model was tested on a CPU to assess its implementation feasibility with limited computational resources. As presented in Table IV, the inference process was conducted on an Ubuntu operating system using Python 3.11.13 and PyTorch 2.6.0, with an Intel Xeon 2.00 GHz 2-core processor. The input resolution was fixed at 640×640 pixels. In addition, the reported FPS measures the model inference time only and does not include preprocessing or postprocessing time.

TABLE V
EVALUATION PERFORMANCE MODEL WITH YOLO SERIES

Model	Parameter	mAP50	mAP50:95
YOLOv5n	2,509,634	0.958	0.700
YOLOv6n	4,238,738	0.959	0.718
YOLOv8n	3,012,018	0.963	0.725
YOLOv10n	2,709,380	0.938	0.699
YOLO11n	2,591,010	0.960	0.705
YOLOv12n[33]	2,539,266	0.959	0.699
YOLO-FAST	2,306,818	0.946	0.669
YOLO-FAST-HDR-CoT (Proposed)	1,732,006	0.952	0.689



Figure 8. Confusion Matrix YOLO-FAST-HDR-CoT.

This inference setting was intended to reflect a more practical deployment scenario in which GPU acceleration may not always be available. Therefore, the evaluation not only measures detection capability but also indicates the model's computational efficiency and suitability for real-world maritime monitoring systems with constrained hardware resources.

III. RESULTS AND DISCUSSION

A. Evaluation on Dataset

This comprehensive evaluation compares recent nano-series YOLO architectures to assess their feasibility for implementation on low-cost devices. The original YOLO-FAST baseline was evaluated against several models, including YOLOv5n [30], YOLOv6n, YOLOv8n [31], YOLOv10n [32], YOLO11n [33], [34] and YOLOv12n [35]. Two primary criteria determine model performance: detection accuracy, measured by mean average precision (mAP), and model efficiency, evaluated by the number of parameters, GFLOPs and FPS. Table V details the quantitative comparison of these metrics.

TABLE VI
PERFORMANCE EVALUATION USING PRECISION, RECALL, AND F1-SCORE ON THE SEASHIPS DATASET

Model	Precision	Recall	F1-Score
YOLO-FAST	0.917	0.899	0.908
YOLO-FAST-HDR-CoT (Proposed)	0.934	0.893	0.913

TABLE VII
COMPARISON OF AP50:95 PERFORMANCE ACROSS ALL OBJECT CLASSES

Model	AP50:95 each class					
	Ore Carrier	Passenger Ship	Container Ship	Bulk Cargo Carrier	General Cargo Ship	Fishing Boat
YOLO-FAST	0.641	0.607	0.792	0.653	0.714	0.610
YOLO-FAST-HDR-CoT (Proposed)	0.664	0.648	0.788	0.687	0.709	0.636

The proposed modified YOLO-FAST architecture is highly efficient, achieving an mAP50:95 score of 0.689 and an mAP50 score of 0.952. In contrast, the original YOLO-FAST baseline achieved 0.669 for mAP50:95 and 0.946 for mAP50. This difference represents a notable improvement of 0.020 in the mAP50:95 score. This accuracy enhancement is particularly significant considering the substantial reduction in model size. The proposed design reduces the number of parameters by approximately 24.9%, dropping from 2,306,818 to 1,732,006.

Although the proposed model achieves a high mAP50 of 0.952, its mAP50:95 is lower at 0.689. This shows that the model can detect ships accurately at an IoU threshold of 0.5, but the bounding box accuracy decreases when stricter IoU thresholds are applied. This indicates that while the proposed model can reliably identify ship objects, the predicted bounding boxes are not always highly precise at stricter thresholds. This behavior may be related to the challenging characteristics of the SeaShips dataset, where ships vary substantially in size and may appear as small or distant objects under complex maritime conditions. Nevertheless, the proposed model still improves the mAP50:95 from 0.669 to 0.689 compared with the original YOLO-FAST, indicating that the integration of HDR-CoT provides more accurate localization while maintaining the lightweight characteristics of the model.

To provide a more comprehensive evaluation, Precision, Recall, and F1-score were analyzed in addition to mAP. As shown in Table VI, the proposed model improved Precision by 1.85% and F1-score by 0.55% compared with YOLO-FAST. However, Recall slightly decreased by 0.67%. It indicates that the proposed model reduces false-positive detections while maintaining a comparable detection capability. Subsequently, Table VII presents an evaluation of AP50:95 for each class to assess the proposed model's impact. The largest improvement was observed for Passenger Ship at 6.75%, while the smallest improvement was 0.82% for Ore Carrier. In contrast, the AP50:95 for Container Ship slightly decreased by 0.51%.

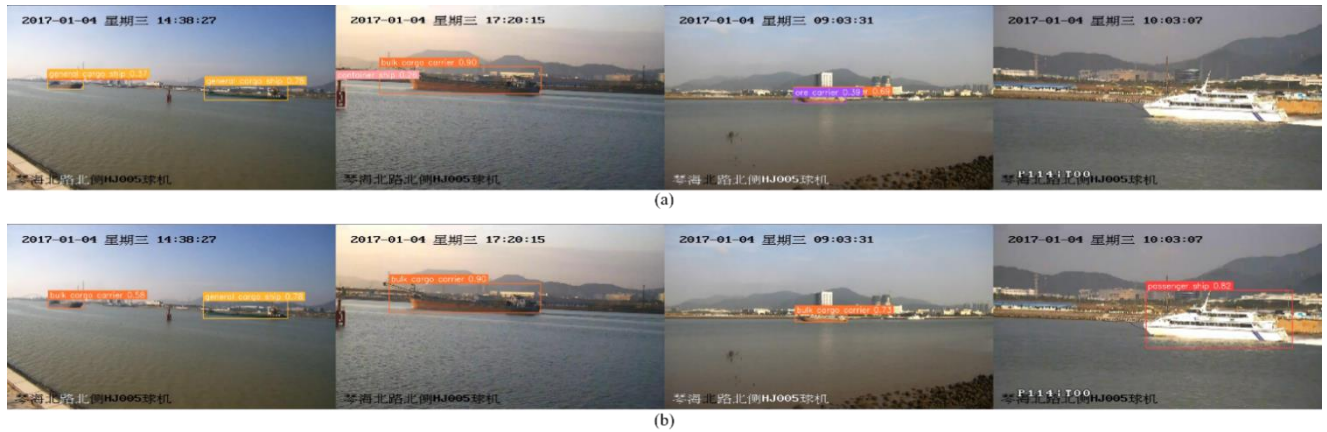


Figure 9. Picture Comparison Model (a) Original YOLO-FAST (b) YOLO-FAST-HDR-CoT.

TABLE VIII
CONSISTENCY EVALUATION ACROSS THREE EXPERIMENTAL RUNS

Model	1st Experiment		2nd Experiment		3rd Experiment		Mean		Standard deviation	
	mAP50	mAP50:95	mAP50	mAP50:95	mAP50	mAP50:95	mAP50	mAP50:95	mAP50	mAP50:95
YOLO-FAST	0.946	0.669	0.948	0.671	0.948	0.671	0.9467	0.6703	0.0012	0.0012
YOLO-FAST-channel optimization	0.945	0.665	0.946	0.667	0.945	0.665	0.9453	0.6657	0.0006	0.0012
YOLO-FAST-HDR-CoT (Proposed)	0.954	0.685	0.952	0.689	0.954	0.685	0.953	0.686	0.0012	0.0023



Figure 10. Examples of failure cases of the proposed YOLO-FAST-HDR-CoT model.

Moreover, compared with YOLO FAST and other lightweight YOLO architectures, the proposed network provides a favorable balance between accuracy and complexity. It improves mAP50:95 by 2.99% over YOLO-FAST while reducing the parameter count by 24.93%, from 2.31M to 1.73M. Although some lightweight architectures achieve higher mAP50:95, they require substantially more parameters, such as YOLOv8n (3.01M parameters) and YOLOv6n (4.24M parameters). These results demonstrate that HDR-CoT improves detection accuracy and efficiency.

The normalized confusion matrix, as illustrated in Figure 8, demonstrates that the proposed model achieves strong true positive rates across various ship categories in the test dataset. Specifically, the model scores 0.93 for ore carriers, 0.83 for

passenger ships, 0.96 for container ships, 0.92 for bulk cargo carriers, 0.92 for general cargo ships, and 0.89 for fishing boats. These results confirm that the model accurately identifies and differentiates between large and small vessels. However, the quantitative analysis also reveals specific limitations. The passenger ship category shows the lowest performance, as the model occasionally misclassifies these vessels as fishing boats at a rate of 0.09 or as background at a rate of 0.06. Fishing boats exhibit a similar trend, occasionally being mistaken for bulk cargo carriers or the background. These misclassifications suggest that feature extraction for differentiating these specific vessel classes requires further refinement.

A comparative visual analysis highlights the superior performance of the proposed architecture in Figure 9b compared to the original YOLO-FAST baseline in Figure 9a. The original model exhibits several limitations across the test scenarios, including misclassifications, false positives, redundant bounding boxes, and missed detections. Specifically, the baseline model misclassifies a bulk cargo carrier in the first scenario and falsely detects a container ship in the second. Additionally, it generates overlapping bounding boxes in the third scenario and completely misses the passenger ship in the fourth.

TABLE IX
ABLATION STUDY OF THE PROPOSED MODIFICATIONS ON THE SEASHIPS DATASET

Model	Parameter	mAP50	mAP50:95
YOLO-FAST	2,306,818	0.946	0.669
YOLO-FAST-channel optimization	1,536,258	0.946	0.667
YOLO-FAST-HDR-CoT (Proposed)	1,732,006	0.952	0.689

TABLE X
ABLATION STUDY OF THE PROPOSED HDR-CoT MODULE ON THE SEASHIPS DATASET

Model	Parameter	mAP50	mAP50:95
YOLO-FAST	2,306,818	0.946	0.669
Proposed w/o Residual	1,732,006	0.954	0.954
Proposed w/o Dilated Convolution	1,732,006	0.685	0.685
Proposed w/o Hadamard Product	1,732,518	0.954	0.954
YOLO-FAST-HDR-CoT (Proposed)	1,732,006	0.952	0.689

The proposed model successfully addresses these limitations by correctly classifying vessels, suppressing false predictions, and accurately detecting the previously unobserved passenger ship. These visual results confirm that the integrated Contextual Transformer module significantly enhances spatial feature representation and detection reliability. To assess the consistency of the proposed model, three independent training runs were conducted. Table VIII summarizes the experimental results in terms of the mean and standard deviation, providing a statistical evaluation of the training consistency. Although the proposed model achieves good overall detection performance, several failure cases are still observed. Figure 10 presents examples of detection failures, including small ships that are not detected, partially occluded ships, and closely positioned ships. Small ships are difficult to detect because their visual features occupy only a small number of pixels. Partial occlusion can also hide important features of the ship, making accurate detection more difficult. In addition, when several ships are located close to each other, their visual features and bounding boxes may overlap, which can lead to missed or inaccurate detections.

B. Ablation Study

This ablation study evaluates the individual contributions of each architectural modification using the SeaShips dataset. Table IX details the stepwise impact of strategic channel tuning and HDR-CoT module integration on both computational efficiency and detection accuracy. The initial evaluation focused on the YOLO-FAST baseline, which achieved an mAP50 of 0.946 and an mAP50:95 of 0.669, utilizing over 2.3 million parameters and 6.4 GFLOPs.

TABLE XI
EVALUATION OF MODEL PERFORMANCE ACROSS DIFFERENT BOUNDING BOX SIZES

Model	Large		Medium		Small	
	mAP50	mAP50:95	mAP50	mAP50:95	mAP50	mAP50:95
YOLO-FAST	0.941	0.677	0.405	0.218	0.214	0.078
YOLO-FAST-HDR-CoT (Proposed)	0.945	0.696	0.423	0.232	0.275	0.100

TABLE XII
ANALYSIS EFFICIENCY MODEL

Model	Parameter	GFLOPs	FPS
YOLOv5n	2,509,634	7.2	15.23
YOLOv6n	4,238,738	11.9	15.12
YOLOv8n	3,012,018	8.2	17.80
YOLOv10n[32]	2,709,380	8.4	16.54
YOLO11n	2,591,010	6.4	19.36
YOLOv12n[33]	2,539,266	6.0	18.64
YOLO-FAST	2,306,818	6.4	19.66
YOLO-FAST-channel optimization	1,536,258	5.7	23.00
YOLO-FAST-HDR-CoT (Proposed)	1,732,006	6.3	20.54

TABLE XIII
COMPREHENSIVE ANALYSIS OF MODEL EFFICIENCY ON A CPU DEVICE

Model	Average Memory Usage (MB)	Latency (ms)	CPU utilization (%)	Peak Memory Consumption (MB)
YOLOv8n	719.69	56.55	96.86	726.41
YOLOv10n[32]	728.37	61.41	97.01	743.93
YOLO11n	712.36	52.88	96.45	726.18
YOLOv12n[33]	682.16	54.13	96.46	697.11
YOLO-FAST	717.73	52.33	96.64	731.71
YOLO-FAST-channel optimization	718.66	44.09	96.15	730.02
YOLO-FAST-HDR-CoT (Proposed)	710.58	48.71	96.42	721.50

In the first modification stage, the architecture applies strategic channel optimization to eliminate network redundancy. This adjustment drastically reduced the parameter count to 1.53 million and lowered the computational cost to 5.7 GFLOPs. Notably, this aggressive compression maintained the mAP50 at 0.946, with only a negligible drop in mAP50:95 to 0.667. This result confirms that the original backbone contained significant parameter redundancy that did not contribute to spatial feature extraction. Subsequently, the tuned network incorporates the proposed Hadamard-Dilated Residual Contextual Transformer (HDR-CoT) module. Although this integration slightly increased the parameter count to 1.73 million and GFLOPs to 6.3 compared to the tuned variant, it provided a substantial enhancement in detection capability. The proposed model achieved an mAP50 of 0.952 and an mAP50:95 of 0.689. Ultimately, the ablation study demonstrates that the proposed architecture outperforms the original baseline in detection accuracy while reducing total parameters by 24.9% and lowering overall computational costs.

Based on Table X, the ablation experiment was conducted to evaluate the contribution of each component in the HDR-CoT module. The results show that the complete HDR-CoT achieves higher mAP50:95 than the original YOLO-FAST, representing a 2.99% improvement. Furthermore, removing the residual connection reduces the performance by 0.58%, while removing the dilated convolution presents similar results. In contrast, removing the Hadamard product results in a larger performance decrease of 1.16% to the proposed module, indicating that the Hadamard product plays an important role in enhancing the extraction performance.

Furthermore, Table XI presents the model performance across different bounding box sizes using a subset of the test dataset. Small, medium, and large bounding boxes are defined based on their areas as $\leq 2,500$, $> 2,500$ to $\leq 10,000$, and $\geq 10,000$ pixels², respectively. The results demonstrate that the proposed module improves detection performance across all bounding box categories, confirming its effectiveness in enhancing the overall detection capability of the proposed network.

C. Runtime Efficiency

A runtime efficiency evaluation was conducted to assess the model's deployment feasibility in practical scenarios requiring fast inference on limited computational resources. As shown in Table XII, the channel-optimized YOLO-FAST achieves the highest inference speed at 23.00 FPS. This performance is expected, as this variant possesses the fewest parameters and lowest GFLOPs among all evaluated models.

However, this increased speed compromises detection accuracy, particularly in the mAP50:95 metric, which drops to 0.667. These findings underscore the inherent trade-off between computational efficiency and detection accuracy. In this regard, the proposed YOLO-FAST Modified model shows a more desirable balance. The proposed model achieves 20.54 FPS, making it the second fastest model after channel optimization and faster than the YOLO-FAST baseline, which reaches 19.66 FPS. At the same time, the proposed model provides higher accuracy than both YOLO-FAST variants. Compared with the YOLO-FAST baseline, FPS increases from 19.66 to 20.54, while mAP50 improves from 0.946 to 0.952, and mAP50:95 improves from 0.669 to 0.689. In relative terms, the speed improvement over the YOLO-FAST baseline is approximately 4.48%, indicating that the proposed architectural modification imposes a negligible runtime penalty and, in fact, simultaneously improves both inference performance and detection quality.

Compared with YOLO-FAST channel optimization, the proposed model experiences a speed reduction from 23.00 FPS to 20.54 FPS, representing a 10.7% decrease. However, this reduction is compensated by a clear gain in accuracy, from an mAP50 of 0.946 to 0.952 and from an mAP50:95 of 0.667 to 0.689. In other words, the proposed model sacrifices a small portion of speed to obtain a more meaningful improvement in detection precision. Such a trade-off is especially important in ship detection, where small errors in

identifying small or occluded objects may significantly affect maritime monitoring performance. Therefore, the proposed model is more suitable than channel optimization when the application requires both speed and detection reliability.

Compared with other benchmark models, the proposed model also demonstrates strong efficiency. YOLOv8, for instance, achieves the highest accuracy but runs at only 17.80 FPS with 8.2 GFLOPs and 3,012,018 parameters. In contrast, the proposed model runs at 20.54 FPS with 6.3 GFLOPs and 1,732,006 parameters. This means the proposed model offers a lower computational burden and faster inference speed, albeit at a slight compromise in absolute accuracy. A similar pattern is also observed when comparing the proposed model with YOLOv5, YOLOv6, and YOLOv10, all of which operate at lower FPS despite having equal or higher computational complexity. These findings reinforce that the proposed design is more appropriate for deployment on edge devices or port-monitoring systems that may not always support GPU acceleration. Moreover, inference was evaluated on a 2-core Intel Xeon CPU running at 2.00 GHz rather than a GPU. This setting makes runtime results more relevant to real-world implementation scenarios, particularly in field-based maritime surveillance systems that often operate with limited hardware resources. In this context, the proposed model's ability to maintain 20.54 FPS indicates that the developed architecture is sufficiently efficient to support near-real-time ship monitoring.

The proposed model uses 1.73 million parameters and 6.3 GFLOPs, compared with more than 2.3 million parameters and 6.4 GFLOPs for the original YOLO-FAST. This represents a 24.9% reduction in parameters and a 1.56% reduction in computational cost. In addition, the proposed model achieves 20.54 FPS on CPU, demonstrating its efficiency for near-real-time ship detection. The inference speed of the proposed HDR-CoT detector was evaluated to assess its suitability for continuous ship monitoring applications. In this study, near-real-time operation is operationally defined as an inference throughput of approximately 20 frames per second (FPS) or higher, corresponding to an average processing time of approximately 50 ms per frame. This criterion is defined according to the requirement for frequent detection updates in continuous visual monitoring rather than strict real-time processing at a predetermined camera frame rate. The proposed HDR-CoT detector achieves an inference speed of 20.54 FPS, corresponding to an average processing time of approximately 48.69 ms per frame.

Furthermore, Table XIII compares the computational efficiency of the proposed model with several lightweight YOLO networks in terms of memory usage, inference latency, and CPU utilization. The proposed model exhibits lower CPU utilization than the competing models, indicating more efficient use of computational resources during inference. The evaluation was conducted on a desktop PC equipped with a 12th Gen Intel® Core™ i7-12700 processor

with a maximum clock frequency of 4.9 GHz and 16 GB of RAM.

Overall, the runtime evaluation confirms that the proposed model achieves a favorable midpoint between computational efficiency, inference speed, and detection performance. Although it is not the absolute fastest model, YOLO-FAST Modified offers a more balanced configuration than both other lightweight benchmarks and previous YOLO-FAST variants. Therefore, it has strong potential for practical deployment in ship detection systems operating on resource-constrained devices in real maritime environments.

IV. CONCLUSION

This study proposes an efficient deep learning architecture for real-time ship detection by modifying the YOLO-FAST model. The primary scientific contribution focuses on analyzing memory and computational efficiency alongside the spatial representation effects of integrating a Contextual Transformer module. The proposed model is significantly more efficient in memory and computation. It requires approximately 24.9 percent fewer parameters than the original YOLO-FAST baseline. Furthermore, the computational demand is only 6.3 GFLOPs, making it 1.56 percent lighter than the original architecture. Regarding spatial representation, integrating the Contextual Transformer block enriches spatial features. This module effectively extracts essential contextual information, particularly for small and distant vessels in complex maritime environments. Balancing high efficiency with rich feature representation yields improved detection performance. The developed system achieves an mAP50:95 score of 0.689, representing a notable 0.020 improvement over the original YOLO-FAST model. Strategic channel optimization in several convolutional layers compresses parameters and computational cost model while reducing the redundancy of unimportant feature maps. Future work will focus on evaluating the robustness of the proposed model under challenging real-world conditions, including severe illumination changes, viewpoint variations, heavy occlusions, and image degradation.

REFERENCES

- [1] L. Qian, Y. Zheng, L. Li, Y. Ma, C. Zhou, and D. Zhang, "A New Method of Inland Water Ship Trajectory Prediction Based on Long Short-Term Memory Network Optimized by Genetic Algorithm," *Applied Sciences (Switzerland)*, vol. 12, no. 8, Apr. 2022, doi: 10.3390/app12084073.
- [2] Y. Mo, C. Li, W. Ren, and W. Wang, "Spotlighting on Objects: Prior Knowledge-Driven Maritime Image Dehazing and Object Detection Framework," *IEEE Journal of Oceanic Engineering*, vol. 50, no. 3, pp. 1978–1992, 2025, doi: 10.1109/JOE.2025.3545289.
- [3] A. Miri Rekavandi, L. Xu, F. Boussaid, A.-K. Seghouane, S. Hoefs, and M. Bennamoun, "A Guide to Image- and Video-Based Small Object Detection Using Deep Learning: Case Study of Maritime Surveillance," *IEEE Transactions on Intelligent Transportation Systems*, vol. 26, no. 3, pp. 2851–2879, 2025, doi: 10.1109/TITS.2025.3530678.
- [4] V. Sanikommu, S. P. Marripudi, H. R. Yekkanti, R. Divi, R. Chandranth, and P. Mahindra, "Edge computing for detection of ship and ship port from remote sensing images using YOLO," *Front. Artif. Intell.*, vol. 8, 2025, doi: 10.3389/frai.2025.1508664.
- [5] M. Zhu, D. Han, B. Han, and X. Huang, "YOLO-HPSD: A high-precision ship target detection model based on YOLOv10," *PLoS One*, vol. 20, no. 5, May, May 2025, doi: 10.1371/journal.pone.0321863.
- [6] L. Shen, T. Gao, and Q. Yin, "YOLO-LPSS: A Lightweight and Precise Detection Model for Small Sea Ships," *J. Mar. Sci. Eng.*, vol. 13, no. 5, May 2025, doi: 10.3390/jmse13050925.
- [7] Y. Li and S. Wang, "EGM-YOLOv8: A Lightweight Ship Detection Model with Efficient Feature Fusion and Attention Mechanisms," *J. Mar. Sci. Eng.*, vol. 13, no. 4, Apr. 2025, doi: 10.3390/jmse13040757.
- [8] L. Min, F. Dou, Y. Zhang, D. Shao, L. Li, and B. Wang, "CM-YOLO: Context Modulated Representation Learning for Ship Detection," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 63, pp. 1–14, 2025, doi: 10.1109/TGRS.2025.3538848.
- [9] C. Qu *et al.*, "DSO-Net: A Lightweight Framework for the Detection of Multiscale Dense Ship Occlusion," *IEEE Internet Things J.*, vol. 12, no. 20, pp. 42638–42651, 2025, doi: 10.1109/JIOT.2025.3593829.
- [10] W. Messaoud, R. Trabelsi, A. Cabani, and F. Abdelkefi, "Adaptive Head Pruning for Attention Mechanism in the Maritime Domain," *IEEE Transactions on Artificial Intelligence*, vol. 6, no. 11, pp. 2966–2976, 2025, doi: 10.1109/TAI.2025.3558724.
- [11] W. Huang *et al.*, "DAShip: A Large-Scale Annotated Dataset for Ship Detection Using Distributed Acoustic Sensing Technique," *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.*, vol. 18, pp. 4093–4107, 2025, doi: 10.1109/JSTARS.2024.3525082.
- [12] Y. Wang, S. Zhang, J. Xu, Z. Cheng, and G. Du, "YOLO-StarLS: A Ship Detection Algorithm Based on Wavelet Transform and Multi-Scale Feature Extraction for Complex Environments," *Symmetry (Basel)*, vol. 17, no. 7, Jul. 2025, doi: 10.3390/sym17071116.
- [13] W. Wu, X. Fan, Z. Hu, and Y. Zhao, "CGDU-DETR: An End-to-End Detection Model for Ship Detection in Day–Night Transition Environments," *J. Mar. Sci. Eng.*, vol. 13, no. 6, 2025, doi: 10.3390/jmse13061155.
- [14] Y. Liu *et al.*, "YOLOv7oSAR: A Lightweight High-Precision Ship Detection Model for SAR Images Based on the YOLOv7 Algorithm," *Remote Sens. (Basel)*, vol. 16, no. 5, Mar. 2024, doi: 10.3390/rs16050913.
- [15] R. D. Kwon, G. R. Nam, J. Tak, J. Shin, H. Cha, and S. W. Lee, "Enhancing Ship Classification in Optical Satellite Imagery: Integrating Convolutional Block Attention Module with ResNet for Improved Performance," Aug. 2024, [Online]. Available: <http://arxiv.org/abs/2404.02135>
- [16] Z. Jiang, L. Su, and Y. Sun, "YOLOv7-Ship: A Lightweight Algorithm for Ship Object Detection in Complex Marine Environments," *J. Mar. Sci. Eng.*, vol. 12, no. 1, Jan. 2024, doi: 10.3390/jmse12010190.
- [17] B. Erum Ayesha, T. Satyanarayana Murthy, P. Ramesh Babu, and R. Kuchipudi, "Ship Detection in Remote Sensing Imagery for Arbitrarily Oriented Object Detection."
- [18] N. Toprak and Y. Yalman, "Ship Detection from Optical Satellite Images Using Convolutional Neural Networks," *Turkish Journal of Engineering*, vol. 9, no. 2, pp. 342–353, Jun. 2025, doi: 10.31127/tuje.1529660.
- [19] S. Kang, Z. Hu, L. Liu, K. Zhang, and Z. Cao, "Object Detection YOLO Algorithms and Their Industrial Applications: Overview and Comparative Analysis," Mar. 01, 2025, *Multidisciplinary Digital Publishing Institute (MDPI)*. doi: 10.3390/electronics14061104.
- [20] Q. Cao *et al.*, "LH-YOLO: A Lightweight and High-Precision SAR Ship Detection Model Based on the Improved YOLOv8n," *Remote Sens. (Basel)*, vol. 16, no. 22, Nov. 2024, doi: 10.3390/rs16224340.

- [21] W. Shi, W. Zheng, and Z. Xu, "Ship-Yolo: A Deep Learning Approach for Ship Detection in Remote Sensing Images," *J. Mar. Sci. Eng.*, vol. 13, no. 4, Apr. 2025, doi: 10.3390/jmse13040737.
- [22] J. Song, J. Xie, Q. Wang, and T. Shen, "An improved YOLO-based method with lightweight C3 modules for object detection in resource-constrained environments," *J. Supercomput.*, vol. 81, no. 5, p. 702, 2025, doi: 10.1007/s11227-025-07187-w.
- [23] Z. Song, X. Zhang, and P. Tan, "YOLO-Fast: a lightweight object detection model for edge devices," *J. Supercomput.*, vol. 81, no. 5, p. 724, 2025, doi: 10.1007/s11227-025-07172-3.
- [24] J. Chen *et al.*, "Run, Don't Walk: Chasing Higher FLOPS for Faster Neural Networks," May 2023, [Online]. Available: <http://arxiv.org/abs/2303.03667>
- [25] Y. Li, T. Yao, Y. Pan, and T. Mei, "Contextual Transformer Networks for Visual Recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 2, pp. 1489–1500, 2023, doi: 10.1109/TPAMI.2022.3164083.
- [26] J. Chen *et al.*, "Run, Don't Walk: Chasing Higher FLOPS for Faster Neural Networks," May 2023, [Online]. Available: <http://arxiv.org/abs/2303.03667>
- [27] P. Zhao and S. Jia, "SPFFNet: Strip Perception and Feature Fusion Spatial Pyramid Pooling for Fabric Defect Detection," in *2025 IEEE International Conference on Big Data (BigData)*, 2025, pp. 2701–2706. doi: 10.1109/BigData66926.2025.11402084.
- [28] J. Hu, L. Shen, and G. Sun, "Squeeze-and-Excitation Networks," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 7132–7141. doi: 10.1109/CVPR.2018.00745.
- [29] Z. Shao, W. Wu, Z. Wang, W. Du, and C. Li, "SeaShips: A Large-Scale Precisely Annotated Dataset for Ship Detection," *IEEE Trans. Multimedia*, vol. 20, no. 10, pp. 2593–2604, Oct. 2018, doi: 10.1109/TMM.2018.2865686.
- [30] X. Sun, H. Wu, G. Yu, and N. Zheng, "Research on the Automatic Detection of Ship Targets Based on an Improved YOLO v5 Algorithm and Model Optimization," *Mathematics*, vol. 12, no. 11, 2024, doi: 10.3390/math12111714.
- [31] K. Lan, X. Jiang, X. Ding, H. Lin, and S. Chan, "High-Efficiency and High-Precision Ship Detection Algorithm Based on Improved YOLOv8n," *Mathematics*, vol. 12, no. 7, Apr. 2024, doi: 10.3390/math12071072.
- [32] A. Wang *et al.*, "YOLOv10: Real-Time End-to-End Object Detection," in *Advances in Neural Information Processing Systems*, A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, Eds., Curran Associates, Inc., 2024, pp. 107984–108011. doi: 10.52202/079017-3429.
- [33] J. Jing and C. Li, "Identification of lightweight rail defects based on YOLOv11 improvement," in *Fifth International Conference on Optical Imaging and Image Processing (ICOIP 2025)*, X. Hao and L. He, Eds., SPIE, 2025, p. 1368823. doi: 10.1117/12.3075801.
- [34] X. Chen, N. Jiang, Z. Yu, W. Qian, and T. Huang, "Citrus leaf disease detection based on improved YOLO11 with C3K2," in *International Conference on Computer Graphics, Artificial Intelligence, and Data Processing (ICCAID 2024)*, X. Xu and A. bin Mohd Zain, Eds., SPIE, 2025, p. 135602Z. doi: 10.1117/12.3061501.
- [35] Y. Tian, Q. Ye, and D. Doermann, "YOLOv12: Attention-Centric Real-Time Object Detectors Latency (ms) MS COCO mAP (%)", doi: 10.0.