scientific reports

Check for updates

OPEN

Modular YOLOv8 optimization for real-time UAV maritime rescue object detection

Beigeng Zhao[1,2], Ye Zhou[2], Rui Song[2], Lizhi Yu[3], Xia Zhang[1,4]✉ & Jiren Liu[1,4]✉

The task of UAV-based maritime rescue object detection faces two significant challenges: accuracy and real-time performance. The YOLO series models, known for their streamlined and fast performance, offer promising solutions for this task. However, existing YOLO-based UAV maritime rescue object detection methods tend to prioritize high accuracy, often at the expense of real-time performance and ease of implementation and expansion. This study proposes a modular plug-and-play optimization approach based on the YOLOv8 framework, aiming to enhance real-time performance while maintaining high accuracy for UAV maritime rescue object detection. The proposed optimization modules are flexible, easy to implement, and extendable. In experiments on the large-scale publicly available SeaDronesSee dataset, our method achieved a 13.53% improvement in accuracy over YOLOv8x while reducing computational cost by 85.63%. Additionally, it surpassed the detection speed of the SeaDronesSee official code's two-stage detector by over 20 times, while maintaining comparable accuracy. Furthermore, our analysis of the experimental results highlights differences in detection difficulty among various objects and potential biases within the dataset.

The task of UAV-based maritime rescue object detection poses two primary challenges for the field of computer vision: accuracy and real-time performance[1–3]. Compared to traditional daily life scene images, maritime rescue images captured from a high-altitude "bird's-eye view" by UAVs lack a fixed "sky above, ground below" orientation[4,5]. The visual characteristics and sizes of boats, personnel, and rescue equipment in this unique scene vary with changes in time, weather, altitude, and angle, presenting significant challenges to detection accuracy[6,7]. Moreover, the high real-time requirements of rescue missions and the constraints of UAV equipment demand streamlined model structures and rapid detection performance[8,9].

To address these challenges, the SeaDronesSee dataset[1], specifically focused on this scenario and task, was constructed and has attracted extensive research. Among the proposed methods, YOLO-based models[10–12] are known for their streamlined structure and real-time performance. However, possibly due to the accuracy-focused ranking mechanism of the SeaDronesSee leaderboard[13], current YOLO-based research[14–16] emphasizes refining model structures to pursue extreme accuracy, somewhat neglecting real-time performance and the ease of implementation and expansion.

In response to these issues, our research is based on the latest stable version of the YOLO series, YOLOv8[17]. Leveraging YOLOv8's modular design, we propose a modular plug-and-play optimization approach, focusing primarily on enhancing real-time performance while maintaining accuracy. We conduct experiments using the official Ultralytics YOLOv8 codebase[18] and the SeaDronesSee dataset[1], performing comprehensive ablation studies, accuracy and speed tests, as well as analyzing the results to validate the effectiveness of our proposed method.

Our contributions are threefold: first, we utilize YOLOv8's modular design to propose a plug-and-play optimization approach, with four modular optimization schemes that are flexible, easy to implement and extend. Second, we scientifically validate the effectiveness of our proposed method on the large-scale UAV maritime rescue object detection dataset, SeaDronesSee. Finally, our analysis of experimental results and dataset characteristics reveals the detection challenges and potential data biases for different object types in the UAV-based maritime rescue scenario presented by SeaDronesSee.

[1]School of Computer Science and Engineering, Northeastern University, Shenyang, China. [2]College of Public Security Information Technology and Intelligence, Criminal Investigation Police University of China, Shenyang, China. [3]Yuhong Sub-bureau, Shenyang Public Security Bureau, Shenyang, China. [4]Neusoft Corporation, Shenyang, China. ✉email: zhangx@neusoft.com; liujiren@mail.neu.edu.cn

Related work

Object detection methods can be mainly divided into two-stage detectors[19], such as Faster R-CNN[20], Mask R-CNN[21] and Cascade R-CNN[22], and single-stage detectors[23], such as YOLO[10–12] and RetinaNet[24]. Two-stage detectors first generate candidate regions and then classify and refine the bounding boxes for these regions. In contrast, single-stage detectors perform object detection directly on the entire image without a candidate region generation stage. Consequently, single-stage detectors generally have lower detection accuracy compared to contemporary two-stage detectors, but they offer faster processing speeds, making them more suitable for real-time applications. This study focuses on YOLOv8[17,18], the latest stable version of the YOLO series known for real-time detection capabilities, with an active research community and promising application prospects.

For training and validating image object detection models, large-scale datasets such as PASCAL VOC[25], MS-COCO[26], ImageNet[27], and Open Images[28] have been constructed. Unlike everyday images, those captured from high altitudes by satellites, aircraft, and drones exhibit unique characteristics. Consequently, an increasing number of datasets focusing on these scenarios have been annotated and made available, including remote sensing datasets like DOTA[29], DIOR[30], and xView[31], as well as UAV detection and tracking datasets such as VisDrone[32]. UAV-based maritime search and rescue scenarios present distinct challenges. Unique visual features and spatial relationships of boats, personnel, and various lifesaving items, along with changes in weather, time, and UAV shooting angles, pose new challenges for image object detection in this specific task. The SeaDronesSee dataset[1] is specifically constructed for this unique task. This study focuses on optimizing YOLOv8 to achieve fast and accurate real-time detection in such scenarios.

The SeaDronesSee leaderboard's[13] ranking strategy prioritizes detection accuracy, which has influenced existing YOLO-related research to also focus primarily on accuracy. Xu et al.[14] introduced an improved method called YoloOW, which once outperformed two-stage detectors to secure the top position on the SeaDronesSee leaderboard. While their method excels in detection accuracy, it poses significant challenges in model architecture and code modification, and its high accuracy relies on input image sizes much larger than YOLO's default, limiting its broader research and application. Similarly, Sea-YOLOv5s[16] and YOLOv7-sea[15], which also prioritize accuracy, showcased improved structures and achieved notable accuracy with higher image resolutions but lacked exploration in speed and real-time performance.

To address the issues of overlooked real-time performance and high modification barriers, our study focuses on leveraging the excellent modular design of the latest stable version of YOLO, YOLOv8. Notably, YOLOv8 adopts an anchor-free detection approach, unlike earlier versions such as YOLOv7 and YOLOv5, which still relied on predefined anchor boxes. The anchor-free design simplifies the model by removing the need for anchor box calculations, reducing both computational complexity and memory requirements, which is particularly beneficial for lightweight aerial image detection tasks[33–35]. This makes YOLOv8 better suited for real-time applications in resource-constrained environments, such as UAV-based maritime rescue scenarios, where balancing detection accuracy and speed is crucial.

We explore easy-to-implement modular plug-and-play modifications to enhance YOLOv8's structure, aiming for real-time, fast, and accurate detection in the maritime rescue scenarios presented in the SeaDronesSee dataset. Our method can be simply realized through YOLOv8's YAML configuration, facilitating subsequent extensions and in-depth research. The effectiveness of our approach is validated through detailed ablation experiments and real-time performance tests. Additionally, we analyze the detection difficulty and potential biases for different objects in the SeaDronesSee dataset based on the experimental results.

Method

Approach overview

As illustrated in Fig. 1, the YOLOv8 model architecture is divided into three main sections: the Backbone section, the Neck section, and the Head section. The Backbone section is responsible for extracting features from images. Located between the Backbone section and the Head section, the Neck section enhances the expressive power of these features by integrating and refining information across different layers. This enhancement enables the Head section to accurately identify the locations and categories of various objects within the image.

The default Backbone section of the YOLOv8 model employs an FPN (Feature Pyramid Network)[36] architecture to extract multi-scale features from the input image. Lower-level features, such as those from P2 and P3, are rich in fine-grained local details, making them particularly effective for detecting small-sized objects. In contrast, higher-level features, such as those from P4 and P5, contain more advanced global semantic information, making them more suitable for detecting larger objects. Unlike typical everyday scenes, aerial images from drones in maritime rescue scenarios contain a large number of small-sized objects. However, the default Neck section of YOLOv8 does not sufficiently utilize these lower-level features that are optimal for detecting small objects. Additionally, as the height and angle of drone photography vary, the size and visual characteristics of objects in the same category can change significantly in the images. Therefore, our optimization strategy for the YOLOv8 model, specifically for this unique scenario, focuses on enhancing the detection capability for small objects while maintaining stability in detecting multi-scale objects with varying features. Based on these considerations, we propose the following modular plug-and-play improvements for different sections of the model.

1. Enhancing the PANet Structure in the Neck Section: We add an extra block to both the top-down and bottom-up pathways of the Path Aggregation Network (PANet)[37] structure, concatenating it to the Backbone FPN's P2 features and adding corresponding detection heads. This enhancement directly leverages the P2 features from the Backbone, which contain low-level local details, beneficial for accurate small object detection.
2. Incorporating Attention Mechanisms Post-Upsample Layers in the Neck Section: Adding attention mechanisms[38] after each upsample layer in the Neck section helps the model focus on important features in both

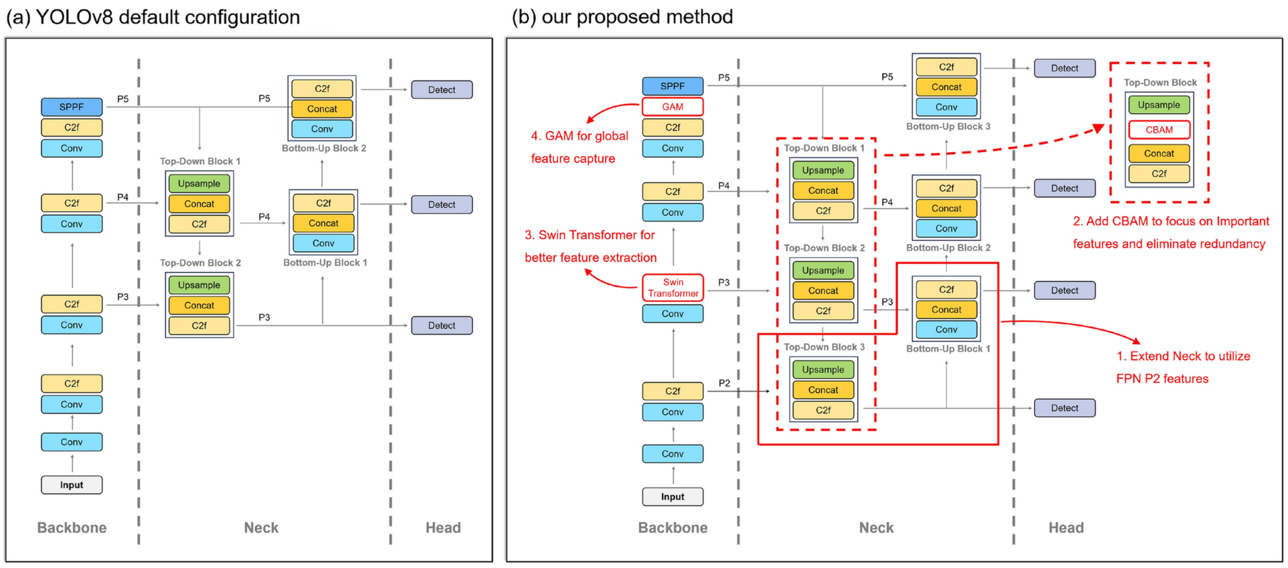
(a) YOLOv8 default configuration
(b) our proposed method
Backbone
Neck
Head
SPPF
C2f
Conv
Input
Upsample
Concat
Detect
Top-Down Block 1
Top-Down Block 2
Top-Down Block 3
Bottom-Up Block 1
Bottom-Up Block 2
Bottom-Up Block 3
P2
P3
P4
P5
GAM
Swin Transformer
CBAM
Top-Down Block
1. Extend Neck to utilize FPN P2 features
2. Add CBAM to focus on Important features and eliminate redundancy
3. Swin Transformer for better feature extraction
4. GAM for global feature capture

Figure 1. Method overview: (**a**) default structure configuration of YOLOv8, (**b**) modular plug-and-play optimization method proposed by us.

the channel and spatial dimensions while suppressing irrelevant and redundant information. In this study, we employ the promising Convolutional Block Attention Module (CBAM)[39] to enhance the model's capability to detect objects of various sizes.

3. Integrating Swin Transformer at the P3 Position of the Backbone FPN: Introducing the Swin Transformer[40] at the P3 position of the Backbone improves the model's feature extraction and generalization capabilities, aiming to enhance the detection of small and multi-scale objects.
4. Introducing GAM at the P5 Position of the Backbone FPN: Incorporating the Global Attention Mechanism (GAM)[41] at the P5 position enables the model to capture long-range dependencies, thereby enhancing its ability to detect large objects.The four modular plug-and-play optimization modules encompass various parts of the YOLOv8 backbone's FPN structure and the Neck section's PANet structure. These modules can be easily implemented, modified, and extended using the YOLOv8 configuration files. Each module is independent, allowing for flexible disassembly and combination in ablation experiments. The following subsections will discuss the implementation details and optimization strategies of each modular plug-and-play module in detail.

Extending neck to utilize FPN P2 features

The Neck section of YOLOv8 employs a PANet[37] structure, which enhances multi-scale features from the Backbone FPN by promoting better information flow and multi-scale feature fusion. As shown in Fig. 1, PANet introduces additional top-down and bottom-up paths to facilitate the flow of information across different levels, enabling the network to aggregate features from various layers more effectively. In the top-down pathway, the upsample module for increasing resolution, the concat module for lateral connections, the C2F module for feature processing and fusion, and the conv module for feature extraction constitute the top-down and bottom-up blocks. These blocks at different positions effectively enhance and fuse features from various levels of the FPN in a dual-pathway manner, from higher to lower levels and back up. This PANet-based Neck structure ensures that both low-level features, which contain fine-grained local details, and high-level features, which capture more abstract semantic information, are fully utilized.

However, as shown in Fig. 1a, in the default YOLOv8 Neck section, the top-down and bottom-up blocks are concatenated to the P3, P4, and P5 layers of the Backbone FPN, but not directly to the P2 layer, which is rich in fine-grained local features. While this streamlined design is effective for detecting common objects in everyday images, the lack of direct concatenation to the P2 layer may result in insufficient fine detection capabilities for small objects in drone aerial imagery. To address this, we propose adding modular plug-and-play top-down and bottom-up blocks in the Neck section, as illustrated by the red solid lines in Fig. 1b. These newly added blocks specifically connect and fuse features from the FPN P2 layer, extending the original PANet pathway in the Neck section, shown in black, and improving small object detection.

Introducing CBAM to focus on key features and eliminate redundancy

The Convolutional Block Attention Module (CBAM)[39] presents a lightweight yet potent approach to enhancing deep neural networks' feature utilization through adaptive emphasis on informative features across channels and spatial locales. Comprised of dual mechanisms—channel attention and spatial attention—CBAM synergistically tunes feature responses, amplifying the network's discriminative prowess. By selectively emphasizing salient data aspects while suppressing less critical details, CBAM fosters a sharper focus on pivotal visual elements, thereby boosting performance in diverse computer vision applications.

CBAM's operation harmoniously intertwines these attention modalities: Channel attention commences with global average pooling (GAP) on the input feature map $X \in \mathbb{R}^{H \times W \times C}$, leading to a channel descriptor vector. Sequentially, two fully-connected layers (FC_1 and FC_2), interlaced with ReLU for non-linearity, refine this vector, ultimately processed by a sigmoid function to output normalized attention weights α_c:

$$\alpha_c = \sigma\left(FC_2\left(\mathrm{ReLU}\left(FC_1\left(\mathrm{GAP}(X)\right)\right)\right)\right) \tag{1}$$

Parallelly, spatial attention M_s underscores vital spatial sites employing a 1×1 convolution after merging global max-pooled (GMP) and GAP outputs from X, reflecting extremal and aggregate channel cues:

$$M_s = \sigma\left(\mathrm{Conv}_{1\times1}\left(\left[\mathrm{GMP}(X), \mathrm{GAP}(X)\right]\right)\right) \tag{2}$$

Ultimately, CBAM consolidates these insights via element-wise multiplication of the input X with broadened channel attention α, reshaped to conform with X, and M_s:

$$X_{CBAM} = X \odot (\alpha \otimes M_s) \tag{3}$$

This fusion leverages the combined advantages of channel attention and spatial attention, yielding a refined feature representation X_{CBAM} acutely attuned to data's essential facets, reinforcing the network's performance capabilities and discriminatory finesse.

As illustrated by the red dashed demarcation in Fig. 1b, our second modular plug-and-play improvement entails the insertion of a CBAM module within each of the top-down blocks constituting the Neck section. Specifically, these modules are introduced immediately following the upsampling operation and prior to the concatenation step in every block. By doing so, each CBAM module selectively accentuates those channel and spatial features within the upsampled feature maps that are most germane to maritime rescue object detection, concurrently suppressing irrelevant information. This targeted enhancement leads to a purer and more efficacious representation of feature maps, ensuring that the successive feature integration phases, including concatenation and the consecutive context-to-feature (C2f) transformation, can more efficiently leverage the optimized feature sets.

Enhancing backbone feature extraction with Swin transformer

In recent advancements of vision transformers, the Swin Transformer[40] has emerged as a groundbreaking architecture, particularly excelling in image recognition and dense prediction tasks. Unlike conventional transformer[42] models that apply self-attention globally across the entire image, Swin Transformer introduces a hierarchical structure with shifted windows, striking a balance between computational efficiency and representational capacity. This design innovation not only scales elegantly to high-resolution inputs but also significantly reduces computational complexity compared to its predecessors.

Central to Swin Transformer is the concept of shifted window-based self-attention, outlined in Eq. (4), where the attention operation is performed locally within non-overlapping windows, followed by a shifting mechanism in consecutive layers to allow cross-window communication. This two-stage process ensures local and global dependencies are captured effectively without compromising computational efficiency:

$$\mathrm{Attention}(Q, K, V) = \mathrm{Softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{4}$$

Here, Q, K, and V denote the query, key, and value matrices derived from the input features within each window, and d_k is the dimensionality of the key vectors, used for scaling. The 'shift' in Swin Transformer refers to the rearrangement of these windows in subsequent layers, where each window slides by a fraction of its size, enabling interaction with neighboring windows and facilitating long-range dependency learning.

As highlighted by the red annotation on the left in Fig. 1b, our third modular enhancement involves replacing the C2f module, responsible for processing FPN P3 features in the Backbone section, with a Swin Transformer module. This substitution allows the model to benefit from Swin Transformer's hierarchical structure and shifted window-based self-attention mechanism, which effectively integrates multi-scale features while maintaining computational efficiency. This capability is particularly crucial in UAV-based maritime search and rescue scenarios, where the objects to be detected-such as boats, people, and rescue equipment-exhibit significant variations in scale and diverse morphological characteristics. Moreover, the Swin Transformer's hierarchical self-attention mechanism can preserve detailed information while filtering out redundant data, which is expected to enhance the model's ability to distinguish and recognize objects in complex environments, such as the ocean surface and objects floating on it, including people, boats, and onboard personnel and equipment, where significant overlap exists between these objects.

Integrating GAM for enhanced global feature capture

GAM[41] is designed to capture long-range dependencies and enhance the global contextual information in feature maps. This mechanism improves the model's ability to detect large objects and understand complex scenes by focusing on globally significant features. The core principle of GAM can be mathematically expressed as follows:

$$M_g(F) = \sigma(\mathrm{Conv}_2(\mathrm{ReLU}(\mathrm{Conv}_1(F)))), \quad (5)$$

where F represents the input feature map, Conv_1 and Conv_2 denote convolutional layers, ReLU is the Rectified Linear Unit activation function, and σ is the sigmoid function. The resulting $M_g(F)$ is the global attention map.

The refined feature map is obtained by applying this attention map to the input feature map through element-wise multiplication:

$$F' = M_g(F) \odot F, \quad (6)$$

where $\odot$ denotes element-wise multiplication. By incorporating GAM, the enhanced feature map F' integrates global context and long-range dependencies, which are crucial for improving object detection performance in the YOLOv8 model.

Our fourth modular plug-and-play enhancement involves introducing the GAM module at the FPN P5 feature level in the Backbone section, as indicated by the red annotation on the left side of Fig. 1b. This enhancement allows the P5 layer features to undergo additional processing and optimization before passing through the SPPF module, providing richer and more informative features for subsequent fusion in the Neck section. In maritime rescue object detection tasks, the size disparity between boats, people, and rescue equipment is significant, with boats in particular being much larger than people and rescue gear in the same image. By incorporating GAM's ability to capture long-range dependencies, the model is expected to better utilize global features when detecting large objects, such as boats, in the image.

Experiment

Dataset

We utilized the large-scale, publicly accessible UAV-based maritime search and rescue object detection dataset, SeaDronesSee[1], to validate our proposed method. The images in this dataset capture various ships, different types of personnel, and rescue equipment in this specialized scenario. These images were taken by various representative UAVs from different altitudes, angles, and times over open waters. They are high in clarity and resolution, reaching up to 3840×2160 or 5456×3632 pixels. Each object within these images is accurately annotated with bounding boxes following the COCO[26] standard. The SeaDronesSee dataset is released under the CC0 1.0 Universal Public Domain Dedication license[43,44], meaning the images and data can be freely used in publications. Representative images and close-ups of various object categories in SeaDronesSee are shown in Fig. 2.

As shown in Fig. 2, the objects in the SeaDronesSee dataset are precisely annotated with rectangular bounding boxes and labels, categorized into six classes: boat, floater, swimmer, floater on boat, swimmer on boat, and life jacket. It can be observed from the figure that the boat class objects are significantly larger in size compared to the other classes within the same image. The swimmer and floater classes represent people in the water, which

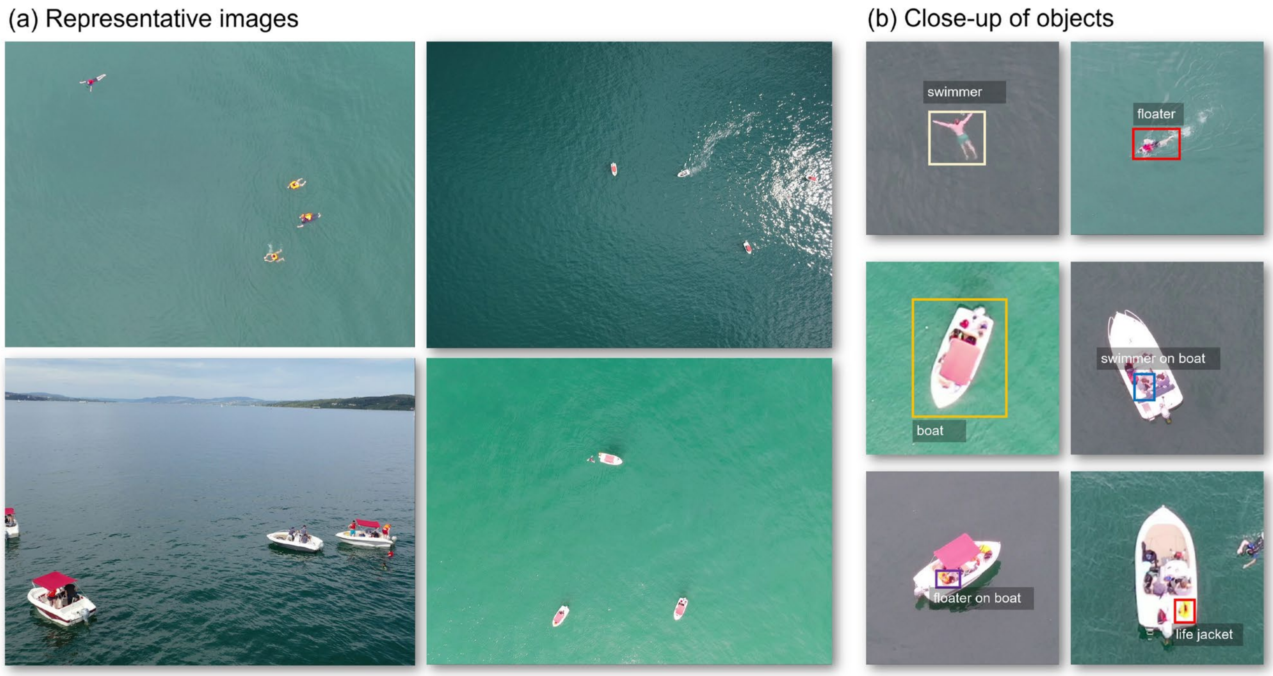

Figure 2. Representative images and close-ups of various object categories in the SeaDronesSee dataset.

are smaller in size than boats and have the ocean as their background. The swimmer on boat and floater on boat classes represent people on boats, which not only exhibit small object characteristics but also overlap with the boat class, adding an extra layer of challenge to the detection task. Additionally, life jacket objects are not only smaller than all the other personnel-related objects, but their position also overlaps with the boat class, making them the most difficult to detect accurately based on visual features.

To conduct experiments more effectively and analyze the results, we statistically analyzed the area distribution (in pixels) of different object categories and the sample counts of different objects in the training and validation sets of the SeaDronesSee dataset, as shown in Table 1. The table reveals significant size differences among various objects. For example, the average size of the boat category is 4 to 7 times larger than that of the swimmer and floater categories, and 31 times larger than that of the life jacket category. Additionally, due to varying UAV shooting altitudes and angles, objects of the same category exhibit considerable size differences in different images. For instance, the size of boat objects varies by up to 2129 times between the smallest and largest instances. These unique characteristics of object appearances in UAV-based maritime SAR scenarios pose distinct challenges for optimizing object detection models.

Furthermore, we conducted a statistical analysis of the annotation counts for different object categories in the training and validation sets, as shown in the last three columns of Table 1. We found that the division between the training and validation sets is imbalanced across categories. For most categories, the ratio of annotations between the training and validation sets ranges from 3 to 4. However, for the category 'swimmer on boat', the ratio reaches 7.68, and for 'life jacket', it reaches a significantly unbalanced 41, with only 2 instances in the validation set. Therefore, we cautiously suspect that this extreme imbalance may introduce some bias into the validation results for the 'life jacket' category. Attention will be paid to this situation in the analysis of the experimental results.

Experimental configuration and evaluation metrics

Experiments were conducted on a Windows 11 machine equipped with an RTX 4090 GPU, utilizing Python 3.9, PyTorch 2.3.0, and the official YOLOv8 code repository[18]. During the training and validation of each YOLOv8-based model, an input image size of 640×640 was used. The experiments were configured with a maximum of 350 training epochs and an early stopping criterion set at 100 epochs.

The models were trained from scratch solely on the SeaDronesSee dataset, without any pre-training on large-scale datasets like COCO. This approach was intentionally chosen to highlight the differences in detection capabilities across object categories within the specific context of UAV-based maritime rescue scenarios presented in the SeaDronesSee dataset, allowing for a more focused evaluation of the effectiveness of different optimization strategies.

To comprehensively evaluate the model's performance, the following evaluation metrics were used:

GFLOPs: This metric measures the computational complexity of the model, specifically the number of floating-point operations executed per second. The formula for GFLOPs is:

$$\text{GFLOPs} = \frac{\text{Total Floating Point Operations}}{10^9} \tag{7}$$

Higher GFLOPs indicate greater computational demands and higher hardware performance requirements.

AP$_{50}$: This metric represents the average precision at an IoU threshold of 0.5, reflecting the model's detection performance under lenient matching conditions. The calculation formula is:

$$\text{AP}_{50} = \int_0^1 \text{Precision}(r)\,\mathrm{d}r \tag{8}$$

where r represents the recall rate. A higher AP$_{50}$ value generally indicates better object recognition capability, as the model is evaluated under relatively forgiving conditions.

AP$_{50\text{-}95}$: This metric provides a more comprehensive evaluation of the model's detection performance by averaging the precision across multiple IoU thresholds, ranging from 0.5 to 0.95 in increments of 0.05. The calculation formula is:

Object	Min area	Max area	Avg area	Train count	Val count	Train-Val ratio
swimmer	110	65601	5368.39	2480	818	3.03
floater	50	172788	6631.17	5963	1615	3.69
boat	165	351330	31416.82	7643	2173	3.52
swimmer on boat	108	35530	4323.04	3501	456	7.68
floater on boat	100	36736	4361.91	1603	405	3.96
life jacket	336	2052	987.34	82	2	41

Table 1. Statistical annotation information of different object categories in the SeaDronesSee dataset, with areas measured in pixels.

$$\mathrm{AP}_{50-95} = \frac{1}{10}\sum_{i=0}^{9} \mathrm{AP}_{\mathrm{IoU}=0.5+0.05\cdot i} \tag{9}$$

This metric offers a thorough assessment of the model's precision under various matching conditions, from lenient to strict. A higher $\mathrm{AP}_{50\text{-}95}$ score indicates the model's ability to maintain good detection performance across different levels of object overlap and localization difficulty, making it a more robust and stringent measure of the model's overall detection capability. The distinction between AP_{50} and $\mathrm{AP}_{50\text{-}95}$ lies in their evaluation scope. AP_{50} measures detection performance under less strict conditions, making it suitable for evaluating the model's ability to detect specific object categories. In contrast, $\mathrm{AP}_{50\text{-}95}$ assesses performance across a wider range of IoU thresholds, offering a more comprehensive and rigorous evaluation of the model's overall detection accuracy and generalization. In our experiments, we use $\mathrm{AP}_{50\text{-}95}$ to report the model's average precision across all objects, while AP_{50} is employed for assessing the detection performance of individual object categories.

FPS: This metric indicates the number of image frames the model processes per second during inference, serving as an important measure of the model's inference speed. Higher FPS implies the model can handle more images in a shorter time, making it suitable for real-time applications. In our experiments, FPS was approximated by dividing the total inference time on the validation set by the total number of images in the validation set.

Ablation study

In this ablation study, we selected the YOLOv8 model with the scale set to 's' for its balance between real-time performance and accuracy. Using the 'n' scale would result in insufficient precision, while choosing 'm' or larger scales would significantly increase the GFLOPs, making real-time applications impractical. By focusing on the 's' scale, we aim to explore the impact of various modular plug-and-play improvements on the model's performance. Specifically, we progressively added each proposed module to evaluate their individual and combined effects on the SeaDronesSee dataset, including PA (adding top-down and bottom-up blocks in the Neck to utilize P2 features), CBAM (incorporating a Convolutional Block Attention Module after the upsample operation in the top-down block), STP3 (replacing the C2F module in the Backbone's FPN P3 with a Swin Transformer), and GAM (adding a Global Attention Mechanism after the FPN P5 in the Backbone). These modular plug-and-play methods, as depicted in Fig. 1b, were systematically tested to determine their impact on model performance.

As shown in Table 2, adding extra top-down and bottom-up blocks in the Neck section to complete the PANet[37] pathway and fully utilize the FPN[36] P2 features significantly enhances the model's detection capabilities. Incorporating the CBAM[39] module after the upsampling operation in the top-down block further improves the detection performance for small objects like swimmer on boat and life jacket. With the addition of other modular plug-and-play improvements, the model's detection performance for all categories, except life jacket, shows notable improvement. Notably, the YOLOv8-PA-STP3-GAM-CBAM-s model, which integrates the most plug-and-play enhancements, achieves excellent results in detecting various objects of different sizes, except for the life jacket category, which was previously discussed in the Dataset subsection as having controversial data bias.

Notably, the YOLOv8-PA-CBAM-s model, which achieves the highest average precision, is not the model with the most integrated modular plug-and-play improvements. Furthermore, it does not achieve the highest detection performance for any individual category. The primary reason for this is the unique characteristics of the life jacket category, which many models fail to detect effectively in the validation set. As detailed in the Dataset subsection, this category not only has small object sizes and overlaps with the boat category, but also has a very limited number of instances, with only 2 examples in the validation set. This scarcity contributes to the evaluation bias in the model's performance.

Evaluation of speed and accuracy

In this subsection, we will validate the advantages in speed and accuracy of the two most representative modular plug-and-play optimized models from the previous subsection: YOLOv8-PA-CBAM-s and YOLOv8-PA-STP3-GAM-CBAM-s. These models represent the highest average precision and the best overall precision, excluding

Model	GFLOPs	$\mathrm{AP}_{50\text{-}95}$	AP_{50}	S_{50}	F_{50}	B_{50}	$S^{\dagger}_{50}$	$F^{\dagger}_{50}$	LJ_{50}
YOLOv8-s	28.5	23.7	39.7	67.8	64.2	94.9	10.4	0.766	0
YOLOv8-PA-s	36.7	28.7	52.0	70.1	78.1	97.1	31.7	34.8	0
YOLOv8-PA-CBAM-s	37.0	**30.2**	**59.1**	67.8	77.8	97.2	32.8	29.7	49.5
YOLOv8-PA-STP3-s	40.1	29.0	52.4	65.7	80.7	97.2	**39.1**	31.9	0
YOLOv8-PA-GAM-s	38.0	29.6	57.5	66.7	78.5	97.6	31.1	21.5	**49.6**
YOLOv8-PA-STP3-CBAM-s	40.4	29.2	53.8	71.0	**81.4**	**97.8**	36.1	**36.4**	0
YOLOv8-PA-STP3-GAM-CBAM-s	41.8	29.4	54.1	**72.2**	**81.4**	97.0	38.2	36.0	0

Table 2. Ablation study results of different Modular Plug-and-Play improvements on the SeaDronesSee validation set. The rightmost six columns represent the average precision at an IoU threshold of 0.5 for six different object categories in the dataset: S (swimmer), F (floater), B (boat), $\mathrm{S}^{\dagger}$ (swimmer on boat), $\mathrm{F}^{\dagger}$ (floater on boat), and LJ (life jacket). The highest score for each metric is indicated in bold.

Model	GFLOPs	AP_{50-95}	AP_{50}	S_{50}	F_{50}	B_{50}	$S^{\dagger}_{50}$	$F^{\dagger}_{50}$	LJ_{50}
YOLOv8-n	8.1	22.7	37.9	62.6	63.0	95.0	6.05	0.519	0
YOLOv8-s	28.5	23.7	39.7	67.8	64.2	94.9	10.4	0.766	0
YOLOv8-m	78.7	25.6	44.7	68.2	71.5	95.5	19.6	13.3	0
YOLOv8-l	164.8	25.6	42.3	69.9	71.4	95.3	15.4	2.37	0
YOLOv8-x	257.4	26.6	46.1	69.9	70.9	95.5	23.9	16.3	0
YOLOv8-PA-CBAM-s	37.0	30.2	59.1	67.8	77.8	97.2	32.8	29.7	49.5
YOLOv8-PA-STP3-GAM-CBAM-s	41.8	29.4	54.1	72.2	81.4	97.0	38.2	36.0	0

Table 3. Comparison of computational cost and accuracy between YOLOv8 models at different scales and two representative models optimized by us. S, F, B, $S^{\dagger}$, $F^{\dagger}$, and LJ represent swimmer, floater, boat, swimmer on boat, floater on boat, and life jacket, respectively.

Model	FPS	Image size	AP_{50-95}	AP_{50}	S_{50}	F_{50}	B_{50}	$S^{\dagger}_{50}$	$F^{\dagger}_{50}$	LJ_{50}
Faster R-CNN with ResNet50 [20,44,45]	18.01	1920×1080	11.3	24.9	36.5	29.2	81.0	2.4	0.2	0
Faster R-CNN with ResNeXt101 [20,44,46]	13.56	1920×1080	12.7	27.6	41.9	36.3	84.1	2.4	0.8	0
Faster R-CNN with ResNeXt101-FPN [20,36,44,46]	12.98	1920×1080	**32.7**	**62.8**	71.6	**88.1**	96.8	**39.2**	30.5	**50.5**
YOLOv8-PA-CBAM-s	333.33	640×640	30.2	59.1	67.8	77.8	**97.2**	32.8	29.7	49.5
YOLOv8-PA-STP3-GAM-CBAM-s	303.03	640×640	29.4	54.1	**72.2**	81.4	97.0	38.2	**36.0**	0

Table 4. Comparison of inference speed, input image size, and accuracy between our optimized model and the two-stage detector in the SeaDronesSee official codebase. The highest values of each accuracy metric are highlighted in bold. S, F, B, $S^{\dagger}$, $F^{\dagger}$, and LJ represent swimmer, floater, boat, swimmer on boat, floater on boat, and life jacket, respectively.

the controversial life jacket category, respectively. The validation method involves comparing the computational cost and accuracy of these two models against various scales of the YOLOv8 models. This comparison aims to demonstrate that our optimizations to the YOLOv8 structure enhance accuracy for low-complexity models. Additionally, we will compare the accuracy and speed of these optimized models with state-of-the-art two-stage models with advanced backbones to verify that our optimization approach yields models capable of real-time high-accuracy detection.

Table 3 presents the computational costs and performance of different scales of YOLOv8 models on the SeaDronesSee object detection dataset. Generally, as the scale increases, the detection capabilities of YOLOv8 models for various object categories also improve. However, models smaller than the 'x' scale exhibit significantly lower detection capabilities for small objects such as swimmer on boat and floater on boat compared to the models optimized with the modular plug-and-play approach. Even the YOLOv8-x model, with a high computational cost of 257.4 GFLOPs, fails to effectively detect the smallest life jacket objects. In contrast, the two 's' scale models optimized with the modular plug-and-play approach achieve an average precision improvement of 13.53% and 10.53% respectively, while only consuming 14.37% and 16.24% of the computational cost of YOLOv8-x. Except for the somewhat contentious life jacket category, the modular plug-and-play optimized models show no weaknesses in detection accuracy across all object categories.

To comprehensively and objectively evaluate the accuracy of the optimized models, we compared their performance with two-stage detectors on the SeaDronesSee dataset, as shown in Table 4. Specifically, we utilized the official baseline code[44] provided by SeaDronesSee to test the performance and speed of Faster R-CNN models with different backbones and FPN structures. Notably, we used the official baseline input image size of 1920× 1080 for training and evaluating the two-stage detectors, while our optimized YOLOv8 models used the default, significantly smaller input size of 640×640. As seen in Table 4, despite having an input image size approximately one-fifth that of the two-stage detectors, the modular plug-and-play optimized 's' scale YOLOv8 models achieved comparable average precision to the most complex Faster R-CNN[20] model with a ResNeXt-101[46] backbone and FPN[36] structure. Additionally, our optimized models outperformed the two-stage detectors in detecting small objects like swimmer, boat, and floater on boat. Remarkably, the detection speed, measured as FPS on the validation set, increased by more than 20 times. Figure 3 visually illustrates this comparison in terms of speed and accuracy.

Category-specific detection analysis

In our experiments, we observed significant differences in the detection difficulty of various objects within the SeaDronesSee dataset, which represents maritime rescue scenarios. To further analyze and understand these differences, we created the heatmap shown in Fig. 4. As illustrated, the boat category is the easiest to detect, with nearly all models achieving high accuracy. Floater and swimmer objects are relatively more challenging to detect, but optimized models show noticeable improvements in accuracy for these categories. The swimmer on boat and floater on boat categories are even more difficult to detect, with performance varying significantly across

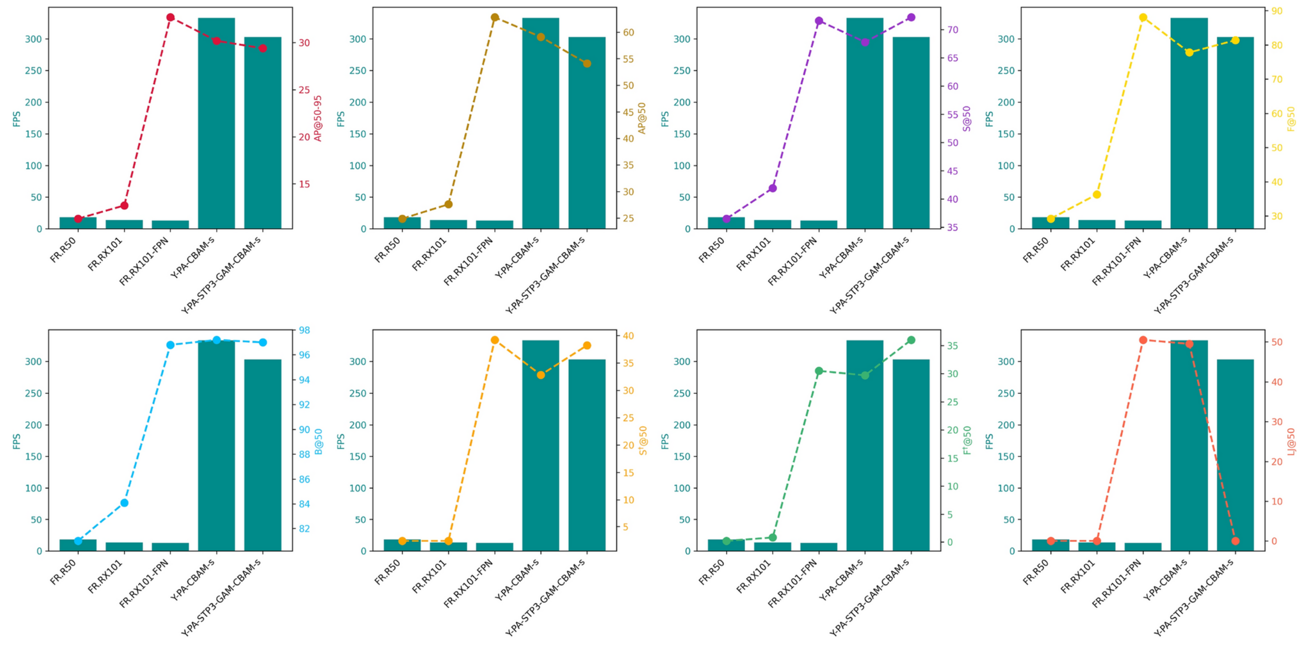

Figure 3. Combination plots illustrating the relationship between different models' speed (FPS) and detection accuracy. Each subplot corresponds to a specific detection accuracy metric. The left vertical axis in each subplot shows the FPS, represented by the height of the bars, where taller bars indicate faster speeds. The right vertical axis shows a specific accuracy metric (i.e., mean precision with IoU thresholds from 0.5 to 0.95, mean precision at IoU 0.5, and mean precision for six individual categories), represented by points on a line, where higher points indicate greater accuracy.

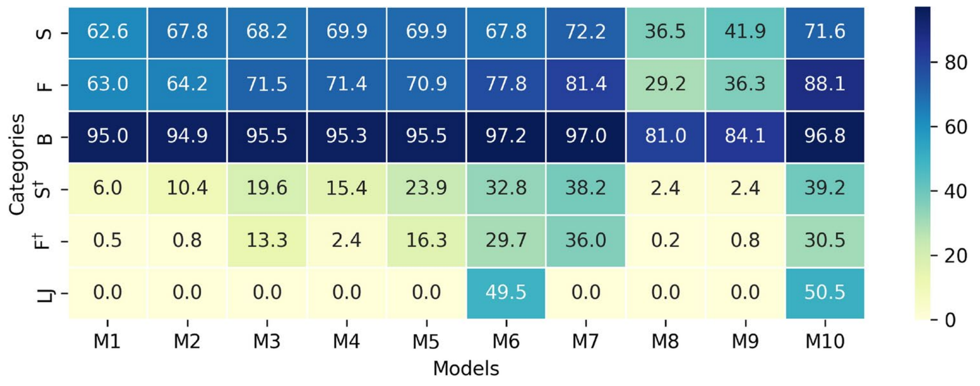

Figure 4. Heatmap of object detection accuracy for different models across various object classes. Each column corresponds to a specific model, where M1 to M7 correspond to the seven models in Table 3, and M8 to M10 correspond to the three two-stage detectors in Table 4. S, F, B, S†, F†, and LJ represent swimmer, floater, boat, swimmer on boat, floater on boat, and life jacket, respectively.

different optimized models. The most challenging and contentious category is the life jacket, with most models failing to detect it accurately, resulting in a detection precision of zero.

The significant differences in detection difficulty across different object categories can be explained by several factors. Based on the statistical analysis of data characteristics and sample distribution for each category presented in the Dataset section, boat objects are the largest, with distinct visual features and the highest number of labeled samples. Their balanced distribution across both data subsets makes them the easiest to detect accurately. Although swimmer and floater objects are smaller and have fewer samples, they are still easier to detect than the remaining categories due to their consistent ocean background, relatively sufficient number of labeled samples, and balanced sample distribution.

In contrast, swimmer on boat and floater on boat objects are more challenging to detect because of their positional overlap with boats. Notably, models detect swimmer on boat more accurately than floater on boat, primarily due to the much larger number of labeled samples for the former.

Finally, life jacket objects pose the greatest challenge. They are the smallest in size, overlap with boat objects, and have far fewer labeled samples than all other categories. Additionally, their sample distribution is highly imbalanced, with 41 times more life jacket samples in the training set than in the validation set.

Discussion

We propose a modular plug-and-play approach to improve the YOLOv8 architecture, aiming to train lightweight models for fast and accurate maritime search and rescue object detection. Experiments on the SeaDronesSee dataset validate our proposal and further reveal the characteristics of this dataset.

Compared to earlier studies on object detection using the SeaDronesSee dataset[14,16,47,48], the model presented in this paper has not achieved breakthroughs in terms of precision and recall. However, our research motivation is not to design a model that simply achieves higher accuracy on this dataset. Instead, we aim to leverage the characteristics of the YOLOv8 framework to propose a method that is user-friendly, flexible, and scalable, specifically designed for the unique challenges of UAV-based maritime search and rescue. Beyond accuracy, we also focus on parameters related to real-time performance. Through a detailed discussion of the proposed modular optimization approach and experiments conducted with the SeaDronesSee dataset, The objectives of our research were realized.

One limitation of this study lies in the gap between experimental data and real-world scenarios. First, due to the difficulty of collecting data in UAV-based maritime search and rescue scenarios, the SeaDronesSee dataset contains sample biases, as discussed in the Dataset subsection, which to some extent restricts the trained model's ability to generalize to unseen real-world data. Second, the complexity of backgrounds and objects in real-world maritime rescue situations surpasses what is captured in the SeaDronesSee dataset, meaning the model's true capabilities require validation in real-world environments. However, given the challenge of acquiring large-scale UAV-based maritime search and rescue images in real scenarios, a feasible and effective solution, as demonstrated in previous work[9,49], is to utilize virtual reality technology to create 3D simulated UAV search and rescue environments, allowing for the evaluation of pre-trained models on open-source datasets such as SeaDronesSee.

Additionally, another limitation of this study is that it focuses on the application of the improved YOLOv8 model for UAV-based maritime search and rescue object detection from an algorithmic perspective, without addressing the integration and deployment of the algorithm with specific UAV hardware and control systems. Regarding the deployment methods for object detection models on UAV hardware, there are two main paradigms[50] in the field of UAV object detection hardware and control systems: cloud-based and edge-based. In the cloud-based paradigm[51,52], UAVs are responsible only for collecting images and uploading them to the cloud, where storage, inference, and deeper analysis are handled by powerful cloud servers. This paradigm requires ample network bandwidth and is thus not well-suited for tasks with high real-time requirements. In contrast, the edge-based paradigm involves UAVs equipped with lightweight computing devices[53–55], such as the NVIDIA Jetson series, Intel Movidius, or Google Coral, which act as the edge, not only capturing images but also processing and inferring part of the data. Meanwhile, the connected edge server provides services like networking, control, and data storage. The improved YOLOv8 model discussed in this paper is known for its lightweight design and real-time responsiveness, making it particularly suitable for deployment in edge computing hardware paradigms, where real-time detection tasks must be performed with limited computational resources.

Future valuable research includes but is not limited to: re-dividing the SeaDronesSee dataset and conducting thorough validation to ensure fair and stable model evaluation; exploring more optimal YOLOv8 models for maritime rescue object detection using the proposed modular plug-and-play method; and deploying the models in real-world scenarios to measure their actual performance.

Conclusions

We proposed a modular plug-and-play approach to optimize the YOLOv8 model for fast and accurate maritime search and rescue object detection. Experiments on the SeaDronesSee dataset confirmed that our optimized models outperform the default YOLOv8 model across multiple detection metrics, achieving state-of-the-art two-stage detector accuracy with over 20 times faster speed. However, this study has some limitations, such as potential evaluation biases due to uneven data distribution, the need for further optimization of the modular approach, and discrepancies between FPS measurements in experimental settings and real-world applications.

Future research directions include re-dividing and validating the SeaDronesSee dataset to ensure fairness and stability in model evaluation; further exploring and optimizing the modular plug-and-play YOLOv8 models to achieve even higher speed and accuracy; and deploying the models in real-world applications to verify their actual performance.

Data availability

The SeaDronesSee dataset is available for download at https://cloud.cs.uni-tuebingen.de/index.php/s/aJQPHLGnke68M52, and additional information can be found on the official website at https://macvi.org/. The YOLOv8 configuration files and core code supporting the methods and results presented in this study can be accessed at https://github.com/bgno1/modular_yolov8_sds.

Received: 9 June 2024; Accepted: 8 October 2024
Published online: 18 October 2024

References

1. Varga, L. A., Kiefer, B., Messmer, M. & Zell, A. Seadronessee: A maritime benchmark for detecting humans in open water. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision* 2260–2270 (2022).
2. Gonçalves, L. & Damas, B. Automatic detection of rescue targets in maritime search and rescue missions using uavs. In *2022 International Conference on Unmanned Aircraft Systems (ICUAS)* 1638–1643 (IEEE, 2022).
3. Vasilopoulos, E. et al. A comparative study of autonomous object detection algorithms in the maritime environment using a uav platform. *Computation* **10**, 42 (2022).
4. Mittal, P., Singh, R. & Sharma, A. Deep learning-based object detection in low-altitude uav datasets: A survey. *Image Vis. Comput.* **104**, 104046 (2020).
5. Yang, Z. et al. Uav remote sensing applications in marine monitoring: Knowledge visualization and review. *Sci. Total Environ.* **838**, 155939 (2022).
6. Fernandes, D. S., Bispo, J., Bento, L. C. & Figueiredo, M. Enhancing object detection in maritime environments using metadata. In *Iberoamerican Congress on Pattern Recognition* 76–89 (Springer, 2023).
7. Chen, M., Sun, J., Aida, K. & Takefusa, A. Weather-aware object detection method for maritime surveillance systems. *Futur. Gener. Comput. Syst.* **151**, 111–123 (2024).
8. Zhang, L. et al. Sg-det: Shuffle-ghostnet-based detector for real-time maritime object detection in uav images. *Remote Sens.* **15**, 3365 (2023).
9. Poudel, R., Lima, L. & Andrade, F. A novel framework to evaluate and train object detection models for real-time victims search and rescue at sea with autonomous unmanned aerial systems using high-fidelity dynamic marine simulation environment. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision* 239–247 (2023).
10. Nazir, A. & Wani, M. A. You only look once-object detection models: a review. In *2023 10th International Conference on Computing for Sustainable Global Development (INDIACom)* 1088–1095 (IEEE, 2023).
11. Redmon, J., Divvala, S., Girshick, R. & Farhadi, A. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* 779–788 (2016).
12. Wang, C.-Y., Bochkovskiy, A. & Liao, H.-Y. M. Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* 7464–7475 (2023).
13. Seadronessee object detection leaderboard (2023, accessed 12 Mar 2023). https://macvi.org/leaderboard/airborne/seadronessee/object-detection.
14. Xu, J. et al. Yoloow: A spatial scale adaptive real-time object detection neural network for open water search and rescue from uav aerial imagery. *IEEE Trans. Geosci. Remote Sens.* (2024).
15. Zhao, H., Zhang, H. & Zhao, Y. Yolov7-sea: Object detection of maritime uav images based on improved yolov7. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision* 233–238 (2023).
16. Wang, X., Pan, Z., He, N. & Gao, T. Sea-yolov5s: A uav image-based model for detecting objects in seadronessee dataset. *J. Intell. Fuzzy Syst.* **2023**, 1–12 (2023).
17. Sohan, M., Sai Ram, T., Reddy, R. & Venkata, C. A review on yolov8 and its advancements. In *International Conference on Data Intelligence and Cognitive Informatics* 529–545 (Springer, 2024).
18. Ultrylytics. Ultralytics yolov8 (2024, accessed 28 Apr 2024). https://github.com/ultralytics/ultralytics.
19. Du, L., Zhang, R. & Wang, X. Overview of two-stage object detection algorithms. *J. Phys. Conf. Ser.* **1544**, 012033 (2020).
20. Ren, S., He, K., Girshick, R. & Sun, J. Faster r-cnn: Towards real-time object detection with region proposal networks. *Adv. Neural Inf. Process. Syst.* **28**, 85 (2015).
21. He, K., Gkioxari, G., Dollár, P. & Girshick, R. Mask r-cnn. In *Proceedings of the IEEE International Conference on Computer Vision*, 2961–2969 (2017).
22. Cai, Z. & Vasconcelos, N. Cascade r-cnn: Delving into high quality object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* 6154–6162 (2018).
23. Zhang, Y., Li, X., Wang, F., Wei, B. & Li, L. A comprehensive review of one-stage networks for object detection. In *2021 IEEE International Conference on Signal Processing, Communications and Computing (ICSPCC)* 1–6 (IEEE, 2021).
24. Lin, T.-Y., Goyal, P., Girshick, R., He, K. & Dollár, P. Focal loss for dense object detection. In *Proceedings of the IEEE International Conference on Computer Vision* 2980–2988 (2017).
25. Everingham, M., Van Gool, L., Williams, C. K., Winn, J. & Zisserman, A. The pascal visual object classes (voc) challenge. *Int. J. Comput. Vis.***88**, 303–338 (2010).
26. Lin, T.-Y. et al. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, Proceedings, Part V 13* 740–755 (Springer, 2014).
27. Deng, J. et al. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition* 248–255 (IEEE, 2009).
28. Krasin, I. et al. Openimages: A public dataset for large-scale multi-label and multi-class image classification. *Dataset* **2**, 18 (2017). https://github.com/openimages.
29. Xia, G.-S. et al. Dota: A large-scale dataset for object detection in aerial images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* 3974–3983 (2018).
30. Li, K., Wan, G., Cheng, G., Meng, L. & Han, J. Object detection in optical remote sensing images: A survey and a new benchmark. *ISPRS J. Photogramm. Remote Sens.* **159**, 296–307 (2020).
31. Lam, D. et al. xview: Objects in context in overhead imagery. arXiv:1802.07856 (2018).
32. Zhu, P., Wen, L., Bian, X., Ling, H. & Hu, Q. Vision meets drones: A challenge. arXiv:1804.07437 (2018).
33. Liu, F. et al. R2yolox: A lightweight refined anchor-free rotated detector for object detection in aerial images. *IEEE Trans. Geosci. Remote Sens.* **60**, 1–15 (2022).
34. Shen, J. et al. An anchor-free lightweight deep convolutional network for vehicle detection in aerial images. *IEEE Trans. Intell. Transp. Syst.* **23**, 24330–24342 (2022).
35. Li, Z., Hou, B., Wu, Z., Ren, B. & Yang, C. Fcosr: A simple anchor-free rotated detector for aerial object detection. *Remote Sens.* **15**, 5499 (2023).
36. Lin, T.-Y. et al. Feature pyramid networks for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* 2117–2125 (2017).
37. Liu, S., Qi, L., Qin, H., Shi, J. & Jia, J. Path aggregation network for instance segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* 8759–8768 (2018).
38. Bahdanau, D., Cho, K. & Bengio, Y. Neural machine translation by jointly learning to align and translate. arXiv:1409.0473 (2014).
39. Woo, S., Park, J., Lee, J.-Y. & Kweon, I. S. Cbam: Convolutional block attention module. In *Proceedings of the European Conference on Computer Vision (ECCV)* 3–19 (2018).
40. Liu, Z. et al. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* 10012–10022 (2021).
41. Liu, Y., Shao, Z. & Hoffmann, N. Global attention mechanism: Retain information to enhance channel-spatial interactions. arXiv:2112.05561 (2021).
42. Vaswani, A. et al. Attention is all you need. *Adv. Neural Inf. Process. Syst.* **30**, 56 (2017).
43. Creative Commons. CC0 1.0 Universal (CC0 1.0) Public Domain Dedication (2009, accesssed 21 Sep 2024).

44. Ben93kie. Seadronessee: Drone-based maritime search and rescue operation code examples (2023, accessed 19 Feb 2024). https://github.com/Ben93kie/SeaDronesSee.
45. He, K., Zhang, X., Ren, S. & Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* 770–778 (2016).
46. Xie, S., Girshick, R., Dollár, P., Tu, Z. & He, K. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* 1492–1500 (2017).
47. Kiefer, B. Leveraging metadata for computer vision on unmanned aerial vehicles. Ph.D. thesis, Universität Tübingen (2023).
48. Zhao, B. et al. Heuristic data-driven anchor generation for uav-based maritime rescue image object detection. *Heliyon***10**, 52 (2024).
49. Kiefer, B., Ott, D. & Zell, A. Leveraging synthetic data in object detection on unmanned aerial vehicles. In *2022 26th International Conference on Pattern Recognition (ICPR)* 3564–3571 (IEEE, 2022).
50. Cao, Z., Kooistra, L., Wang, W., Guo, L. & Valente, J. Real-time object detection based on uav remote sensing: A systematic literature review. *Drones* **7**, 620 (2023).
51. Salamí, E., Gallardo, A., Skorobogatov, G. & Barrado, C. On-the-fly olive tree counting using a uas and cloud services. *Remote Sens.* **11**, 316 (2019).
52. Alsamhi, S. H. et al. Computing in the sky: A survey on intelligent ubiquitous computing for uav-assisted 6g networks and industry 4.0/5.0. *Drones* **6**, 177 (2022).
53. Li, F. et al. A remote sensing and airborne edge-computing based detection system for pine wilt disease. *IEEE Access* **9**, 66346–66360 (2021).
54. Yağ, İ & Altan, A. Artificial intelligence-based robust hybrid algorithm design and implementation for real-time detection of plant diseases in agricultural environments. *Biology* **11**, 1732 (2022).
55. Huda, S. A. & Moh, S. Survey on computation offloading in uav-enabled mobile edge computing. *J. Netw. Comput. Appl.* **201**, 103341 (2022).

Author contributions
B.Z. was primarily responsible for the project's main conceptualization, experimentation, and manuscript writing. Y.Z. conducted data analysis, figure editing, and contributed to manuscript writing. R.S. assisted with data analysis, figure editing, and manuscript review. L.Y. contributed to resource management, data validation, and manuscript review. X.Z. and J.L. provided supervision, methodological support, and reviewed the manuscript. All authors reviewed the manuscript.

Competing Interests
The authors declare no competing interests.

Additional information
Correspondence and requests for materials should be addressed to X.Z. or J.L.