sustainability

MDPI

Article

AI-RCAS: A Real-Time Artificial Intelligence Analysis System for Sustainable Fisheries Management

Seung-Gyu Kim, Sang-Hyun Lee and Tae-Ho Im *

Department of Information and Communication Engineering, Hoseo University, 20, Hoseo-ro79beon-gil, Baebang-eup, Asan-si 31499, Republic of Korea; ksg980920@naver.com (S.-G.K.); tkdgus0880@naver.com (S.-H.L.)
* Correspondence: taehoim@hoseo.edu; Tel.: +82-540-9642

Abstract: This study proposes an Artificial Intelligence-based Real-time Catch Analysis System (AI-RCAS) for sustainable fisheries management. The AI-RCAS, implemented on a Jetson board, consists of fish recognition using YOLOv10, tracking with a ByteTrack algorithm optimized for marine environments, and a counting module. Experiments in actual fishing environments showed significant improvements, with species recognition rates of 74–81%. The system supports the efficient operation of the total allowable catch (TAC) system through real-time analysis, addressing the limitations of the existing Electronic Monitoring (EM) systems. However, challenges remain, including object-tracking difficulties and performance issues in unstable marine environments. Future research should focus on optimizing the fishing process, improving video processing, and expanding the dataset for better generalization.

Keywords: electronic monitoring; artificial intelligence; YOLOv10; ByteTrack; total allowable catch; sustainable fisheries

check for updates

Citation: Kim, S.-G.; Lee, S.-H.; Im, T.-H. AI-RCAS: A Real-Time Artificial Intelligence Analysis System for Sustainable Fisheries Management. *Sustainability* **2024**, *16*, 8178. https://doi.org/10.3390/su16188178

Academic Editor: Hossein Bonakdari

Received: 15 August 2024
Revised: 13 September 2024
Accepted: 18 September 2024
Published: 19 September 2024

Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

The global consumption of fishery resources continues to increase. According to the FAO (2024) report, world fisheries production reached a record high of 185.40 million tons in 2022, with 89% (164.60 million tons) consumed as food [1]. Annual per capita fish consumption has significantly increased from 9.1 kg in 1961 to 20.6 kg in 2021, with the growth rate of fish consumption (3.0% annually) far exceeding population growth (1.6% annually) [1]. This rapid increase in demand has led to an expansion of fishing activities, consequently elevating the risk of overfishing. To meet the growing demand, fishing operations have become more intensive, resulting in many species being caught at levels exceeding sustainable thresholds. This poses a serious threat to marine ecosystems. Overfishing, which involves catching fish faster than they can reproduce, causes extensive changes to the structure and function of marine ecosystems [2–7]. The continuous fishing pressure due to increased demand is making it difficult for many fish stocks to recover. Reflecting this situation, the FAO (2024) report indicates that the proportion of biologically sustainable fish stocks decreased by 2.3% compared to 2019, to 62.3% [1]. This demonstrates that the increase in demand and the subsequent overfishing are directly affecting the sustainability of marine ecosystems.

To address these issues, countries worldwide have introduced various policies for sustainable fisheries [8]. Among these, the total allowable catch (TAC) system and Individual Transferable Quotas (ITQs) are prominent. The TAC system sets an annual total allowable catch for specific species, which is effective for overall resource management but does not directly control individual vessel catches [9]. ITQs, based on TAC, allocate quotas to individual fishers or companies [10]. Fisheries that have implemented ITQs have shown a significant reduction in resource collapse risk [11,12]. Accurate catch measurement is essential for the effective operation of these systems [13]. The observer program was introduced for accurate catch measurement [14]. This involves trained observers directly

boarding vessels to record catch amounts, bycatch, and compliance with regulations. However, this method has limitations in terms of personnel costs and difficulty in application to all vessels. In response to these challenges, Electronic Monitoring (EM) systems have gained traction [15]. These systems employ cameras and sensors installed on vessels to monitor fishing activities in real time, providing data comparable to human observers [16]. Despite their advantages, the current EM systems face a significant hurdle: the substantial human and time costs associated with analyzing collected footage. The manual analysis of videos remains a time-consuming and resource-intensive process.

To mitigate these issues, researchers have turned to artificial intelligence, particularly deep learning technologies, for automation. While EM systems enhanced with deep learning have shown promise in increasing efficiency and reducing costs through automated video analysis, they have not fully resolved the time and resource constraints. A notable remaining challenge is the time delay in retrieving and analyzing video from EM systems, which impedes real-time analysis. For instance, vessels engaged in long-term fishing operations of six months or more still require extensive time for footage analysis, even with EM systems in place. This delay hampers the real-time management of the TAC system and swift decision-making processes. In light of these persistent challenges, this study introduces an innovative solution: the Artificial Intelligence-based Real-time Catch Analysis System (AI-RCAS). This system is designed to enable real-time catch measurement directly in the field, addressing the limitations of the current EM systems and paving the way for more efficient and timely fisheries management. The AI-RCAS is designed to analyze catch in real time on-site using a low-power embedded board, specifically the Jetson. This system consists of three main modules: fish recognition, tracking, and counting, which are expected to support the efficient operation of the TAC system and ultimately contribute to sustainable fisheries and marine ecosystem conservation.

The structure of this paper is as follows: Section 2 examines the concept of EM systems, current usage cases, and related research trends. Section 3 explains in detail the structure and operating principles of the proposed AI-RCAS. Section 4 presents the experimental design and results to verify the performance of the AI-RCAS, and discusses its significance and limitations. Finally, Section 5 summarizes the conclusions of the study and suggests future research directions.

2. Electronic Monitoring (EM) Systems: Overview and Research Trends

The history of Electronic Monitoring (EM) systems began in 1999 with the Area "A" crab fishery in British Columbia, Canada. Initially used to monitor gear restrictions and catch control, its application has since expanded and is now used in various fishery management practices worldwide. Globally, 100 EM pilot projects and twelve fully implemented programs have been identified, primarily used in North America, Australia, New Zealand, Europe, and some tropical regions [15]. EM systems offer benefits such as cost-effectiveness, extensive fleet coverage, and detailed recording capabilities of fishing activities and locations. However, EM systems also have limitations including privacy concerns, technical complexity, and issues with analyzing large volumes of data [17–19]. In particular, the manual analysis of the collected video data is a major issue that requires significant time and cost [15]. Recent advancements in computer vision and artificial intelligence technologies are gradually addressing these limitations. Deep learning-based automated image analysis techniques are greatly improving the efficiency and accuracy of EM systems, further enhancing their cost-effectiveness [20].

Recent studies have focused on developing automated species identification and counting systems for EM systems by applying artificial intelligence, particularly deep learning technologies. Among computer vision technologies, object detection models are gaining particular attention. Object detection is a technique that identifies the presence and location of the objects of interest in images or video frames, broadly categorized into One-stage Detectors and Two-stage Detectors. One-stage Detectors, such as YOLO (You Only Look Once) [21] or SSD (Single Shot Detector) [22], predict the location and class

of objects simultaneously, characterized by fast processing speeds. On the other hand, Two-stage Detectors, like Faster R-CNN [23] or Mask R-CNN [24], first propose object regions and then perform classification, generally providing higher accuracy. These models offer high efficiency, enabling real-time processing by predicting object locations and classes through a single network. In EM systems, these object detection models play a crucial role in real-time species identification and location detection.

Huang (2019) developed a system to detect fish in stereo videos using an object detection model based on Faster R-CNN [23,25]. The detected objects were tracked in a 3D space using a Kalman filter, and a method was proposed to readjust detection scores using Kalman filter predictions. Tseng and Kuo (2020) developed a method to detect and segment fish using Mask R-CNN [24,26]. For fish counting, they used time threshold and distance threshold methods. This approach was used to track position changes in the detected fish in consecutive frames to prevent duplicate counting and accurately determine numbers. Qiao et al. (2021) developed a system that detects fish and fishermen frame by frame using a deep learning-based object detection framework and applies a time filter to detect fishing events [27]. They conducted comparative studies on various CNN architectures including ResNet [28], GoogLeNet [29], and DenseNet [30]. This system considered a fishing event when fish and fishermen were simultaneously detected in consecutive frames. Khokher et al. (2022) developed a system to detect and classify target species and bycatch species groups using Convolutional Neural Networks (CNNs) [31]. They also developed a visual tracking system to perform fish counting. This study proposed a method to interpolate fish trajectories in video sections where detections were missed using a correlation tracker.

While previous studies have demonstrated the potential for automated catch monitoring by applying deep learning technologies to EM systems, they mostly assumed high-performance GPU environments, limiting their application in actual fishing environments. Moreover, as mentioned in Section 1, the issue of having to collect footage from EM systems still remains, making real-time analysis and quick decision making difficult. These limitations highlight the need for a new approach that can overcome these challenges in real fishing environments. The following chapter introduces a novel system designed to address these issues and provide real-time catch analysis capabilities in the field.

3. Materials and Methods: AI-RCAS Design and Implementation

3.1. The AI-RCAS Configuration

The core of this study is the Artificial Intelligence-based Real-time Catch Analysis System (AI-RCAS), designed to process video collected from EM systems operating on fishing vessels in real time. The AI-RCAS is implemented on the Jetson board, a low-power embedded board, enabling efficient real-time processing even in the limited power environment of fishing vessels. The AI-RCAS is composed of multiple interconnected modules that work in tandem to achieve accurate and real-time catch analysis. At its heart lies an object recognition module that utilizes a lightweight model based on YOLOv10 [32] to detect fish in the video stream in real time. This module is optimized for performance on low-power embedded systems, allowing for efficient processing without compromising accuracy. Working in concert with the object recognition module is an advanced object-tracking module. This component employs an improved version of the ByteTrack algorithm [33], specifically enhanced to handle the unique challenges of tracking fast-moving fish in dynamic marine environments. The improvements made to the algorithm allow for more robust and consistent tracking, even in situations where fish may temporarily disappear from view or move rapidly across the frame. The final component of the AI-RCAS is the object-counting module. This module operates by defining specific areas within the camera's field of view as counting zones. When a tracked fish object passes through these predefined lines or areas, the system increments the catch count. This approach allows for the accurate measurement of catch while minimizing false positives and negatives. The AI-RCAS is configured as an integral part of the broader EM system. For instance, as illustrated in Figure 1, it can receive video data in real-time streaming from CCTV cameras

filming a predefined fishing area distinguished by lines. These predefined lines serve dual purposes: they represent the boundaries of the actual fishing area and act as reference points for the fish counting module.

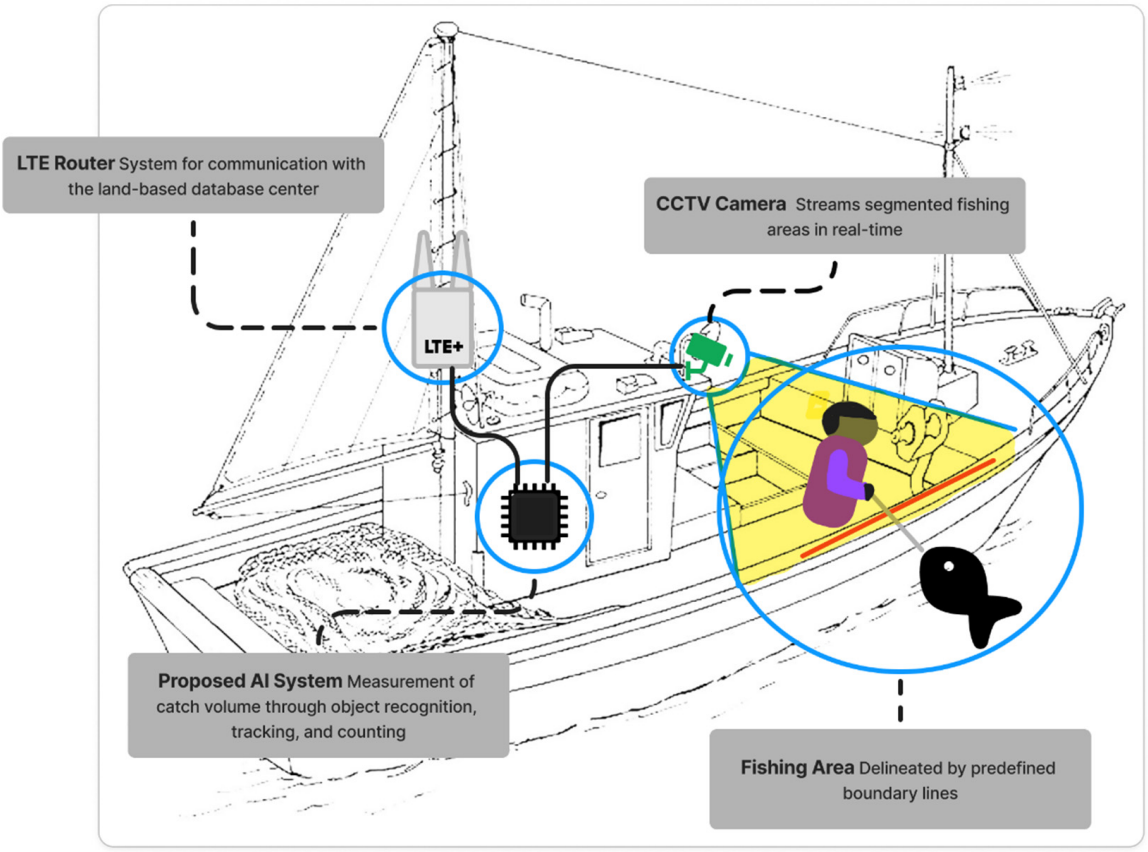


Figure 1. Configuration of the AI-RCAS integrated with the EM system.

3.2. Fish Detection AI Model in AI-RCAS

3.2.1. Dataset Construction for AI-RCAS

For the development of the AI-RCAS's fish detection capabilities, we constructed a dataset for species recognition based on the video data collected through EM (Electronic Monitoring) systems in Korean coastal and offshore fishing environments. Among the species managed under Korea's TAC (total allowable catch) system, four species were selected: cutlassfish, red crab, crab, and skate. The data collection and preprocessing process was as follows: Individual frames were extracted from 1920 × 1080 resolution videos continuously recorded through the EM system and converted into image data. Due to the characteristics of marine fishing environments, noise such as motion blur, defocus blur, and illumination imbalance were observed due to camera movement, water droplet splashes, and lighting changes [34]. As such noise can degrade the performance of object detection models, the images of significantly low quality were removed. After noise processing, the remaining images were labeled with bounding boxes for species. In this process, the accuracy of labeling was ensured through verification by fisheries experts. After labeling was completed, a special preprocessing method was applied to minimize information loss that could occur during image resizing. Conventional YOLO models use methods like Letterboxing or Adaptive image scaling for image resizing [35]. However, these methods have the disadvantage of potentially losing important details of small fish objects. To address this, this study adopted a method of cropping and extracting 640 × 640-pixel areas based on the labeled bounding boxes. This method minimizes pixel information loss due to resizing, preserves the important visual features of small fish

objects, and removes unnecessary background information, allowing the model to focus more on fish objects. No information loss occurred due to this method (Figure 2).



Figure 2. Main labeling and optimized cropping process in the dataset construction stage. The red lines indicate the guiding lines showing the process of cropping from the original image to a size of 640, while the green lines represent the labeling bounding boxes.

After completing the labeling and optimized cropping process, we decided to compare single-class and multi-class models as an approach to optimize the model's performance. This decision was based on the concern that while multi-class models have the advantage of recognizing multiple species simultaneously, performance could be degraded due to the similarity of features between classes or data imbalance. Especially considering that most catches in Korea's TAC system operation target single species, the effectiveness of single-class models was expected to be high. Against this background, we constructed two types of datasets to compare the performance of single-class and multi-class models. We built independent single-class datasets for each species, and simultaneously composed a multi-class dataset including all the species. Through this, we aimed to analyze the advantages and disadvantages of each approach and evaluate their applicability in real fishing environments. The final constructed dataset was divided into Train, Validation, and Test sets in a 7:2:1 ratio. Table 1 shows the composition of single-class datasets for each species and the multi-class dataset.

Table 1. Dataset composition by species.

Dataset Type	Species	Train	Validation	Test	Total
Single-Class	Cutlassfish	2727	779	391	3897
	Skate	2580	737	368	3685
	Crab	2338	668	335	3341
	Red crab	2794	798	400	3992
Multi-Class	All species	10,439	2982	1494	14,915

3.2.2. AI Model Training for AI-RCAS

In this study, we implemented a species recognition model for the AI-RCAS using YOLOv10 [32], the latest object detection architecture. YOLOv10 is an architecture that inherits the advantages of the existing YOLO series while greatly improving performance and efficiency. This model forms a crucial component of the AI-RCAS's fish detection capabilities.

The main features of YOLOv10 are as follows:

1. Consistent Dual Assignment strategy: It proposes a method to train without NMS (Non-Maximum Suppression), improving inference speed. This allows for rich learning signals and efficient inference by simultaneously using one-to-many and one-to-one matching.
2. Efficiency-centered model design: It greatly reduces computational redundancy and increases efficiency through lightweight classification heads, spatial-channel separate downsampling, and rank-based block design.
3. Accuracy-centered model design: It improves model performance by introducing large kernel convolutions and partial self-attention modules.

The structure of YOLOv10 consists of a backbone, a neck, and a head, and an attention mechanism is applied to each part to focus on the critical features. In this study, an experiment is planned using nano, small, and medium models called YOLOv10-n, YOLOv10-s, and YOLOv10-m for performance evaluation under various computational resource environments.

Table 2 illustrates the key differences between YOLOv10 nano, small, and medium models in terms of parameters and FLOPs (Floating Point Operations per Second). The number of parameters indicates the model's complexity and memory requirements, while FLOPs represent computational complexity.

Table 2. Comparison of YOLOv10 model variants.

Model	Parameters (M)	FLOPs (G)
YOLOv10-Nano	2.3	6.7
YOLOv10-Small	7.2	21.6
YOLOv10-Medium	15.4	59.1

As model size increases from nano to medium, both parameters and FLOPs increase significantly. This results in a trade-off between computational efficiency and potential accuracy. The nano model, being lightweight, offers faster processing but potentially lower accuracy. The medium model, while computationally more intensive, has the capacity for higher accuracy. The small model strikes a balance between the two.

These models are expected to perform differently depending on their respective sizes and characteristics. In our study, we evaluated these models in real fishing environments to determine the optimal balance between accuracy and speed for the AI-RCAS system. The hyperparameters used for model training are shown in Table 3, and the same parameters were applied to all three models to ensure a fair comparison.

Table 3. YOLOv10 model training hyperparameters.

Hyper-Parameter	Value
Epochs	200
Batch-size	64
Input image-size	640 × 640 pixel
Workers	8
Learning-rate	0.002
Optimizer	AdamW

All the models are trained in the same hardware environment, and the specifications of the training computers used are shown in Table 4.

Table 4. Training computer specifications.

Component	Specification
CPU	12 cores
Memory	96 GB (including 32 GB shared memory)
GPU	NVIDIA A100 (Nvidia: Santa Clara, CA, USA)

The performance of the models is evaluated based on mAP (mean Average Precision). mAP is a key indicator for evaluating the performance of object detection models, defined as the average of AP (Average Precision) for all the classes [36]. mAP is calculated as follows:

$$\text{mAP} = \frac{1}{\text{n}} \sum_{\text{i}=1}^{\text{n}} \text{AP}_\text{i} \quad (1)$$

where n is the number of classes, and AP_i is the Average Precision of the i-th class. AP is calculated as the area under the precision–recall curve. In this study, we use mAP@0.5:0.95 according to the COCO dataset evaluation protocol [37] to evaluate the model performance. This metric represents the average of AP calculated by varying the IoU (Intersection over Union) threshold from 0.5 to 0.95 at 0.05 intervals.

mAP@0.5:0.95 is a more stringent and comprehensive evaluation method than using a single IoU threshold (e.g., mAP@0.5). This method comprehensively evaluates model performance at various IoU thresholds, allowing for a more robust evaluation of the accuracy of object location prediction. It is particularly suitable for evaluating the overall model performance in tasks where accurate location identification is important, such as species recognition.

However, it should be noted that the mAP performance may not necessarily correspond to the actual counting performance. In particular, considering the applicability in real fishing environments, we will evaluate the balance between processing speed and accuracy on the Jetson board. Through this, we expect to be able to select the most effective model and determine the appropriate approach for implementing the actual EM system.

3.3. Fish Tracking Algorithm in AI-RCAS

Fish tracking is essential for accurate catch measurement in the AI-RCAS. In this study, we implemented the tracking module of the AI-RCAS based on ByteTrack [33], the latest tracking algorithm. ByteTrack uses the BYTE (Bi-directional Young tracklet Embedding) method and improves tracking performance by utilizing detection results with low confidence. ByteTrack achieved the performance of MOTA at 80.3% and IDF1 at 77.3% on the MOT17 dataset, with a speed of 29.6 FPS on an RTX 2080Ti GPU (Nvidia: Santa Clara, CA, USA) [33].

The AI-RCAS requires real-time processing and must operate on low-power embedded systems such as the Jetson board. ByteTrack's high accuracy, real-time processing capability, and simple structure were deemed suitable for meeting these requirements. However, to address the specific challenges of tracking fish in marine environments, we carefully optimized two key parameters of the ByteTrack algorithm. Firstly, we increased the buffer size of ByteTrack. Fish movements in fishing environments are very fast, which can result in losing objects in the middle of tracking. By modifying the algorithm to increase ByteTrack's buffer, we allowed lost objects to be retained longer. This modification enables the system to track fast-moving fish objects for extended periods, improving the overall tracking continuity. Secondly, we adjusted the similarity score threshold. Considering the situation where fish are thrown in actual fishing fields, we increased the model's sensitivity by adjusting this threshold. This allows for stable tracking even when the shape of the fish changes rapidly or similarity decreases due to sudden movements or temporary occlusions. These optimizations were designed to enhance the algorithm's performance in the specific context of fishing environments, where rapid movements, temporary disappearances, and sudden shape changes in fish are common occurrences. The

optimized ByteTrack algorithm was implemented on the Jetson board to ensure real-time processing capabilities in the constrained computational environment of fishing vessels. The specific parameter adjustments and resulting performance differences will be discussed in detail through the experimental results in Section 4. Through this, we will verify how effective the modified ByteTrack algorithm proposed in this study is for the catch analysis system. These parameter adjustments, while seemingly simple, were the result of extensive experimentation to address the unique challenges of marine environments, such as rapid fish movements and varying lighting conditions.

3.4. Fish Counting Algorithm in AI-RCAS

AI-RCAS requires algorithms that effectively count tracked fish populations in order to accurately measure catches. The AI-RCAS counting module adopts a method of counting fish based on predefined fishing grounds. In the initial installation stage of the system, defining the fishing grounds is carried out first. In this process, a virtual line is set by considering the deck structure of the fishing boat and the way the fishing grounds are located. These lines represent the boundaries of the actual fishing grounds and act as a reference point for fish counting. The core principle of the counting algorithm is line crossing detection using the vector dot product. Equation (2) is a mathematical representation of the algorithm.

$$C_t = C_{t-1} + \sum_{i=1}^{n} \delta\left(\text{sign}(\vec{v} \cdot \overrightarrow{u_{i,t-1}}), \text{sign}(\vec{v} \cdot \overrightarrow{u_{i,t}})\right) \quad (2)$$

where

- C_t is the total number of fish at time t;
- C_{t-1} is the total number of fish at time $t-1$ (initial value is 0);
- n is the number of fish objects being tracked in the current frame;
- $\vec{v}$ is the vector of the predefined counting line;
- $\vec{u_{i,t}}$ is the vector from the start point of the line to the center point of the i th fish object at time t;
- $\delta(\cdot,\cdot)$ is the Kronecker delta function, which returns 1 if the two arguments are equal, and 0 if they are different.

Equation (2) determines whether a line crossing has occurred by examining the change in the sign of the dot product between each fish's movement vector and the reference line vector. When a sign change occurs, it is considered that the fish has crossed the reference line, and the count is incremented. This method allows for the efficient verification of line crossing for all the tracked fish in each frame. The computational complexity of the proposed algorithm is O(n), where n is the number of fish being tracked. This is because the calculation is performed once for each fish in every frame. Therefore, it has efficiency suitable for real-time processing. Figure 3 visually illustrates the operating principle of this algorithm. It shows the relationship between the fish's movement trajectory and the reference line, as well as the count increase at the intersection point.

Figure 3 visually illustrates the core principles of this algorithm. Based on these principles, Algorithm 1 presents the pseudocode for the entire fish counting system. This algorithm encompasses the calculation process described in Equation (2) and demonstrates the overall flow of the system. As shown in Algorithm 1, the process begins with the YOLOv10 model detecting fish entities in each video frame. These detected entities are then tracked across frames using the ByteTrack algorithm. The count increases when tracked entities cross a predefined line, with each entity assigned a unique ID to prevent duplicate counting. The following pseudocode details this fish-counting system algorithm:

Algorithm 1 Fish Counting System

```
Input: Video frames, Predefined line
Output: Fish count
procedure FishCountingSystem
    Define fishing area: Set a virtual line considering the deck structure and fishing method
    while video frames are available do
        Frame ← get next frame from video
        DetectedObjects ← YOLOv10 DetectObjects(Frame)
        TrackedObjects ← ByteTrack TrackObjects(DetectedObjects)
        for each object in TrackedObjects do
            if object crosses the predefined line then
                Increment fish count
                Assign unique ID to the object to avoid duplicate counting
            end if
        end for
    end while
end procedure
function YOLOv10 DetectObjects(Frame) ▷ Use YOLOv10 to detect fish objects in the frame
    return List of detected fish objects
end function
function ByteTrack TrackObjects(DetectedObjects) ▷ Use ByteTrack to track the position of each fish object
    return List of tracked fish objects with unique IDs
end function
```

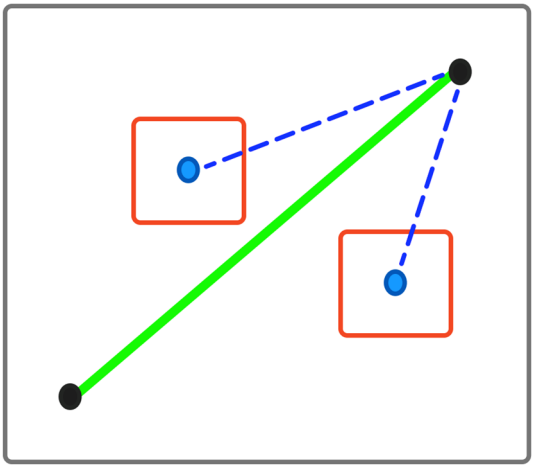


Figure 3. Visual representation of the fish-counting algorithm based on Equation (2). The green line represents the predefined reference line, with the black dots indicating its start and end points. The red boxes depict the bounding boxes of the tracked fish objects, with the blue dots at their centers. The blue dashed lines connect the reference line's start point to the center of each fish object. This illustration demonstrates how the algorithm calculates the dot product between the reference line vector (green) and the vectors to the fish objects (blue dashed) to detect line crossings and count fish.

However, this method heavily relies on the accuracy of line setting, so careful attention is required during the initial setup. Also, periodic calibration may be necessary as the line's position may change due to the movement of the fishing vessel or changes in camera angle. The proposed counting algorithm, combined with the ByteTrack-based tracking algorithm, constitutes a high-accuracy real-time catch analysis system. This is expected to support the effective implementation of the TAC system and ultimately contribute to the realization of sustainable fisheries.

4. Results and Discussion

4.1. Experimental Setup for AI-RCAS Evaluation

In this study, we designed experiments to evaluate the performance of the developed AI-RCAS in actual fishing environments. The main purpose of the experiment was to verify the stability and real-time processing capability of the AI-RCAS in real fishing environments. The experiments were conducted by selecting three representative fishing vessels from each of Korea's West Sea, South Sea, and East Sea. The main target species for each sea area were blue crab in the West Sea, hairtail and skate in the South Sea, and red snow crab in the East Sea. The EM (Electronic Monitoring) system, including the proposed AI system introduced in Section 3, was installed on the selected vessels. The EM system consisted of an NVR, cameras, an LTE router, and the proposed AI system based on the Jetson Xavier NX. During the installation process, virtual lines were set to accurately distinguish the actual fishing area through collaboration with the captain. These lines served as reference points for the AI system to accurately count the catch. After completing the EM system installation, the system was operated during actual fishing operations. The video captured by the cameras during fishing was continuously recorded in the NVR while simultaneously being streamed to the AI system in real time. The AI system performed species recognition and catch counting by processing the input video in real time. The catch data analyzed by the AI system was immediately transmitted to an onshore database server via the LTE router and stored in real time. This allowed the real-time analysis of each vessel's catch status from onshore.

After the experiment, the fishing footage stored in each vessel's NVR was collected. Based on this footage, experts manually analyzed the catch. The actual catch data obtained in this way was compared and analyzed with the catch data measured by the AI system and stored in the database. Finally, the species recognition rate was calculated using the following equation:

$$\text{Species Recognition Rate}(\%) = 100 - (|R_n - C_n|/R_n) \times 100 \quad (3)$$

where R_n is the actual number of target species caught, and C_n is the number estimated by the AI system. Through this process, we objectively evaluated the performance of the proposed AI system and verified its applicability and accuracy in actual fishing environments. We also comprehensively evaluated the stability and reliability of the system by analyzing changes in system performance according to sea areas, species, and various marine environmental conditions. The main equipment and environment used in the experiment are shown in Tables 5 and 6. Figure 4 shows the actual installation photos.

Table 5. AI system specifications.

Component	Specification
Target Board	Nvidia Jetson Xavier NX Development Kit (16 GB) (Nvidia: Santa Clara, CA, USA)
CPU	6-core NVIDIA Carmel ARM®v8.2 64-bit (Nvidia: Santa Clara, CA, USA)
GPU	384-core NVIDIA Volta™ (Nvidia: Santa Clara, CA, USA)
Memory	16 GB LPDDR4x
Storage	1 TB M.2 NVMe SSD
Operating System	Ubuntu 20.04 LTS
CUDA	11.4
TensorRT	8.5.2
OpenCV	4.5.4
Network	LTE Router (Bandwidth: Max 100 Mbps)

Table 6. Input system specifications.

Component	Specification
Camera System	Resolution: 1920 × 1080 (Full HD) Frame Rate: 30 FPS
NVR	Storage Capacity: 4 TB Recording and Streaming (RTSP)



(**a**) (**b**)

Figure 4. (**a**) Installation of the AI system on a fishing vessel. The green circle indicates the proposed AI system, which is housed in an external enclosure designed to withstand marine environments. (**b**) CCTV camera installation on the fishing vessel, highlighted by a red circle.

4.2. Performance Evaluation of AI-RCAS

4.2.1. AI Model Training Results

In this study, we trained and evaluated three variants of the YOLOv10 model (nano, small, and medium) for the AI-RCAS. The training was conducted using the dataset described in Section 3.2.1, which included four species managed under Korea's TAC system: cutlassfish, red crab, crab, and skate. We compared the performance of single-class and multi-class models to determine the most effective approach for species recognition in fishing environments. Figure 5 shows the epoch-wise mAP changes for the single-class and multi-class versions of the YOLOv10-n, YOLOv10-s, and YOLOv10-m models. The graphs demonstrate that in all the models, the mAP values generally increase and stabilize as training progresses. Notably, YOLOv10-s shows the highest mAP values overall and tends to converge the fastest.

Table 7 summarizes the final mAP performance of each model variant for both single-class and multi-class configurations. Analyzing the results in Table 7, the YOLOv10-s model showed the highest mAP performance overall. While there was a significant performance difference from YOLOv10-n, the difference with YOLOv10-m was not large. Based on these results, we set two main hypotheses.

Table 7. mAP performance comparison of YOLOv10 variant models.

Model	Multi-Class	Cutlassfish	Skate	Crab	Red-Crab
YOLOv10-nano	0.833	0.887	0.797	0.754	0.86
YOLOv10-small	0.855	0.89	0.813	0.773	0.861
YOLOv10-Midum	0.849	0.869	0.817	0.782	0.848

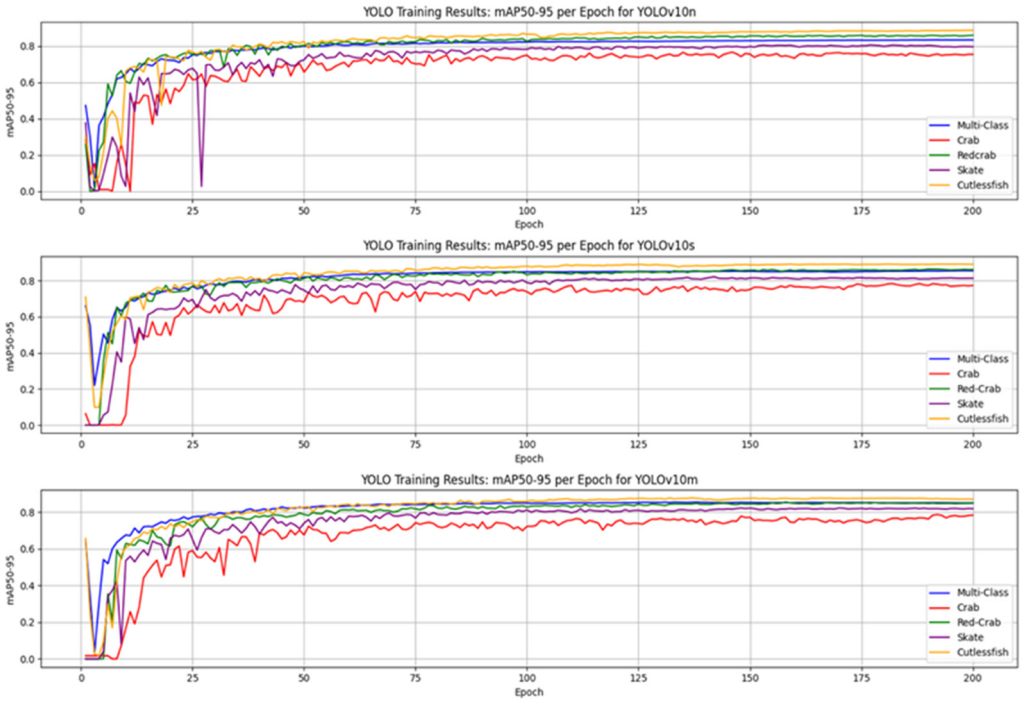


Figure 5. Epoch-wise mAP changes for single-class and multi-class YOLOv10 models.

Our first hypothesis posits that the YOLOv10-s model will exhibit superior counting performance compared to both YOLOv10-n and YOLOv10-m models in real-world applications. This expectation is grounded in the consistently high mAP values the YOLOv10-s model achieved across all the classes. Moreover, taking into account the computational constraints of the Jetson board, we anticipate that the YOLOv10-s model will offer the most effective balance between real-time processing capabilities and accuracy.

Our second hypothesis addresses the comparison between the single-class and multi-class models. While the results in Table 7 do not allow for a direct performance comparison between these model types, we propose that the single-class models will demonstrate better performance on low-power embedded systems such as the Jetson board. This hypothesis is based on the inherent structural simplicity of single-class models, which typically require fewer computational resources and less memory compared to their multi-class counterparts. Given the limited computing power of the Jetson board, we expect this difference in complexity to significantly impact the real-time processing performance. Consequently, we anticipate that the single-class models, with their higher computational efficiency, will prove more effective in actual catch-counting tasks.

To validate these hypotheses, it is crucial to conduct comprehensive experiments on the Jetson board under real fishing conditions. These experiments will allow us to assess the actual performance of each model variant in terms of processing speed, accuracy, and power efficiency. By testing our models in authentic marine environments, we can determine whether the mAP performance observed during training translates effectively to real-world fish counting scenarios. The results of these experiments, which will be presented and analyzed in the following sections, will be instrumental in identifying the most suitable model configuration for our AI-RCAS, ensuring optimal performance in practical fishing applications.

4.2.2. ByteTrack Algorithm Performance Optimization for Marine Environments in AI-RCAS

In this study, we optimized the ByteTrack algorithm's parameters for marine applications to enhance the tracking performance of the AI-RCAS and overcome the limitations of the existing algorithm. The performance of the optimized ByteTrack algorithm was

evaluated in actual fishing environments. The results showed a significant improvement in the average species recognition rate. Compared to the existing algorithm's average recognition rate of 40%, the optimized algorithm achieved an average recognition rate of 81%. This represents a 2.02-fold performance improvement.

This significant performance improvement, from an average recognition rate of 40% to 81%, is primarily attributed to the careful optimization of buffer size and similarity threshold for marine conditions. These parameter adjustments, rather than fundamental algorithm changes, demonstrate the importance of environment-specific tuning in real-world applications. By increasing the buffer size, we were able to track fast-moving fish objects for longer periods, and by adjusting the similarity threshold, we could track objects without losing them even in situations where fish are thrown. This ultimately led to more accurate species recognition. In particular, it enabled more robust tracking in situations common in fishing environments such as sudden movements, temporary occlusions, and fish throwing, greatly improving overall recognition performance.

4.2.3. Model Comparison Results for AI-RCAS

In this study, we compared the performance of the three variant models of YOLOv10 (nano, small, and medium) running on the Jetson Xavier NX board as part of the AI-RCAS. We also analyzed performance differences according to precision by applying FP16 and FP32 formats for each model. The species recognition rate and FPS (Frames Per Second) were used as performance evaluation metrics, and power consumption and current usage were also measured. Table 8 summarizes the performance of each model.

Table 8. Performance comparison of YOLOv10 variant models on Jetson Xavier NX.

Model	Precision	Class Type	Power Consumption (W)	Current Usage (A)	FPS	Species Recognition Rate (%)
YOLOv10-nano	FP16	Single	10.23	1.41	25	81
		Multi	10.25	1.44	23	78
	FP32	Single	11.44	2.32	24~25	81
		Multi	11.51	2.35	22~23	78
YOLOv10-small	FP16	Single	12.51	2.14	16	77
		Multi	12.62	2.2	15	75
	FP32	Single	14.67	2.95	15~16	77
		Multi	14.88	2.99	14~15	75
YOLOv10-medium	FP16	Single	15.17	3.12	13	73
		Multi	15.23	3.17	12	73
	FP32	Single	17.59	3.53	8	68
		Multi	17.6	3.53	7	69

The experimental results show that when using the FP16 format, power consumption and current usage decreased in all the models. In particular, for the YOLOv10-nano model, the current usage significantly decreased from 2.3 A in FP32 to 1.4 A in FP16, and for the YOLOv10-small model, it decreased from 2.9 A to 2.1 A. This demonstrates that the FP16 format provides considerable advantages in terms of power efficiency. In terms of FPS performance, only minimal differences were observed between FP16 and FP32 in the YOLOv10-nano and YOLOv10-small models. However, for the YOLOv10-medium model, using FP16 showed a significant performance improvement of about 5 FPS with 13 FPS compared to 8 FPS in FP32. Notably, in terms of species recognition rate, there was no difference between FP16 and FP32 in the YOLOv10-nano and YOLOv10-small models. However, for the YOLOv10-medium model, using FP16 achieved a recognition rate of 73%, showing a 5 percentage point improvement compared to 68% in FP32. Particularly for the YOLOv10-medium model, the throttling problem that occurred when using the FP32 format was resolved through the FP16 format. This resulted in significant improvements

in both processing speed and recognition rate. However, the 13 FPS performance of the YOLOv10-medium model and the 16 FPS performance of the YOLOv10-small model still fall short of the 30 FPS of the input video, confirming that data loss due to frame dropping is occurring. The YOLOv10-nano model showed a processing speed of 25 FPS in the FP16 format, demonstrating performance closest to the frame rate of the input video among the three models. This suggests that the data loss due to frame dropping can be minimized.

These experimental results clearly show the impact of model size, complexity, and computational precision on the actual performance, power consumption, and current usage in embedded environments like the Jetson Xavier NX. In particular, FPS performance was found to have a significant impact on the overall performance of the species recognition system.

4.2.4. Species-Specific Performance of Optimal AI-RCAS Model

We selected the YOLOv10-nano model (FP16), which showed the best performance, to evaluate the recognition rate for each species in the AI-RCAS. Since the performance difference between single-class and multi-class models was minimal, we conducted experiments for both cases. Table 9 shows the species-specific performance of this model.

Table 9. Species-specific performance of YOLOv10-nano model (FP16).

Species	Single-Class SRR (%)	Multi-Class SRR (%)
Cutlassfish	81%	78%
Red-Crab	77%	76%
Skate	79%	78%
Crab	74%	74%

Note: SSR stands for Species Recognition Rate.

Looking at the species-specific performance, in both model configurations, the recognition rate for cutlassfish was the highest, followed by skate, red crab, and blue crab. The performance difference between the single-class and multi-class models was minimal, about 1 percentage point for each species.

4.3. Discussion of AI-RCAS Limitations and Future Improvements

The AI-RCAS proposed in this study showed significant performance improvements in actual fishing environments. However, while analyzing the experimental results, several important limitations and areas for improvement were identified. Firstly, difficulties in object tracking occurred due to operator interference. As the current system was developed to fit the existing fishing environments, the problem of operators' movements frequently obscuring fish objects in the camera's view occurred. This acted as a major factor hindering the continuity of object tracking. Such issues could be improved through the optimization of the fishing process, and future research will need to develop a workflow that is advantageous for securing camera views through collaboration with fishing experts. Secondly, performance degradation issues occurred due to marine environments. In situations with heavy waves, camera footage experienced shaking due to vessel movement, resulting in a surge in data processing. This led to decreased FPS (Frames Per Second) performance and consequently lowered species recognition rates. To overcome this limitation, it seems necessary to improve FPS performance through video processing acceleration techniques. Thirdly, there was a problem with the lack of diversity and scale in the dataset. The current dataset is limited to specific species and environments, which may limit generalization performance for various marine environments and species. In particular, it was confirmed that more diverse and larger datasets are essential to increase the recognition accuracy due to the fast movement of fish and the influence of the surrounding lights. Also, to more accurately verify the performance of the multi-class models, it was found necessary to build datasets for more species than currently available. To overcome these limitations, cooperative data collection and dataset expansion seem necessary.

These limitations are important factors to consider when applying the system proposed in this study in practice. Future research will need to concretize ways to overcome these limitations and further improve the robustness and reliability of the system through long-term experiments in actual fishing environments. Through this, more effective and reliable catch analysis will be possible, and it will greatly contribute to sustainable fisheries management and marine ecosystem conservation.

5. Conclusions

This study proposed an Artificial Intelligence-based Real-time Catch Analysis System (AI-RCAS) for sustainable fisheries. In a situation where the global consumption of fishery resources is increasing, accurate and efficient catch measurement is essential for marine ecosystem conservation. To this end, this study presented a new approach utilizing deep learning technology in the form of the AI-RCAS.

The AI-RCAS is based on the Jetson Xavier NX board and consists of three main modules: fish recognition using YOLOv10, fish tracking using a ByteTrack algorithm optimized for marine environments, and a counting module based on predefined lines. The experimental results in actual fishing environments showed that the optimized ByteTrack algorithm in the AI-RCAS significantly increased the average species recognition rate from 40% to 81%. Additionally, the YOLOv10-nano model (FP16) demonstrated optimal performance on the Jetson board, achieving a processing speed of 25 FPS and species recognition rates of 74–81% depending on the species. The main significance of the AI-RCAS is as follows: First, it can support the efficient operation of the TAC (total allowable catch) system through real-time catch analysis. This is expected to greatly contribute to the sustainable management of fishery resources. Second, through real-time processing using a low-power embedded system, the AI-RCAS has solved the problems of manpower and time costs, which were the limitations of the existing EM (Electronic Monitoring) systems. Third, by applying AI technology to the fisheries sector, it has contributed to promoting the digital transformation of fishery resource management. However, the AI-RCAS also has several limitations. First, difficulties in object tracking due to operator movements were observed. Second, unstable marine environments (e.g., waves and lighting changes) affected system performance. Third, the current dataset is limited to specific species and environments, which may limit generalization ability to various situations.

To overcome these limitations, we propose the following directions for future research and improvement of the AI-RCAS: First, a workflow advantageous for securing camera views should be developed through the optimization of the fishing process. Second, it is necessary to improve FPS performance in unstable marine environments through video processing acceleration techniques. Third, a more diverse and larger-scale dataset needs to be built through collaborative data collection. Additionally, the robustness and reliability of the AI-RCAS should be further improved through long-term experiments.

In conclusion, the AI-RCAS proposed in this study can be an important tool for realizing sustainable fisheries. If this technology further develops and is widely applied in the future, it is expected that the efficiency and accuracy of fishery resource management will be greatly improved. Furthermore, this can contribute not only to simple catch analysis but also to marine ecosystem conservation, climate change response, and strengthening food security. Therefore, continuous research and investment in the convergence of AI technology and the fisheries sector, as exemplified by the AI-RCAS, will be necessary in the future.

Author Contributions: Conceptualization, T.-H.I.; methodology, T.-H.I.; software, S.-G.K.; validation, S.-G.K. and S.-H.L.; formal analysis, S.-G.K. and T.-H.I.; investigation, S.-G.K. and T.-H.I.; resources, S.-G.K. and T.-H.I.; data curation, S.-G.K. and S.-H.L.; writing—original draft preparation, S.-G.K.; writing—review and editing, T.-H.I.; visualization, S.-G.K.; supervision, T.-H.I.; project administration, T.-H.I.; funding acquisition, T.-H.I. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the Korea Institute of Marine Science & Technology Promotion (KIMST) funded by the Ministry of Oceans and Fisheries (20210499 and 20183084). This work was partly supported by the Institute of Information & Communications Technology Planning & Evaluation (IITP)-Innovative Human Resource Development for Local Intellectualization program grant funded by the Korean government (MSIT) (IITP-2024-RS-2024-00436765).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data analyzed in the study within the article; further inquiries can be directed to the corresponding author.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. FAO. *The State of World Fisheries and Aquaculture 2024: Towards Blue Transformation*; FAO: Rome, Italy, 2024. Available online: https://openknowledge.fao.org/items/3bffafd3-c474-437b-afd4-bb1182feeea6 (accessed on 21 June 2024).
2. Jackson, J.B.C.; Kirby, M.X.; Berger, W.H.; Bjorndal, K.A.; Botsford, L.W.; Bourque, B.J.; Bradbury, R.H.; Cooke, R.; Erlandson, J.; Estes, J.A.; et al. Historical overfishing and the recent collapse of coastal ecosystem. *Science* **2001**, *293*, 629–637. [CrossRef] [PubMed]
3. Shakouri, B.; Yazdi, S.K.; Fashandi, A. Overfishing. In Proceedings of the 2010 2nd International Conference on Chemical, Biological and Environmental Engineering, Cairo, Egypt, 2–4 November 2010; IEEE: Piscataway, NJ, USA; pp. 229–234.
4. Coll, M.; Libralato, S.; Tudela, S.; Palomera, I.; Pranovi, F. Ecosystem overfishing in the ocean. *PLoS ONE* **2008**, *3*, e3881. [CrossRef] [PubMed]
5. Marine Stewardship Council. What Is Overfishing. Available online: https://www.msc.org/what-we-are-doing/oceans-at-risk/overfishing (accessed on 15 June 2024).
6. World Wildlife Fund. What Is Overfishing? Facts, Effects and Overfishing Solutions. Available online: https://www.worldwildlife.org/threats/overfishing (accessed on 15 June 2024).
7. Frontiers in Marine Science. End Overfishing and Increase the Resilience of the Ocean to Climate Change. 2018. Available online: https://www.frontiersin.org (accessed on 15 June 2024).
8. Marchal, P.; Andersen, J.L.; Aranda, M.; Fitzpatrick, M.; Goti, L.; Guyader, O.; Haraldsson, G.; Hatcher, A.; Hegland, T.J.; Le Floc'H, P.; et al. A comparative review of fisheries management experiences in the European Union and in other countries worldwide: Iceland, Australia, and New Zealand. *Fish Fish.* **2016**, *17*, 803–824. [CrossRef]
9. Beddington, J.R.; Agnew, D.J.; Clark, C.W. Current problems in the management of marine fisheries. *Science* **2007**, *316*, 1713–1716. [CrossRef]
10. Costello, C.; Gaines, S.D.; Lynham, J. Can catch shares prevent fisheries collapse? *Science* **2008**, *321*, 1678–1681. [CrossRef]
11. Grafton, R.Q.; Squires, D.; Fox, K.J. Private property and economic efficiency: A study of a common-pool resource. *J. Law Econ.* **2000**, *43*, 679–713. [CrossRef]
12. Motu Economic and Public Policy Research. New Zealand's Quota Management System: A History of the First 20 Years. 2007. Available online: https://www.motu.org.nz (accessed on 15 June 2024).
13. National Research Council. *Improving the Collection, Management, and Use of Marine Fisheries Data*; National Academies Press: Washington, DC, USA, 2000. Available online: https://nap.nationalacademies.org/catalog/9969/improving-the-collection-management-and-use-of-marine-fisheries-data (accessed on 15 June 2024).
14. Brooke, S.G. Federal fisheries observer programs in the United States: Over 40 years of independent data collection. *Mar. Fish. Rev.* **2012**, *74*, 1–25. [CrossRef]
15. van Helmond, A.T.M.; Mortensen, L.O.; Plet-Hansen, K.S.; Ulrich, C.; Needle, C.L.; Oesterwind, D.; Kindt-Larsen, L.; Catchpole, T.; Mangi, S.; Zimmermann, C.; et al. Electronic monitoring in fisheries: Lessons from global experiences and future opportunities. *Fish Fish.* **2020**, *21*, 162–189. [CrossRef]
16. Emery, T.J.; Noriega, R.; Williams, A.J.; Larcombe, J.; Nicol, S.; Brock, D. The use of electronic monitoring within tuna longline fisheries: Implications for international data collection, analysis and reporting. *Rev. Fish Biol. Fish.* **2019**, *29*, 861–879. [CrossRef]
17. Bartholomew, D.C.; Mangel, J.C.; Alfaro-Shigueto, J.; Pingo, S.; Jimenez, A.; Godley, B.J. Remote electronic monitoring as a potential alternative to on-board observers in small-scale fisheries. *Biol. Conserv.* **2018**, *219*, 35–45. [CrossRef]
18. Michelin, M.; Elliott, M.; Bucher, M.; Zimring, M.; Sweeney, M. *Catalyzing the Growth of Electronic Monitoring in Fisheries*; California Environmental Associates and The Nature Conservancy: Sacramento, CA, USA, 2020; (Update 2020). Available online: https://fisheriesem.com (accessed on 2 August 2024).
19. EDF Fishery Solutions Center. Fisheries Monitoring Roadmap. Available online: https://fisherysolutionscenter.edf.org/resources/fisheries-monitoring-roadmap (accessed on 21 June 2024).
20. French, G.; Fisher, M.H.; Mackiewicz, M.; Needle, C.L. Convolutional neural networks for counting fish in fisheries surveillance video. In Proceedings of the Machine Vision of Animals and Their Behaviour (MVAB), Swansea, UK, 10 September 2015; BMVA Press: Durham, UK; pp. 7.1–7.10.

21. 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 (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 779–788.
22. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.Y.; Berg, A.C. SSD: Single Shot MultiBox Detector. In Proceedings of the European Conference on Computer Vision (ECCV), Amsterdam, The Netherlands, 11–14 October 2016; pp. 21–37.
23. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In Proceedings of the Advances in Neural Information Processing Systems (NIPS), Montreal, QC, Canada, 7–12 December 2015; pp. 91–99.
24. He, K.; Gkioxari, G.; Dollár, P.; Girshick, R. Mask R-CNN. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017; pp. 2961–2969.
25. Huang, T.-W.; Wang, D.; Tuck, G.N.; Punt, A.E.; Gerner, M. Automatic Video Analysis for Electronic Monitoring of Fishery Activities. *ICES J. Mar. Sci.* **2020**, *77*, 1367–1378.
26. Tseng, C.-H.; Kuo, Y.-F. Detecting and Counting Harvested Fish and Identifying Fish Types in Electronic Monitoring System Videos Using Deep Convolutional Neural Networks. *ICES J. Mar. Sci.* **2020**, *77*, 1367–1378. [CrossRef]
27. Qiao, M.; Wang, D.; Tuck, G.N.; Little, L.R.; Punt, A.E.; Gerner, M. Deep Learning Methods Applied to Electronic Monitoring Data: Automated Catch Event Detection for Longline Fishing. *ICES J. Mar. Sci.* **2021**, *78*, 25–35. [CrossRef]
28. 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 (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
29. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going Deeper with Convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 7–12 June 2015; pp. 1–9.
30. Huang, G.; Liu, Z.; van der Maaten, L.; Weinberger, K.Q. Densely Connected Convolutional Networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 4700–4708.
31. Khokher, M.R.; Little, L.R.; Tuck, G.N.; Smith, D.V.; Qiao, M.; Devine, C.; O'Neill, H.; Pogonoski, J.J.; Arangio, R.; Wang, D. Early lessons in deploying cameras and artificial intelligence technology for fisheries catch monitoring: Where machine learning meets commercial fishing. *Can. J. Fish. Aquat. Sci.* **2022**, *79*, 257–266. [CrossRef]
32. Wang, A.; Chen, H.; Liu, L.; Chen, K.; Lin, Z.; Han, J.; Ding, G. YOLOv10: Real-Time End-to-End Object Detection. *arXiv* **2024**, arXiv:2405.14458.
33. Zhang, Y.; Sun, P.; Jiang, Y.; Yu, D.; Weng, F.; Yuan, Z.; Luo, P.; Liu, W.; Wang, X. ByteTrack: Multi-Object Tracking by Associating Every Detection Box. *arXiv* **2021**, arXiv:2110.06864.
34. Nair, J.J.; Topiwala, P.N. Image Noise Modeling, Analysis and Estimation. In Proceedings of the SPIE 7744, Visual Information Processing XIX, Orlando, FL, USA, 7 May 2010; p. 77440N.
35. Kopf, J.; Kienzle, W.; Drucker, S.; Kang, S.B. Quality Prediction for Image Completion. *ACM Trans. Graph.* **2012**, *31*, 131. [CrossRef]
36. Padilla, R.; Netto, S.L.; da Silva, E.A.B. A Survey on Performance Metrics for Object-Detection Algorithms. In Proceedings of the 2020 International Conference on Systems, Signals and Image Processing (IWSSIP), Niterói, Brazil, 1–3 July 2020; pp. 237–242.
37. Lin, T.Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; Zitnick, C.L. Microsoft COCO: Common Objects in Context. In *Computer Vision—ECCV 2014*; Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T., Eds.; Lecture Notes in Computer Science; Springer: Cham, Switzerland, 2014; Volume 8693, pp. 740–755.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.