AI

MDPI

Article

Comparison of CNN-Based Architectures for Detection of Different Object Classes

Nataliya Bilous [1,2,*], Vladyslav Malko [1], Marcus Frohme [2,*] and Alina Nechyporenko [1,2]

[1] Computer Science Faculty, Kharkiv National University of Radio Electronics, 14 Nauky Ave., 61166 Kharkiv, Ukraine; vladyslav.malko@nure.ua (V.M.); nechyporenko@th-wildau.de (A.N.)
[2] Division Molecular Biotechnology and Functional Genomics, Technical University of Applied Sciences Wildau, 1 Hochschulring, 15745 Wildau, Germany
* Correspondence: nataliya.bilous@nure.ua (N.B.); mfrohme@th-wildau.de (M.F.); Tel.: +380-679190676 (N.B.)

Abstract: (1) Background: Detecting people and technical objects in various situations, such as natural disasters and warfare, is critical to search and rescue operations and the safety of civilians. A fast and accurate detection of people and equipment can significantly increase the effectiveness of search and rescue missions and provide timely assistance to people. Computer vision and deep learning technologies play a key role in detecting the required objects due to their ability to analyze big volumes of visual data in real-time. (2) Methods: The performance of the neural networks such as **You Only Look Once** (YOLO) v4-v8, Faster R-CNN, Single Shot MultiBox Detector (SSD), and EfficientDet has been analyzed using COCO2017, SARD, SeaDronesSee, and VisDrone2019 datasets. The main metrics for comparison were mAP, Precision, Recall, F1-Score, and the ability of the neural network to work in real-time. (3) Results: The most important metrics for evaluating the efficiency and performance of models for a given task are accuracy (mAP), F1-Score, and processing speed (FPS). These metrics allow us to evaluate both the accuracy of object recognition and the ability to use the models in real-world environments where high processing speed is important. (4) Conclusion: Although different neural networks perform better on certain types of metrics, YOLO outperforms them on all metrics, showing the best results of mAP-0.88, F1-0.88, and FPS-48, so the focus was on these models.

Keywords: deep learning; object detection; neural network; YOLO; SSD; EfficientDet

Citation: Bilous, N.; Malko, V.; Frohme, M.; Nechyporenko, A. Comparison of CNN-Based Architectures for Detection of Different Object Classes. *AI* **2024**, *5*, 2300–2320. https://doi.org/10.3390/ai5040113

Academic Editor: Arslan Munir

Received: 16 September 2024
Revised: 25 October 2024
Accepted: 4 November 2024
Published: 11 November 2024

1. Introduction

The detection of people and technical objects in various critical situations, such as natural disasters and military operations, plays a key role in search and rescue operations and ensures the safety of civilians. For example, in floods, recognition technologies help find people who may be cut off from safe areas or stranded in flooded areas, allowing rescue services to quickly help. During warfare, such systems are used to search for the wounded, locate soldiers in a combat zone, and recognize enemy vehicles to help plan operations and protect civilians. On roads, recognition systems can detect pedestrians in dangerous areas, preventing accidents at intersections and ensuring safety for both pedestrians and drivers. Fast and accurate search for people and equipment significantly increases the efficiency of rescue missions and enables timely assistance to victims. Modern computer vision and deep learning technologies, with their ability to analyze large amounts of visual data in real-time, are becoming indispensable tools in these processes. In recent years, there has been significant progress in the development of neural networks, which has opened new possibilities for image analysis and processing. This study aims to comparatively analyze several state-of-the-art neural networks to identify the most productive models for object recognition tasks in complex environments. The COCO2017 [1], SARD [2], SeaDronesSee [3], and VisDrone2019 [4] datasets were used as data sources, which provide diverse and

realistic scenarios for testing the models. The neural networks considered for the study were **You Only Look Once** (YOLO)v4-v8 [5–10], Faster R-CNN [11], SSD (Single Shot MultiBox Detector) [12], and EfficientDet [13]. The main criteria for comparison were the average detection accuracy (mAP), precision (Precision), completeness (Recall), F1-Score, and real-time ability of the neural network. These metrics allow for a comprehensive evaluation of both object recognition accuracy and model performance in real-world environments where high processing speed is crucial. The results of the comparative analysis showed that different neural networks exhibit different levels of performance depending on the type of metric. However, the YOLO family of models (v4–v8) showed superior performance in all key metrics. YOLO is characterized by high accuracy, fast processing speed, and robustness to noise and interference, making these models most suitable for use in real-world scenarios requiring high performance. This research highlights the importance of applying neural networks to object recognition tasks in complex and critical environments. State-of-the-art models such as YOLO can greatly improve the efficiency of search and rescue operations and provide higher civilian safety. Further development and deployment of these technologies open new opportunities for automation and efficiency improvements in various fields, such as agriculture, autonomous transport, and environmental monitoring. The aim of the research is to compare neural network models to identify the best one for detecting people and technical objects in various critical situations (floods, military operations). The subject of the research is the neural network models YOLOv4-v8, Faster R-CNN, and SSD. The object of the research is the metrics for evaluating the effectiveness of neural network models, such as mean accuracy of detection (mAP), precision (Precision), completeness (Recall), F1-Score, and the ability of the neural network to work in real-time. The relevance of the research stems from the growing need for efficient detection and monitoring systems that can operate in real-time in various environments, including water and road scenes. Such systems are important for security, rescue operations, autonomous vehicles, and other critical applications.

2. Review of the Literature

Object detection in images using neural networks has become one of the key challenges in computer vision. In recent years, many researchers have aimed at developing and improving deep learning methods for recognizing different classes of objects. Let us review the main advances and approaches that formed the basis of our research. The YOLO model [5], proposed by Joseph Redmon and colleagues in 2016, was a breakthrough in object detection. YOLO is characterized by high-speed image processing and real-time capability, which makes it particularly useful for tasks requiring fast responses. Subsequent versions (YOLOv2, YOLOv3, YOLOv4, and onward to YOLOv8) have significantly improved accuracy and robustness to various lighting conditions and noise. Studies have shown that YOLO works effectively in a variety of applications including traffic monitoring [14–16], security [17,18], and autonomous [19,20] systems. Faster R-CNN [11], proposed by Shao Jin and colleagues in 2015, is an improved version of R-CNN and Fast R-CNN. The main difference between Faster R-CNN and R-CNN is the inclusion of the Region Proposal Network (RPN) in Faster R-CNN, which streamlines the generation of region proposals and significantly improves detection speed. In contrast, R-CNN relies on a slower, separate region proposal step. The SSD model [12] proposed by Wei Liu and colleagues in 2016 also focuses on high-speed image processing. SSD utilizes multi-resolution feature maps to detect objects at different scales, achieving high accuracy with low latency. SSD is widely used in tasks related to autonomous driving and robotics. EfficientDet [13], proposed in 2020, is a family of models based on EfficientNet. The main focus of EfficientDet is on computational efficiency and resource efficiency, making it ideal for use in computationally constrained environments. EfficientDet shows high accuracy and performance, especially in the context of mobile and embedded systems.

Saieshan Reddy [21], in his research, has conducted a comprehensive analysis of three convolutional neural network (CNN)-based object detection algorithms for vehicle

detection tasks. The algorithms considered include Faster R-CNN, YOLO v3, and SSD. Reddy's results show that SSD outperforms other algorithms in terms of average accuracy (mAP = 0.92) and detection time (0.5 s per image), despite higher values of average loss. YOLO v3 also performs well, striking a balance between accuracy and speed (mAP = 0.81 and 1.16 s per image). Faster R-CNN, although it has high accuracy (mAP = 0.76), is inferior in terms of processing speed (5.1 s per image), which limits its real-time applications. Chaoyue Sun et al. [22] presented the MRD-YOLO model for object detection in challenging road conditions using multispectral images. Current visible light-based algorithms often face problems of misses and false positives in poor visibility. To address these problems, multispectral images combining RGB and infrared channel data were used. MRD-YOLO includes a BIC-Fusion module for efficient information fusion, a SAConv module for improved detection of objects of different scales, and an AIFI framework for better utilization of semantic information. Experiments on FLIR_Aligned and M3FD datasets have shown that MRD-YOLO outperforms existing algorithms in terms of detection accuracy in challenging road conditions, avoiding misses and false positives (Figure 1). Thus, MRD-YOLO significantly improves object detection in autonomous vehicle systems by providing reliable detection in a variety of challenging environments.

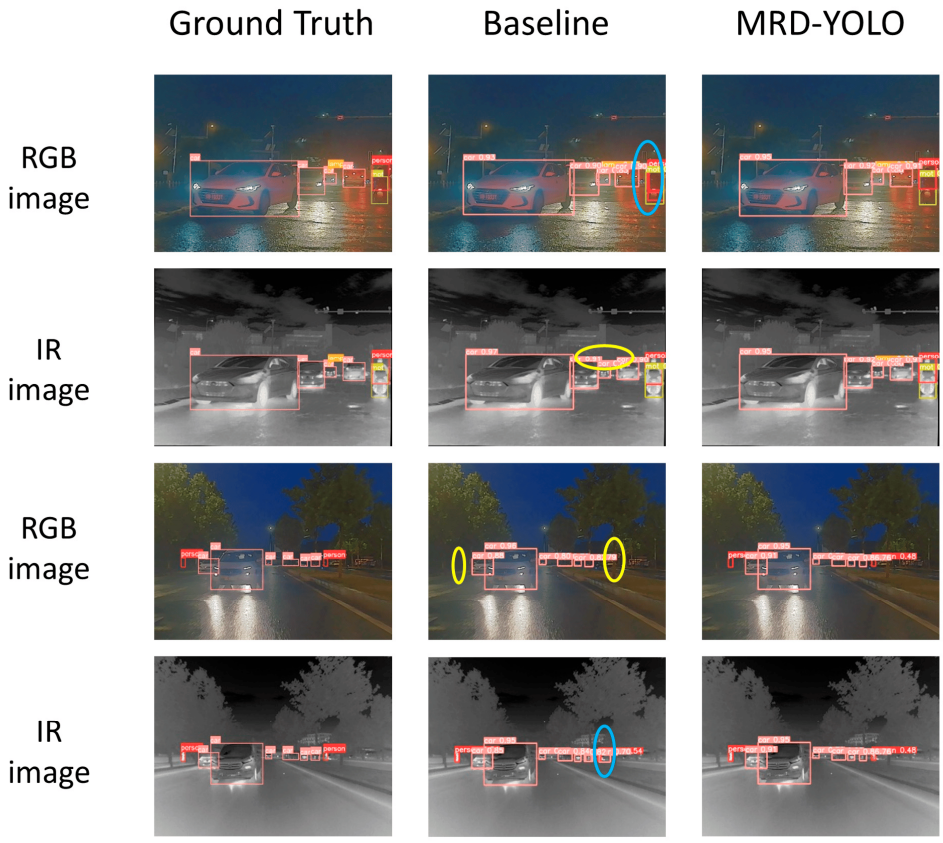

Figure 1. Visual comparison of detection results. Missed targets are marked with yellow ellipses, whereas false detections are indicated with blue ellipses.

Yanyan Dai et al. [23] proposed a novel approach to object detection and tracking for autonomous vehicles by combining YOLOv8 and LiDAR. Traditional methods based on computer vision often encounter difficulties under varying illumination and complex environments. To improve results, the authors combined YOLOv8, which provides high-accuracy object detection in RGB images, with LiDAR, which provides 3D distance information. The paper presents a solution for calibrating data between sensors, filtering ground points from LiDAR clouds, and managing computational complexity. A filtering algorithm trained on YOLOv8, a calibration method for converting 3D coordinates to pixel

coordinates, and object clustering based on the merged data are developed. Experiments on an Agilex Scout Mini robot with Velodyne LiDAR and Intel D435 camera confirmed the effectiveness of the approach, improving the performance and reliability of object detection and tracking systems. Nataliya Bilous et al. [24] developed a method to identify and compare human postures and exercises based on 3D joint coordinates. This method is intended to analyze human movements, especially in medicine and sports, where it is important to evaluate the correctness of exercise performance. The method developed by the authors uses pose descriptions in the form of logical statements and is robust to data errors, regardless of the angle of view and human body proportions. Joint coordinates are adjusted according to bone length, which improves positioning accuracy and eliminates the need for outlier processing (Figure 2).

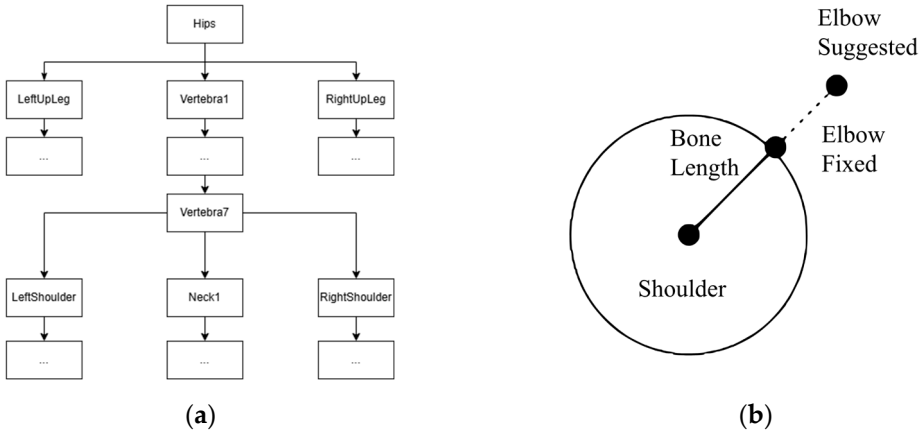

Figure 2. Tree diagram of joints (**a**). Correction of the position of a joint (**b**).

To correct errors, a graph averaging method is applied along each axis, which groups consecutive points so that the difference between the maximum and minimum values does not exceed the error. The groups are then filtered, leaving only those where both points are smaller or both are larger. The proposed method requires only a modern smartphone and does not impose restrictions on the way the exercise video is captured. The method has been tested on UTKinect-Action3D, SBU-Kinect-Interaction v2.0, Florence3D, JHMDB, and NTU RGB+D 120 datasets, as well as on our own dataset collected using ARKit and BlazePose. The results showed high motion tracking accuracy and robustness to data errors, making the method suitable for use in areas such as fitness, rehabilitation, and health monitoring. Daud Khan et al. [25] proposed the integration of YOLOv3 and MobileNet SSD algorithms to improve real-time object detection. The research aims to overcome the problems of blur, noise, and rotational distortion that affect detection accuracy in real-world environments. YOLOv3 provides high speed and accuracy by detecting multiple objects in a single image, while MobileNet SSD balances speed and accuracy on resource-constrained devices. The authors attempt to improve the accuracy of object localization and the performance of algorithms in various applications such as augmented reality, robotics, surveillance systems, and autonomous vehicles. The integration of these technologies improves safety, provides immediate response to threats, and allows robots to better interact with their environment. The authors also compared lightweight versions of YOLOv3 and YOLOv4 in terms of daytime and nighttime accuracy. Experiments showed significant improvements in the performance and reliability of object detection systems when these algorithms were integrated. Further optimization including hardware acceleration and object tracking techniques is recommended to improve real-time detection capabilities. In their research, Dinesh Suryavanshi et al. [26] compare the YOLO and SSD algorithms for real-time object detection. The objective is to carry out a comparative analysis of the accuracy, speed, and efficiency of two popular deep learning-based algorithms. The COCO (Common Objects in Context) dataset was employed for the evaluation of performance.

The YOLO algorithm is distinguished by its high speed and capacity to detect multiple objects in a single image in a single iteration of the network, rendering it well-suited for real-time, performance-intensive tasks such as traffic monitoring and autonomous vehicles. YOLO is distinguished by its high accuracy and low background error, rendering it an effective choice for a diverse array of applications. In contrast, the SSD (Single Shot Detector) algorithm is capable of detecting multiple objects in a single iteration. However, in contrast to YOLO, it employs a more intricate architectural design comprising multiple convolutional layers, with the objective of enhancing the precision of the detection process. The SSD algorithm is particularly effective for tasks that require more accurate object recognition, such as forensic analysis and landmark detection. The research revealed that YOLO exhibits superior speed and real-time performance compared to SSD; however, SSD displays enhanced accuracy in complex environments. The selection of an appropriate algorithm is dependent upon the specific accuracy and speed requirements of the intended application. Therefore, YOLO is more appropriate for applications that necessitate rapid detection, whereas SSD may be preferable for tasks that demand more precise object recognition. Akshatha K.R. et al. [27] investigated the use of Faster R-CNN and SSD algorithms for detecting people in aerial thermal images. The main objective of the research is to evaluate the performance of these algorithms under conditions where objects (people) are small in size and images have low resolution, which often causes problems for standard object detection algorithms. The research utilized two standard datasets, OSU (Ohio State University) Thermal Pedestrian [28] and AAU PD-T (Aalborg University person detection dataset) [29]. The algorithms were tuned with different network architectures such as ResNet50 [30], Inception-v2 [31], and MobileNet-v1 [32] to improve detection accuracy and speed (Figure 3). The results showed that the Faster R-CNN model with ResNet50 architecture achieved the highest accuracy, showing an average detection accuracy (mAP) of 100% for the OSU test dataset and 55.7% for the AAU PD-T dataset.

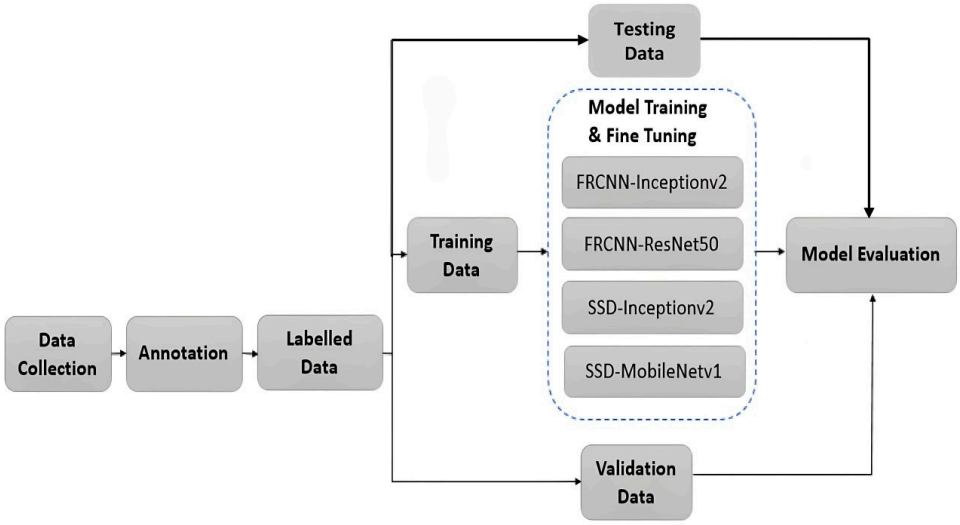

Figure 3. Proposed approach to compare various deep learning-based human detection models.

At the same time, the SSD with MobileNet-v1 architecture delivered the fastest detection rate of 44 frames per second (FPS) on the NVIDIA GeForce GTX 1080 GPU. Algorithm parameters such as anchor scaling and network steps were fine-tuned to improve performance. These tweaks resulted in a 10% increase in mAP accuracy for the Faster R-CNN ResNet50 model and a 3.5% increase for the Inception-v2 SSD. Thus, the research shows that Faster R-CNN is more suitable for tasks requiring high accuracy, while SSD is better suited for applications requiring high real-time detection rates. Heng Zhang et al. [33] presented an improved Faster R-CNN algorithm for real-time vehicle detection based on frame difference and spatiotemporal context. Unlike the original approach that uses anchors, the new method uses the frame difference to highlight regions of interest. The

spatial context enhances the expression of target information, while the temporal context improves the detection performance. Experiments showed the high performance of the proposed method with low sensitivity to background changes. Shengxin Gao et al. [34] investigated the performance of a pedestrian detection model based on the Faster R-CNN algorithm. The model was trained on the Caltech Pedestrian dataset [35] and showed an average precision (AP) of 51.9%, indicating high accuracy in pedestrian detection tasks. The processing time per image was only 0.07 s, making the model suitable for real-time applications. The model also consumes only 158 MB of memory, making it easy to implement on resource-constrained devices. The research emphasizes that even with relatively low AP, the model demonstrated fast processing speed, making it useful for applications such as surveillance systems and autonomous vehicles. The Faster R-CNN algorithm incorporates a region proposal network (RPN) to generate regions likely to contain pedestrians and uses a convolutional neural network for accurate localization and classification. The research applied data augmentation techniques such as random flipping, scaling, and pruning to improve the model's generalization ability. The results showed that the model achieved the highest accuracy at the 45th training epoch, demonstrating an AP of 51.9%. Despite this, the model managed to maintain a high inference rate and compact memory size, which emphasizes its suitability for real-world applications. The authors emphasize that future enhancements to the model's performance can be achieved through the implementation of more advanced architectures and meticulous tuning of hyperparameters. This research provides valuable insights for further developments in computer vision and emphasizes the potential of using the Faster R-CNN algorithm in real-world pedestrian detection applications. Zbigniew Omiotek et al. [36] in their research investigated the application of the Faster R-CNN algorithm to detect dangerous objects in surveillance camera images. The researchers developed a customized dataset including images of baseball bats, pistols, knives, machetes, and rifles under different lighting and visibility conditions. Using different backbone networks, the best results were achieved with ResNet152, which showed an average detection accuracy (mAP) of 85% and AP of 80% to 91% depending on the object. The average detection rate was 11–13 frames per second, making the model suitable for public safety systems. The research emphasizes that most existing solutions overestimate their effectiveness due to the insufficient quality of the datasets used. The dataset created presents objects in a variety of conditions, including partial overlap, blur, and poor lighting, to better reflect real-world conditions. This improves the robustness of the results and the detection accuracy. The Faster R-CNN algorithm with different backbone networks such as ResNet152 has been tested and shown to perform well. The average accuracy for baseball bat detection was 87.8%, 91.3% for pistol, 80.8% for knife, 79.7% for machete, and 85.3% for rifle. Despite the high accuracy, the model managed to keep the processing speed at 11–13 frames per second. The experiments showed that the Faster R-CNN with ResNet152 backbone network is most effective for public safety monitoring tasks, which allows us to recommend it for use in video surveillance systems. The authors also note that further research may include increasing the number of training examples and using data augmentation techniques to improve the accuracy and reliability of the model. Tong Bai et al. [37] presented a method for recognizing vehicle types in images using an improved Faster R-CNN model. As the number of vehicles increases, traffic jams, accidents, and vehicle-related crimes increase, which requires improved traffic management methods. The research improves three aspects of the model: combining features from different layers of the convolutional network, using contextual features, and optimizing bounding boxes to improve recognition accuracy. A dataset with images of cars, SUVs, and vans was used in the experiments. The results showed that the improved model effectively identifies vehicle types with high accuracy. The average recognition accuracy (AP) for the three vehicle types was 83.2%, 79.2%, and 78.4%, respectively, which is 1.7% higher than the traditional Faster R-CNN model. These improvements make the proposed model suitable for use in intelligent transportation systems, improving the accuracy and reliability of traffic management and safety. Yanfeng Wang et al. [38] presented a new approach to

aircraft detection and classification in aerial images called TransEffiDet, which is based on a combination of EfficientDet and Transformer. The main goal of the research is to improve the accuracy of object detection in challenging environments, such as military operations, where reliable and accurate aircraft detection at large scales and in changing environments is essential. The TransEffiDet model uses the powerful EfficientDet backbone network to fuse multi-scale feature maps efficiently, as well as a transformer module to model long-term dependencies in images. The authors developed a feature fusion module that combines local and global contexts extracted from different layers of the transformer and EfficientDet. This allows the system not only to recognize objects at long distances but also to capture their small details accurately. One of the key achievements of the work is Models and Data Analysis Initiative (MADAI) [39], a public dataset for aircraft detection in aerial images, which includes five types of aircraft: fighters, armed helicopters, bombers, early warning aircraft, and passenger aircraft. The set contains 2558 images and will be made available with the article, which will contribute to the further development of technology in this area. In the results of the study, TransEffiDet demonstrated a significant improvement in mean accuracy (mAP) to 86.6%, which is 5.8% higher than the result of EfficientDet. The experiments also showed that the model is more effective at detecting aircraft, including fighter and early warning aircraft, and is more robust to external influences than other approaches (Figure 4).

Figure 4. Examples of aircraft detection results in the MADAI dataset.

This work opens up new possibilities for the use of deep neural networks in military and civilian tasks related to aircraft detection over large areas and can serve as a basis for future research in aviation security and military intelligence. Munteanu D. et al. [40] have carried out research to develop a real-time automatic sea mine detection system using three deep learning models: YOLOv5, SSD, and EfficientDet. In the current armed conflicts and geopolitical tensions, navigation safety is jeopardized by the large number of sea mines,

especially in maritime conflict areas. The research utilized images captured from drone, submarine, and ship cameras. Due to the limited number of sea mine images available, the researchers applied data augmentation techniques and synthetic image generation, creating two datasets: for floating mines and underwater mines. The models were trained and tested on these datasets and evaluated for accuracy and processing time. The results showed that the YOLOv5 algorithm provided the best detection accuracy for both floating and underwater mines, achieving high performance with a relatively short training time. The SSD model also performed well, although it required more computational resources. The EfficientDet model, one of the newest neural networks, showed high performance in detection tasks due to innovations introduced in previous models. Tests on handheld devices such as the Raspberry Pi have shown that the system can be used in real-world scenarios, providing mine detection with a latency of about 2 s per frame. These results highlight the potential of using deep learning to improve maritime safety in conflict situations. The research presented by O. Hramm et al. [41] describes a customizable solution for cell segmentation using the Hough transform and watershed algorithm. The paper emphasizes the importance of developing flexible algorithms that can adapt to different datasets and imaging conditions to improve recognition accuracy and reliability. The algorithm proposed by the authors has many tunable parameters that can optimize the segmentation process for different types of images. The Hough transform is used to define circles, which helps in clustering overlapping cells, while the watershed algorithm segments cells even if they overlap or are in complex conditions. Major achievements of the research include the creation of a robust algorithm that can be effectively applied to image processing with large amounts of noise and artifacts. This is particularly relevant to biological and medical analysis tasks where image quality often leaves much to be desired. The applicability of this research to our work is to use the proposed methods to improve the accuracy and robustness of deep learning models in detecting people and technical objects in complex environments such as water, roads, and other environments. The results and conclusions of this work will be used to optimize our models and adapt them to different application scenarios, thus ensuring high accuracy and robustness of monitoring and security systems. Thus, the research provides valuable data and methods that can be directly applied in our research to improve the performance of neural network models, providing high accuracy and reliability in real-world scenarios.

In their work, Bilous et al. [42] presented methods for detecting and comparing body poses in video streams, focusing on the evaluation of different libraries and methods designed to analyze and compare body poses in real-time. The authors investigated the performance and accuracy of libraries such as OpenPose [43], PoseNet [44], and BlazePose [45] in order to identify the most effective solutions for pose recognition. They emphasized that accurate detection and comparison of body poses is important for applications such as exercise monitoring, safety, and medical diagnostics. The research involved testing each of the libraries on different datasets to evaluate their performance and accuracy. BlazePose combined with the weighted distance method performed best, outperforming OpenPose and PoseNet on a number of measures. The authors emphasize that the methods and algorithms proposed in their work can be used in a variety of applications, including video surveillance systems, exercise control, and medical monitoring. The results of the research confirm that BlazePose is a powerful and efficient solution for real-time pose recognition. In conclusion, the authors emphasize the importance of selecting appropriate libraries and methods to achieve high accuracy and performance in real-world applications. In their work, Bilous N. et al. [46] presented a method for tracking human head movements using control points. This method reduces the number of vectors to be recorded to the minimum number needed to describe head movements. The research includes a comparison of existing face vector detection methods and demonstrates the advantage of regression methods, which show significant accuracy and independence from illumination and partial face occlusion. The main goal of the work is to create a method that can track head movements and record only significant head direction vectors. The authors propose a

control point method that significantly reduces the set of head direction vectors describing motion. According to the results, the regression-based methods showed significantly better accuracy and independence from illumination and partial face occlusion. The results of the research confirm the applicability of the control point method for human motion tracking and show that head vector detection methods from 2D images can compete with Red-Green-Blue-Depth (RGBD)-based methods in terms of accuracy. Thus, when combined with the proposed approach, these methods present fewer limitations than RGBD-based methods. This method can be useful in various fields such as human–computer interaction, telemedicine, virtual reality, and 3D sound reproduction. Moreover, head position detection can be used to compare the exercises performed by a person to a certain standard, which is useful for rehabilitation facilities, fitness centers, and entertainment venues. In their paper, Yi Xiao et al. [47] proposed a novel approach called TTST (Top-k Token Selective Transformer) to enhance the resolution of remote sensing images. The main problem faced by existing transformer models in this context is the use of all tokens to calculate self-consciousness, which leads to excessive consumption of computing resources and the involvement of irrelevant information. The authors have solved this problem by developing a token selection mechanism based on the top k most important tokens, which allows adaptive selection of the most relevant components for building long-term dependency models. This approach significantly reduces computational costs while maintaining the accuracy of image restoration. In addition, the authors have developed a Multi-scale Feed-forward Layer (MFL) that provides better interaction between objects of different scales, which is especially important for remote sensing images where objects can vary greatly in size. Using this technique, TTST is able to more accurately restore the scale correlations of objects, which significantly improves the quality of the final image. Special attention is paid to the Global Context Attention (GCA) module, which allows for better aggregation of information on large spatial scales. The GCA module dynamically adapts to the size of objects and helps to provide deeper and more detailed image reconstruction. This allows TTST to take into account global image features, which is critical for remote sensing tasks such as land cover classification or change detection. Experimental results show that TTST significantly outperforms other state-of-the-art approaches, including CNN and transform models. On the Aerial Image Dataset (AID) [48], the TTST model demonstrated a 0.16 dB improvement in PSNR compared to HAT-L, while reducing computational costs by almost half. Testing on real data was also conducted, where TTST outperformed other methods on metrics such as Natural Image Quality Evaluator (NIQE) and Average Gradient (AG), confirming the model's ability to achieve high results in real-world conditions with image degradation. In visual comparisons, TTST also showed a better ability to recover textures and details in images, especially in cases where the information in low-quality images was severely degraded. For example, in the case of the image 'playground_270' from the AID dataset, the TTST model was able to recover clearer and more detailed object contours than other transform models such as TransENet [49] and HAT-L [50]. The conclusions of this work emphasize that TTST is a promising solution for image resolution enhancement in remote sensing, especially in cases where computational resource savings without quality loss are important. The authors also point out that the model can be used in a wide range of applications, from land surface analysis to change monitoring and object classification, making it a versatile tool for improving image quality in many fields. In their article, Kui Jiang et al. [51] propose an innovative method for removing rain streaks from images, which they call the Progressive Conjugate Network (PCNet). The main problem faced by modern computer vision systems is that rainy conditions significantly reduce the quality of images, which in turn negatively affects the accuracy of tasks such as object recognition and segmentation. Therefore, the authors set out to create a model that combines high image processing accuracy and speed, enabling real-time use, especially on mobile devices with limited resources. The key innovation in their approach is the use of a conjugate representation module (CRM) that allows for the simultaneous analysis of rain streaks and their interaction with clean parts of the image. The CRM learns to detect common

features of rainbands and non-rain content, which allows it to separate unwanted elements while leaving important details intact more accurately. This makes the model particularly effective for tasks where textures and image details need to be preserved. The PCNet model applies progressive image processing, where the quality of rain streak extraction is gradually improved at each stage. To achieve this goal, the authors use a cascade of coupled CRM modules to gradually separate rainstreaks from clean parts of the image, which gives more accurate results than traditional methods that work with all image elements simultaneously. In addition, to reduce computational costs, the PCNet model uses depth-wise separable convolutions and an efficient U-shaped structure, which reduces the number of model parameters and makes it suitable for use on mobile devices. This solution makes PCNet particularly attractive for applications requiring fast real-time image processing. Experiments conducted by the authors have demonstrated a significant outperformance of PCNet compared to state-of-the-art treeing models. In particular, the model showed a significant improvement in both image reconstruction accuracy and processing speed, making it up to eight times faster than other popular models such as Multi-scale Progressive Fusion Network (MSPFN). Additionally, a lighter version of PCNet, called PCNet-fast, achieves real-time performance while maintaining high accuracy of the tree, making it extremely useful for mobile devices and other systems with limited computing resources. PCNet has also been successfully applied to other tasks, such as object recognition and image segmentation, proving its versatility and effectiveness in real-world environments where image quality can be significantly reduced by rain or other degradation factors.

3. Materials and Methods

3.1. Datasets

Four main datasets, COCO2017, SARD, SeaDronesSee, and VisDrone2019, were used to train the selected neural networks and evaluate their performance in different environments. These datasets provide unique images and annotations that allow the models to learn to recognize and classify objects in different scenarios. COCO2017 (Common Objects in Context) [1] is one of the most widely used and well-known datasets in computer vision. It includes over 200,000 images annotated for more than 80 object categories such as people, cars, bicycles, animals, and household objects. The annotations include not only bounding boxes but also segmentation masks, allowing COCO2017 to be used for object detection and segmentation tasks. This dataset is characterized by a variety of scenes and objects collected in real-world environments with different variations in lighting, poses, and backgrounds. This makes COCO2017 ideal for the comprehensive evaluation of object recognition algorithms and provides models with a rich training experience. SARD [2] is a specialized dataset designed for maritime object detection tasks. The dataset includes drone and ship images that cover a wide range of scenarios related to search and rescue operations. SARD includes annotations for various objects such as boats, life rafts, and people in the water. The images in this dataset vary in lighting and weather conditions, allowing models to learn to recognize objects in complex and variable marine environments. This dataset is particularly valuable for the development of algorithms designed for marine rescue and aquatic monitoring. The SeaDronesSee [3] dataset is designed for monitoring aquatic areas using drones. It contains images of people on water, various marine objects, and natural phenomena such as waves and foam. Annotations in SeaDronesSee include object detection and segmentation tasks, allowing models to recognize and classify different objects in complex aquatic environments. This dataset is used to develop algorithms that can perform effectively under the changing lighting and dynamic scenes that characterize marine environments. SeaDronesSee provides realistic scenarios for testing and improving models aimed at safety and rescue in the water. VisDrone2019 is one of the largest and most diverse datasets collected from drones. It includes images captured under a variety of lighting and weather conditions, making it ideal for evaluating algorithm performance in complex traffic scenes and urban environments. VisDrone2019 [4] contains annotations for more than 10 object categories, including cars, pedestrians, cyclists, and other vehicles.

The dataset is used for object detection, tracking, and segmentation tasks, allowing for a comprehensive evaluation of the algorithms in real-world environments. Annotations include bounding boxes, segmentation masks, and object traces, making it one of the most comprehensive and versatile datasets for drone-related computer vision tasks.

3.2. Data Preparation

Data preparation is a critical step to ensure successful training of neural network models. In this research, several key steps were performed to prepare the data, which ensured high-quality input data and contributed to improved model performance. The first step was image preprocessing. All images were normalized to bring them to a uniform scale and range of pixel values. Normalizing the images improved the stability and speed of model training by reducing the impact of differences in brightness and contrast. In addition, all images were resized to a uniform size, ensuring compatibility with the neural network model architecture and allowing the use of fixed-size batches. Data augmentation played a key role in increasing the size of the training set and improving the generalization ability of the models. Various techniques such as rotation, scaling, shifting, and changing the brightness and contrast of the images were used in the augmentation process. These techniques generated additional examples from the original images, which reduced the risk of overtraining the models and improved their ability to recognize objects under different conditions.

After data preprocessing and augmentation, the datasets were divided into three parts: training, validation, and test samples in an 80:10:10 ratio. The training sample was used to train the models directly, the validation sample was used to monitor the training process and prevent overtraining, and the test sample was intended for the final evaluation of the model's performance. The data were partitioned using stratified random sampling, ensuring that the proportions of object classes were maintained in each subset. This method provided an unbiased evaluation of the models. Special attention was paid to the processing of annotations for images. The annotations contained information about the location and types of objects in the images (Figure 5), which is necessary to train the models on detection and classification tasks. The image shows that the partitioning includes both foreground and background, which provides data diversity and helps the model learn to distinguish objects in complex environments. This is especially important when creating datasets for real-time autonomous control and monitoring tasks, where recognition accuracy plays a key role. This annotation helps models improve object recognition in complex scenes where overlaps, partial visibility, and different viewing angles are present. All annotations were brought to a common format compatible with the deep learning frameworks used, which simplified the process of loading and using the data in the models. Special scripts were developed to efficiently load the data into memory and transfer it to the GPU. These scripts provided fast and seamless loading of large amounts of data, minimizing latency and providing high performance when training models. The scripts also included methods for dynamic data augmentation during training, further increasing the diversity of training examples and improving the generalizability of the models.

Utilizing an integrated approach to data training that included preprocessing, augmentation, proper partitioning into samples, and efficient data loading played a key role in achieving high model performance. This approach allowed us to create high-quality input data for training, which in turn ensured the stability and accuracy of neural network models under different conditions.

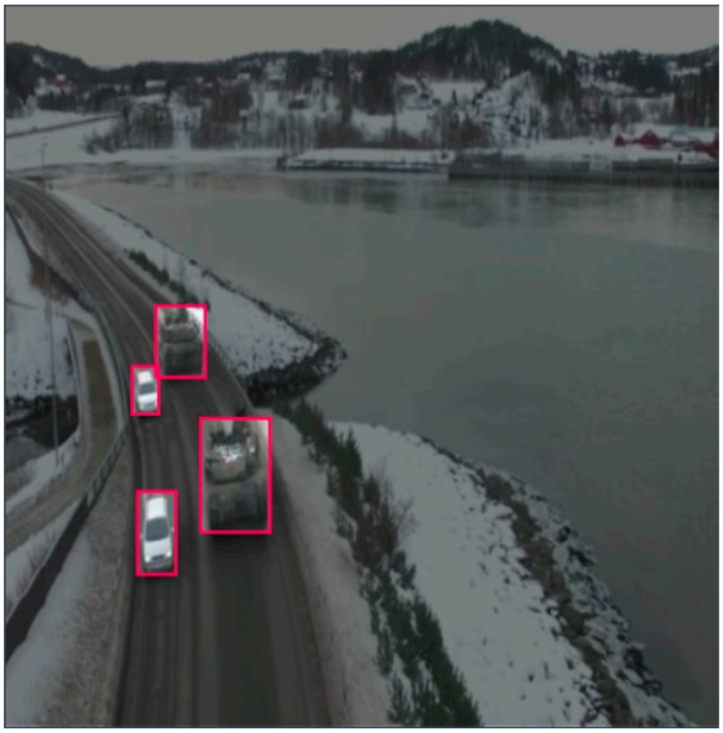

Figure 5. Annotated image of a technical object for object detection model training.

3.3. Neural Network Models

For this research, several modern neural network models have been selected, each with unique characteristics and advantages that make them suitable for different object detection tasks. The models considered include YOLOv4-v8, Faster R-CNN, SSD, and EfficientDet. The YOLO model was chosen for its high-speed image processing and real-time capability. This model is characterized by the fact that it processes the whole image in one pass of the network, which allows high speed. YOLOv4-v8 versions were used in this research. Each subsequent version of YOLO brought improvements in accuracy and robustness to different lighting conditions and noise, making these models the most productive in real-world environments. YOLOv4 [6] offered improvements over previous versions by introducing a new detection method and utilizing techniques such as CSPNet and PANet, greatly improving the accuracy and speed of the model. In the YOLOv5 [7] version, image processing techniques and learning algorithms have been improved to achieve even better performance. The YOLOv6 [8] and YOLOv7 [9] versions continue to enhance the model architecture, including improvements in handling different object scales and optimizing computational resources. YOLOv8 [10] is the latest version in this research, which integrates the latest developments in deep learning and provides the highest accuracy and speed among all YOLO versions.

The Faster R-CNN [11] model was chosen for its high accuracy in object detection tasks. The main advantage of Faster R-CNN is the use of Region Proposal Network (RPN), which significantly speeds up the object detection process compared to previous versions of R-CNN. However, despite its high accuracy, the processing speed of this model is slower compared to YOLO, which limits its application in tasks requiring real-time processing. Faster R-CNN uses RPN to generate region suggestions, which are then processed by a convolutional network to localize and classify objects accurately. The model finds wide application in tasks where high accuracy is important, such as video surveillance, autonomous driving, and image analysis in medical applications.

SSD [12] was selected for its focus on high-speed image processing. The model utilizes multiple feature maps to detect objects at different scales to achieve a high level of accuracy with low latency. SSD is widely used in tasks related to autonomous driving and robotics

due to its ability to work efficiently in real-world environments. The SSD utilizes multiple feature maps of different resolutions, which enables the model to efficiently detect objects of different sizes and scales in the image. The SSD model demonstrates high performance with low latency, making it suitable for applications requiring real-time processing.

EfficientDet [13] is a model based on the EfficientNet architecture that emphasizes computational and resource efficiency. This model is ideal for use in environments with limited computational resources. EfficientDet exhibits high accuracy and performance, especially in the context of mobile and embedded systems, making it an important component for resource-intensive tasks. EfficientDet utilizes a novel BiFPN (Bidirectional Feature Pyramid Network) architecture to efficiently fuse features from different layers to achieve high accuracy at low computational cost. EfficientDet is widely used in mobile and embedded systems where it is important to minimize resource consumption while maintaining high object recognition accuracy.

Each of these models was trained and evaluated using training datasets. Popular deep learning frameworks such as TensorFlow and PyTorch were used in the training process. The main metrics for evaluating the performance of the models were mAP (mean accuracy), precision, accuracy, completeness, F1-Score, and FPS (frames per second), which enabled a comprehensive evaluation of both object recognition accuracy and the models' ability to perform in real-world environments. Thus, the research provided a comparative analysis of the performance of state-of-the-art neural network models, which allowed us to identify their strengths and weaknesses in different application scenarios.

3.4. Training Models

Training neural network models is a key step in the process of developing and optimizing neural networks. For each selected model, experiments were performed to train and evaluate them. The model training process involved several steps. First, the models were initialized with pre-configured hyperparameters. They were then trained on the training dataset, with regular evaluation on the validation dataset to monitor progress and prevent overfitting. After training was completed, the models were evaluated on the test dataset on key metrics such as mAP (mean accuracy), precision, accuracy, completeness, F1-Score, and FPS (frames per second). These metrics allowed for a comprehensive evaluation of both object recognition accuracy and the models' ability to perform in real-world conditions.

3.4.1. Computational Platform

A high-performance computing platform was utilized in this research to ensure efficient training and performance evaluation of neural network models. The choice of hardware and software plays a critical role in performing complex deep learning tasks, especially when dealing with large datasets and complex model architectures. Compute nodes med-IoT-KI serve equipped with powerful 4x NVIDIA Tesla A100 GPUs with 80 GB of memory were used to train the models. This GPU was chosen for its high performance and ability to process large amounts of data. The CPU used was a dual-processor AMD EPYC 7413 with 24 cores at 2.65 GHz, providing multi-threading and high performance for data processing and GPU coordination. The system was also equipped with 512 GB of RAM to handle large batches of data and maintain high processing speeds. For data storage, 2 TB SSD disks were used to provide fast access to data and models, reducing latency during training. Modern software tools and libraries were used for model development, training, and evaluation. The operating system chosen was Ubuntu 20.04 LTS for its stability, security, and extensive support for scientific computing tools. The deep learning frameworks used were TensorFlow 2.4 and PyTorch 1.8, which provide powerful tools for developing and training neural networks and also support GPU operation. The NumPy, Pandas, OpenCV, and Scikit-learn libraries were used for data preprocessing, result analysis, and visualization. Environment optimization and tuning were performed to ensure the stable and efficient operation of the computing platform. NVIDIA CUDA 11.2 and cuDNN 8.1 drivers were installed to ensure compatibility with the used GPUs and

deep learning frameworks. TensorFlow and PyTorch were configured to use all available GPUs, maximizing performance and speeding up model training. All necessary libraries and dependencies were installed and updated to ensure their relevance and compatibility with deep learning frameworks.

Preparing the data for training involved several key steps. First, the OpenCV and NumPy libraries were used to normalize, resize, and augment the images. Next, the datasets were divided into training, validation, and test samples to provide an unbiased assessment of model performance. Finally, scripts were developed to efficiently load the data into memory and transfer it to the GPU for model training. The use of a high-performance computing platform and optimized software significantly accelerated the model training process and ensured high accuracy and stability of the results. The environment setup and data preparation played a key role in achieving high performance, which confirms the importance of choosing the right hardware and software tools for deep learning tasks.

3.4.2. Configuring Hyperparameters

Hyperparameter tuning is a key step in the training of neural network models, as choosing the right hyperparameter values directly affects the performance, accuracy, and generalization ability of the model. In this research, optimal hyperparameter values were selected for each of the models based on multiple experiments and analyzing the results. The initial learning rate for all models was set at 0.01. This value provided a sufficient convergence rate, allowing the model to train quickly while preventing jump changes in weights that can occur if the learning rate is too high. During training, the learning rate could be adjusted based on monitoring the errors and convergence of the model to ensure the best results. The batch size, which determines the number of images processed per iteration, was chosen to be 64. This value was chosen considering the availability of computational resources and the necessary balance between gradient estimation accuracy and memory utilization efficiency. Larger packet sizes produce more accurate gradient estimates but require more memory, whereas smaller packets produce noisier estimates but require fewer resources. The number of epochs to train each model was set to 30. This value was chosen based on learning curve analysis and validation. An insufficient number of epochs can lead to undertraining of the model when it fails to learn all the patterns in the data. On the other hand, too many epochs can lead to overfitting when the model starts to overfit the training data, losing its ability to generalize to new data. Therefore, 30 epochs were the optimal value, providing a balance between training and generalization. The Adam optimizer, which combines the advantages of adaptive learning rate and moment methods, was used to update the model weights. Adam automatically adjusts the learning rate for each parameter, allowing the model to converge faster and adapt to different types of data. This optimizer has shown high performance and stability on a wide range of deep learning tasks.

To prevent overfitting in the YOLOv4-v8 models, a Dropout method with a neuron outage probability of 0.5 was used. This method consists of randomly turning off a certain percentage of neurons during training, which prevents the model from being dependent on specific features and improves its generalization ability. Dropout helps models avoid overfitting and improves their performance on validation and test data. The Faster R-CNN model used an SGD (Stochastic Gradient Descent) optimizer with a momentum of 0.9. This optimizer allows past gradients to be taken into account when updating the weights, resulting in more stable and faster training. L2-regularization with a factor of 0.0005 was also applied to the model to prevent increasing weights and overfitting. L2-regularization adds a penalty to the loss function for large values of weights, which helps to control the complexity of the model and improve its generalization ability. Dropout with probability 0.4 was applied in the SSD model, which helped to improve the generalization ability of the model. The RMSprop optimizer used in this model provided adaptive updating of weights and good convergence. RMSprop automatically adjusts the learning rate for each parameter, allowing the model to adapt to the data faster and improve its performance.

EfficientDet, based on the EfficientNet architecture, used L2 regularization with a factor of 0.0001 to prevent increasing weights and overfitting. This regularization helped control the complexity of the model and improve its ability to generalize. EfficientDet emphasizes computational and resource efficiency, making it ideal for use in computationally constrained environments.

An early stopping method was used for all models, which stopped training if performance on the validation dataset stopped improving over 10 epochs. This method avoided unnecessary training time and prevented overfitting of models while ensuring high accuracy and stability. Hyperparameter tuning was conducted by trial and error using cross-validation to ensure the best results. Optimizing the hyperparameters achieved a balance between the accuracy, performance, and generalization ability of the models, which was critical for real-world object recognition tasks.

4. Results

4.1. Performance Evaluation

Experimental results showed that the models exhibit different levels of performance in human and technical object detection tasks under different conditions. The table below summarizes the comparative accuracy, completeness, average accuracy at threshold 0.5 (mAP@0.5), and F1-Score for each architecture after 30 training epochs (Table 1):

Table 1. Architectures performance.

Architecture	Precision	Recall	mAP@0.5	F1-Score
YOLOv4	0.81	0.83	0.82	0.82
YOLOv5	0.73	0.73	0.73	0.73
YOLOv6	0.62	0.62	0.62	0.62
YOLOv7	0.68	0.68	0.68	0.68
YOLOv8	0.87	0.89	0.88	0.88
Faster R-CNN	0.84	0.82	0.83	0.83
SSD	0.76	0.75	0.75	0.75
EfficientDet	0.82	0.80	0.81	0.81

From the above data, we can see that YOLOv8 demonstrates the highest performance among all models, showing the best results in all key metrics. Faster R-CNN also shows high accuracy but is inferior to YOLOv8 in processing speed.

4.2. Processing Speed

One of the key aspects of real-time object detection tasks is processing speed. The table below shows the FPS (frames per second) for each architecture (Table 2):

Table 2. Processing speed for each architecture.

Architecture	FPS
YOLOv4	40
YOLOv5	45
YOLOv6	42
YOLOv7	43
YOLOv8	48
Faster R-CNN	10
SSD	35
EfficientDet	30

YOLOv8 demonstrates the highest processing speed among all architectures, which makes it the most suitable for tasks requiring real-time. Faster R-CNN, despite its high accuracy, is significantly inferior in terms of speed, which limits its application in such tasks.

4.3. Classification Errors

The analysis of object classification errors has revealed important aspects of model optimization. Partially detected objects, misclassification, and completely missed objects are typical errors that occur during the detection of people and technical objects in different environments. The table below summarizes the error analysis for different models (Table 3):

Table 3. Detection of errors for each architecture.

Architecture	Partially Detected	Misclassified	Missed
YOLOv4	10%	7%	6%
YOLOv5	11%	8%	6%
YOLOv6	21%	12%	11%
YOLOv7	10%	6%	6%
YOLOv8	8%	4%	4%
Faster R-CNN	9%	5%	6%
SSD	14%	9%	8%
EfficientDet	11%	7%	7%

YOLOv8 significantly outperforms other models in all error rates, which indicates its high efficiency and accuracy in the task of detecting people and technical objects in various environments. Figure 6 shows a street scene where the YOLOv8 model recognized several cars. On the left is a black car, which the model recognized with a probability of 0.92, circled by an orange rectangle and labeled "car". In front of the black car is a silver car, recognized by the model with a probability of 0.66, also circled by an orange rectangle and labeled "car". Partially visible in the right foreground is a silver car, identified by the model with a probability of 0.93, circled by a green rectangle and labeled "truck". The image demonstrates how YOLOv8 performs in a complex street traffic environment. It is important to note here that YOLOv8 can handle objects of different types and sizes, detecting them in real-time. This allows YOLOv8 to be used in traffic control systems, autonomous vehicles, and security applications where instant and accurate object recognition is required. The scene shown also highlights YOLOv8's ability to handle different lighting and background conditions, which is essential for urban applications. This allows the system to recognize and respond quickly to dynamic changes in road conditions.

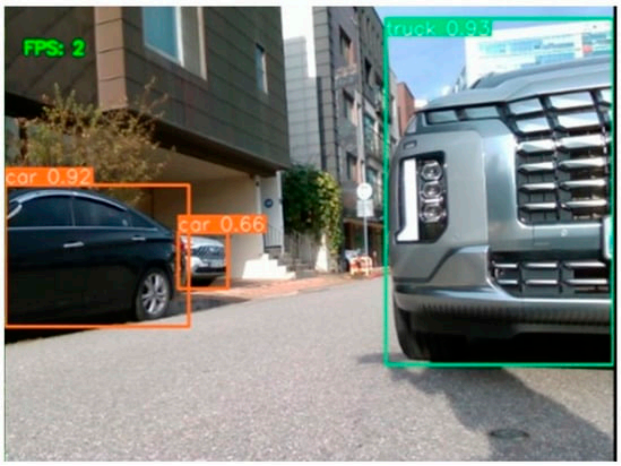

Figure 6. Detection and classification of vehicles on an urban road scene using the YOLOv8 model.

Figure 7 shows a scene at a military training ground where the YOLOv8 model recognized two tanks. In the foreground is a large tank, which the model identified with a probability of 0.925. The tank is circled by a red rectangle and labeled "tank". In the distance, on the road leading up the hill, you can see a second tank, recognized by the model with a probability of 0.907, also circled by a red rectangle and labeled "tank".

Figure 7. Detection and classification of military vehicles on a terrain using the YOLOv8 model.

The image demonstrates the YOLOv8 model's performance in recognizing and classifying military vehicles in open terrain, with probabilities for each recognized object.

Experimental results confirmed that YOLOv8 is the most effective model for detecting people and technical objects in different environments, providing high accuracy, speed, and ability to adapt to different scenarios. Faster R-CNN showed high accuracy, but its low processing speed limits its application in real-time. SSD and EfficientDet also showed decent results but are inferior to YOLOv8 in terms of accuracy and processing speed. Thus, YOLOv8 is the preferred choice for security monitoring and complex environment analysis tasks, providing new opportunities for further improvements and developments in computer vision and deep learning.

5. Discussion

The results of our research showed that YOLOv4-v8, Faster R-CNN, SSD, and EfficientDet models exhibit different levels of performance in human and technical object detection tasks in different environments. YOLOv8 proved to be the most efficient model, providing high accuracy and processing speed, making it the preferred choice for real-time tasks. Faster R-CNN showed high accuracy, but its low processing speed significantly limits its application in tasks where responsiveness is important. SSD and EfficientDet also performed well but are inferior to YOLOv8 in challenging environments such as water and road scenes. Our experiments showed that environmental conditions have a significant impact on the performance of the models. For example, complex water scenes with reflections and variable lighting conditions pose additional challenges for the models. In such environments, YOLOv8 performed best due to its ability to adapt to a variety of conditions and its high object recognition accuracy. On roads with many moving objects and a variety of backgrounds, YOLOv8 also performed well, outperforming other models in terms of accuracy and processing speed. This makes it ideal for applications in traffic monitoring systems and autonomous vehicles. The classification error analysis showed that the main problems arise from partially detected objects, misclassification, and missed objects. Partially detected objects are often found in complex environments such as water, where reflections and variable lighting conditions are present. Misclassification can be due to insufficient image clarity, or the complexity of scenes where multiple objects are present. Missing objects are more common in low visibility conditions or where there are multiple overlaps. Despite the high performance of YOLOv8, there are several areas for further improvement. One of them is to improve the handling of complex scenes with many objects and overlaps. This can be achieved by using additional training data and applying reinforcement learning techniques. It is also worth considering using more complex architectures to improve accuracy and reduce classification errors. The metrics used in our study—mean average precision (mAP), F1-score, and frames per second (FPS)—play a crucial role in evaluating the real-world applicability of the models. These metrics not only measure the accuracy of object recognition but also determine the model's usability in scenarios requiring rapid processing. While different neural networks perform better

on specific metrics, YOLO consistently outperforms them across the board. With a mean average precision (mAP) of 0.88, F1-score of 0.88, and an impressive processing speed of 48 frames per second (FPS), YOLOv8 is the most well-rounded model. This high processing speed (FPS) makes YOLOv8 especially suitable for real-time applications such as traffic monitoring, security systems, and autonomous vehicles, where timely decision-making is critical. Its ability to maintain high accuracy while processing data quickly makes it stand out compared to other models, such as Faster R-CNN, which sacrifices speed for accuracy, or SSD, which may not match YOLOv8's precision and recall in complex environments. The models used in this research have a wide range of applications in various fields. YOLOv8, due to its high accuracy and processing speed, is ideal for security monitoring tasks such as water and road surveillance, as well as for applications in autonomous vehicle systems. Faster R-CNN, despite its low processing speed, can be effective in applications where high accuracy is required and real-time is not necessary, such as medical imaging and video surveillance. SSD and EfficientDet can also be useful in computationally constrained environments due to their efficiency and speed. In conclusion, our research shows that YOLOv8 is the most effective model for human and technical object detection tasks in various environments due to its high accuracy, processing speed, and ability to adapt to a variety of scenarios. Faster R-CNN, SSD, and EfficientDet also performed well, but are inferior to YOLOv8 in key performance metrics. These results emphasize the importance of selecting the right model for specific tasks and conditions, which is critical for the successful application of computer vision and deep learning technologies in real-world scenarios.

6. Conclusions

In this research, comprehensive experiments were performed to compare the performance of different neural network models in human and technical object detection tasks under a variety of conditions. The models considered included YOLOv4-v8, Faster R-CNN, SSD, and EfficientDet. The experimental results allowed us to draw several key conclusions. First, the YOLOv8 model performed the best among all the models considered, demonstrating high accuracy and processing speed. These characteristics make it the preferred choice for real-time tasks such as water and road safety monitoring and autonomous vehicle systems. YOLOv8 has also shown high robustness to challenging environmental conditions such as reflections on water and variable lighting conditions. Second, Faster R-CNN demonstrated high object recognition accuracy, but its low processing speed limits real-time applications. This model can be effective in applications where high accuracy is required but processing speed is not critical, such as medical imaging and video surveillance. SSD and EfficientDet also performed well, especially in computationally constrained environments. These models strike a balance between accuracy and performance, making them suitable for mobile and embedded systems where resource efficiency is important. The analysis of classification errors showed that the main problems arise from partially detected objects, misclassification, and missing objects. These errors are most common in complex environments such as water and road scenes with many objects. Further improvement of the models requires a focus on making them more robust to such conditions, which can be achieved by using additional training data and applying reinforcement learning techniques. Data training played a key role in the success of model training. Normalization, augmentation, and proper partitioning of data into training, validation, and test samples ensured high-quality input data and contributed to improved model performance. Efficient data loading and optimization of the computational platform also played an important role in achieving high performance. In conclusion, the results of this research emphasize the importance of selecting the appropriate model for specific tasks and conditions. YOLOv8 is the most effective model for human and technical object detection tasks in various environments due to its high accuracy, processing speed, and ability to adapt to a variety of scenarios. These findings are critical for the successful application of computer vision and deep learning technologies in real-world scenarios, providing new opportunities for further improvements and developments in this field.

Author Contributions: Conceptualization, N.B. and V.M.; methodology, N.B.; software, V.M.; validation, N.B., A.N. and V.M. formal analysis, N.B. and M.F.; investigation, A.N., N.B. and V.M.; resources, M.F.; data curation, N.B.; writing—original draft preparation, N.B. and V.M.; writing—review and editing, M.F.; visualization, V.M. supervision, N.B.; project administration, M.F.; funding acquisition, M.F. All authors have read and agreed to the published version of the manuscript.

Funding: The financial support of the Volkswagen Foundation, Grant No 9D167.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The original contributions presented in this study are included in the article. Further inquiries can be directed to the corresponding author(s).

Acknowledgments: The research was performed in collaboration with the research laboratory "Information Technologies in Learning and Comuter Vision Systems" of Kharkiv National University of Radio Electronics and Technical University of Applied Sciences Wildau.The server used for research is med-IoT-KI, TH Wildau, project 85053663 "AI-supported IoT environment for medical examinations" MWFK program to promote the infrastructure for research, development, and innovation (InfraFEI).

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

References

1. Lin, T.-Y.; Maire, M.; Belongie, S.; Bourdev, L.; Girshick, R.; Hays, J.; Perona, P.; Ramanan, D.; Zitnick, C.L.; Dollár, P. Microsoft COCO: Common Objects in Context. In Proceedings of the Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, 6–12 September 2014.
2. Sambolek, S.; Ivasic-Kos, M. Search and Rescue Image Dataset for Person Detection—SARD. 2021. Available online: https://ieee-dataport.org/documents/search-and-rescue-image-dataset-person-detection-sard (accessed on 24 October 2024).
3. Varga, L.A.; Kiefer, B.; Messmer, M.; Zell, A. SeaDronesSee: A Maritime Benchmark for Detecting Humans in Open Water. *arXiv* **2021**. [CrossRef]
4. Zhu, P.; Wen, L.; Du, D.; Bian, X.; Fan, H.; Hu, Q.; Ling, H. Detection and Tracking Meet Drones Challenge. *IEEE Trans. Pattern Anal. Mach. Intell.* **2021**, *44*, 7380–7399. [CrossRef] [PubMed]
5. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You Only Look Once: Unified, Real-Time Object Detection. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 779–788.
6. Bochkovskiy, A.; Wang, C.-Y.; Liao, H.-Y.M. YOLOv4: Optimal Speed and Accuracy of Object Detection. *arXiv* **2020**, arXiv:2004.10934.
7. Jocher, G. Ultralytics YOLOv5. 2020. Available online: https://github.com/ultralytics/yolov5 (accessed on 24 October 2024).
8. Li, C.; Li, L.; Jiang, H.; Weng, K.; Geng, Y.; Li, L.; Ke, Z.; Li, Q.; Cheng, M.; Nie, W.; et al. YOLOv6: A Single-Stage Object Detection Framework for Industrial Applications. *arXiv* **2022**, arXiv:2209.02976. [CrossRef]
9. 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. *arXiv* **2022**, arXiv:2207.02696.
10. Jocher, G.; Chaurasia, A.; Qiu, J. Ultralytics YOLOv8. 2023. Available online: https://github.com/ultralytics/ultralytics (accessed on 24 October 2024).
11. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 1137–1149. [CrossRef]
12. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.-Y.; Berg, A.C. SSD: Single Shot MultiBox Detector. *arXiv* **2015**, arXiv:1512.02325. [CrossRef]
13. Tan, M.; Pang, R.; Le, Q.V. EfficientDet: Scalable and Efficient Object Detection. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 14–19 June 2020; pp. 10778–10787.
14. Neamah, S.B.; Karim, A.A. Real-Time Traffic Monitoring System Based on Deep Learning and YOLOv8. *ARO* **2023**, *11*, 137–150. [CrossRef]
15. Kalva, A.R.; Chelluboina, J.S.; Bharathi, B. Smart Traffic Monitoring System Using YOLO and Deep Learning Techniques. In Proceedings of the 2023 7th International Conference on Trends in Electronics and Informatics (ICOEI), Tirunelveli, India, 11–13 April 2023; pp. 831–837.
16. Kunekar, P.; Narule, Y.; Mahajan, R.; Mandlapure, S.; Mehendale, E.; Meshram, Y. Traffic Management System Using YOLO Algorithm. *Eng. Proc.* **2023**, *59*, 210.
17. Ağdaş, M.T.; Gülseçen, S. Automatic Weapon and Knife Detection System on Security Cameras: Comparative YOLO Models. *Avrupa Bilim Teknol. Derg.* **2022**, *41*, 16–22. [CrossRef]
18. Wang, M.; Yang, B.; Wang, X.; Yang, C.; Xu, J.; Mu, B.; Xiong, K.; Li, Y. YOLO-T: Multitarget Intelligent Recognition Method for X-Ray Images Based on the YOLO and Transformer Models. *Appl. Sci.* **2022**, *12*, 11848. [CrossRef]

19. Azevedo, P.; Santos, V. YOLO-Based Object Detection and Tracking for Autonomous Vehicles Using Edge Devices. In *ROBOT2022: Fifth Iberian Robotics Conference*; Tardioli, D., Matellán, V., Heredia, G., Silva, M.F., Marques, L., Eds.; Lecture Notes in Networks and Systems; Springer International Publishing: Cham, Switzerland, 2023; Volume 589, pp. 297–308, ISBN 978-3-031-21064-8.
20. Özcan, İ.; Altun, Y.; Parlak, C. Improving YOLO Detection Performance of Autonomous Vehicles in Adverse Weather Conditions Using Metaheuristic Algorithms. *Appl. Sci.* **2024**, *14*, 5841. [CrossRef]
21. Reddy, S.; Pillay, N.; Singh, N. Comparative Evaluation of Convolutional Neural Network Object Detection Algorithms for Vehicle Detection. *J. Imaging* **2024**, *10*, 162. [CrossRef] [PubMed]
22. Sun, C.; Chen, Y.; Qiu, X.; Li, R.; You, L. MRD-YOLO: A Multispectral Object Detection Algorithm for Complex Road Scenes. *Sensors* **2024**, *24*, 3222. [CrossRef]
23. Dai, Y.; Kim, D.; Lee, K. An Advanced Approach to Object Detection and Tracking in Robotics and Autonomous Vehicles Using YOLOv8 and LiDAR Data Fusion. *Electronics* **2024**, *13*, 2250. [CrossRef]
24. Bilous, N.; Svidin, O.; Ahekian, I.; Malko, V. A Skeleton-Based Method for Exercise Recognition Based On 3D Coordinates of Human Joints. *IJ-AI* **2024**, *13*, 581. [CrossRef]
25. Khan, D.; Waqas, M.; Tahir, M.; Islam, S.U.; Amin, M.; Ishtiaq, A.; Jan, L. Revolutionizing Real-Time Object Detection: YOLO and MobileNet SSD Integration. *J. Comput. Biomed. Inform.* **2023**, *6*, 41–49.
26. Suryavanshi, D. A Comparative Study of Object Detection Using YOLO and SSD Algorithms. *Int. J. Sci. Res. Eng. Manag.* **2023**, *7*, 1–7. [CrossRef]
27. Akshatha, K.R.; Karunakar, A.K.; Shenoy, S.B.; Pai, A.K.; Nagaraj, N.H.; Rohatgi, S.S. Human Detection in Aerial Thermal Images Using Faster R-CNN and SSD Algorithms. *Electronics* **2022**, *11*, 1151. [CrossRef]
28. Davis, J.; Keck, M. A Two-Stage Approach to Person Detection in Thermal Imagery. In Proceedings of the IEEE Workshop on Applications of Computer Vision, Breckenridge, CO, USA, 5–7 January 2005.
29. Huda, N.U.; Hansen, B.D.; Gade, R.; Moeslund, T.B. The Effect of a Diverse Dataset for Transfer Learning in Thermal Person Detection. *Sensors* **2020**, *20*, 1982. [CrossRef]
30. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016.
31. Ioffe, S.; Szegedy, C. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. *arXiv* **2015**, arXiv:1502.0316.
32. Howard, A.G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; Adam, H. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. *arXiv* **2017**, arXiv:1704.04861.
33. Zhang, H.; Shao, F.; Chu, W.; Dai, J.; Li, X.; Zhang, X.; Gong, C. Faster R-CNN Based on Frame Difference and Spatiotemporal Context for Vehicle Detection. *Signal Image Video Process.* **2024**, *18*, 7013–7027. [CrossRef]
34. Gao, S. Exploration and Evaluation of Faster R-CNN-Based Pedestrian Detection Techniques. *Appl. Comput. Eng.* **2024**, *32*, 185–190. [CrossRef]
35. Dollar, P.; Wojek, C.; Schiele, B.; Perona, P. Pedestrian Detection: A Benchmark. In Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, USA, 20–25 June 2009; IEEE: Miami, FL, USA, 2009; pp. 304–311.
36. Omiotek, Z.; Zhunissova, U. Dangerous Items Detection in Surveillance Camera Images Using Faster R-CNN. 2024, *preprints*. Available online: https://www.preprints.org/ (accessed on 24 October 2024).
37. Bai, T.; Luo, J.; Zhou, S.; Lu, Y.; Wang, Y. Vehicle-Type Recognition Method for Images Based on Improved Faster R-CNN Model. *Sensors* **2024**, *24*, 2650. [CrossRef]
38. Wang, Y.; Wang, T.; Zhou, X.; Cai, W.; Liu, R.; Huang, M.; Jing, T.; Lin, M.; He, H.; Wang, W.; et al. TransEffiDet: Aircraft Detection and Classification in Aerial Images Based on EfficientDet and Transformer. *Comput. Intell. Neurosci.* **2022**, *2022*, 2262549. [CrossRef]
39. Wang, Y.; Wang, T.; Zhou, X.; Cai, W.; Liu, R.; Huang, M.; Jing, T.; Lin, M.; He, H.; Wang, W.; et al. MADAI Dataset 2022. Available online: https://github.com/wangyanfeng231/TransEffiDet (accessed on 24 October 2024).
40. Munteanu, D.; Moina, D.; Zamfir, C.G.; Petrea, Ș.M.; Cristea, D.S.; Munteanu, N. Sea Mine Detection Framework Using YOLO, SSD and EfficientDet Deep Learning Models. *Sensors* **2022**, *22*, 9536. [CrossRef]
41. Hramm, O.; Bilous, N.; Ahekian, I. Configurable Cell Segmentation Solution Using Hough Circles Transform and Watershed Algorithm. In Proceedings of the 2019 IEEE 8th International Conference on Advanced Optoelectronics and Lasers (CAOL), Sozopol, Bulgaria, 6–8 September 2019; pp. 602–605.
42. Bilous, N.V.; Ahekian, I.A.; Kaluhin, V.V. Determination and Comparison Methods of Body Positions on Stream Video. *Radio Electron. Comput. Sci. Control* **2023**, 52–60. [CrossRef]
43. Liu, Y. OpenPose-Based Yoga Pose Classification Using Convolutional Neural Network. *Highlights Sci. Eng. Technol.* **2022**, *23*, 72–76. [CrossRef]
44. Kendall, A.; Grimes, M.; Cipolla, R. PoseNet: A Convolutional Network for Real-Time 6-DOF Camera Relocalization. In Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, 7–13 December 2015; pp. 2938–2946.
45. Bazarevsky, V.; Grishchenko, I.; Raveendran, K.; Zhu, T.; Zhang, F.; Grundmann, M. BlazePose: On-Device Real-Time Body Pose Tracking. *arXiv* **2020**, arXiv:2006.10204.

46. Rakova, A.O.; Bilous, N.V. Reference Points Method for Human Head Movements Tracking. *Radio Electron. Comput. Sci. Control* **2020**, 121–128. [CrossRef]
47. Xiao, Y.; Yuan, Q.; Jiang, K.; He, J.; Lin, C.-W.; Zhang, L. TTST: A Top-*k* Token Selective Transformer for Remote Sensing Image Super-Resolution. *IEEE Trans. Image Process.* **2024**, *33*, 738–752. [CrossRef] [PubMed]
48. Xia, G.-S.; Hu, J.; Hu, F.; Shi, B.; Bai, X.; Zhong, Y.; Zhang, L.; Lu, X. AID: A Benchmark Data Set for Performance Evaluation of Aerial Scene Classification. *IEEE Trans. Geosci. Remote Sens.* **2017**, *55*, 3965–3981. [CrossRef]
49. Lei, S.; Shi, Z.; Mo, W. Transformer-Based Multistage Enhancement for Remote Sensing Image Super-Resolution. *IEEE Trans. Geosci. Remote Sens.* **2022**, *60*, 5615611. [CrossRef]
50. Chen, X.; Wang, X.; Zhou, J.; Qiao, Y.; Dong, C. Activating More Pixels in Image Super-Resolution Transformer. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Vancouver, BC, Canada, 18–22 June 2023.
51. Jiang, K.; Wang, Z.; Yi, P.; Chen, C.; Wang, Z.; Wang, X.; Jiang, J.; Lin, C.-W. Rain-Free and Residue Hand-in-Hand: A Progressive Coupled Network for Real-Time Image Deraining. *IEEE Trans. Image Process.* **2021**, *30*, 7404–7418. [CrossRef]