http://dx.doi.org/10.21622/ACE.2023.03.2.064

OBJECT DETECTION IN INLAND VESSELS USING COMBINED TRAINED AND PRETRAINED MODELS OF YOLO8

Ahmad A. Goudah[1], Maximilian Jarofka[2], Mohmed El-Habrouk[3], Dieter Schramm[4] and Yasser G. Dessouky[5]

[1,5] Arab Academy for Science, Technology and Maritime Transport

[3] Department of Electrical Engineering, Faculty of Engineering, Alexandria University, Alexandria, Egypt,

[1,2,4] Institute of Mechatronics University Duisburg-Essen D-47057 Duisburg, Germany,

Emails: { ahmad.goudah@aast.edu, maximilian.jarofka@uni-due.de, eepgmme1@yahoo.co.uk, dieter.schramm@uni-due.de, ygd@aast.edu}

Received on, 14 June 2023 - Accepted on, 11 September 2023 - Published on, 20 November 2023

ABSTRACT

One of the central challenges within the domain of computer vision is object detection, encompassing the identification and localization of specific entities within an image. Introducing a pioneering approach, the YOLO (You Only Look Once) algorithm emerged in 2015, executing object detection within a singular neural network. This innovation triggered a profound transformation within object detection, ushering in remarkable advancements beyond the capacities of the preceding decade. Subsequently, YOLO underwent successive iterations, culminating in eight versions that have earned prominent stature among leading object identification algorithms. This recognition is attributed to YOLO's integration of state-of-the-art concepts prevalent in the realm of computer vision research. Particularly noteworthy is the latest iteration, YOLOv8, which demonstrates superior performance in terms of both accuracy and speed when juxtaposed with YOLOv7 and YOLOv5. This study delves into the most recent strides in object detection as an important field of computer vision, which has been seamlessly assimilated into YOLOv5, YOLOv7, YOLOv8, and their antecedents. The introductory section, delineating the foundational importance of object detection, aligns seamlessly with the research's overall narrative. The elucidation of object detection's significance within diverse contexts, such as vehicle identification across varying scales and environments, underscores its multifaceted utility. The refinement process further enhances the discernment of YOLO's progression through its iterations, elucidating the evolution from the pre-eminent YOLOv1 to the recent apex represented by YOLOv8. Notably, the text now highlights YOLOv8's distinc- tive advancements in accuracy and speed over YOLOv7 and YOLOv5, lending heightened clarity to the incremental evolution of the algorithm. The augmentation extends to the exploration of YOLOv8's amalgamation with contemporary computer vision concepts. These concepts' incorporation is now underscored, demonstrating how YOLOv8 benefits from the strides made in computer vision research. The final passage captures the thrust of the research, examining the application of the developed object detection models within the specific context of inland waterway vessels. The distinct stages of detection, the addition of new classes, manual annotation, and the process of network training are now presented with greater precision,

ensuring a lucid understanding of the methodology. Moreover, the description of the combined model's competence to detect all 85 classes with a measure of accuracy enhances the comprehensiveness of the study's contributions.

Index Terms: Object Detection, YOLO, YOLO8, Autonomous

I. INTRODUCTION

BJECT detection is a well-known research topic that hasbeen extensively studied in computer vision systems. Object detection objective in many fields is to identifythe location and classify the objects that are of intereston the scene. The analysis of detection methods involvesthe utilization of different techniques to extract features that are primarily designed for detecting vehicle objects at multiple scales and in various environmental conditions. In this context, the term "object" typically refers to entities thatare either human-made or possess a high degree of structure(such as vehicles, buildings, ships, etc.), and which aredistinguishable from complex background environments andlandscapes. Over the past two decades, the improved accuracy of image interpretation in these applications has allowed themto fulfill the necessary requirements in real-world situations,which has in turn greatly advanced the development of Earthobservation technologies and object-detection methodologies.Many computer vision fields rely on moving object detectionas a fundamental research component. Over the last fewdecades, numerous detection methods have been suggested.Existing surveys have mostly concentrated on the accuracyof detection, but practical detection tasks were not takeninto account. However, in various application tasks, the training modes and requirements differ significantly. Moving object detection serves as the initial stage in numerous computer vision processes aimed at identifying movingobjects that are not part of a scene, known as the foreground. Afterward, the objects are isolated from the background through segmentation. Several intelligent monitoring tasksrely on moving object detection and foreground segments, including but not limited to target tracking, behavior analysis, traffic monitoring, visual surveillance, and human-machine interaction [1]–[4].

The field of computer vision is currently experiencing a widespread use of deep learning models, thanks to the development of Deep Convolutional Neural Networks (DC- NNs) and the increasing computational power of GPUs. Object detection aims to identify visual objects belonging to specific classes, such as TV/monitor, books, cats, humans, etc.,and determine their location by enclosing them in bounding boxes. Once located, these objects are then classified into their respective categories. Object detection is a task that involves detecting and categorizing a diverse range of objects within an image. It involves identifying the location of an object in an image, drawing a bounding box around it, and then determining the category it belongs to. Object detection also emphasizes the recognition of instances belonging to predefined categories. The advancement of object detection can be divided into two distinct historical phases. The period before 2014 was dominated by traditional methods, whereas the era following 2014 was characterized by the emergence of deep learning-based methods. The architectures of these two phases differ in terms of accuracy, speed, and hardware resources required. When compared to traditional techniques, Convolutional Neural Networks (CNNs) have superior architecture and are significantly more expressive, which contributes to their improved performance [5].

In this paper, different methods of object detection willbe -roughly- overviewed. YOLO (You Only Look Once) as a popular object detection method, will be compared -briefly-to other state-of-the-art object detection methods through some relevant studies. The rest of this paper will cover the application of detecting objects for inland waterway vessels through three stages and combined trained and pretrained model. The first one, is to detect the objects listed in the ready trained list from the coco dataset that contains 80 classes and includes objects like:(person, bird, boat, … and etc.). The next stage of this system is to add five classes which are more popular in the environment of the inland vessels through their waterway path and start to train and validate the output of a certain set. The selected classes which are added at this step are as follows:

- L Shore
- R Shore
- Bridge Pillar
- Crane
- Buoyer

After doing the steps of manual annotation and labeling, the images set will be divided to two sub-sets: train and validate sets. The training of the YOLO8 custom network is done and labels are generated accordingly. The third stage of the system here is responsible for combining the detected output from the ready pretrained YOLO8 network with the new output generated from the next level of the 5 classes trained network. Combined model will be able to detect all 85 classes with a measures' accuracy as shown on results part of this paper.

II. OBJECT DETECTION METHODS: HISTORY AND STATE OF THE ART

Object detection is a fundamental task in computer vision that involves detecting the presence and location of objects in an image or video. Over the years, various object detection methods have been developed, each with its own strengths and limitations. These methods will be roughly overviewed in the next part of this paper section, and they could be categorized as two main big groups, as [6] states:

A. Traditional Methods.

B. Deep Learning Based Methods.

A. TRADITIONAL METHODS

Traditional methods of object detection involve using com- puter vision techniques to analyze the image and extract in- formation about the objects within it. These methods typically involve a series of image processing steps, including feature extraction, object classification, and object localization. These steps are processed step by step as follows:

1) Feature extraction: The first step in traditional object detection involves extracting features from the image that can be used to identify and classify objects. Common fea- ture extraction techniques include edge detection, corner detection, and texture analysis.

2) Object classification: Once features have been extracted, the next step is to classify them into different categories. This can be done using a variety of machine learning algorithms, such as support vector machines (SVMs) or decision trees.

3) Object localization: The final step in traditional object detection is to localize the objects within the image. This typically involves identifying the position and size of the object relative to the image frame.

There are several traditional methods of object detection that are commonly used in computer vision research and ap- plications, as shown in figure 1. These methods are including:

1) Feature-based methods:

- SIFT (Scale-Invariant Feature Transform) features: SIFT features are a feature extraction technique that is commonly used in object detection and recognition. The technique involves identifying distinctive features in an image, such as corners and edges, and then matching those features to a database of known objects.SIFT detects and describes local features that are invariant to scale, rotation, and illumination changes. SIFT works by identifying key points in an image that are both stable and distinctive, and then describing the surrounding region of the image in a way that is invariant to rotation and scale changes. SIFT is quite accurate and robust to noise and illumination changes, but can be slow to compute and is patented, which maylimit its use in some contexts [7].

- HOG (Histogram of Oriented Gradients) features: HOG features are a feature extraction technique that involves calculating the gradients of an image and then constructing a histogram of the gradient orientations. HOG computes a histogram of gradient orientations in an image to capture edge information and shape features. This technique has been used to successfully detect objects in a variety of applications, including pedestrian detection and face detection [8].

- SURF (Speeded Up Robust Features): SURF features are similar to SIFT features in that they involve identi- fying distinctive features in an image. However, SURFfeatures are designed to be faster and more robustthan SIFT features, making them well-suited to real- time object detection applications. SURF detects and describes local features that are invariant to scale, rotation, and affine distortion using a modified Hessianmatrix. [9].

Fig. 1. Traditional Methods Based Object Detection Techniques.

http://dx.doi.org/10.21622/ACE.2023.03.2.064

- ORB (Oriented FAST and Rotated BRIEF): a fusion of the FAST (Features from Accelerated Segment Test) keypoint detector and the BRIEF (Binary Robust Independent Elementary Features) descriptor that is faster and more robust than SIFT and SURF. ORB is less accurate than SIFT, but is more robust to lighting changes and has a lower memory footprint. ORB is also free to use and does not have any patent restrictions. Overall, both SIFT and ORB are effective methods for object detection, but they differ in their trade-offs between accuracy, speed, and robustness. The choice of which method to use will depend on the specific application and the properties of the images being analyzed [10].

Detailed overview of the ORB algorithm and its ap- proach to object detection could be listed as:

a) Feature Detection: ORB starts by detecting features within an image, such as corners or edges, that are likely to be distinctive and repeatable. ORB uses a modified version of the FAST algorithm for feature detection, which is designed to identify features with high-contrast edges.

b) Feature Description: Once features are detected, ORB then extracts a compact binary descriptor for each feature. The descriptors are designed to be robust to changes in lighting and scale, making them suitable for object detection across a wide range of conditions.

c) Feature Matching: After descriptors have been ex- tracted for each image, ORB then matches the descriptors between the target object and the scene image. This is typically done using a nearest- neighbor search, where each descriptor in the target object is compared to descriptors in the scene image to find the best match.

d) RANSAC-based Pose Estimation: Once feature matches have been identified, ORB uses a robust estimation algorithm called RANSAC (Random Sample Consensus) to estimate the pose (position and orientation) of the target object within the scene. RANSAC iteratively samples subsets of the feature matches and computes the pose estimate for each subset, selecting the estimate with the highest number of inliers (matches that agree with the estimated pose) as the final estimate.

e) Object Localization: Finally, once the object pose has been estimated, ORB can localize the object within the scene by drawing a bounding box around the object and overlaying it on the scene image.

Overall, the ORB algorithm is an effective and efficient traditional method for object detection, especially in scenarios where deep learning approaches may not be feasible due to limited data or hardware resources. However, it may not perform as well as state-of-the-art deep learning methods in complex and varied scenes or with highly similar objects [11] [12].

2) Template matching-based methods:

- Template matching: Template matching is a simple method of object detection that involves comparing an image to a template image of the object being detected. If the two images match, the object is considered to be present in the image. Template matching compares a template image with a larger search image at different locations and scales to find the best match, using meth- ods such as cross-correlation or normalized correlation[13].

3) Edge-based methods: These methods are often used for real-time applications, as they are computationally effi- cient and can work well in low light conditions [14].

- Canny edge detector: One of the earliest and most widely used edge detection algorithms which was introduced by John Canny in 1986. The Canny detector applies a series of filters to an image to identify edges, and then uses non-maximum suppression and hysteresis thresholding to produce the final edge map. The algorithm has been shown to produce high-quality edge maps with low false positive rates. It detects sharp changes in intensity or color, which are then used to define the boundaries of objects [15].

- Sobel operator
- Prewitt operator
- Laplacian of Gaussian (LoG) filter

The Sobel and Prewitt operators are simple gradient- based edge detectors that compute the gradient mag- nitude of an image and threshold it to obtain an edge map. The LoG filter is a more sophisticated approach that convolves an image with a Gaussian filter followedby the Laplacian operator to identify edges [16] [17].

4) Contour-based methods:

- Contour-based methods are a type of image processing technique that focus on identifying and analyzing the edges or contours in an image. These methods are com- monly used in computer vision, pattern recognition, and object detection. As [18] proposes, new algorithm for contour detection that combines both bottom-up and top-down approaches to achieve more accurateresults.

5) Region-based methods:

- Selective search: generates a large set of region pro- posals in an image based on color, texture, and size similarity, which can then be classified as object or background regions. This algorithm involves identify- ing and localizing objects within an image by dividing the image into regions and analyzing each region for the presence of an object. These methods are com- monly used in computer vision applications such as object recognition and tracking [11].

6) Scale-invariant methods:

- SIFT, SURF, ORB: as described above. Scale-invariant methods are a type of object detection technique that aim to identify objects at different scales within an image. These methods are commonly used in computer vision applications such as object recognition, tracking,and surveillance [19].

7) Scale-space methods: refer to techniques that analyze images at different scales to detect objects of varying sizes. These methods are based on the principle that objects in images can appear at different scales due to their size, distance from the camera, and other factors. By analyzing images at multiple scales, scale-space methods can detect objects regardless of their size or location in the image. One popular scale-space approach is:

- The Laplacian of Gaussian (LoG) operator, which involves convolving the image with a Gaussian filter at different scales and then applying the Laplacian operator to the filtered image. The resulting imagehighlights regions with high intensity variations at different scales, which can be used to detect objects of different sizes [20].

8) Correlation-based methods: Correlation-based methods are a type of object detection technique that involve computing the correlation between an image and a tem-plate to identify objects. Correlation-based methods for object detection describe several variations of methods, including:

- Normalized cross-correlation (NCC): a method for comparing two images by computing their correlation coefficients at each pixel location to find the best match.

- Mean Squared Error (MSE) correlation
- Phase-only correlation (POC)

These methods are commonly used in computer vi- sion applications such as face recognition, tracking, and surveillance. correlation-based methods can be extended to handle variations in lighting, pose, and occlusion [21].

9) Frequency domain-based methods:

- Fourier-Mellin transform: a technique for matching objects in an image by analyzing their frequency content and phase relationships. Frequency domain-based methods are a type of object detection techniquethat analyze the frequency content of an image to identify objects. These methods are commonly used in computer vision applications such as image processing,texture analysis, and pattern recognition. Frequency- based method for object detection that uses a bank of Gabor filters to analyze the frequency content of an image. The method is able to detect objects at differentscales and orientations by analyzing the responses of the Gabor filters across the image. the effectiveness of this method on several object detection shows that it is able to achieve high accuracy with low computational complexity [22].

10) Color-based methods: Color-based methods are a typeof object detection technique that use color informationto identify objects in an image. These methods are commonly used in computer vision applications such as traffic monitoring, object tracking, and image retrieval.

- Color histograms: Represent the distribution of color values in an image using a histogram, which can be used to identify objects with specific color character- istics.
- Color Moments: are a type of feature extraction methodcommonly used in color-based object recognition and image retrieval. Color moments are statistical descrip- tors that capture the statistical properties of color distributions in an image. There are several types of color moments, including:
  - The first-order moments (mean)
  - Second-order moments (variance)
  - higher-order moments (skewness, kurtosis, etc.) These moments are computed separately for each colorchannel (e.g., red, green, blue) and can be combined into a feature vector to represent the color distribution of an image.Color moments are popular because they are simple to compute, invariant to translation and scal-ing, and can capture higher-order statistical properties of the color distribution. However, they can be sensitive to noise and may not capture spatial information about the object [23].
- Color Coherence Vectors: are a type of feature extrac- tion method used in color-based object recognition andimage retrieval. Color coherence vectors capture the spatial coherence of color distributions in an image by measuring the degree to which neighboring pixels havesimilar color values.

To compute color coherence vectors, an image is first segmented into regions or objects, and the color distribution of each region is represented as a color histogram. Next, the spatial coherence of the colordistribution is measured by computing the similarity between adjacent pixels or regions. This similarity measure can be based on various metrics, such as:

  - Euclidean distance
  - Mahalanobis distance

Volume 3, Issue 2, December 2023- ISSN 2735-5985
http://dx.doi.org/10.21622/ACE.2023.03.2.064

– correlation coefficient

The result of the computation is a set of vectors that represent the spatial coherence of the color distribution in each region. These vectors are called color coher- ence vectors and can be used as features for object recognition or image retrieval.

Color coherence vectors are effective in capturing the spatial coherence of color distributions and can handle variations in lighting, shadows, and object pose. However, they can be sensitive to image noise and may not capture global color information [24].

11) Texture-based methods: refer to techniques that utilize the texture or patterns in an image to detect objects. These methods are based on the principle that objects in images often have distinctive textures or patterns that can be used to identify them. Texture-based methods are typically used in scenarios where the objects of interest have similar colors or shapes to the background, making it difficult to detect them using traditional color or shape-based methods.

One popular texture-based approach is:

- The Local Binary Pattern (LBP) operator: which in- volves comparing the intensity values of pixels in an image with their neighboring pixels and encoding the results as binary patterns. The resulting patterns can be used to detect texture variations in the image and identify objects with distinctive textures. that works by computing a binary pattern for each pixel in an image based on the values of its surrounding pixels. The Local Binary Pattern (LBP) operator binary patterns are then used to describe the texture of the image. LBP can be used for object detection by comparing the texture of the object to the texture of the background. [25].

12) Motion-based methods: refer to the techniques that utilize the motion of objects to identify and track them in video sequences. These methods are typically used in scenarios where objects are moving in a dynamic environment and traditional static object detection techniques may not be sufficient.

One popular motion-based approach is:

- Optical flow: which computes the displacement of pixels between consecutive frames in a video sequence. Optical flow can be used to track objects by detecting the regions where the flow vectors are consistent over time. Another approach is background subtraction, which involves subtracting a background image from each frame in the video sequence to highlight moving objects [26].

In conclusion, traditional methods for object detection have their own strengths and weaknesses, and the choice of method depends on the specific application and the nature of the objects to be detected.

While deep learning-based methods have achieved state- of-the-art performance in many areas, traditional methods are still useful in situations where data or hardware resources are limited. Table I: shows the comparison of grouped traditional methods of object detection into different categories and lists their pros and cons.

B. DEEP LEARNING BASED METHODS

There are many Convolutional Neural Networks architec- tures which are developed by the time to solve object detection problem in a good, accurate and fast way. These architectures could be listed as:

4) Region-based Convolutional Neural Networks (RCNNs): RCNNs are one of the earliest and most popular object detection methods [43]. They use a two-stage approach, where the first stage proposes a set of object regions in the

image, and the second stage classifies each region as containing an object or not [52] [11]. The most famous RCNN models are Faster RCNN, RFCN and Mask RCNN, which have achieved state-of-the-art results on various benchmark datasets [52] [53].

5) Single-Shot Detectors (SSDs): SSDs are a one-stage ap- proach that simultaneously predicts the class and location of objects in an image [44]. Unlike RCNNs, they do not require a separate region proposal step, making them faster and more efficient [47]. Some of the popular SSD models are YOLO (You Only Look Once), RetinaNet, and EfficientDet [48]. EfficientDet is a family of object detection models developed by Google Research that uses efficient architectures and training techniques to achieve high accuracy while using fewer computational resources. It achieves state-of-the-art performance on several bench- mark datasets at 2020 [48].

6) Anchor-Free Detectors: Anchor-free detectors are a new class of object detectors that do not rely on predefined anchor boxes for object localization [54]. Instead, they use a set of learnable points to predict object locations and sizes [55]. Some examples of anchor-free detectors are CornerNet, FCOS (Fully Convolutional One-StageDetector), and RepPoints [56].

7) Transformer-based Detectors: Transformer-based object detectors use self-attention mechanisms to capture long- range dependencies between different parts of an image [50]. These models have achieved impressive results on various object detection benchmarks. Some popular transformer-based detectors are DETR (DEtection TRans- former), Deformable DETR, and Sparse R-CNN [51].

8) Hybrid Approaches: Hybrid object detectors combine multiple detection methods to improve detection accuracyand efficiency. For example, CenterNet combines the effi- ciency of SSDs with the accuracy of RCNNs [57], while Cascade RCNNs use multiple stages of classification to refine object detection results [58].

As figure 2 shows, these different types of object detection methods which are depending on Deep Learning Networks, could be summarized and discussed as follows:

1) Two-stage detectors: These methods include Faster R- CNN [43], R-FCN [59], and Mask R-CNN [60].

2) One-stage detectors: Examples of these methods include YOLO [44], SSD [46], and RetinaNet [47].

3) Multi-scale detectors: These methods include FPN [47], RetinaNet [47], and Cascade R-CNN [58].

4) Anchor-based detectors: Examples of these methods in- clude Faster R-CNN [43], RetinaNet [47], and YOLOv3 [61].

5) Anchor-free detectors: Examples of these methods in- clude FCOS [55], CornerNet [54], and CenterNet [62].

6) 3D object detectors: These methods include PointNet [63], MV3D [64], and AVOD [65].

7) Video object detectors: Examples of these methods in- clude Tube-CNN [66], SlowFast [67], and SiamRPN [68].

8) Few-shot object detectors: Examples of these methods include Meta R-CNN [69], FSOD [70], and Few-Shot Object Detection via Feature Reweighting [71].

 Volume 3, Issue 2, December 2023- ISSN 2735-5985
http://dx.doi.org/10.21622/ACE.2023.03.2.064

III. OBJECT DETECTION TECHNIQUES AND DEVELOPMENT TIMELINE

Object detection is an active area of research and develop- ment, with many new methods and techniques emerging on a regular basis. This section of the paper covers the developmenttimeline of Object Detection techniques briefly to give the pig picture needed when studying this important and widely used field of computer vision. a brief overview of some of the major milestones in the development of object detection methods over the past few decades could be listed through the time lime as:

- 1990s: The first object detection methods based on hand-crafted features, such as Histograms of Oriented Gra-dients (HOG) and Haar-like features, were introduced. These methods used traditional machine learning algo-rithms, such as Support Vector Machines (SVM), to detect objects in images [29].

- 2000s: The use of deep neural networks for object detection began to gain popularity. The seminal work on this topic was the Viola-Jones algorithm, which useda boosted cascade of simple classifiers to achieve real- time face detection [72]. In the later years of the decade, methods such as Deformable Part Models (DPM) and Fast R-CNN: a Fast Region-based Convolutional Network were introduced, which combined deep neural networks with traditional machine learning techniques to improve accuracy and speed [73] [42].

TABLE I
HISTORY OF SOME FAMOUS TRADITIONAL OBJECT DETECTION METHODS WITH PROS AND CONS

Method	Year	Pros	Cons	Refer-ences
Feature-based methods	2001	Robust to occlusion. Can detect objects in cluttered environments.	Sensitive to noise. Com- putationally expensive.	[27]
	1999	Can detect objects of vari- ous sizes and orientations.Can handle occlusion and cluttered scenes.	Sensitive to image noise and illumination changes. Computational complexity is high.	[28]
Color-based methods	1999	Can detect objects based on color information.	Limited to objects with distinctive colors. Sensi- tive to changes in illumi-nation.	[29]
Region-based methods	1998	Can detect objects of vari-ous sizes and shapes. Can handle occlusion and clut-tered scenes.	Computationally expensive. Sensitive to image noise and illumination changes.	[30]
Scale-invariant methods	1998	Can detect objects at dif-ferent scales.	Sensitive to image noise and illumination changes. Computationally expensive.	[31]

Contour-based methods	1995 1994	Can detect objects with well-defined contours. Computationally efficient. Can handle objects with smooth boundaries. Can handle occlusion and clut-tered scenes.	Sensitive to noise. Can fail to detect objects with poorly defined contours. Sensitive to noise and poor edge detection in cluttered scenes. Compu-tationally expensive.	[32] [33]
Scale-space method	1987	Can detect objects at dif-ferent scales.	Computationally expensive.	[34]
Template match-ing-based methods	2000 1983	Simple and easy to imple- ment. Can detect objects in cluttered environments. Simple and efficient. Can detect objects in cluttered scenes.	Sensitive to changes in lighting and viewpoint. Limited to detecting ob-jects with a similar ap- pearance to the template image. Limited to specific objects and can't handle varia- tions in scale and orienta-tion.	[35] [36]
Texture-based methods	1983	Can detect objects based on texture information.	Limited to objects with distinctive textures. Sensi- tive to changes in illumi-nation.	[37]
Motion-based methods	1980	Can detect moving objects in video sequences.	Limited to moving ob- jects. Sensitive to camera motion.	[38]
Correla-tion-based methods	1980	Can detect objects in clut-tered scenes.	Sensitive to image noise and illumination changes. Limited to specific ob- jects.	[39]
Edge-based methods	1986 1979	Can detect objects with well-defined edges. Com- putationally efficient. Can detect objects with sharp boundaries. Compu-tationally efficient.	Sensitive to noise. Can fail to detect objects with poorly defined edges. Sensitive to noise and poor edge detection in cluttered scenes.	[40] [40]
Frequency domain-based methods	1987	Can detect objects based on their frequency con- tent and handle occlu-sion and cluttered environ- ments. Can handle noisy images and detect periodic patterns objects.	Computationally expensive. Sensitive to changes in lighting and viewpoint. Limited to specific objects. Can't handle n o n - periodic patterns.	[41]

http://dx.doi.org/10.21622/ACE.2023.03.2.064

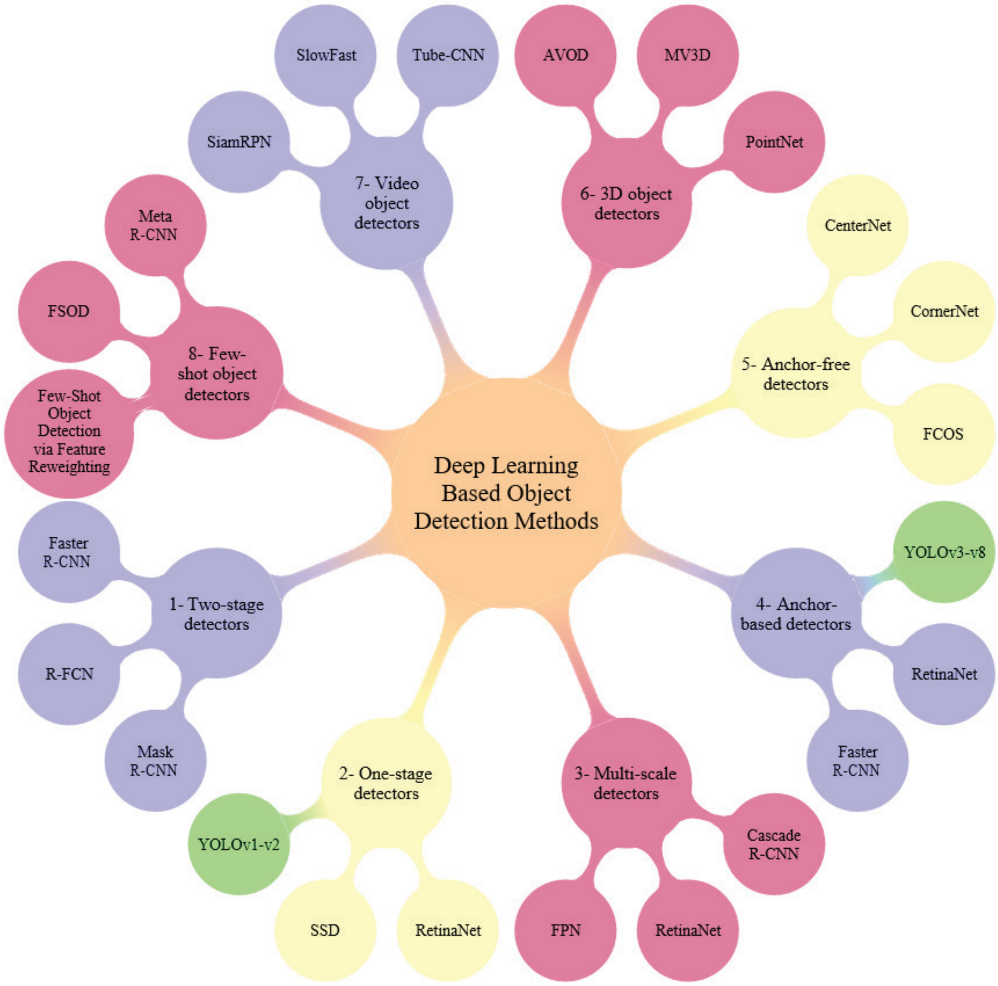

Fig. 2. Deep Learning Based Object Detection Methods.

- 2010s: This decade saw the rise of two-stage object de- tection methods, such as Faster R-CNN: Region Proposal Networks (RPN) and Mask R-CNN, which separated ob- ject proposal generation from object classification. Faster R-CNN: Region Proposal Networks (RPN) are used to generate region proposals, and a Fast R-CNN networkis used for object detection. These methods achieved state-of-the-art performance on benchmark datasets such as COCO and Pascal VOC [43]. Mask R-CNN: addsa branch to Faster R-CNN for predicting object masks in addition to bounding boxes and class labels [60]. In the later years of the decade, single-stage methods such as YOLO (You Only Look Once) and SSD (Single Shot Detector) were introduced, which achieved real-time performance by directly predicting object bounding boxesand class labels in a single pass of the network. [44] [46].

- 2020s: In recent years, the focus of object detection research has shifted towards improving efficiency and robustness. Methods such as EfficientDet (starting by EfficientDet-D0 through EfficientDet-D7 models) [74] and DETR (DEtection TRansformer) have achieved state- of-the-art performance on benchmark datasets while re- ducing the computational cost and improving the gen- eralization ability of the models. Additionally, there has been increasing interest in using self-supervised and un- supervised learning methods for object detection, as these approaches can leverage large amounts of unlabeled data to improve performance. In this decade, many algorithms have been introduced to be a normal result for the tedious and continues work to achieve better and better performance, For example:

- YOLOv4: an improvement over YOLOv3, achieving high accuracy with faster inference time [75].

- YOLOv5: is a family of object detection models de- veloped by Ultralytics that builds on the success of the previous YOLO models. YOLOv5 uses a novel neural network architecture and training techniques to achieve state-of-the-art performance on several bench- mark datasets [45].

- SpineNet: a family of object detectors that use a new architecture to achieve high accuracy with efficient computation and memory usage. It is developed by Facebook AI Research that uses a novel neural network architecture to achieve high accuracy while using fewercomputational resources. It achieves state-of-the-art performance at 2020 on several benchmark datasets [49].

- DETR(Detection Transformer): is a novel object detec- tion architecture that uses a transformer-based neural network to perform object detection. It achieves state- of-the-art performance at 2020 on several benchmark datasets while using a simple and unified approach[50].

- Deformable DETR: is an extension of the DETR architecture in 2021 that uses deformable convolutional layers to perform feature extraction. It achieves state- of-the-art performance on several benchmark datasets while improving the detection of small objects [51].

- Detectron2: is a popular open-source object detec- tion framework developed by Facebook AI Research (FAIR). It builds on the success of the original De- tectron framework and offers a modular and flexible platform for developing and training state-of-the-art object detection models at 2020. Detectron2 supports a wide range of model architectures and can be easily customized for different tasks and it provides a user- friendly interface and extensive documentation. On the other hand, it requires significant computational resources for training and inference and it may have a steeper learning curve for users unfamiliar with the PyTorch framework [76] [77].

- Sparse R-CNN: is an object detection method that uses a sparse convolutional neural network to perform feature extraction. It achieves state-of-the-art perfor- mance on several benchmark datasets at 2021 while using fewer computational resources compared to other methods. It achieves high accuracy with fewer compu- tational resources and can be fine-tuned for specific tasks. Furthermore, it is highly modular and it can be easily customized for different tasks. On the other hand, it may require specialized hardware to achieve real-time performance and it can be sensitive to the choice of hyperparameters [78].

TABLE II
HISTORY OF SOME FAMOUS DEEP LEARNING-BASED OBJECT DETECTION METHODS WITH PROS AND CONS

Method	Year	Pros	Cons	Refer- ences
R-CNN (Re- gion-based Con- volutional Neural Networks)	2014	Good detection ac- curacy, high recall, good for com- plex images.	Slow training and testing, re- quires selective search for re- gion proposals, not real-time.	[11]

http://dx.doi.org/10.21622/ACE.2023.03.2.064

Fast R-CNN	2015	Faster than R-CNN, end- to-end training, good de- tection accuracy.	Still requires re- gion pro-posals, not real-time.	[42]
Faster R-CNN	2015	Faster than R-CNN and Fast R-CNN, end-to-end training, good detection accuracy.	Still requires re- gion pro-posals, not real-time.	[43]
YOLO (You Only Look Once)	2016	Real-time, good detection accuracy, and end-to-end training.	Can miss small objects, and less accurate than other methods on complex images.	[44]
YOLOv5	2020	Achieves high accuracy with faster inference time compared to previous YOLO models. Has a lightweight architecture that requires fewer computational resources. Can be fine-tuned for specific tasks.	May not be as accurate as some other methods. May require specialized hard- ware to achieve real-time performance.	[45]
SSD (Single Shot MultiBox Detector)	2016	Real-time, good detection accuracy,and end-to-end training.	Can miss small objects, less accurate than other methods on complex images.	[46]
RetinaNet	2017	Good detection accuracy for small objects, less prone to false negatives.	Can miss large objects, slower than some other methods.	[47]
EfficientDet	2020	Achieves high accuracy with less computational resources. Has a good balance between accuracy and efficiency. Can be fine-tuned for specific tasks.	May require specialized hardware to achieve real- time performance. Can be sensitive to hyperparameters and initialization.	[48]
SpineNet	2020	Achieves high accuracy with less computational resources. Has a good balance between accuracy and efficiency. Is highly modular and can be eas- ily customized for differ- ent tasks.	May require specialized hardware to achieve real- time performance. Can be sensitive to hyperparameters and initialization.	[49]
DETR (DEtection TRansformer)	2020	No need for region pro- posals, end-to-end train- ing, good accuracy for small objects, efficient.	Less accurate for large objects, slower than some other methods.	[50]

http://dx.doi.org/10.21622/ACE.2023.03.2.064

Deformable DETR	2021	Achieves high ac- curacy with im- proved detection of small objects. Can de- tect objects of different sizes and shapes. Is highly modular and can be eas- ily customized for differ- ent tasks.	May require more com- puta- tional resources com- pared to other methods. Can be sensitive to the choice of hyperparame- ters.	[51]
DETR (DEtection TRansformer)	2020	No need for region pro- posals, end-to-end train- ing, good accuracy for small objects, efficient.	Less accurate for large ob- jects, slower than some other methods.	[50]

In summary, there has been significant progress and im- provement in object detection over the past few years. The state-of-the-art in object detection has been rapidly advancing in recent years, with new methods continually emerging that achieve better accuracy, efficiency, and robustness. Overall, these neural network-based methods of object detection repre- sent significant advancements in the field and offer a range of pros and cons depending on the specific application and use case, As shown in Table II.

Furthermore, there is an ongoing process of discovery and innovation in the field of object detection, with researchers and developers constantly exploring new approaches and tech-niques. Through these techniques:

- Accuracy: The ability of an object detection system to correctly identify objects in an image or video is extremely increased.
- Efficiency: The speed and computational resources re- quired to perform object detection
- Robustness: The ability of an object detection system to perform well under various conditions, such as changes in lighting, camera angle, or object orientation.

The ability of an object detection system to correctly identify objects in an image or video is extremely increased, and new methods and techniques are being developed that areable to achieve these goals more effectively than previous approaches. In addition, a number of benchmark datasets have been instrumental in advancing the field of object detection. These include the Caltech [79], KITTI [80], ImageNet [81], PASCAL VOC [82], MS COCO [83], and Open Images V5 [84] datasets. Recently, a new drone-based dataset was introduced as part of the ECCV VisDrone 2018 contest [85]. This dataset is comprised of a significant amount of images and videos captured from a drone platform [86].

IV. YOLO (YOU ONLY LOOK ONCE) OBJECT DETECTION ALGORITHM: HISTORY AND STATE OF THE ART

YOLO (You Only Look Once) is a popular object detectionalgorithm that has undergone several iterations since its initial release. Furthermore, it is concluded from the previous sectionwith its subsections and from [74], that the primary single- stage detection networks are:

- Single Shot multiBox Detector (SSD) [87].
- EfficientDet [88].
- You Only Look Once (YOLO) series of networks [89]. Although SSD has

good detection accuracy, it lacks sufficient low-level feature convolution layers, resulting in inadequate feature extraction, which makes it less sensitive to small target detection. EfficientDet-D0 through EfficientDet-D7 modelscan achieve higher accuracy, but this comes at performane weakness factors such that the cost of higher memory con- sumption and slower inference. In [90], YOLOv5 was used for some object detection applications (palm tree detection) using UAV images and was quantitatively compared with main-stream networks such as YOLOv3, YOLOv4, and SSD300, with YOLOv5 demonstrating the best accuracy. The YOLOv5 network adjusts the perceptual field size and enhances the feature extraction ability, indicating its potential for detection in regions with high canopy coverage [74]. Here are the key differences between each version:

1) YOLOv1: This was the first version of YOLO released in 2015. It used a single deep neural network to perform both object classification and localization in a single stage. It achieved competitive accuracy and speed on the PASCAL VOC 2007 detection dataset. However, it suffered from low recall due to the "grid cell" structure, which made it difficult to detect small objects [44].

2) YOLOv2: Released in 2016, YOLOv2 addressed some of the limitations of the original version. It introduced anchor boxes, which improved the detection of small objects, and a multi-scale feature extraction network, which improved performance on objects of different sizes. It also incorporated batch normalization and residual con- nections for improved training and performance. further- more, it addressed the shortcomings of the first version. It achieved a very good results on multiple detection datasets and improved the accuracy and speed of the algorithm [91].

3) YOLOv3: The third version of YOLO further improved the algorithm by introducing various enhancements such as multi-scale prediction, feature pyramid networks, and improved training techniques. It achieved state-of-the-art results on multiple detection benchmarks and improved the accuracy, speed, and robustness of the algorithm. YOLOv3 is released in 2018, YOLOv3 introduced several improvements over YOLOv2. It increased the number of anchor boxes, improved the feature extraction network, and introduced skip connections to help detect small objects. It also introduced the concept of "darknet-53," a more complex feature extraction network that improved detection accuracy [61].

4) YOLOv4: Released in 2020, It introduced a number of new features, including the "CSPDarknet-53" archi- tecture, which improved feature extraction and training efficiency. It also introduced the "Mish" activation func- tion, which outperforms traditional activation functions like ReLU. YOLOv4 also introduced a variety of train- ing techniques, including self-adversarial training, Drop-Block regularization, and improved data augmentation. The fourth version of YOLO introduced a number of enhancements such as a CSPDarknet53 backbone, spatial attention, and dynamic anchor assignment, among others. It achieved fabulous results on multiple detection bench- marks and improved the accuracy, speed, and efficiency of the algorithm at 2020, as mentioned in [75].

5) YOLOv5: Released in 2020, YOLOv5 is an independent project by Ultralytics that is based on the YOLOv4 architecture. It introduced a number of improvements, including a redesigned backbone network, a "Swish" activation function, and improved anchor box placement. It also incorporated a new training pipeline that made it easier to train on custom datasets [92].

6) YOLO Nano: Released in 2020, YOLO Nano is a lightweight version of YOLO designed for resource-constrained devices. It achieves real-time performance on mobile devices with limited computing power by using a simplified architecture and fewer layers. It was presented at the 2020 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays [93].

http://dx.doi.org/10.21622/ACE.2023.03.2.064

7) YOLOv5s, YOLOv5m, YOLOv5l, YOLOv5x: Ultralytics has released several versions of YOLOv5 with varying levels of computational complexity. These versions have different backbone architectures and feature extraction networks, allowing users to choose the version that best suits their needs based on the available hardware re- sources. The fifth version of YOLO introduced a number of significant improvements such as a new architecture, self-attention, and a novel anchor-free object detection method. It achieved state-of-the-art results on multiple detection benchmarks and improved the accuracy, speed, and simplicity of the algorithm [92] [94] [95] [96] [97].

8) YOLOv6: YOLOv6 is verified to bring more improve- ments to the YOLO architecture. The sixth version of YOLO introduced various improvements such as custom architecture search, scaled-YOLOv4, and ensemble of models. It achieved results on multiple detection bench- marks and improved the accuracy, speed, and efficiency of the algorithm [98].

9) YOLOv7: The seventh version of YOLO introduced a trainable bag-of-freebies (BoF) module that can be added to any object detection architecture to improve its per- formance. It achieved state-of-the-art results on multiple detection benchmarks and improved the accuracy, speed, and efficiency of the algorithm [99].

10) YOLOv8: The eighth and most recent version of YOLO introduced various improvements such as deformable convolutional networks, Scaled-YOLOv5, and ensemble of models with different input resolutions. It achieved state-of-the-art results on multiple detection benchmarks and improved the accuracy, speed, and efficiency of the algorithm. The YOLOv8 network is utilized for address- ing classification, object detection, and image segmen- tation challenges. These various approaches enable the identification of objects in images or videos through distinct means. In addition to providing the object type and probability, the neural network for object detection also outputs the coordinates of the object on the image, including its x and y position, width, and height. This information is demonstrated in the second image. More- over, object detection neural networks have the capability to identify multiple objects in an image and determine their respective bounding boxes [100].

Furthermore, it is noted that some of the versions mentioned above are not officially released by the original authors of YOLO and might have some variations or differences in their implementation.

V. YOLO 5,7, AND 8 DETECTION RUN COMPARISON FOR OBJECT DETECTION PROCESS OF VEHICLES AND INLAND VESSELS SCENCES

The YOLO model is currently the most widely used real- time object detector due to its lightweight network architecture, effective feature fusion methods, and more accurate de- tection results. Among the YOLO algorithm variants, YOLOv5 and YOLOv7 have gained significant acceptance in current usage.

YOLOv5 utilizes deep learning technology to achieve real- time and efficient object detection tasks. It improves upon YOLOv4 by enhancing the model structure, training strategy, and overall performance. YOLOv5 adopts the CSP (Cross- Stage Partial) network structure, which effectively reduces redundant calculations and enhances computational efficiency. However, YOLOv5 still has room for improvement in detect- ing small objects and dense object scenarios, as well as in handling complex situations like occlusion and pose changes. YOLOv7 introduces a novel training strategy, at that time (2022), called Trainable Bag of Freebies (TBoF) to enhance the performance of real-time object detectors. TBoF incor- porates various trainable tricks, including data augmentation and MixUp, which significantly improve the accuracy and generalization ability of object detection. However, YOLOv7 is constrained by training data, model structure, and hy- perparameters,

http://dx.doi.org/10.21622/ACE.2023.03.2.064

leading to performance degradation in some cases. Additionally, the proposed method requires more com- putational resources and training time to achieve optimal performance [101]. Figure V, shows the YOLO Publications Timeline before YOLOv8.

In this section, a comprehensive evaluation is conducted on selected versions of the YOLO platform applied to a specific selected dataset sample of images. A subset of representative images is carefully chosen to determine the most effective YOLO versions for efficient object detection. The selection criteria consider the diversity of scenes, encompassing chal- lenging scenarios where objects may be combined or difficult to detect. The dataset is thoughtfully balanced, featuring wa- terway scenes for inland vessels, street traffic views capturing vehicles, pedestrians, birds, horses, and traffic lights, among others.

The results of the runs reveal distinct identification out- comes for each object in all selected versions of YOLO. A total of thirty-three (33) images, as illustrated in Figure V, are carefully curated, comprising 25 images related to inland vessels scenes and 8 particularly challenging images representing street traffic views with multiple objects. The subsequent part of this section delves into the detailed analysis of the object detection results, highlighting the capabilities of each selected YOLO version in various challenging conditions.

The selected dataset images, as shown in figure V, are selected as:

- 25 images for inland vessels related images.
- 8 (tricky) images are selected for the other (street traffic views for vehicle, pedestrians, one for a bird, one for a horse, traffic lights, and etc.)

The rest part of this section will cover the different output and shows the ability of each version of the selected ones to detect object in a various set of different conditions.

A. COMMON OBJECTS IN CONTEXT DATASET (COCO)

The Common Objects in Context (COCO) dataset has emerged as a significant milestone in the field of computer vision, specifically for object recognition and detection tasks [83]. With its vast collection of images and meticulously annotated objects, the COCO dataset provides researchers and developers with a standardized benchmark to evaluate and advance the performance of object detection algorithms [102]. By offering a diverse range of object categories and complex scenes, COCO has become an invaluable resource for training and testing state-of-the-art models, pushing the boundaries of object recognition and detection capabilities.

Object recognition and detection form the fundamental building blocks of many computer vision applications, includ- ing autonomous driving, surveillance systems, robotics, and augmented reality. The ability to accurately identify and local- ize objects within an image is crucial for understanding visual scenes and enabling intelligent decision-making. However, this task is inherently challenging due to variations in object appearance, scale, occlusion, and cluttered backgrounds.

The COCO dataset addresses these challenges by providing a large-scale and diverse collection of images spanning 80 object categories, as shown in table IV [83].

These categories encompass a wide range of everyday objects such as people, animals, vehicles, household items, and more. The dataset consists of over 200,000 images from complex real-world scenes, covering a diverse set of visual contexts and capturing a wide variety of object instances. Each image is meticulously annotated with multiple objects bounding boxes, segmentations, and corresponding category labels, enabling fine-grained analysis and evaluation of object detection algorithms [83].

http://dx.doi.org/10.21622/ACE.2023.03.2.064

The COCO dataset is a comprehensive and extensive re- source for object detection, segmentation, and captioning tasks. It encompasses a range of notable features, including [83]:

1) Object segmentation: The dataset provides precise de- lineation of objects through segmentation, allowing for pixel-level ground truth information.

2) Recognition in context: COCO captures objects within their surrounding scenes, enabling recognition in diverse visual contexts and promoting a better understanding of object interactions.

3) Super pixel stuff segmentation: The dataset includes detailed annotations for stuff segmentation, providing insights into the distribution and boundaries of regions like sky, water, and vegetation.

4) Scale and diversity: COCO comprises a substantial col- lection of 330,000 images, encompassing a wide variety of scenes and object instances. This large-scale nature allows for robust training and evaluation of models.

5) Abundance of object instances: With approximately 1.5 million annotated object instances, the COCO dataset offers a rich variety of examples for each object cate- gory, facilitating comprehensive analysis and algorithm development.

6) Extensive object and stuff categories: The dataset covers 80 object categories, including people, animals, vehicles, and household items, providing a broad range of objects to detect and recognize. Additionally, it includes 91 stuff categories, capturing various contextual elements.

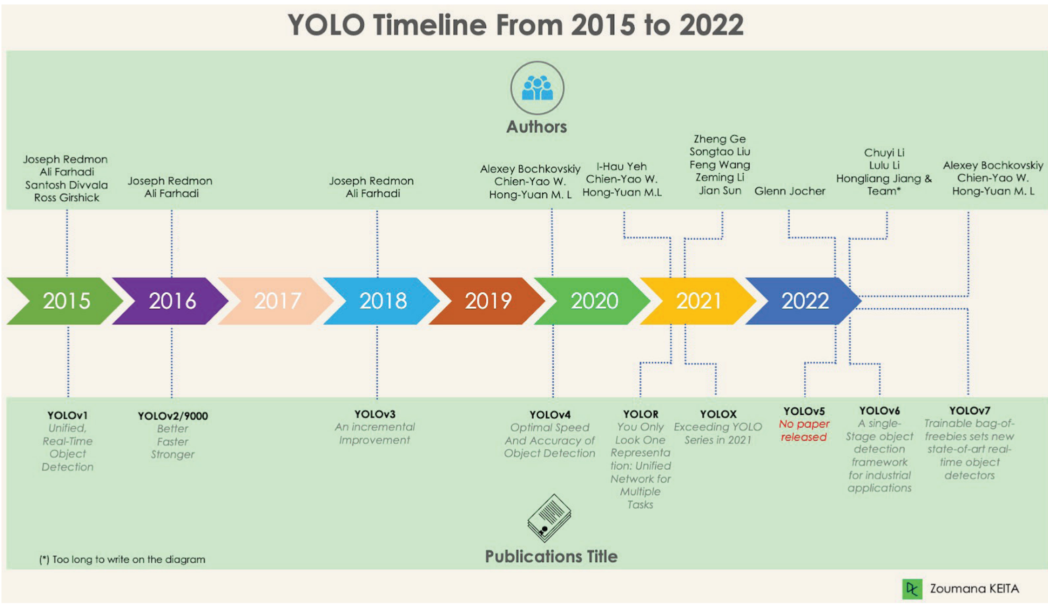

Fig. 3. YOLO Publications and Authors Timeline from 2015 to 2022 (Zoumana KEITA)

7) Caption annotations: Each image in the COCO dataset is associated with five captions, enabling research and development in image captioning and language under- standing tasks.

8) Key points annotations: COCO provides annotations for key points, specifically for approximately 250,000 peo- ple, facilitating the development and evaluation of pose estimation algorithms.

The combination of these features makes the COCO dataset a highly valuable resource for advancing research in computer vision, enabling the development

http://dx.doi.org/10.21622/ACE.2023.03.2.064

and evaluation of state-of- the-art models across multiple tasks, including object detec- tion, segmentation, captioning, and pose estimation. One of the distinguishing features of the COCO dataset is its emphasis on accurately localizing objects of interest. In addition to category labels, each annotated object is precisely delineated using segmentation masks, providing pixel-level ground truth information [83]. This level of detail allows researchers to explore advanced detection techniques that go beyond simple bounding box estimation, enabling more precise object local- ization and instance segmentation.

The availability of such a rich and comprehensive dataset has fostered significant advancements in object recognition and detection. Researchers and developers have leveraged the COCO dataset to train and benchmark state-of-the-art models, resulting in remarkable progress in object detection accuracy and efficiency [103] [104]. The dataset's widespread adoption has fueled the development of sophisticated algorithms, in- cluding deep learning approaches, which have demonstrated exceptional performance in detecting and recognizing objects across a wide variety of challenging scenarios [105] [106].

Furthermore, the COCO dataset has facilitated the develop- ment of robust and generalizable object detection frameworks. By training models on COCO, researchers can leverage the dataset's diversity to enhance the models' ability to handle complex scenes, occlusion, and small object instances [105]. The evaluation metrics provided by COCO, such as aver- age precision (AP) and intersection over union (IoU), offer standardized benchmarks for comparing the performance of different object detection algorithms, promoting fair and objec- tive evaluations [102]. The COCO dataset has revolutionized the field of object recognition and detection by providing a comprehensive benchmark for training, testing, and evaluating algorithms. Its large-scale collection of images, diverse object categories, precise annotations, and detailed evaluation metrics have accelerated progress in the development of state-of-the- art models. As computer vision applications continue to evolve and demand increasingly accurate and robust object detection capabilities, the COCO dataset will undoubtedly remain a vital resource in driving advancements and pushing the boundaries of object recognition technology [83].

http://dx.doi.org/10.21622/ACE.2023.03.2.064

Fig. 4. Dataset samples for vehicles with various behavior patterns, including mixed traffic situations and inland waterway vessels

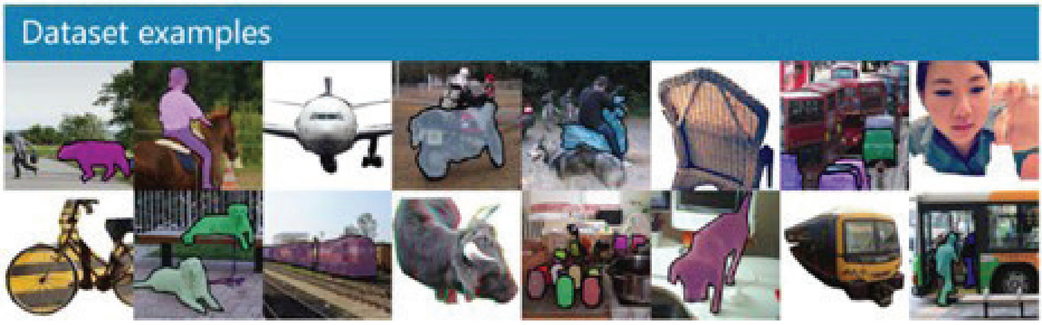

Fig. 5. COCO Objects Examples [83]

B. APPLYING YOLOV5 X, L, S, AND M MODELS FOR THE SELECTED IMAGES SET.

YOLOv5 provides a range of options with four available models: s, m, l, and x. Each model offers distinct levels of de- tection accuracy and performance, as illustrated in figure V-B Although, there are more models for YOLOv5 but they are dealing with the images width of 1280 pixels. All considered models here to be applied is 460 pixels in size. YOLOv5s,

YOLOv5m, YOLOv5l, and YOLOv5x is used here with the selected 33 images set to test the ability of the system to detect the different objects pretrained in COCO dataset 80 classes.

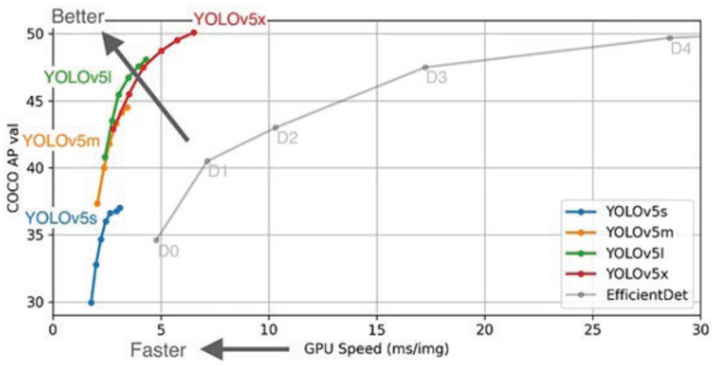

Fig. 6. YOLOv5 Four Models s, m, l, and x Accuracy and Performance (Source: https://github.com/ultralytics/yolov5)

This subsection here aims to analyze and compare the differences between s, m, l, and x models of YOLOv5, employing the COCO 80 classes model. The resulting output provides valuable insights into the variations and characteris- tics exhibited by these weight models in terms of their object detection capabilities. Detecting of objects in thirty-three (33) images by YOLOv5 models x and l are applied. The utilization of YOLOv5 models x and l yields the capability to accurately detect objects within the sample of the 33 images. Employing YOLOv5 models x and l brings forth a multitude of advantages tailored to precise object detection within the realm of 33 images. Model x excels in achieving faster inference times, allowing for real-time detection in dynamic scenarios. Its optimized architecture efficiently balances accuracy and speed, making it ideal for applications requiring swift decision-mak- ing. On the other hand, model l boasts exceptional accuracy in object localization and recognition. Its deeper architecture and refined feature extraction enable it to discern intricate details within the images, ensuring a high level of detection precision, especially in complex scenes with overlapping objects or varying scales. By harnessing the strengths of both models x and l, the overall detection process becomes comprehensive and robust, encompassing scenarios that demand both speed and precision in object identification within the set of the 33 images

Detecting of objects in 33 images by YOLOv5 models sand m are applied. YOLOv5 model s is designed with a focus on speed and efficiency. Its architecture is optimized for rapid inference times, enabling real-time object detection in dynamic scenarios. This advantage is particularly valuable in applica- tions requiring quick decision-making and tracking, such as video surveillance and robotics. Model s efficiently balances speed and accuracy, making it well-suited for scenarios where timely detection is essential. On the other hand, YOLOv5 model m strikes a balance between speed and accuracy. With a slightly deeper architecture and enhanced feature extraction, it excels in accurately localizing and recognizing objects within images. This accuracy advantage is especially useful in scenarios with complex scenes, overlapping objects, or varying object scales. Model m's versatility makes it suitable for a wide range of applications, including autonomous vehicles, industrial automation, and quality control. By leveraging the strengths of both models s and m, the overall object detection process is enriched, catering to scenarios that demand different priorities in terms of speed and accuracy within the specified previous sample set of 33 images

The remaining part of this subsection displays several images generated by four YOLOv5 models: x, l, s, and m. The detected objects in these images are enclosed within bounding boxes and labeled with the class assigned by the respective model. Additionally, an F1 score will be calculated on this sample test to demonstrate the performance and accuracy of these models.

Figures 7 (a), (b), (c), and (d) display the output achieved of applying YOLOv5 model x, l, s, and m respectively, and show true positive detection instant for one bird and false positive for the other one in some cases. Furthermore, the existence of showing multiple labels for the same object by two different classes, one is true which is "car" and the other one is false as well and it is "boat". The existence of water near to the car do this sort of ambiguity and the presence of some wood bars in front of the car, leads to detect the wood as bench in model m, as shown in 7 (d), and it leads falsely the s model to detect the car as a train on the other hand, as Figure 7 (c) illustrates.

A bird in the sky is detected only in one model of YOLOv5 which is x model as appears in 8 (a). but the hand back on the person's shoulder is detected in model x and l as 8 shows in (a) and (b) respectively. True positive person detection is achieved in all models of 8 as well. False Detection of a boat is only achieved in this image when model x applies as 8 (a) illustrates. Model x at this image is most accurate one in this case to detect person, bird, backpack objects very professionally as shown in 8 (a) and (b).

Bridge pillar is detected as boat, as Figures 9 (b) and (c) show. The same vessel is detected as one boat in Figures 9 (c) and (d), but it is counted as two boats in (a) and (b) respectively. Yellow buoy is not detected at all and needs to be added in the training model to be differentiated to the boat. Figures 10 (a), (b), (c), and (d) show that white car on top of the vessel is not detected except the case (a). While the persons in the images are detected accurately.

The trailer part contains the cabin is only detected from the vessel in model x of Figure 11 (a). The person on the vessel surface is not detected in all models except of Model m as Figure 11 (d) illustrates. Building with the boat is delt as one object, as shown in Figure 11 (c). Model x and m are detecting a back shore building as a boat, as shown in Figure 11 (a), (d) respectively. A vessel is sailing in the opposite direction and delt as multiple objects in (b), (d) and as boat partially in (a) as Figure 11 shows.

 Volume 3, Issue 2, December 2023- ISSN 2735-5985
http://dx.doi.org/10.21622/ACE.2023.03.2.064

Fig. 7. YOLOv5 detection run for selected image 1.

Fig. 8. YOLOv5 detection run for selected image 2.

Fig. 9. YOLOv5 detection run for selected image 3.

C. APPLYING YOLOV7 X MODEL FOR THE SELECTED IMAGES SET.

YOLOv7 demonstrates superior performance in 2022 in terms of both speed and accuracy across a range of frame rates, from 5 FPS to 160 FPS. It achieves the highest accuracy, with an average precision (AP) of 56.8%, among all real-time

http://dx.doi.org/10.21622/ACE.2023.03.2.064

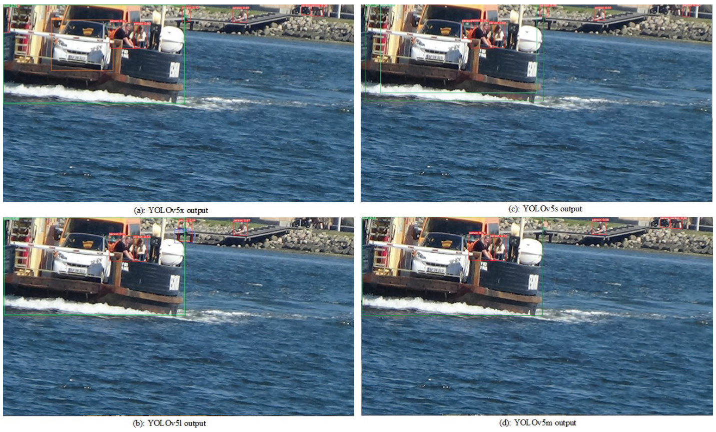

Fig. 10. YOLOv5 detection run for selected image 4.

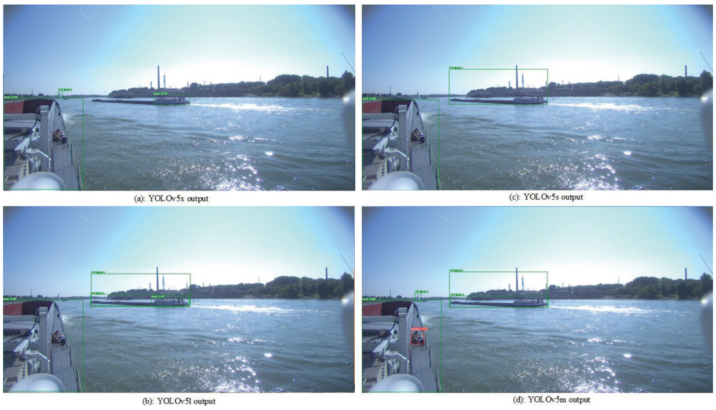

Fig. 11. YOLOv5 detection run for selected image 5.

Deformable DETR, DINO-5scale-R50, ViT-Adapter-B, and several others, in terms of both speed and accuracy as 12 shows Only one model YOLOv7x is used here and as with the previous subsection, limitations of object detection are applied here too with YOLOv7x. The run process to detect objects in 33 selected data set is also applied here by using YOLOv7x. One of the main YOLOv7x model advantagesis could be considered as one of the fastest object detection algorithms available. It can detect objects in real time, making it ideal for applications such as autonomous driving and robotics. YOLOv7x is also very accurate, achieving state-of-the-art results on a variety of object detection benchmarks. This makes it a good choice for applications where accuracy is important, such as medical imaging and security. YOLOv7x has fewer parameters than previous versions of YOLO, making it easier to train and deploy on resource-constrained devices. This makes it a good choice for applications where speed and accuracy are both important, such as mobile phone cameras.

http://dx.doi.org/10.21622/ACE.2023.03.2.064

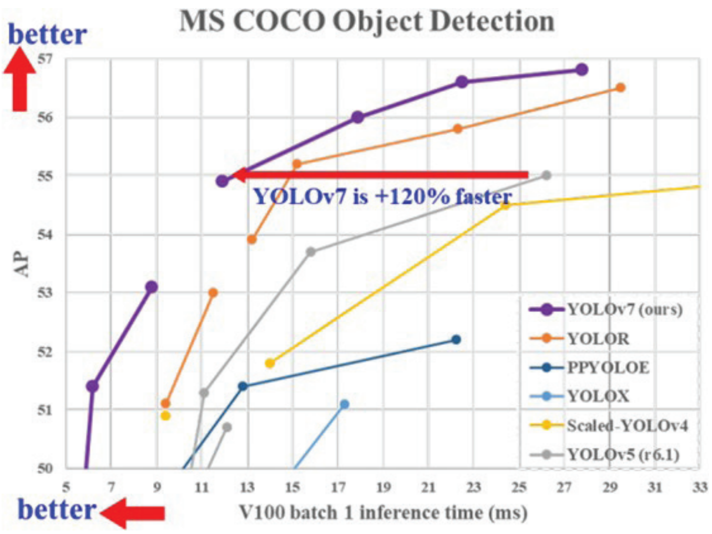

Fig. 12. YOLOv7 in comparison with other real-time object detectors (https://github.com/WongKinYiu/yolov7).

YOLOv7x can detect objects of different sizes at the same time. This makes it a good choice for applications where objects can vary in size, such as traffic or surveillance. YOLOv7x is robust to noise and occlusion. This makes it a good choice for applications where objects may be partially obscured, such as in poor lighting conditions or with moving objects. YOLOv7x achieves its speed by using a simplified model architecture with fewer parameters. This means that the model is less computationally expensive to train and run. YOLOv7x also uses a more efficient implementation of the convolution operations, which further improves its speed

In terms of accuray, YOLOv7x achieves its accuracy by using a larger training dataset with more diverse objects. This helps the model to learn to identify a wider variety of objects. YOLOv7x also uses a more powerful loss function that pe- nalizes inaccurate predictions. This helps the model to learn to make more accurate predictions. YOLOv7x has fewer pa- rameters than previous versions of YOLO because it uses a simplified model architecture. This makes it easier to train and deploy on resource-constrained devices, such as mobile phones or embedded systems. YOLOv7x can detect objects of different sizes at the same time by using a technique called anchor boxes. Anchor boxes are a set of pre-defined boxes that are used to represent the possible sizes and aspect ratios of objects. YOLOv7x predicts the probability of each object being present in each anchor box, as well as the coordinates of the object's bounding box. YOLOv7x is robust to noise and occlusion because it uses a technique called spatial pyra- mid pooling. Spatial pyramid pooling divides the image into a grid of cells, and then averages the predictions from each cell. This helps to reduce the impact of noise and occlusion on the predictions Applying this model on many images is done and Figure 13 shows one image from the selected set and also the same limitations apply here as well.

Detected images can differ based on model accuracy and previous training process: The accuracy of an object detection model can vary depending on various factors, including the training data, model architecture, and optimization techniques used. A model that has been trained on a large, diverse, and well-annotated dataset is likely to perform better in detecting objects compared to a model trained on a smaller or less diverse dataset. Additionally, the training process itself, such as the choice of hyperparameters and training duration, can impact the accuracy of the model.

http://dx.doi.org/10.21622/ACE.2023.03.2.064

Fig. 13. YOLOv7 Image Run.

Objects detected multiple times with different classes: It is not uncommon for an object detection model to produce mul- tiple bounding boxes for the same object instance, especially if the object is large or has complex geometry. However, if the model assigns different classes to these multiple detections of the same object, it could indicate issues with the training process or the quality of the training data. In such cases, reevaluating the training pipeline and dataset annotations may help improve the model's performance.

Misclassifying buoys as boats: Object detection models are trained on large datasets containing various object classes. If a model misclassifies buoys as boats, it might indicate a lack of specific training examples for buoys or similarities between the visual features of buoys and boats. Adding more diverse training examples of buoys and refining the training process can help address this issue.

Need for adding different classes and training the system: If you want the system to detect objects from rivers, waterways, and streets, you would need to include specific classes for those objects during the training process. This would require augmenting the training dataset with relevant images and annotations for objects like riverbanks, bridges, street signs, etc. By training the system on a comprehensive dataset with a wide range of classes, you can enhance its ability to detect and classify objects accurately in different environments.

Based on the results obtained from running the four YOLOv5 models on each image of the sample set in the previ- ous subsection, and after contrast them with the performance of the other x model of YOLOv7, several observations can be drawn:

- The accuracy of detected objects in an image can vary based on the model's accuracy and the training process it underwent.
- In certain cases, an object detection model may detect the same object multiple times but assign different classes to those detections.
- For instance, buoys might be misclassified as boats in- stead of being recognized as buoys.
- To enhance the system's capability to detect objects from rivers, waterways, and streets, it is necessary to add and train the model in different classes, enabling it to identify various objects in these environments.
- In certain models, static objects like cranes may be included within the bounding box of a boat, while in other cases, they may not be included.
- Sometimes, a single long boat may be treated as a single object, while in other instances, it may be detected as two separate boats.
- Treating a part of an object as a complete object, such as considering a

partial boat as a whole boat, is a problem that needs to be addressed.

- Some objects may be misidentified due to misleading neighboring objects. For example, a car with pieces of wood in front might be incorrectly classified as a train, boat, or another car in different runs of the model.

From all these outcomes and other results, it is clear that object detection models can have varying performance depending on their architecture, training data, and optimization techniques. Addressing these issues may involve refining the training process, augmenting the dataset with diverse exam- ples, and fine-tuning the model to improve its accuracy and robustness in detecting and classifying objects correctly in both environments, driverless vehicles, and pedestrian streets and for waterway inland vessels, as well.

IV. YOLO8 APPLIED TRAINING AND DETECTING USING COMBINED DATASET OBJECTS FOR DRIVERLESS VEHICLES AND INLAND VESSELS

A. YOLO8 FEATURES AND NETWORK STRUCTURE

The latest iteration in the YOLO model series, YOLOv8, was recently introduced by Ultralytics. Although a reviewed paper is yet to be published, examination of the repository reveals several notable features that distinguish it from other object detection models. In terms of architecture, as shown in Figure 15, YOLOv8 incorporates significant changes, particu- larly in how it receives and analyzes visual data. Unlike previ- ous YOLO models like YOLOv4, YOLOv8 adopts an anchor-free approach. This is like other variations in the YOLO model series, such as YOLOX, which aims to streamline performance while maintaining high accuracy. Empirically, the best anchor- free approaches have demonstrated comparable or improved performance. However, from a theoretical standpoint, there are certain trade-offs. Anchor-free approaches offer greater flexibility in object detection as they directly identify objects without relying on preset anchors, which can be biased based on previous training and fail to generalize well to new data. However, this flexibility may also lead to biased and mislead- ing predictions that lack the logical foundation inherent in the more traditional process of human object detection and observation.

YOLOv8 implements Anchor-Free instead of Anchor-Based object detection. It employs a dynamic Task Aligned Assigner for the matching strategy, calculating the alignment degree of Anchor-level for each instance. YOLOv8 achieves better accuracy than YOLOv5, making it the most accurate detector to date.

http://dx.doi.org/10.21622/ACE.2023.03.2.064

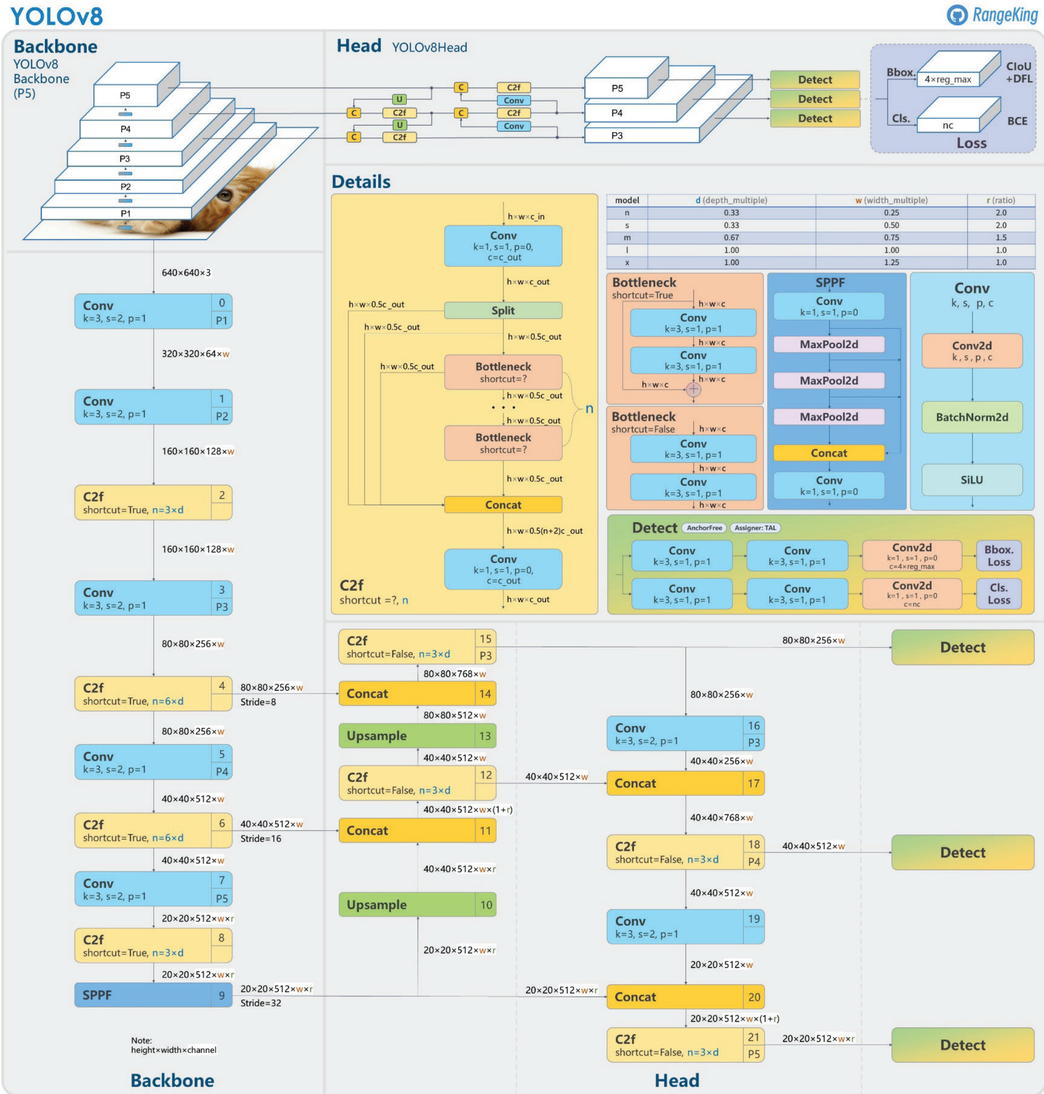

Fig. 14. YOLOv8 Architecture, visualization made (GitHub user Range King)

One key feature of YOLOv8 is its extensibility. It is de- signed to seamlessly work with all YOLO versions and allows researchers to switch between them, facilitating performance comparisons. Therefore, YOLOv8 was selected as the baseline version used in work implementation.

Furthermore, YOLOv8 benefits from the extensive commu- nity support it has garnered, which has contributed to its popu- larity and widespread usage. This community involvement has facilitated empirical investigations into more effective training schedules and methods. For instance, YOLOv8 does not ad- here to the same training strategy throughout the entire training process. One notable example is the mosaic augmentation, which stitches together images to train the model to detect objects with varying combinations and locations. However, it has been observed that employing this augmentation towards the end of the training process can degrade performance. Consequently, YOLOv8 employs a carefully selected training setup based on empirical experimentation to achieve optimal results.

YOLOv8 offers support for all versions of YOLO but with a focus on speed, accuracy, and user-friendliness, as shown by 15. YOLOv8 presents itself as an exceptional option for addressing diverse requirements in object detection and tracking, instance segmentation, image classification, and pose estimation tasks. and allows seamless switching between different versions, providing researchers with flexibility in their experiments. Furthermore, YOLOv8 is compatible with various hardware platforms (CPU-GPU), enhancing its versa- tility.

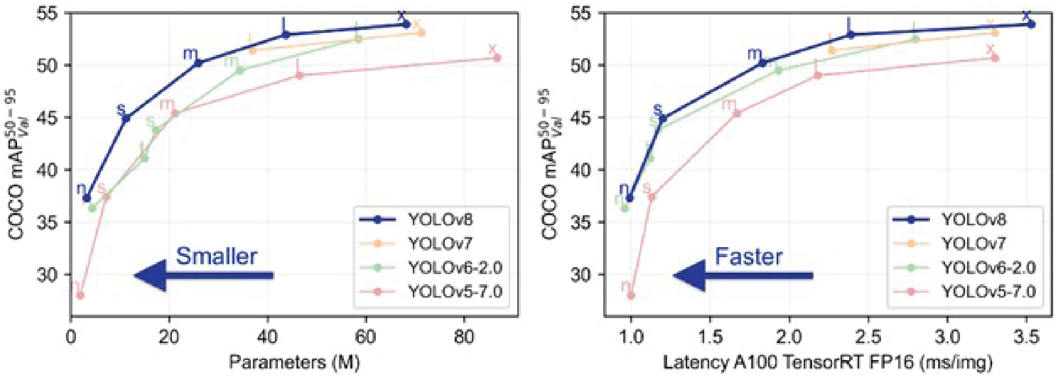

Fig. 15. YOLOv8 is a cutting-edge, state-of-the-art (SOTA) model (https://github.com/ultralytics)

YOLOv8 represents a collection of neural network mod- els developed and trained using PyTorch, which have been exported as files with the .pt extension. These models can be categorized into three types: Classification, Detection, and Segmentation, each serving a distinct purpose. Additionally, there are five models available for each type, differing in size, as shown in Table III.

TABLE III
YOLO8 Three types of models and 5 models of different sizes for each type

Classification	Detection	Segmentation	Kind
yolov8n-cls.pt	yolov8n.pt	yolov8n-seg.pt	Nano
yolov8s-cls.pt	yolov8s.pt	yolov8s-seg.pt	Small
yolov8m-cls.pt	yolov8m.pt	yolov8m-seg.pt	Medium
yolov8l-cls.pt	yolov8l.pt	yolov8l-seg.pt	Large
yolov8x-cls.pt	yolov8x.pt	yolov8x-seg.pt	Huge

B. YOLO8 BASED DETECTION RUN ON A SET OF IMAGES FOR INLAND VESSELS USING COCO 80 CLASSES DATASET READY TRAINED MODEL

In the subsequent sections of this paper, object detection will be performed using YOLOv8. The YOLOv8 models specifi- cally designed for object detection come with pre-training on the COCO dataset, which includes a wide range of images covering 80 different categories. As a result, for the purpose of detecting objects on the street for driverless vehicles, the pre-trained models based on the COCO dataset fulfil the requirements, and there is no need for custom training. The pre-trained models listed in Table III are sufficient for this environment without requiring any additional training. By applying YOLOv8x on objects detecting for the same images, It is noticed that detection is much better than the previously tested versions, as Figure 16 shows, and the existing x model is enough to base on at the train phase to add new ability for detecting the new object.

YOLOv8 employs a novel architecture known as BiFPN (Bidirectional Feature Pyramid Network), which synergisti- cally amalgamates features from various network strata. This integration equips YOLOv8 with enhanced capabilities in managing objects of varying dimensions and aspect ratios. While YOLOv7 also employs BiFPN, YOLOv8 distinguishes itself by incorporating a higher count of channels within the Feature Pyramid Network (FPN) layers, affording it the ca- pacity to assimilate more intricate features. The adaptability of YOLOv8 is exemplified by its capacity to accommodate higher input resolutions during training compared to its predecessors. This elevated resolution empowers YOLOv8 to discern and precisely identify smaller objects. Though YOLOv7 exhibits similar adaptability, YOLOv8's superior performance in managing higher resolutions is evident

The adoption of the ResNet-50 backbone network in YOLOv8 represents a notable advancement over previous versions. This deeper and more expansive architecture en- ables YOLOv8 to distill complex features that contribute to refined object detection. Although YOLOv7 also integrates the ResNet-50 backbone, YOLOv8's augmented depth furnishes it with an advantage in feature extraction. YOLOv8 leverages anchor-free detection for bounding box prediction, a technique that surpasses predefined anchor boxes in terms of flexibility and precision. While YOLOv7 embraces anchor-free detec- tion, YOLOv8 employs a superior approach termed CenterNet, showcasing enhanced accuracy

The expanded repertoire of detectable object classes in YOLOv8 stems from its utilization of a broader and more diverse training dataset. This augmentation enables YOLOv8 to surpass its predecessors, including YOLOv7, in terms of class diversity. YOLOv8 employs focal loss as a pivotal training mechanism, effectively mitigating the impact of mis- classified objects and elevating accuracy. While YOLOv7 also integrates focal loss, YOLOv8 deploys an advanced version, demonstrative of its commitment to precision

The innovative amalgamation of Focal Loss and Smooth L1 Loss functions in YOLOv8 addresses misclassification and inaccurate bounding boxes, resulting in superior performance. The unique weighting of these loss functions in YOLOv8 is carefully tailored for optimal efficacy, distinguishing it from YOLOv7's similar approach. YOLOv8 employs the Swish activation function, known to heighten the precision of object detection algorithms. This function is shared with YOLOv7, albeit contributing to YOLOv8's consistent accuracy enhancements Auto Augment serves as YOLOv8's preprocessing technique, incorporating an automated application of diverse transformations to images. This feature is an evolution over YOLOv7's utilization of Auto Augment, with YOLOv8's iteration displaying a refined effectiveness. Post-processing, YOLOv8 employs Non-Maximum Suppression (NMS) to re- fine detection outcomes by eliminating redundancy and over- laps. This NMS approach, while comparable to YOLOv7's application, reflects YOLOv8's superior optimization for enhanced efficacy. Overall, YOLOv8 is a significant improvement over previous versions of YOLO. It achieves better accuracy, handles objects of different sizes and aspect ratios better, can be trained on images with a higher input resolution, and can detect more object classes, as Table V shows.

The accuracy of the YOLO object detection algorithms has steadily improved over the years. mAP (mean Average Preci- sion) is a metric used to evaluate the performance and accuracy of object detection algorithms. It is calculated by averaging the average precision (AP) scores for each object category in a dataset. The AP score for a single category is calculated by first calculating the precision and recall scores for that cat- egory. Precision is the fraction of predicted bounding boxes that actually contain the object. Recall is the fraction of ground truth bounding boxes that are predicted by the algorithm. The AP score is then calculated as the area under the precision- recall curve. The mAP score is a more robust metric than the AP score because it averages the AP scores for all object categories in a dataset. This helps to reduce the impact of any individual category that may have a low AP score. The mAP score is typically used to compare the performance of different object detection algorithms. A higher mAP score indicates that the algorithm is more accurate. Overall, mAP is a useful metric for evaluating the performance of object detection algorithms. Nevertheless, a crucial consideration necessitates an awareness of its limitations during the interpretation of the outcomes YOLOv8 is the most accurate version of YOLO, achieving an mAP of 65.7% on the COCO dataset, as shown in Table VI.

The COCO dataset is a challenging dataset for object detection, as it contains a wide variety of objects in a variety of settings. The mAP metric measures the accuracy of an object detection algorithm by calculating the intersection over union (IoU) between the predicted bounding boxes and the ground truth bounding boxes. An mAP of 100% indicates that the predicted bounding boxes perfectly match the ground truth bounding boxes. It is rarely achieved in real-world scenarios due to various challenges. Object detection algorithms should aim for high mAP values, but it's essential to balance precision and recall. Striking the right balance depends on the specific application and may require fine-tuning and optimization for optimal results.

The improvement in accuracy from YOLOv1 to YOLOv8 can be attributed to a number of factors, including:

- The use of deeper and wider neural networks
- The use of more data augmentation techniques
- The use of more power- ful loss functions
- The use of better optimization techniques

YOLOv8 is the most accurate version of YOLO, but it is also the most computationally expensive.

COCO 80 classes are covering all the objects that are most likely appeared on the street for these autonomous vehicles and for their scene view. Each object that can be detected by the neural network is associated with a numeric ID. In the case of a YOLOv8 pre-trained model, there are 80 different object types, each assigned a unique ID ranging from 0 to 79, as mentioned in Table IV. The object classes in the COCO dataset are widely recognized. Testing this COCO trained model on the detection of the Inland vessels' environment objects is efficient for the objects appeared in both environments like (person, bird, etc.) or already included inside the COCO dataset like (boat, surfboard, etc…), but it will not be sufficient in case of waterway objects like: buoys, shores, cranes and bridge pillars. These added classes will be discussed deeply in the next subsection.

Five more classes from inland vessels' environment by extrapolation, as a contribution of this study, are added to the COCO dataset. These classes are:

- Left Shore (L Shore) with ID 80
- Right Shore (R Shore) with ID 81
- Bridge Pillar with ID 82
- Crane with ID 83
- Buoy with ID 8

TABLE IV
COCO dataset 80 Classes with preassigned ID's

ID	Object	ID	Object	ID	Object	ID	Object	ID	Object	ID	Object	ID	Object	ID	Object
0	person	1	bicy-cle	2	car	3	motor-cycle	4	airplane	5	bus	6	train	7	truck
8	boat	9	traffic light	10	fire hy-drant	11	stop sign	12	parking meter	13	bench	14	bird	15	cat
16	dog	17	horse	18	sheep	19	cow	20	elephant	21	bear	22	zebra	23	giraffe
24	backpack	25	um-brella	26	handbag	27	tie	28	suitcase	29	fris-bee	30	skis	31	snow-board
32	sports ball	33	kite	34	baseball bat	35	baseball glove	36	skate-board	37	surf-board	38	tennis racket	39	bottle
40	wineglass	41	cup	42	fork	43	knife	44	spoon	45	bowl	46	banana	47	apple
48	sandwich	49	or-ange	50	broccoli	51	carrot	52	hot dog	53	pizza	54	donut	55	cake
56	chair	57	couch	58	potted plant	59	bed	60	dining table	61	toilet	62	tv	63	laptop
64	mouse	65	re-mote	66	keyboard	67	cell-phone	68	micro-wave	69	oven	70	toaster	71	sink
72	refrigerator	73	book	74	clock	75	vase	76	scissors	77	teddy bear	78	hair drier	79	tooth brushes

http://dx.doi.org/10.21622/ACE.2023.03.2.064

Fig. 16. YOLOv8 model x object detection run.

D. COMBINED MODEL AND RESULTS

To combine the trained 80 classes dataset with the new classes data set, a set of processes starting with annotation, then training and predicting are applies in a certain sequence to reduce the annotation time and get the results accurately in both driverless and inland vessels environments. the next subsection shows the block diagram of the applied processes and discussed the methodology of detecting objects base on trained model for combined dataset.

In order to combine the trained model with 80 classes dataset and give it the ability to detect the new classes as well, a set of processes should be used. It is very important to do this as a building block and automate the process as can as possible. To achieve this goal, two main building blocks are needed as figures 17 and 18 show. First of all, as Figure 17 shows, the first block consists of three steps:

1) Annotate the dataset with the new objects with 5 Classes (Buoy, L Shore, R Shore, Crane, and Bridge Pillar).

2) Train the new dataset with new labels generated by the manual annotation.

3) Validate the performance of the new trained model for the newly added five objects.

For this purpose, a dataset of 300 images containing these new objects is chosen and annotated manually. labelImg program is used here for this manual annotation process, as shown in Figure 19, and the program output is with YOLO format for labels. This format is class code, and the location of the bounded box of this object with x and y coordinates for the up left corner and down right on of the box.

These labels are saved with the same name of the images as txt file. After finish all of the annotation process, a 300 text files for labels will be saved in the output folder. All objects in these files are coded from 0 to 4, while the main COCO set objects are coded from 0 to 79. It is clear here that it is a must to avoid misleading codes to shift the codes generated in the new set from 0 to 4 to be from 80 to 84. A python program code is written for this purpose. This will make them combined in one list to be ready for the whole training for all objects in the scene for both street and waterway environments.

However, the training process takes a long time (in terms of days) on CPU based machines, and it is very fast in GPU's base ones (12 hours). Google Co-lab Pro package is used in training epochs. Utilizing the Google Colab Pro package for YOLO training offers substantial benefits. First, it's cost-efficient, providing access to powerful GPUs without substantial upfront investment. This accelerates training, cru- cial for YOLO's computational demands. Second, Colab Pro ensures ample resources, mitigating memory-related issues and enabling efficient handling of larger datasets. Its integration with Jupyter Notebooks fosters collaborative and interactive model development. Lastly, pre-installed libraries expedite setup, and seamless integration with Google Drive aids version control and sharing.

After the training epochs finished, confusion matrix, the losses, and F1 score metric are calculated to measure the model performance, accordingly, as illustrated in Figures 20, 21,22, 23,24, and 25 respectively.

In this stage and after training the 5 classes model with acceptable confidence for the detection of each class and low losses converges until the end of epochs it is time to prepare the dataset for the combined model. This data set includes both 80 classes models labels and the newly generated 5 classes ones. Manual efforts done here but with a very important reward for the system. One of the famous problems here is the double detecting behavior for the same object with the old 80 classes and in the 5 classes system too. Then a label correction phase is required here to distinguish between the old detection (boat, for example) and the new and correct one (buoy, in this case) as figure 26 shows.

The last preparation step here, the training for all of the system depending on the corrected labels and combined labels for each image is required to generate the .pt file which be the best weights file for the neural network to depend on it for further prediction in street environment or in waterway inland vessels one as well. The training and epochs are generated, as the previous training step, as figure 4.36 shows and the losses, confusion matrix and F1 score metric is calculated to measure the model performance, accordingly, as illustrated in Figures 4.41, 4.42, and 4.43 respectively.

The F1 score is a widely used evaluation metric in object detection tasks that combines precision and recall into a single value. It provides a measure of the overall performance and accuracy of an object detection model. In the context of detecting 85 classes on the street and on inland vessels, calculating the F1 score helps assess the model's ability to correctly identify and classify objects within these environments. The F1 score takes into account both the model's ability to correctly detect true positive instances (precision) and its ability to find all relevant instances (recall).

A high F1 score indicates that the model is effectively de- testing and classifying objects across 85 classes. It reflects a good balance between precision and recall, where both the number of correctly identified objects and the number of false positives and false negatives are minimized. It is important to note that achieving a high F1 score for such a diverse range of classes in different environments can be challenging. It requires a comprehensive and well-curated training dataset that includes sufficient examples for each class. Additionally, fine- tuning the model architecture, optimizing hyperparameters, and carefully selecting training techniques can contribute to improving the F1 score.

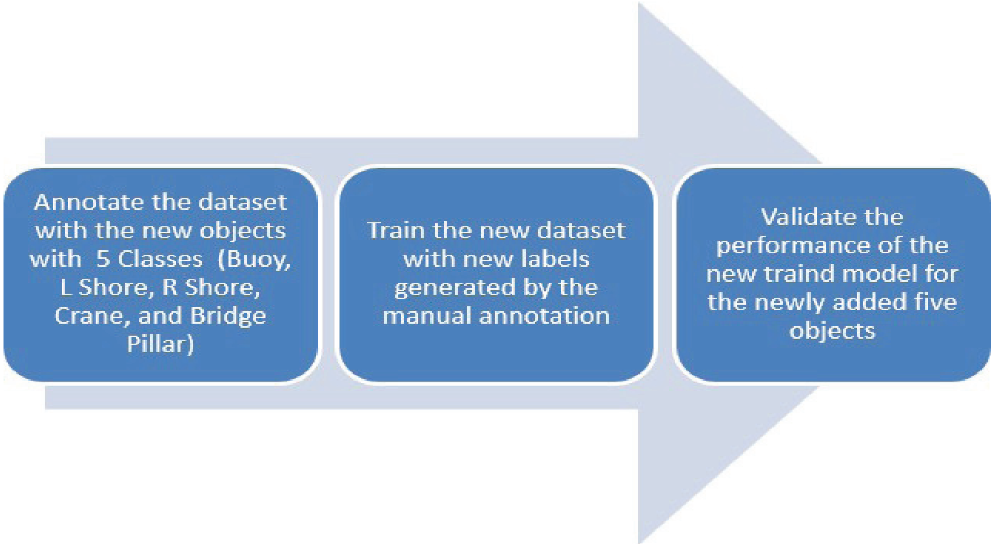

Fig. 17. Training for the new 5 classes model.

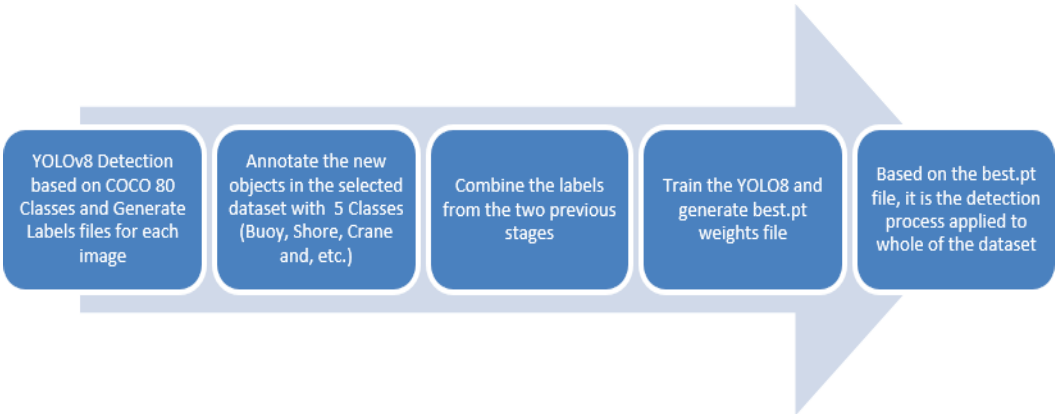

Fig. 18. Combined model process flow.

Regular evaluation of the F1 score is crucial during the development and improvement stages of the object detection combined model. It helps identify areas where the model may struggle, such as detecting specific classes or dealing with challenging scenarios. By analyzing and addressing the factors affecting the F1 score, developers can continuously enhance the model's performance and accuracy in detecting the 85 classes on the street and on inland vessels.

V. CONCLUSION

This study has undertaken an in-depth exploration of object detection methodologies, with a particular emphasis on the application of YOLOv7 and YOLOv8 models within street and inland vessel environments. The evaluation of these models' performance and accuracy was conducted using a comprehensive array of evaluation metrics, notably including the F1 score.

The results demonstrated that YOLOv8 surpassed existing object detectors in terms of both speed and accuracy across a range of frame rates. It achieved the highest accuracy among real-time object detectors operating at 30 FPS or higher on the GPU V100. The YOLOv8 models, pre-trained on the COCO dataset, showed promising performance for detecting 85 combined classes in street and inland vessel scenarios, eliminating the need for custom training.

http://dx.doi.org/10.21622/ACE.2023.03.2.064

The findings unveiled that YOLOv8 surpassed prevailing object detection methods in terms of both speed and accuracy across a diverse range of frame rates. Particularly notable was its achievement of the highest accuracy among real-time object detectors operating at 30 FPS or higher on the GPU V100. The YOLOv8 models, having been pre-trained on the COCO dataset, exhibited promising performance in detecting 85 combined classes across scenarios involving street and inland vessels, obviating the necessity for custom training. Nonetheless, it is imperative to acknowledge that the attainment of elevated accuracy and performance in object detection tasks remains a multifaceted challenge. The process necessitates meticulous model fine-tuning, meticulous hyperparameter optimization, and the curation of a diverse and well-annotated training dataset. These considerations bear significant weight in enhancing both the F1 score and the overarching performance metrics. Future research trajectories might encompass the exploration of advanced architectural paradigms, the integration of supplementary contextual information, and the resolution of context-specific challenges intrinsic to street and inland vessel domains. Addressing these challenges and pursuing innovation stands poised to yield more robust and accurate object detection models, thereby capacitating the adept identification and classification of objects within a spectrum of scenarios.

In summation, this paper profoundly contributes to the field of object detection by rigorously evaluating the performance of the combined YOLOv8 model within the contexts of street and inland vessel applications. This endeavor underscores their potential efficacy in real-time scenarios, thereby charting a course towards safer and more operationally efficient autonomous systems within these domains. The revelations emanating from this study, enriched by insights into the intricacies of YOLOv8, pave a discernible path for the progressive evolution of object detection technology. The implications of these findings are poised to catalyze the advancement of this field, fostering the development of more sophisticated and precise models conducive to safer and more proficient autonomous systems.

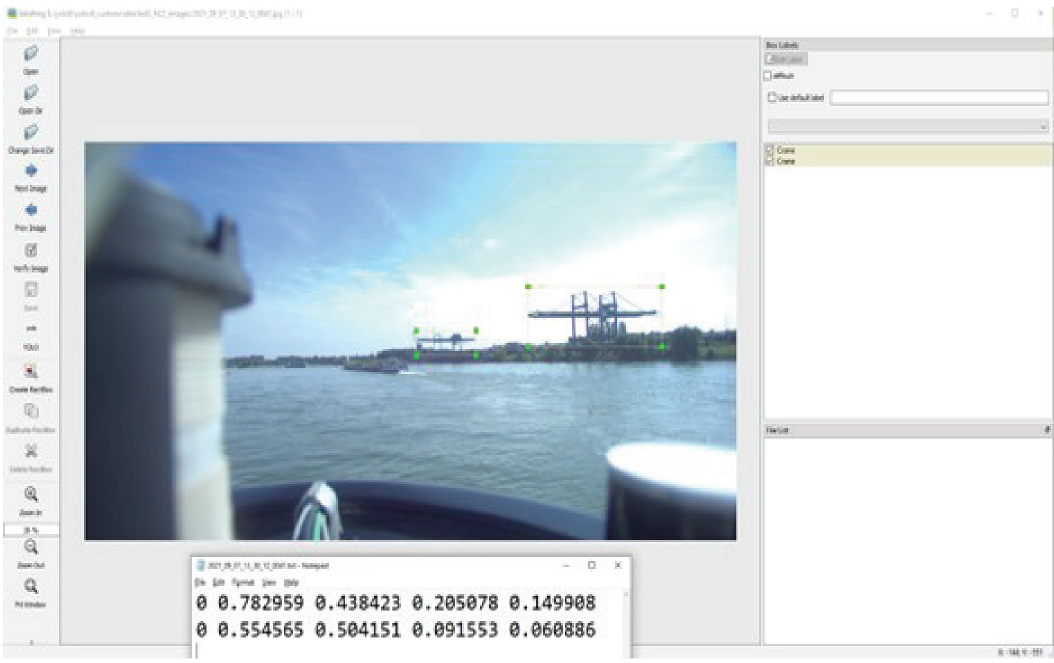

Fig. 19. Annotate new objects in the scene.

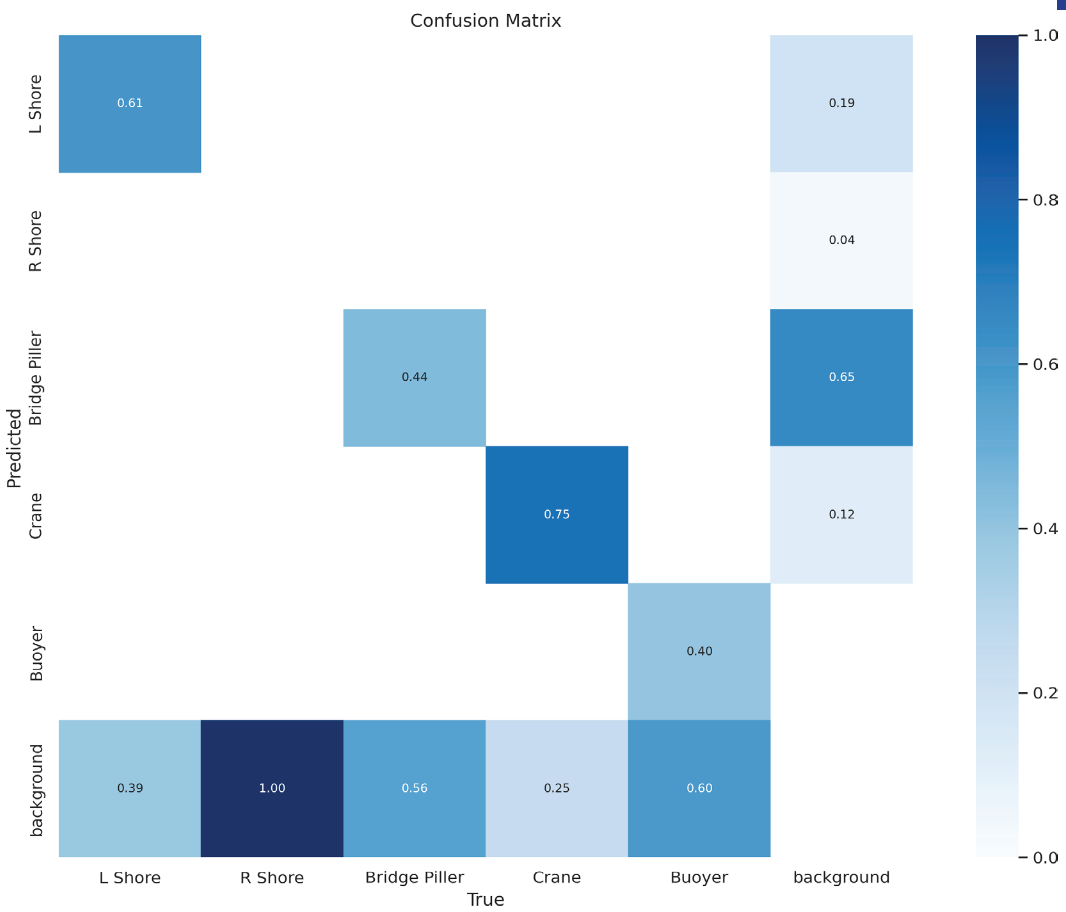

Fig. 20. Confusion Matrix of the 5 Classes Model.

Volume 3, Issue 2, December 2023- ISSN 2735-5985
http://dx.doi.org/10.21622/ACE.2023.03.2.064

TABLE V
YOLO8 and Evolution of Previous YOLO Algorithms

Key features	YOLO (V1)	YOLO (V2)	YOLO (V3)	YOLO (V4)	YOLO (V5)	YOLO (V6)	YOLO (V7)	YOLO (V8)
Architecture	Single neural network for both object localization and classification	Single neural network for both object localization and classification	Darknet-53 backbone and three detection heads with feature maps of different resolutions	Single-shot detection neural network for object localization and classification	Single neural network for both object detection and classification	Single-stage object detection framework	ELAN- efficient layer aggregation network	The modified version of the SPP-YOLO
Input resolution	448 X 448 pixels	416 X 416 pixels	Configurable input resolution up to 608 X 608 pixels	640 X 640 pixels	The configurable resolution, typically 640 X 640	Can be trained on multiple resolutions (up to 640 X 640)	Can be trained on multiple resolutions (1280 X 1280)	Can be trained on multiple resolutions (608 X 608)
Backbone network	"Darknet-19" as its backbone network	Darknet-19 architecture as its backbone network	Darknet-53 architecture as its backbone network	CSPDarknet53 or CSPRes- NeXt50 (depending on configuration)	CSPDarknet53 (improved Darknet network)	EfficientRep	CBS, E-ELAN, MP, and SPPCSPC modules	CSPDarknet53 architecture
Bounding boxes	Predicts bounding boxes with class probabilities	Predicts bounding boxes with class probabilities and anchor boxes	Predicts bounding boxes with class probabilities and objectness score Object classes	Predicts bounding boxes with class probabilities	Predicts bounding boxes with class probabilities and confidence score	Predicts bounding boxes with class probabilities and confidence score	Predicts bounding boxes with class probabilities and confidence score	Predicts bounding boxes and class probabilities for an object
Object classes	20	80	COCO dataset with 80 object classes	80 (COCO dataset)	Customizable, depending on the dataset	80	80	80
Training	Trained on VOC 2012 and ImageNet datasets using darknet framework	Trained on COCO dataset using darknet framework with data augmentation	Trained on COCO dataset using darknet framework	Trained on COCO dataset using the PyTorch framework	Trained using PyTorch framework with various datasets, such as COCO, VOC, and Open Images	Trained on COCO dataset using PyTorch framework	v1	Trained on COCO dataset using PyTorch framework

Volume 3, Issue 2, December 2023- ISSN 2735-5985
http://dx.doi.org/10.21622/ACE.2023.03.2.064

Loss function	Includes three main components: Localization Loss (Lloc), Confidence Loss (Lconf), and Classification Loss (Lcls)	includes several main components: Localization Loss (Lloc), Confidence Loss (Lconf), Classification Loss (Lcls), and Class Probability Loss (Lprob).	The multi-scale loss combines binary cross-entropy, confidence loss, and regression loss	IoU loss, GIoU loss, and objectness loss	Combined loss function consisting of focal loss, binary cross-entropy, and smooth L1 loss	yolov8n-seg.pt	YOLOv6 used Varifocal loss (VFL) for classification and Distribution Focal loss (DFL) for detection	VFL Loss as classification loss and DFL loss + CIOU loss as classification loss
Activation function	Leaky ReLU	Leaky ReLU	Leaky ReLU	Mish activation	SiLU (Swish)	SiLU	Leaky Relu	Leaky Relu
Pre-processing	Resizing and normalization	Resizing and normalization with data augmentation	Random resizing and data augmentation	Resizing and normalization plus Random crop, resize with letter boxing, and color distortion	Random scaling, translation, rotation techniques are used	Padding Gray borders	Compound scaling method	Mosaic data augmentation and class-specific anchor boxes
Post-processing	Non-max suppression	Non-max suppression with region proposal network (RPN) and anchor boxes	Non-max suppression with dynamic threshold based on objectness score and IoU threshold	NMS with an threshold of 0.5	Non- maximum suppression and confidence thresholding	Non- maximum suppression	Non- maximum suppression	Non- maximum suppression

TABLE VI
Comparison of YOLO8 with Previous Versions Accuracies

YOLO version	mAP on COCO (test-dev)
YOLOv1	21.2%
YOLOv2	33.1%
YOLOv3	35.5%
YOLOv4	43.5%
YOLOv5	48.6%
YOLOv6	50.4%
YOLOv7	57.9%
YOLOv8	65.7%

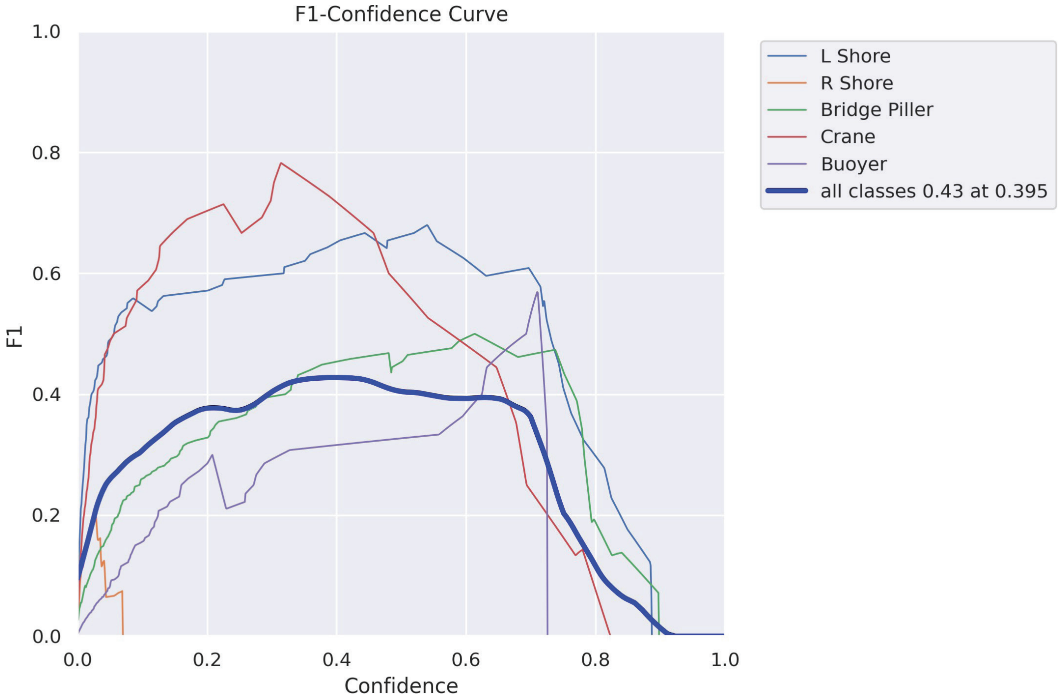

Fig. 21. F1-Confidence curve for the 5 classes model.

http://dx.doi.org/10.21622/ACE.2023.03.2.064

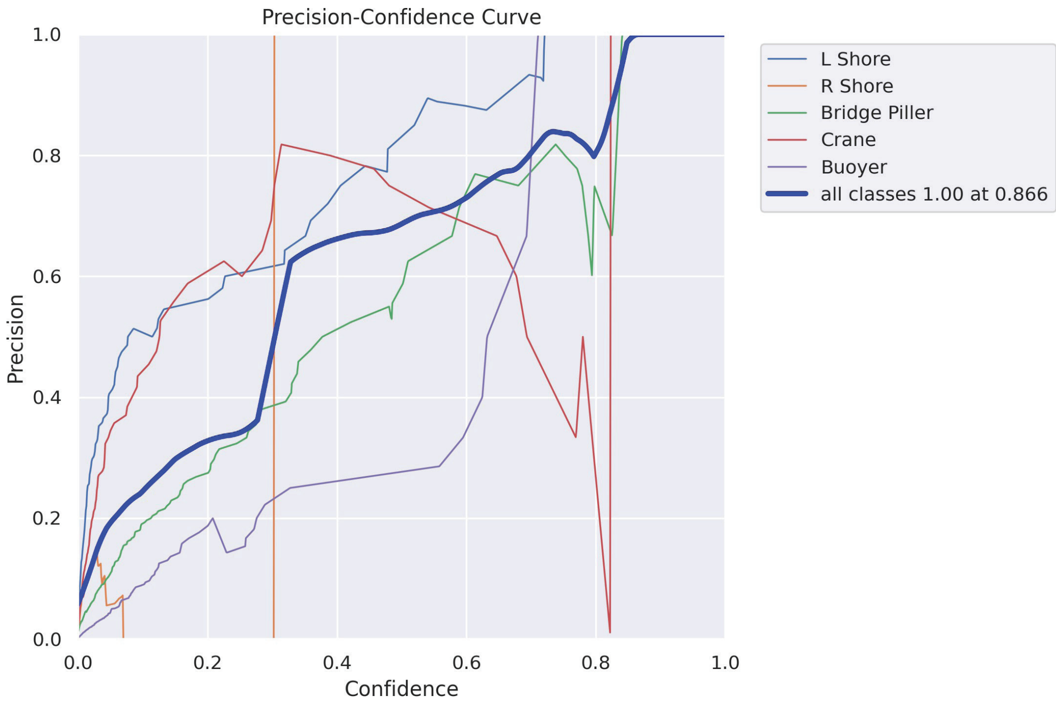

Fig. 22. Precision-Confidence curve of the 5 Classes Model

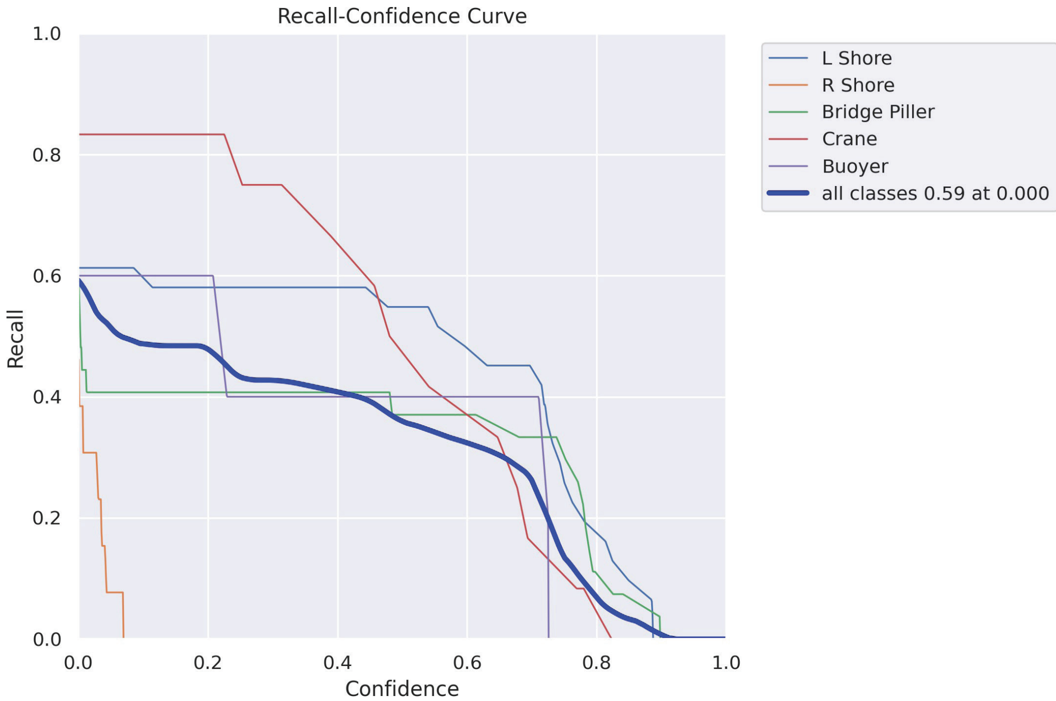

Fig. 23. Recall-Confidence curve of the 5 Classes Model.

http://dx.doi.org/10.21622/ACE.2023.03.2.064

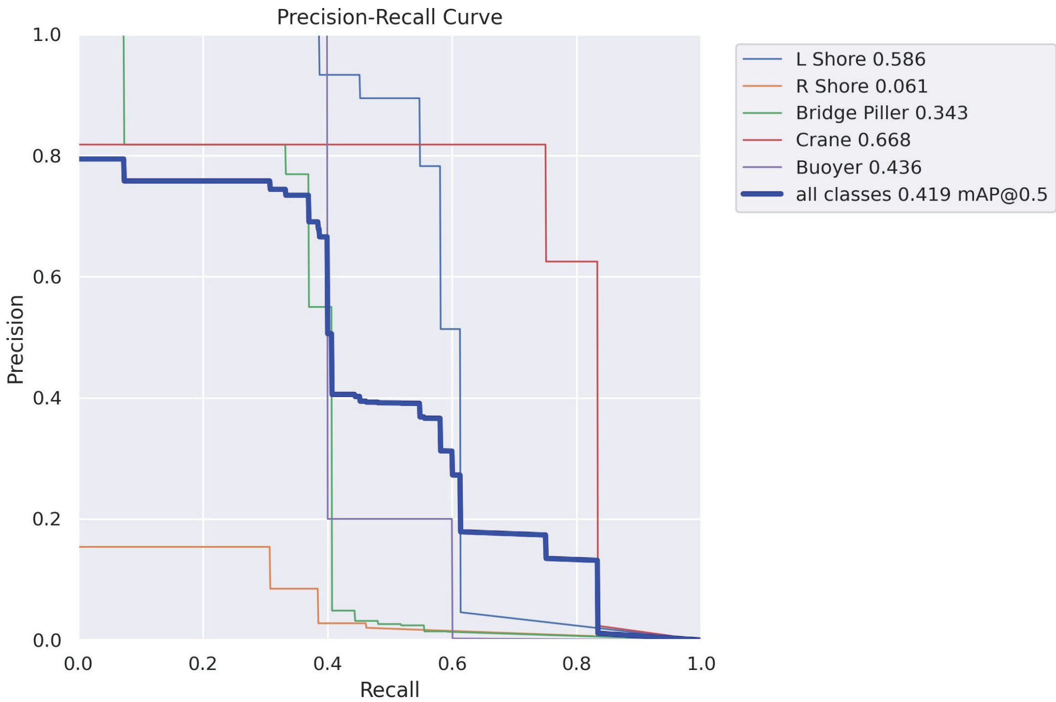

Fig. 24. Precision-Recall curves of the 5 Classes Model.

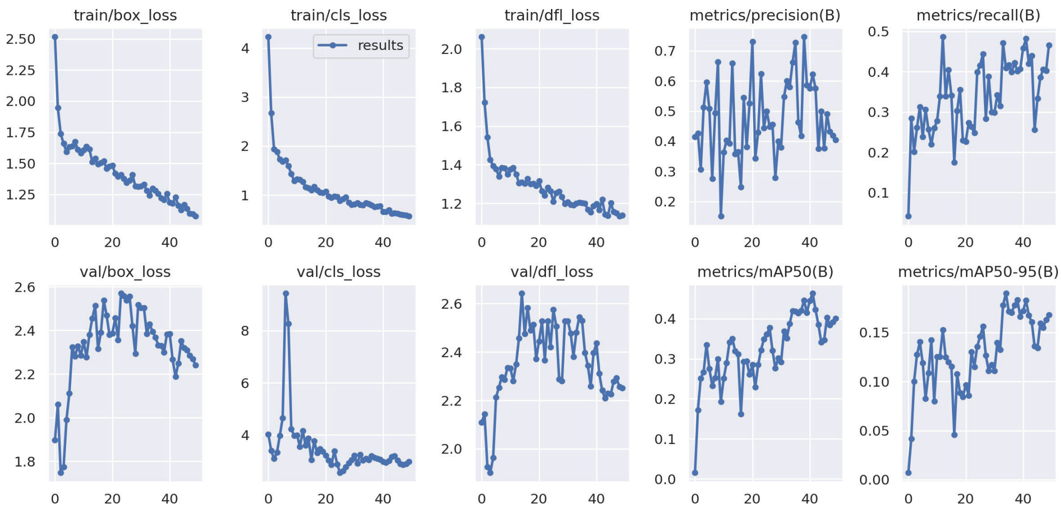

Fig. 25. Losses different curves of the 5 Classes Model

Volume 3, Issue 2, December 2023- ISSN 2735-5985
http://dx.doi.org/10.21622/ACE.2023.03.2.064

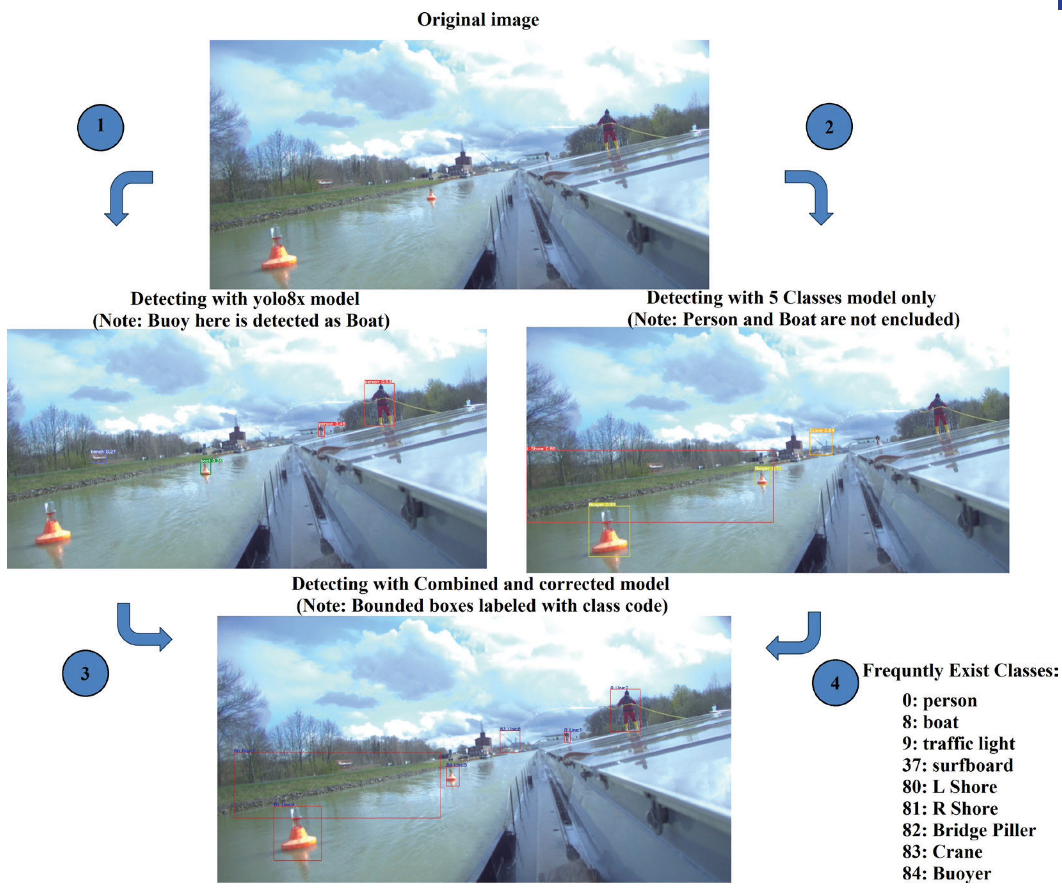

Fig. 26. Correcting multiple labels error and construct the combined model.

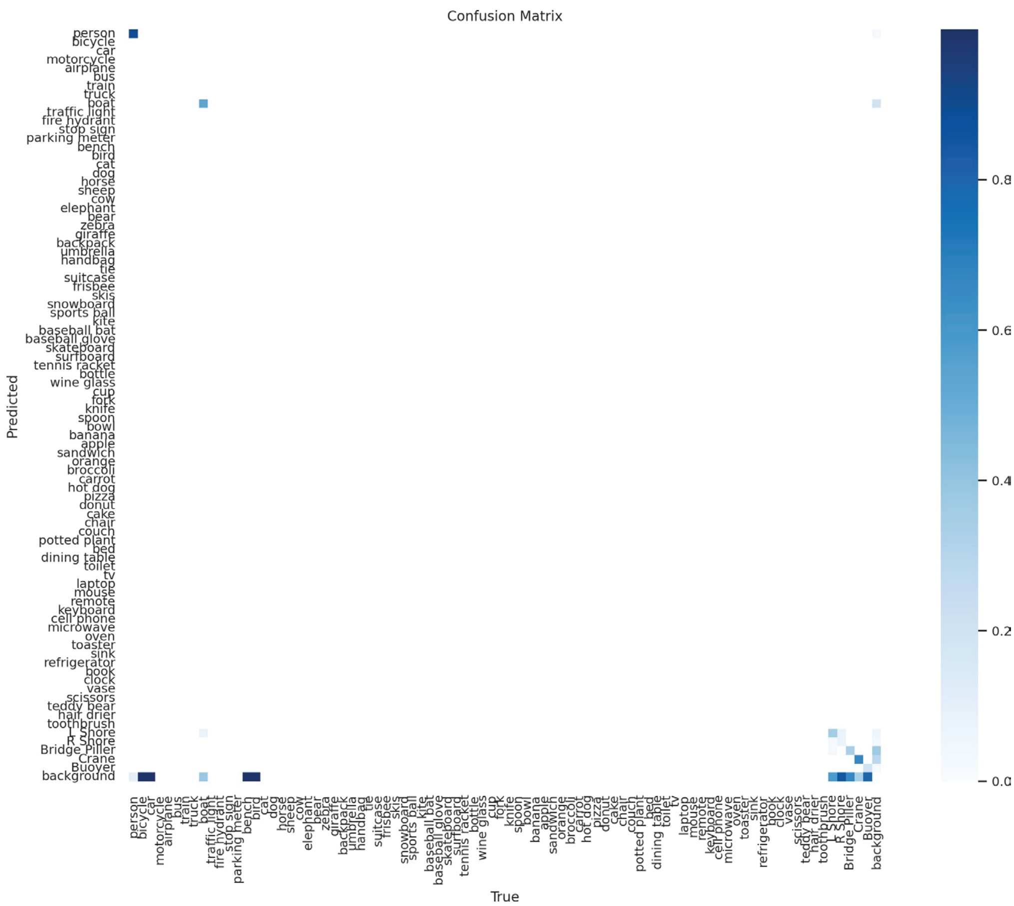

Fig. 27. Confusion Matrix of the 85 Classes Model.

http://dx.doi.org/10.21622/ACE.2023.03.2.064

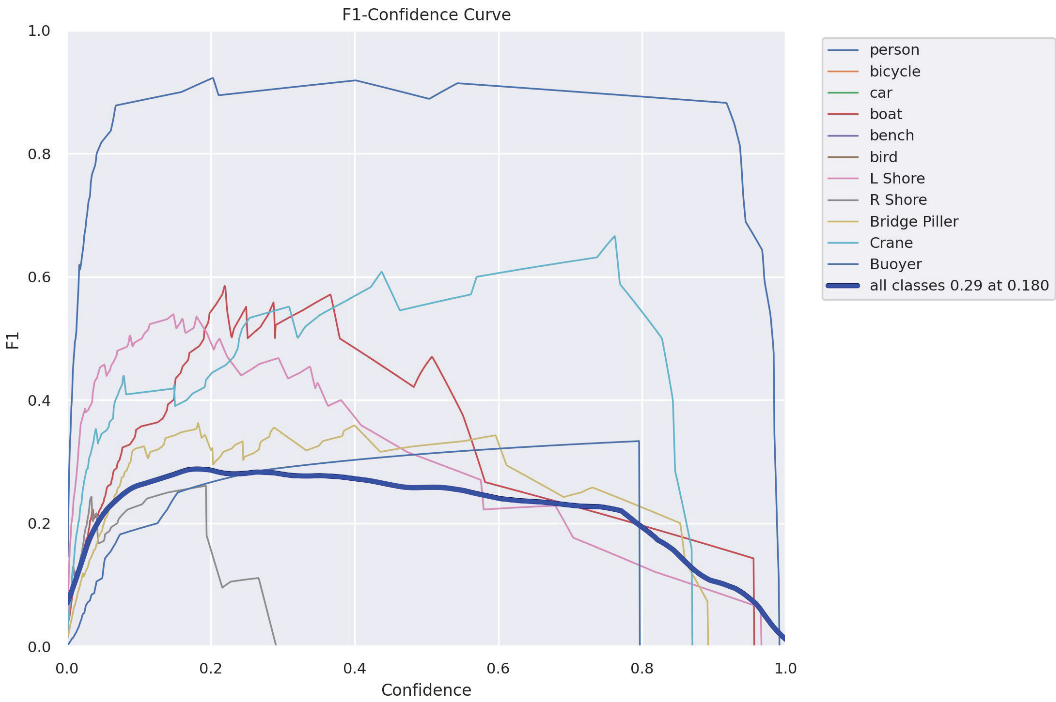

Fig. 28. F1-Confidence curve for the 85 classes model.

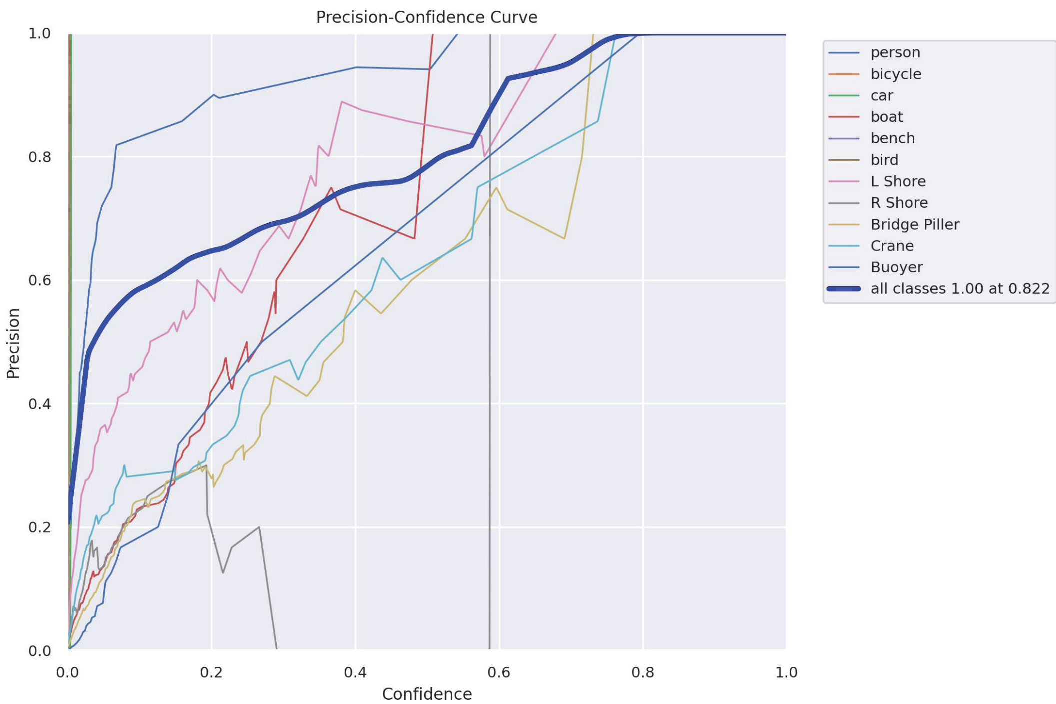

Fig. 29. Precision-Confidence curve of the 85 Classes Model

http://dx.doi.org/10.21622/ACE.2023.03.2.064

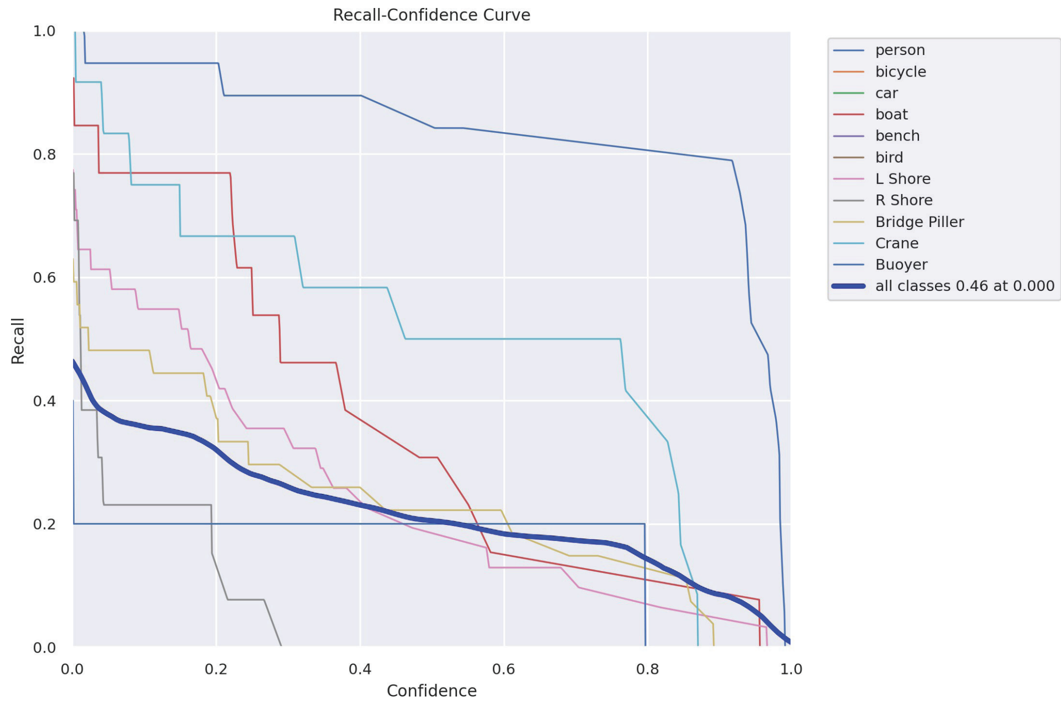

Fig. 30. Recall-Confidence curve of the 85 Classes Model.

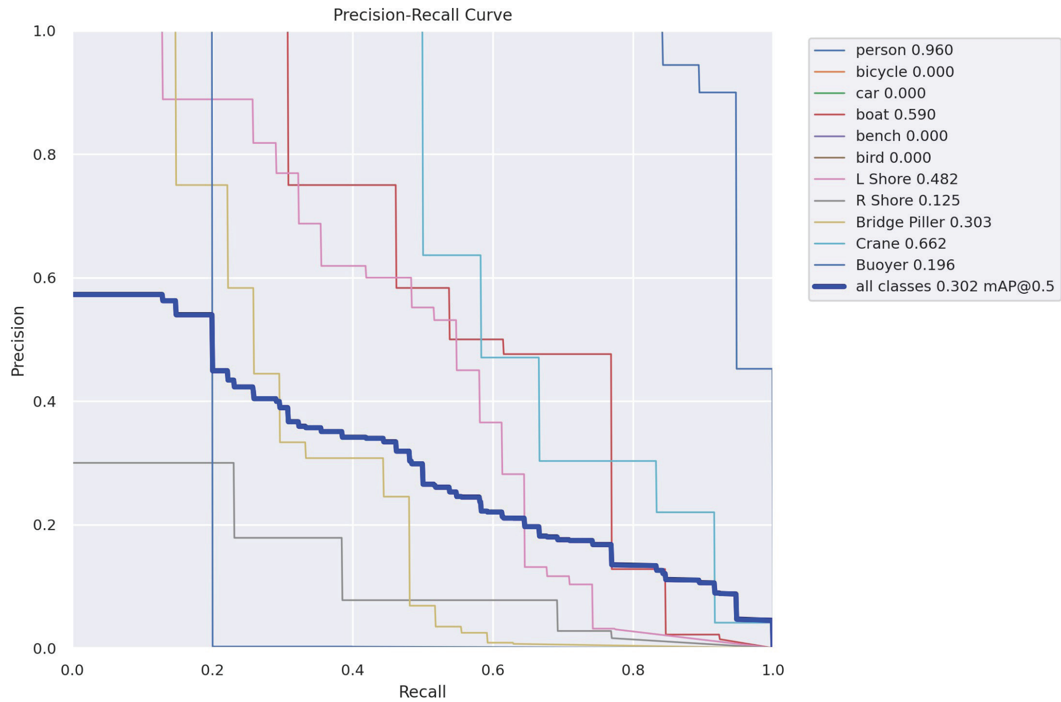

Fig. 31. Precision-Recall curves of the 85 Classes Mode

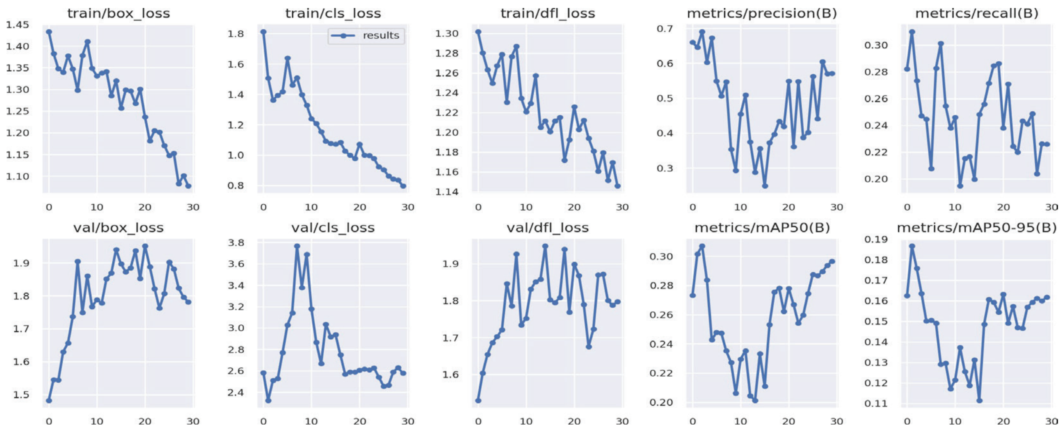

Fig. 32. Losses different curves of the 85 Classes Model

VI. REFERENCES

[1] X. Zhao, G. Wang, Z. He, and H. Jiang, "A survey of moving object detection methods: A practical perspective," Neurocomputing, vol. 503, pp. 28–48, 2022. [Online]. Available: https://doi.org/10.1016/j. neucom.2022.06.104

[2] W. Han, J. Chen, L. Wang, R. Feng, F. Li, L. Wu, T. Tian, and J. Yan, "Methods for Small, Weak Object Detection in Optical High- Resolution Remote Sensing Images: A survey of advances and chal- lenges," IEEE Geoscience and Remote Sensing Magazine, vol. 9, no. 4,pp. 8–34, 2021.

[3] M. N. Chan and T. Tint, "A Review on Advanced Detection Meth- ods in Vehicle Traffic Scenes," Proceedings of the 6th International Conference on Inventive Computation Technologies, ICICT 2021, pp. 642–649, 2021.

[4] M. S. Karis, N. R. A. Razif, N. M. Ali, M. A. Rosli, M. S. M. Aras, and M. M. Ghazaly, "Local Binary Pattern (LBP) with application to variant object detection: A survey and method," Proceeding - 2016 IEEE 12th International Colloquium on Signal Processing and its Applications, CSPA 2016, no. March, pp. 221–226, 2016.

[5] R. Kaur and S. Singh, "A comprehensive review of object detection with deep learning," Digital Signal Processing, vol. 132, p. 103812, 2022. [Online]. Available: https://doi.org/10.1016/j.dsp.2022.103812

[6] K. Li and L. Cao, "A review of object detection techniques," in 2020 5th International Conference on Electromechanical Control Technology and Transportation (ICECTT), 2020, pp. 385–390.

[7] D. G. Lowe, "Distinctive image features from scale-invariant key- points," International Journal of Computer Vision, vol. 60, no. 2, pp. 91–110, 2004.

[8] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in CVPR, 2005.

[9] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," Computer Vision and Image Understanding, vol. 110,no. 3, pp. 346–359, 2008.

[10] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "Orb: An efficient alternative to sift or surf," Proceedings of the IEEE International Conference on Computer Vision, p. 2564, 2011.

[11] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for

http://dx.doi.org/10.21622/ACE.2023.03.2.064

accurate object detection and semantic segmentation," IEEE Conference on Computer Vision and Pattern Recognition, pp. 580–587, 2014.

[12] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," Communications of the ACM, vol. 24, no. 6, pp. 381–395, 1981.

[13] B. Zitova and J. Flusser, "Image registration methods: a survey," Image and Vision Computing, vol. 21, no. 11, pp. 977–1000, 2003.

[14] P. Kaur and S. Singh, "A survey of edge detection techniques," International Journal of Advanced Research in Computer Science and Software Engineering, vol. 2, no. 11, pp. 274–281, 2012.

[15] J. Canny, "A computational approach to edge detection," IEEE Trans- actions on Pattern Analysis and Machine Intelligence, vol. PAMI-8,no. 6, pp. 679–698, Nov. 1986.

[16] R. C. Gonzalez and R. E. Woods, Digital image processing, 3rd ed. Pearson Prentice Hall, 2008.

[17] D. Marr and E. Hildreth, "Theory of edge detection," Proceedings of the Royal Society of London B, vol. 207, no. 1167, pp. 187–217, 1980.

[18] P. Arbeláez, M. Maire, C. Fowlkes, and J. Malik, "Contour detection and hierarchical image segmentation," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 33, no. 5, pp. 898–916, 2011.

[19] C. H. Lampert, H. Nickisch, and S. Harmeling, "Learning to detect objects with a scale-invariant template," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 31, no. 11, pp. 2109–2119, 2009.

[20] T. Lindeberg, "Feature detection with automatic scale selection," in International journal of computer vision, vol. 30, no. 2. Springer, 1998, pp. 77–116.

[21] V. Balasubramanian and S. Z. Li, "Object detection using correlation filters," IEEE Signal Processing Magazine, vol. 29, no. 4, pp. 75–86, 2012.

[22] P. Dollar, Z. Tu, P. Perona, and S. Belongie, "Frequency-based object detection," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. IEEE, 2006, pp. 311–318.

[23] T. Arici, S. Dikbas, and Y. Altunbasak, "Color moments for content- based image retrieval: A review," Signal Processing, vol. 88, no. 8, pp.2008–2021, 2008.

[24] J. Liu, Y. Xu, and H. Zhang, "Color coherence vector based image retrieval using multivariate gaussian model," IEEE Transactions on Multimedia, vol. 10, no. 4, pp. 629–640, 2008.

[25] T. Ojala, M. Pietik"ainen, and T. M"aenp"a"a, "Multiresolution gray- scale and rotation invariant texture classification with local binary patterns," IEEE Transactions on Pattern Analysis and Machine Intelli- gence, vol. 24, no. 7, pp. 971–987, 2002.

[26] A. Yilmaz, O. Javed, and M. Shah, "Object tracking: A survey," in

ACM Computing Surveys (CSUR), vol. 38, no. 4. ACM, 2006, p. 13.

[27] D. G. Lowe, "Object recognition from local scale-invariant features," in

Proceedings of the Seventh IEEE International Conference on Computer Vision. IEEE, 2001, pp. 1150–1157.

[28] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in Proceedings of the 2001 IEEE Computer SocietyConference on Computer Vision and Pattern Recognition, vol. 1, 1999,pp. I–511–I–518.

[29] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in Proceedings of the 2005 IEEE Computer Society Con- ference on Computer Vision and Pattern Recognition, vol. 1, 2005, pp. 886–893.

[30] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient graph-based image segmentation," in Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 2, 2004,pp. II–IV.

[31] D. G. Lowe, "Object recognition from local scale-invariant features," in Proceedings of the Seventh IEEE International Conference on Computer Vision, vol. 2. IEEE, 1999, pp. 1150–1157.

[32] S. Shaheen, "Object recognition using contour-based representation,"

[33] Image and vision computing, vol. 13, no. 9, pp. 751–757, 1995.

[34] D. Mumford and J. Shah, "Boundary detection by minimizing func- tionals," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 1994, pp. 22–26.

[35] T. Lindeberg, "Scale-space theory: A basic tool for analyzing struc- tures at different scales," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 16, no. 6, pp. 610–625, 1994.

[36] G. Bradski and A. Kaehler, "The opencv library," Dr. Dobb's Journal of Software Tools, 2000.

[37] J. Matas, O. Chum, M. Urban, and T. Pajdla, "Robust wide-baseline stereo from maximally stable extremal regions," Image and vision computing, vol. 22, no. 10, pp. 761–767, 2004.

[38] T. Ojala, M. Pietik"ainen, and T. M"aenp"a"a, "Multiresolution gray- scale and rotation invariant texture classification with local binary patterns," IEEE Transactions on Pattern Analysis and Machine Intelli- gence, vol. 24, no. 7, pp. 971–987, 2002.

[39] Z. Zivkovic, "Improved adaptive gausian mixture model for back- ground subtraction," in Proceedings of the 17th International Confer- ence on Pattern Recognition, vol. 2, 2004, pp. 28–31.

[40] P. J. Burt and E. H. Adelson, "The laplacian pyramid as a compact image code," Proceedings of the IEEE, vol. 72, no. 10, pp. 1273–1284, 1983.

[41] J. Canny, "A computational approach to edge detection," IEEE Transac- tions on Pattern Analysis and Machine Intelligence, no. 6, pp. 679–698, 1986.

[42] D. Gabor, "Theory of communication. part 1: The analysis of informa- tion," Journal of the Institution of Electrical Engineers-Part III: Radio and Communication Engineering, vol. 93, no. 26, pp. 429–441, 1946.

[43] R. Girshick, "Fast r-cnn," in ICCV, 2015.

[44] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real- time object detection with region proposal networks," in Advances in neural information processing systems, 2015, pp. 91–99.

[45] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 779– 788.

[46] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "Yolov5: Improved real-time object detection," arXiv preprint arXiv:2011.08036, 2020.

[47] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in European conference on computer vision. Springer, 2016, pp. 21–37.

[48] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in Proceedings of the IEEE international conference on computer vision, 2017, pp. 2980–2988.

[49] M. Tan, B. Chen, R. Pang, V. Vasudevan, M. Sandler, A. Howard,

[50] M. Zhu, and Q. V. Le, "Efficientdet: Scalable and efficient object detec- tion," IEEE Conference on Computer Vision and Pattern Recognition, pp. 10 781–10 790, 2020.

[51] X. Du, Z. Zhang, H. Li, L. Zhang, W. Wen, R. Wang, and M. Tan, "Spinenet: Learning scale-permuted backbone for recognition and localization," in CVPR, 2020.

[52] N. Carion, F. Massa, A. Kirillov, and R. Girshick, "End-to-end ob-ject detection with transformers," European Conference on Computer Vision, pp. 213–229, 2020.

[53] X. Zhu, W. Su, L. Lu, B. Li, X. Wang, and J. Dai, "Deformabledetr: Deformable transformers for end-to-end object detection," IEEE Conference on Computer Vision and Pattern Recognition, pp. 13 152– 13 161, 2021.

[54] J. Dai, Y. Li, K. He, and J. Sun, "R-fcn: Object detection via region- based fully convolutional networks," Advances in Neural Information Processing Systems, pp. 379–387, 2016.

[55] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," IEEE Conference on Computer Vision and Pattern Recognition, pp. 2117–2125, 2017.

[56] H. Law and J. Deng, "Cornernet: Detecting objects as paired key- points," in Proceedings of the European Conference on Computer Vision. Springer, 2018, pp. 734–750.

[57] Z. Tian, C. Shen, H. Chen, and T. He, "Fcos: Fully convolutional one- stage object detection," in Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 9627–9636.

[58] K. Yang, W. Zhang, Z. Li, P. Sun, H. Liu, and C.-C. J. Kuo, "Reppoints: Point set representation for object detection," IEEE International Conference on Computer Vision.

http://dx.doi.org/10.21622/ACE.2023.03.2.064

[59] X. Zhou, D. Wang, and P. Krähenbühl, "Objects as points," IEEE Conference on Computer Vision and Pattern Recognition, pp. 9626- 9635, 2019.

[60] Z. Cai and N. Vasconcelos, "Cascade r-cnn: Delving into high quality object detection," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 6154–6162.

[61] J. Dai, Y. Li, K. He, and J. Sun, "R-fcn: Object detection via region- based fully convolutional networks," in Advances in neural information processing systems, 2016, pp. 379–387.

[62] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in Proceedings of the IEEE international conference on computer vision, 2017, pp. 2961–2969.

[63] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," arXiv preprint arXiv:1804.02767, 2018.

[64] S. Zhou, Q. Huang, J. Yao, and Y. Guo, "Multi-modal multi-scale deep learning for large-scale image annotation," IEEE Transactions on Image Processing, vol. 28, no. 3, pp. 1276–1289, 2019.

[65] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 652–660.

[66] Z. Chen, Y. Wang, Y. Peng, Z. Zhang, G. Yu, and J. Sun, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 1382–1391.

[67] J. Ku, M. Mozafari, J. Lee, A. Harati, and V. Bhat, "Inception-v4, inception-resnet and the impact of residual connections on learning," in Proceedings of the AAAI Conference on Artificial Intelligence, vol. 32, no. 1, 2018, pp. 4278–4285.

[68] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 770–778.

[69] C. Feichtenhofer, H. Fan, J. Malik, and K. He, "Slowfast networks for video recognition," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 6202- 6211.

[70] Z. Li, C. Ma, and Y. Sheikh, "Deepgcns: Can gcns go as deep as cnns?" in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 9267–9276.

[71] Z. Yan and T. Kanade, "Learning to segment under various forms of weak supervision," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 926–935.

[72] B. Deng, K. Li, X. Liu, and D. Tao, "Arcface: Additive angular margin loss for deep face recognition," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2020, pp. 4690- 4699.

[73] H. Li, C. Li, G. Li, and L. Chen, "A real-time table grape detection method based on improved YOLOv4-tiny network in complex background," Biosystems Engineering, vol. 212, pp. 347–359, 2021. [Online]. Available: https://doi.org/10.1016/j.biosystemseng.2021.11.011

[74] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," CVPR, 2001.

[75] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," in PAMI, 2010.

[76] X. Wang, Q. Zhao, P. Jiang, Y. Zheng, L. Yuan, and P. Yuan, "LDS-YOLO: A lightweight small object detection method for dead trees from shelter forest," Computers and Electronics in Agriculture, vol. 198, no. January, p. 107035, 2022. [Online]. Available: https://doi.org/10.1016/j.compag.2022.107035

[77] A. Bochkovskiy, C.-Y. Wang, and H.-Y. Liao, "Yolov4: Optimal speed and accuracy of object detection," IEEE Conference on ComputerVision and Pattern Recognition, pp. 10 934–10 944, 2020.

[78] Y. Wu, A. Kirillov, F. Massa, W.-Y. Lo, and R. Girshick, "Detectron2," in NeurIPS, 2019.

[79] Y. Zhou, Y. Cui, M. Wang, and J. Xu, "Detectron2: A pytorch-based modular object detection library," arXiv preprint arXiv:2006.03137, 2020.

[80] P. Wang, P. Chen, Y. Yuan, D. Liu, Z. Huang, X. Hou, and G. Cottrell,"Sparse r-cnn: End-to-end object detection with learnable proposals,"in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 14 783–14 792.

[81] P. Dollar, C. Wojek, B. Schiele, and P. Perona, "Pedestrian detection: An evaluation of the state of the art," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 34, no. 4, pp. 743–761, 2012.

[82] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in 2012 IEEE Conferenceon Computer Vision and Pattern Recognition. IEEE, 2012, pp. 3354– 3361.

[83] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma,

Z. Huang, A. Karpathy, A. Khosla, M. Bernstein et al., "Imagenet large scale visual recognition challenge," International Journal of Computer Vision, vol. 115, no. 3, pp. 211–252, 2015.

[84] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisser- man, "The pascal visual object classes (voc) challenge," International Journal of Computer Vision, vol. 88, no. 2, pp. 303–338, 2010.

[85] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan,

P. Dollar, and C. L. Zitnick, "Microsoft coco: Common objects in context," in Computer Vision–ECCV 2014. Springer, 2014, pp. 740– 755.

[86] A. Kuznetsova, H. Rom, N. Alldrin, J. Uijlings, I. Krasin, J. Pont- Tuset, S. Kamali, S. Popov, M. Malloci, T. Duerig, and V. Ferrari, "The open images dataset v4: Unified image classification, object detection, and visual relationship detection at scale," arXiv preprint arXiv:1811.00982, 2018.

[87] P. Zhu, L. Wen, X. Bian, H. Ling, and Q. Hu, "Vision meets drones: A challenge," arXiv preprint arXiv:1804.07437, 2018.

[88] L. Jiao, F. Zhang, F. Liu, S. Yang, L. Li, Z. Feng, and R. Qu, "A survey of deep learning-based object detection," IEEE Access, vol. 7, pp. 128 837–128 868,

2019.

[89] X. Lou, Y. Huang, L. Fang, S. Huang, H. Gao, L. Yang, Y. Weng, and I.-K. Hung, "Measuring loblolly pine crowns with drone imagery through deep learning," J. For. Res., vol. 33, no. 1, pp. 227–238, 2021.

[90] A. Ammar, A. Koubaa, and B. Benjdira, "Deep-learning-based auto- mated palm tree counting and geolocation in large farms from aerial geotagged images," Agronomy, vol. 11, no. 8, p. 1458, 2021.

[91] W. Chen, S. Lu, B. Liu, G. Li, T. Qian, and J. Huang, "Detecting citrus in orchard environment by using improved yolov4," Sci. Program., vol.2020, pp. 1–13, 2020.

[92] T. Jintasuttisak, E. Edirisinghe, and A. Elbattay, "Deep neural network based date palm tree detection in drone imagery," Comput. Electron. Agric., vol. 192, p. 106560, 2022.

[93] J. Redmon and A. Farhadi, "Yolo9000: Better, faster, stronger," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 7263–7271.

[94] G. Jocher, A. Wong, I. Vasilev, and M. Green, "Yolov5: A universal object detection framework," in Proceedings of the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 2021, pp. 429–437.

[95] B. Wong, S. Wu, Q. Lin, J. Liang, W. Cheng, Z. Li, and J. Ho, "Yolo nano: a highly compact you only look once convolutional neural network for object detection," in Proceedings of the 2020 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays. ACM, 2020, pp. 161–170.

[96] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, "Yolov5s: Improved version of yolov4 tiny," arXiv preprint arXiv:2103.16857, 2021.

[97] Wang, Chien-Yao and Bochkovskiy, Alexey and Liao, Hong-Yuan Mark, "Yolov5m: Yolov5 with more parameters," arXiv preprint arXiv:2104.08501, 2021.

[98] Wang, Chien-Yao and Bochkovskiy, Alexey and Liao, Hong-Yuan Mark, "Yolov5l: Stronger yolov5 with longer anchor boxes," arXiv preprint arXiv:2104.10472, 2021.

[99] Wang, Chien-Yao and Bochkovskiy, Alexey and Liao, Hong-Yuan Mark, "Yolov5x: Improved yolov5 backbone with dropblock," arXiv preprint arXiv:2105.04217, 2021.

[100] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "Yolov6: Re- designing yolo for improved accuracy and speed," arXiv preprint arXiv:2104.06580, 2021.

[101] Wang, Chien-Yao and Bochkovskiy, Alexey and Liao, Hong-Yuan Mark, "Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors," arXiv preprint arXiv:2105.04206, 2021.

[102] D.-Y. Chen, W.-C. Chiu, Y.-D. Chen, Y.-H. Liu, S.-H. Lai, S.-H.Lai, and Y.-T. Chen, "Yolov8: A scalable object detection system for complex environments," arXiv preprint arXiv:2211.00623, 2022.

[103] H. Lou, X. Duan, J. Guo, H. Liu, J. Gu, L. Bi, and H. Chen, "Dc- yolov8: Small-size object detection algorithm based on camera sensor," Journal of Computer Science and Technology, 2023.

[104] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and

C. L. Zitnick, "Microsoft coco: Common objects in con-text," IEEE Transactions on Pattern Analysis and Machine Intelligence(TPAMI), vol. 39, no. 12, pp. 2414–2425, 2017.

[105] S. Gupta, R. Girshick, P. Arbeláez, and J. Malik, "Learning rich features from rgb-d images for object detection and segmentation," in European Conference on Computer Vision (ECCV), 2014.

[106] H. Zhao, Y. Zhang, S. Liu, J. Shi, C. C. Loy, and D. Lin, "Psanet: Point- wise spatial attention network for scene parsing," in IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018.

[107] X. Chen, H. Ma, J. Wan, B. Li, and T. Xia, "Multi-view 3d object detection network for autonomous driving," in IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017.

[108] Y. Li, K. He, and J. Sun, "R-fcn: Object detection via region-based fully convolutional networks," in Neural Information Processing Systems (NIPS), 2016.

[109] R. M. Alamgir, A. A. Shuvro, M. A. Mushabbir, M. A. Raiyan, N. J. Rani, M. M. Rahman, M. H. Kabir, and S. Ahmed, "Performance Analysis of YOLO-based Architectures for Vehicle Detection from Traffic Images in Bangladesh," 2022. [Online]. Available: http://arxiv.org/abs/2212.09144

[110] M. C. Keles, B. Salmanoglu, M. S. Guzel, B. Gursoy, and G. E.Bostanci, "Evaluation of YOLO Models with Sliced Inference for Small Object Detection," arXiv e-prints, p. arXiv:2203.04799, 2022. [Online]. Available: http://arxiv.org/abs/2203.04799

[111] C. Wang, A. Bochkovskiy, and H. Liao, "Yolov7: Trainable bag-of- freebies sets new state-of-the-art for real-time object detectors," in Proceedings, 2022.

[112] U. Sirisha, S. P. Praveen, P. N. Srinivasu, P. Barsocchi, and A. K.Bhoi, "Statistical analysis of design aspects of various yolo-based deep learning models for object detection," International Journal of Computational Intelligence Systems, vol. 16, no. 1, p. 126, 2023.

[113] A. Bochkovskiy, H.-Y. Liao, and C.-Y. Wang, "Yolov8: Towards real- time object detection," arXiv preprint arXiv:2301.05417, 2023.

[114] A. Bochkovskiy, C.-Y. Wang, H.-Y. Liao, and R. Girshick, "Yolov5: A unified, fast, and accurate object detection system," arXiv preprint arXiv:2004.10934, 2020.

[115] A. Bochkovskiy, H.-Y. Liao, and C.-Y. Wang, "Yolov6: An improved implementation of yolov5," arXiv preprint arXiv:2101.08697, 2021.

[116] J. Redmon and A. Farhadi, "Yolov4: Optimal speed and accuracy trade-offs for object detection," arXiv preprint arXiv:1804.04381, 2018.

[117] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "Yolo: Real-time object detection," arXiv preprint arXiv:1506.02640, 2015.

[118] R. Girshick, J. Malik, and T. Darrell, "Mean average precision (map) for object detection," IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), 2014. [Online]. Available: https://arxiv.org/abs/1412.0076