SMALL TARGET DETECTION ON AN UNCREWED SURFACE VESSEL

Final Report

THE UNIVERSITY OF SYDNEY

Group Based 5703 Capstone Project

Group Members

1. Yinglong Chen (530548896)
2. Shikun Tong (520520965)
3. Kaiyue Wang (530402325)
4. Weibin Liu (520415063)
5. Simiao Yu (520428764)
6. Bozhao Zhang (530464862)

CONTRIBUTION STATEMENT

Our group, taking project CS07-1, with group members Yinglong Chen,Shikun Tong,Kaiyue Wang,Weibin Liu,Simiao Yu and Bozhao Zhang , would like to state the contributions each group member has made for this project during this semester:

- Yinglong Chen: Project coordinator, assist with DETR model building and evaluation, project communication, report writing and presentation recording.
- Shikun Tong: DETR model building and performance evaluation, report writing, data annotator.
- Kaiyue Wang: YOLO-V8 model building and performance evaluation , report writing , data annotator.
- Weibin Liu: Maskrcnn model building and training, group shared document manager, data preprocessing, report writing and presentation completion.
- Simiao Yu: YOLO V5 model building and evaluate, write final report, data annotator.
- Bozhao Zhang: Faster R-CNN modelling and evaluation, experiment management and analysis.

All group members agreed on the contributions listed on this statement by each group member.

Signatures: Yinglong Chen, Shikun Tong, Kaiyue Wang, Weibin Liu, Simiao Yu and Bozhao Zhang

ABSTRACT

Uncrewed surface vessels can drive on the surface of the water through remote control without requiring human pilots on board. Such characteristics make it possible to perform tasks in complex and dangerous environments without exposing personnel to risks. At the same time, data can be collected continuously due to its long operation. It has a broad prospect to be used in related target detection tasks. Therefore, it will be widely used in Marine security and environmental monitoring, expanding the Navy's monitoring and reconnaissance capabilities to help detect Marine pollution, climate change and other issues. However, there are some challenges in target detection through images collected by uncrewed surface vessels, because its detection range is different in the size of the target, vulnerable to the impact of the surface environment resulting in blurred images, and its own energy consumption is fast, which will make the quality of the collected images vary. In order to address these challenges, the capabilities of different models for water object detection will be compared in this report. The project will first train and evaluate each model on the MS COCO dataset, which allows the model to be exposed to a wealth of visual features during training, allowing the generalization ability of each model to be compared even when the project dataset is limited. Second, each model was trained and its performance evaluated using the project's water object dataset. The models selected are YOLO-V5, YOLO-V8, Mask RCNN, Fast RCNN, DETR. The mAP indicators of each model will be evaluated during the evaluation process. After comprehensive evaluation, the YOLO-V8 model outperforms other object detection algorithms on COCO and project data sets. For particular situations when efficiency and real-time processing are crucial, the faster R-CNN provides a compromise between speed and accuracy. Our findings make clear the advantages and disadvantages of every model, which helps us select the one that best fits our requirements for ocean monitoring.

Table of Contents

1. INTRODUCTION

The uncrewed surface vessel is a vessel that can be remotely controlled to perform a variety of tasks on or above the surface of the water without the need for human piloting on board. The uncrewed surface vessels are usually equipped with cameras and other devices to record the surrounding environment. Therefore, it is a type of equipment that can be used on the water to perform reconnaissance tasks, such as target detection or environmental monitoring, at a lower cost and efficiency (Zhou, Hu, Li, Jing, & Qu, 2022). Because of the rapid development of the field of target detection, the work of uncrewed surface vessels in the identification of objects on the water has become more effective. (Cheng et al., 2022).

At this stage, there is already some relevant work for the detection of water objects. According to Cheng et al. (2022), a water target dataset was established. Then, various techniques for optimizing the YOLOv4 algorithm are considered to enable the target recognition system to operate in complex water target environments while maintaining its stability, speed, and accuracy. Shin, Lee, and Lee (2020) also proposed a data enhancement method that can automatically expand the target detection dataset in ocean images, which makes the target detection results of unmanned surface ships more accurate. In addition, Yu et al. (2023) proposed a small-target enhanced YOLOv7 method aimed at improving the multi-scale target detection capability of unmanned surface ships and solving the problem of identifying targets of different sizes.

In this project, the target company is called Ocius. The company has a fleet of robotic sailing vessels called "Bluebottle", which integrates advanced visual perception, image analysis and artificial intelligence technologies to provide a strong technical support for real-time monitoring and security of maritime activities. It is mainly used in the scientific and defense industries. The Bluebottle had several cameras installed to look for ships that weren't supposed to be there. By continuously shooting and returning image data, it provides a large amount of original information for the intelligent analysis system in the background. Through the rapid analysis of the information recorded by the camera, it helps to quickly identify potential hazards,

so that emergency response can be made quickly, and ultimately ensure the safety of maritime activities.

Figure 1: The Bluebottle by OCIUS

This project requires the team to create a target detection model that will enable Bluebottle to perform surveillance missions more effectively, automatically detect potential threats in the vast ocean, improve maritime safety, and provide important data for science and national defense. However, because it is expensive, small, and vulnerable to shaking at sea, it presents unique target detection challenges. Improving the target detection capabilities of these vessels will help improve their cost-effectiveness and surveillance effectiveness. The bluebottle's surveillance capabilities can be enhanced by developing more efficient and accurate detection models to identify potential threats faster. At the same time, developing high-performance models can help alleviate the problem of ship wobbling caused by wind and waves at sea and the blurring of targets in different weather conditions. By addressing these issues, the number of vessels required can be reduced and the company can operate more cost-effectively.

2. Related Literature

2.1 Literature Review

Recent advances in computer vision and deep learning have had a significant impact on maritime surveillance, especially for unmanned surface vessels such as the Bluebottles developed by Ocius. These vessels face unique challenges such as motion blurring due to sea conditions, wide detection radius, and energy constraints due to reliance on solar power. ensembles of deep learning models such as YOLOv5, YOLOv8, DETR, Mask R-CNN, and Faster R-CNN are becoming increasingly important in addressing these challenges.

Zhou et al. (2021) highlight the effectiveness of the YOLOv5 model in maritime environments, addressing issues such as motion blur and wide-area surveillance - common challenges faced by vessels such as the Bluebottles.The YOLO model introduced by Redmond et al. (2016) signalled a shift towards real-time object detection, which is critical for autonomous operation of USVs.

The DETR model discussed by Carion et al. (2020) uses a Transformer-based architecture that eliminates the need for hand-crafted components such as non-maximum suppression, thus simplifying the detection process and potentially reducing computational overheads, which is critical for energy constraints on solar vessels.

The Mask R-CNN proposed by He et al. (2017) extends the Faster R-CNN by adding instance segmentation branches, allowing for accurate depiction of objects, which facilitates the detection of small or partially occluded maritime targets.The Facebook AI team (Kirillov et al., 2019) developed Detectron2 to further improve its efficiency and accuracy in complex environments.

The Faster R-CNN detailed by Ren et al. (2015) combines a regional proposal network to achieve efficient and accurate detection by focusing computational resources on likely object locations, an approach suitable for the variety of conditions encountered by maritime surveillance vessels.

In terms of computational efficiency, Howard et al. (2017) and Sandler et al. (2018) explored MobileNets and MobileNetV2, which are critical for maintaining high levels of detection performance within the stringent energy constraints of USVs such as Bluebottles. These models optimize the trade-off between latency and accuracy and are suitable for deployment in energy-constrained environments.

This review highlights the dynamic development of deep learning for marine object detection, focusing on the unique operational challenges faced by uncrewed surface vessels.The continued development of models such as YOLOv5 and emerging technologies for model efficiency play a critical role in enhancing the autonomous capabilities of USVs, ensuring they meet stringent maritime security and surveillance requirements.

3. Project Problems

3.1 Project Aims & Objectives

The primary objective of the project was to include a target detection model into Bluebottle's surveillance system in order to improve the capabilities of the previously established surveillance system over water. The object detection model needs to be able to analyze the image data collected by Bluebottle, automatically generate bounding boxes around detected objects, and detect the categories of objects in the image. For example, boats. Therefore, in this project, the team will use the technology of machine learning and deep learning to build a model for target detection of the data set provided by the project. The model will be trained using the MS COCO dataset and evaluated for its generalization. After that, the model will be trained and evaluated using the dataset provided by the project to test the effectiveness and results of the final model. The final deliverables of the project will be the model code and a detailed experimental report. The report includes the methodology, resources used, experiment results and discussion, limits and future work, and a literature review.

3.2 Project Questions

There will be some problems during the implementation of the project, which will bring challenges.

First of all, Bluebottle has a small number of vessels due to its high price, and to ensure cost-effectiveness, it has a large detection radius (about 5km). This means that the water targets it detects will appear to vary in size depending on the distance. It will be difficult to detect some small objects on the water because some small objects cannot be identified by the human eye, which makes it difficult for each deep learning model to learn these small objects and leads to detection failure.

In addition, due to the small size of the Bluebottle hull, it is vulnerable to sea and wind waves. This can cause motion blur in the photos taken by the ship's monitoring equipment, which can lead to degraded image quality. To solve this problem, the project must consider adopting a more resilient object recognition model that can accurately identify objects in low-resolution photos by utilizing deep learning methods combined with data enhancement and transfer learning.

Furthermore, due to Bluebottle's reliance on solar energy, careful energy management is required. The model must operate with minimal resource and energy consumption. The need to build efficient models will reduce energy consumption, and extending its uninterrupted operation time in water will be a key challenge.

3.3 Project Scope

For the scope of the project, the final deliverable will be Python code for object detection models against images collected by Bluebottle, which can run smoothly in the client's environment and meet the client's needs, and will not be involved in integrating the model into Bluebottle's system. At the same time, the team will provide a detailed experimental report. In this report, the target detection model used will be introduced in detail, and the final test results obtained will be discussed and analyzed. Depending on the project requirements, the team can choose to provide multiple models that meet different requirements. In consideration of Bluebottle's various challenges and particularities, the delivered model should strike a balance between high precision and high efficiency, and the model should have performance in detecting small and fuzzy objects. For high precision, the team will select several

evaluation indicators for explanation and discussion in the following chapters, and conduct detailed comparison and analysis of the results of different models. For efficiency, the team will focus on comparing inference times across models.

4. METHODOLOGIES

4.1 Methods

4.1.1 Yolo V5

Yolo V5 is a highly efficient deep learning model designed specifically for real-time detection of small objects. Glenn Jocher released Yolo V5 in 2020. Yolo V5 expands the model architecture of the previous YOLO series algorithms. As depicted in the diagram below, the model consists of four main parts: the input module, the backbone network for feature extraction, the neck network for cross-scale feature fusion, and the prediction network for completing object detection(Lei et al., 2022).

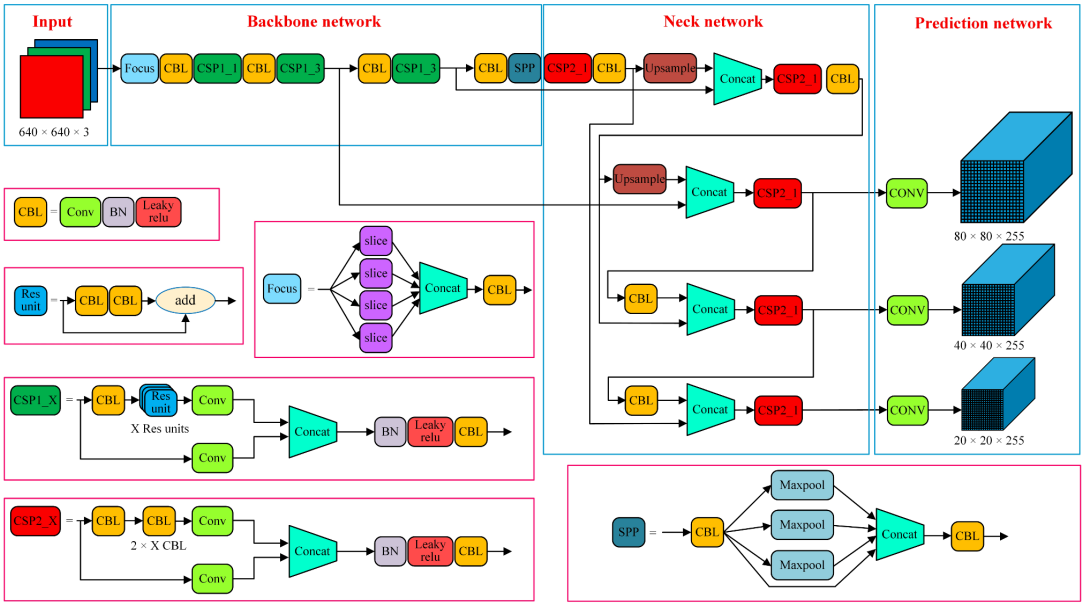


Figure 2: The network structure of Yolo V5

The input section is responsible for initialising the image data in preparation for subsequent deep feature extraction.The input section of the YOLO network handles images with dimensions of 640x640x3 , indicating that it processes colour images with three RGB channels. This initial layer is crucial for setting up the spatial

dimensions and depth of the image data, preparing it for entering the network and performing complex feature extraction.

The backbone network processes image features through multi-layer convolution and residual concatenation to extract basic visual information for object detection. It starts with a ‘focus’ module, which may adjust the focus on key features or change the image size to optimize processing. Afterwards, the image data is passed through several CBL modules and two important CSP modules, CSP1 and CSP2, which are designed to enhance feature propagation and integration in the network through various residual units. At the end of the backbone, an SPP module uses spatial pyramid pooling to capture multi-scale contextual information, which is crucial for detecting objects of different sizes.

The neck network enhances and refines features through up-sampling and feature fusion techniques in preparation for accurate detection. In the neck network, the feature map is enhanced by up-sampling and joining processes that increase resolution and combine features from different network depths. This part of the network utilises multiple up-sampling steps followed by connectivity with lower resolution feature maps through the CSP2 and CBL modules. These steps are crucial to refine the features prior to prediction, allowing the network to maintain highly detailed feature recognition across a range of object sizes and complexities.

The prediction network uses the processed features for object specific prediction of class, location and confidence. This part uses convolutions of different scales (80x80x255, 40x40x255, 20x20x255) to accommodate objects of various sizes, converting deep features into spatial predictions. Each convolution output provides key detection information including category probabilities, bounding box coordinates, and confidence levels, which are critical for accurate object localisation and recognition.

4.1.2 Yolo V8

The YOLO-V8 model is one of the most state-of-the-art real-time object detection models nowadays , which has been developed and used in various applications or hardware such as the human detection through the surveillance camera and vehicles detecting with high accuracy and high interface speed suitable for our small target detection on vessel task . Also the lightweight design or

implementation for hardware is suitable for the resource limitation task like our project which is only powered by solar energy.

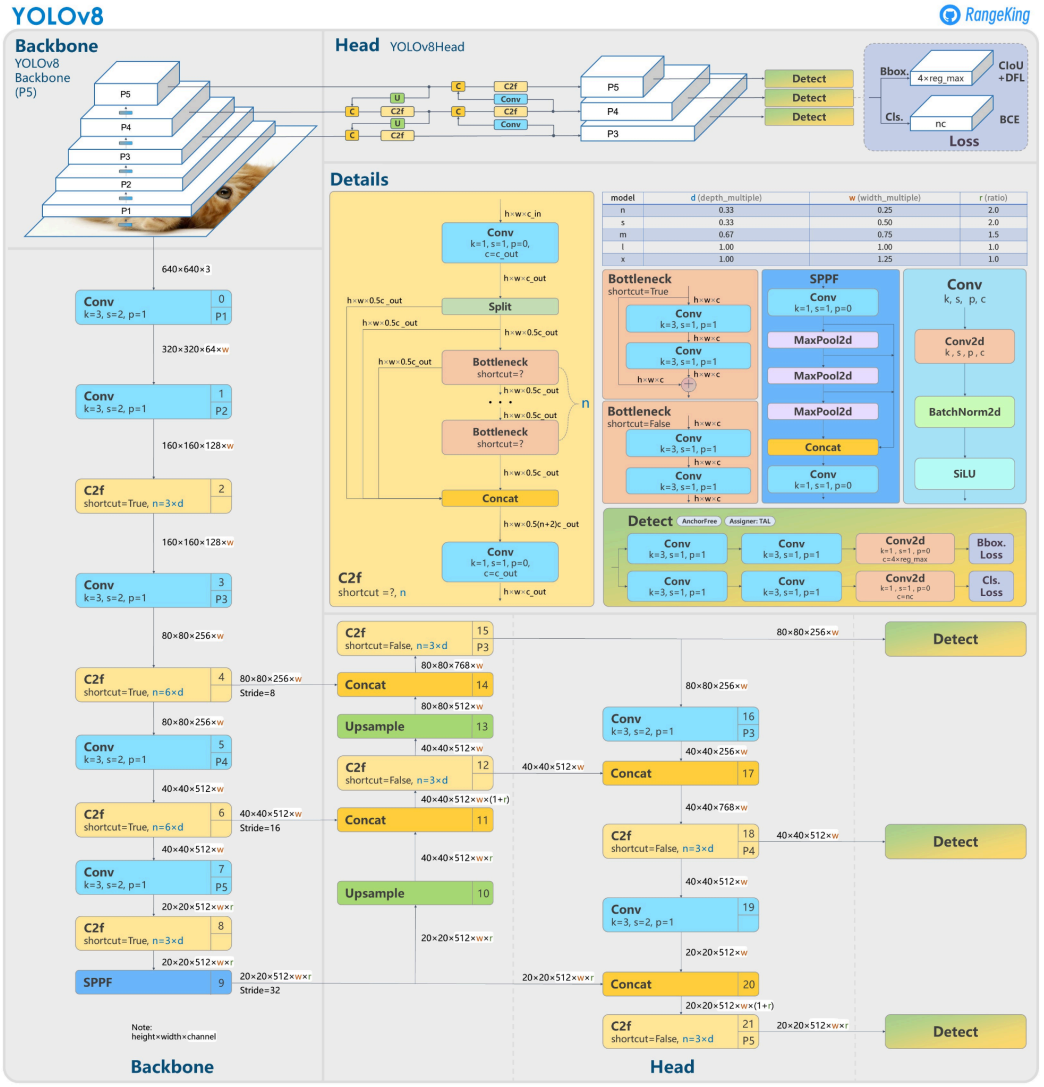


Figure 3 : Architecture of YOLO-V8

According to the architecture of YOLO-V8 which is published on GitHub (King,2023) which is presented above , there are three main blocks which are Backbone , Neck and Head . Backbone can extract the features out of the images, which are made up by multiple convolutional layers and employ the cross-stage partial connections to improve the flow between the different layers (Mehra, 2023), so that multiple levels of the meaningful feature can be extracted.

Neck is the bridge between Backbone and Head, it can collect the feature map of different sizes from the Backbone and concatenate the features of different scales to improve the ability to express various features.

Head is the architecture which contains multiple convolutional layers and the fully connected layers to output the bounding box and classification and make the final object detection. The self-attention mechanism is one of the key features in Head which allows model to adjust the importance of different features based on their relevance to the task (Mehra, 2023) which allows the model to detect the small or large object at different scale by utilizing a feature pyramid network made up by several convolutional layers .

According to Jocher, Chaurasia, and Qiu (2023) , compared to YOLO-V5 , YOLO-V8 has higher inference speed which is below 100 ms at most times. Also the architecture of YOLO-V8 is optimized by utilizing more convolutional layers with more parameters, residual block and self-attention mechanism and achieving stronger ability to extract the features and higher accuracy of object detection especially for the small object detection suitable for our task . As the performance of YOLO-V5 and V8 shown below , YOLO-V8 has higher mAP and higher inference speed across various parameters than YOLO-V5.

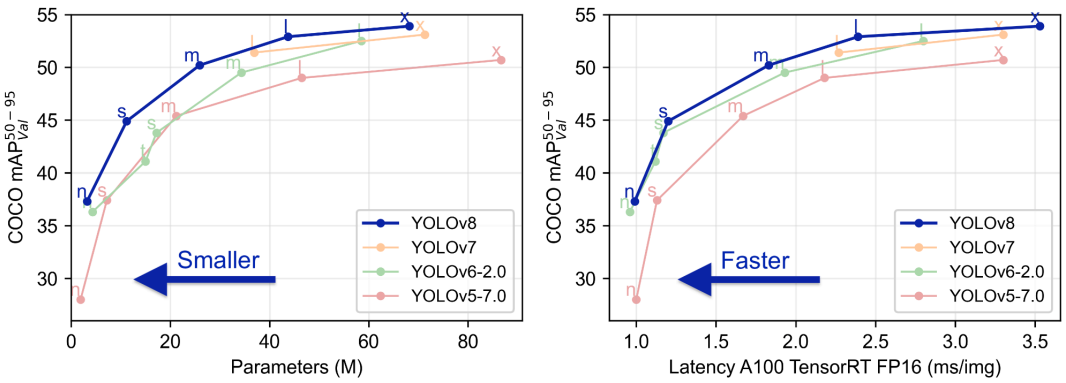
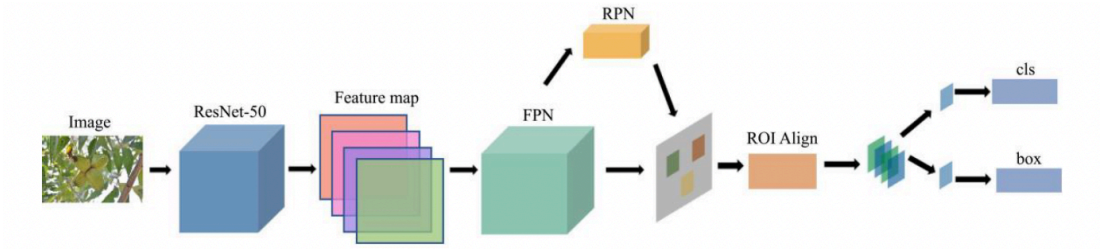


Figure 4 : Comparison between YOLO-V5 and YOLO-V8

4.1.3 Faster R-CNN

Faster R-CNN (Region-based Convolutional Neural Network) with Feature Pyramid Networks (FPN) is an efficient and widely used model for target detection, especially for small target detection tasks. Faster R-CNN is a two-stage target detection framework that first generates candidate regions through a region proposal network (RPN), and then classifies and regresses these candidate regions (Hu & Wang, 2020). The model combines the powerful feature extraction capabilities of convolutional neural networks (CNNs), enabling it to handle complex detection

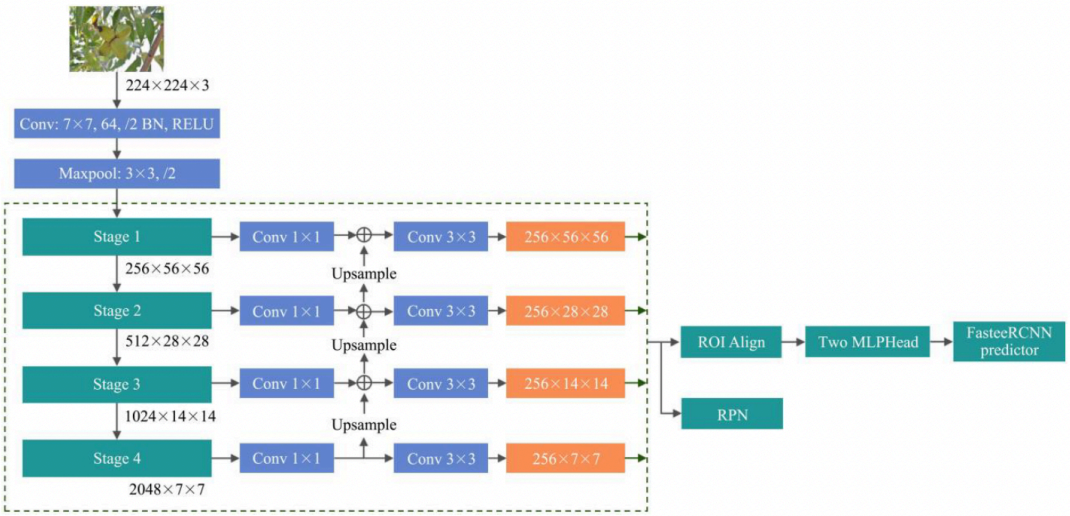
tasks. The introduction of FPNs further enhances its ability to detect multi-scale targets (Hu et al., 2022). FPNs generate feature pyramids at multiple scales by fusing shallow and deep features through top-down paths and lateral connections, which significantly improves small target detection.



Note: ResNet-50: Residual networks with 50 layers; RPN: Region Propose Network; FPN: Feature Pyramid Networks; ROI: Regions of Interest; cls: Classification; bbox: bounding box.

Figure 5: Faster R-CNN R50-FPN

Architecturally, Faster R-CNN uses a residual network (ResNet-50) as its backbone network. The residual network solves the problem of gradient vanishing in deep network training by introducing jump connections, which improves the efficiency and accuracy of feature extraction. In addition, FPN realizes the effective fusion of feature maps with different resolutions by attaching 3×3 convolutional layers and 1×1 convolutional layers at each scale (Ren, Zhu, & Xiao, 2018).



Note: Conv: Convolution; MLP Head: Multi-Layer Perceptron Head; Faster RCNN: Faster Region Convolutional Neural Network.

Figure 6: Structure between ResNet-50 and FPN

Another key improvement of Faster R-CNN is the introduction of ROI Align in place of traditional ROI Pooling, which eliminates quantization errors and improves the localization accuracy of candidate regions. This improvement makes

the model perform better when dealing with small targets with irregular shapes and fuzzy edges (Xie, Cheng, Wang, Yao, & Han, 2021).

In the unmanned submarine project, the Faster R-CNN with FPN model is able to achieve accurate detection of small targets in complex marine environments through efficient feature extraction and multi-scale fusion. For example, in the fruit detection experiments, the model's mean average precision (mAP) significantly outperforms other classical network structures, showing its superior performance in complex environments (Cheng et al., 2018).

4.1.4 Mask R-CNN

The Mask R-CNN model is one of the most advanced object detection and instance segmentation models, and is widely used in various application scenarios and hardware devices, such as human detection through surveillance cameras and high-precision detection of vehicles. Its efficient interface speed and high accuracy make it very suitable for small object detection tasks.

According to the paper published by He et al. in 2017 and the public implementation on GitHub (Abdulla, 2017), the architecture of Mask R-CNN mainly consists of three parts: Backbone, Region Proposal Network (RPN) and Multi-task Heads. The backbone network is responsible for extracting features from the image. It consists of multiple convolutional layers and combines features at different levels through the Feature Pyramid Network (FPN) to extract meaningful features at multiple levels (He et al., 2017).

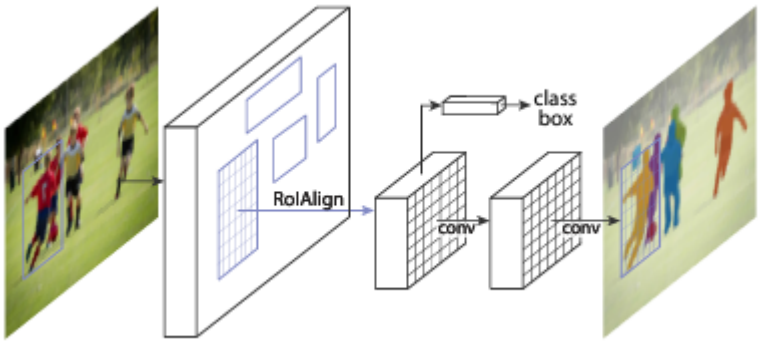


Figure 7. Schematic diagram of the operation process of the Mask R-CNN model

RPN is the bridge connecting the backbone network and the multitask head, responsible for generating candidate regions. RPN generates a large number of

candidate boxes on the feature map through a sliding window mechanism, and classifies (foreground or background) and regresses (adjusts the size and position of the candidate box) each candidate box, thereby improving the accuracy of detection.

The multi-task head includes ROIAlign, classification head and mask header. ROIAlign effectively improves the segmentation accuracy by accurately aligning the features of the candidate region. The classification head and regression head are used to classify and accurately locate the bounding box of each candidate region, while the mask header generates the segmentation mask of each candidate region (He et al., 2017).

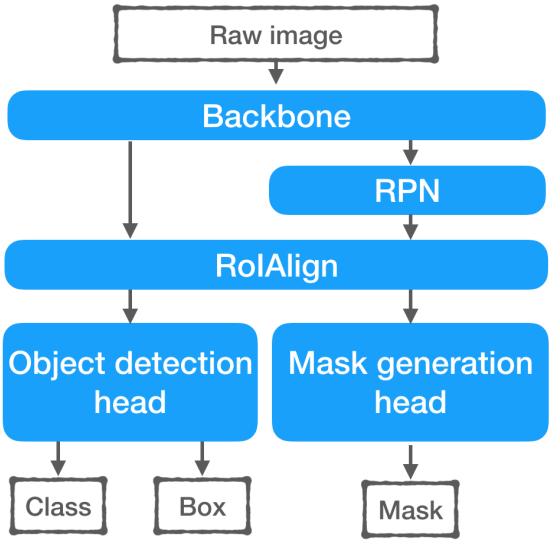


Figure 8. Architecture diagram of the Mask R-CNN model

Compared with Faster R-CNN, Mask R-CNN has higher accuracy and better performance in instance segmentation tasks. Its architecture introduces more convolutional layers and more complex feature extraction mechanisms, making the model particularly good at handling complex backgrounds and small target detection. Detectron2 is an open source framework developed by Facebook AI Research (FAIR) to efficiently implement Mask R-CNN, providing rich tools and flexible interfaces for researchers to conduct experiments and model optimization (Kirillov et al., 2019). By using Detectron2 for training and optimization, Mask R-CNN not only excels in accuracy, but also has significant improvements in inference speed and resource efficiency, making it very suitable for application scenarios that require high precision.

4.1.5 DETR

DETR is a target detection model based on a transformer architecture, originally proposed in the paper "End-to-End Object Detection with Transformers".Vedaldi, Bischof, Brox, and Frahm (2020) state that the DETR model consists of three main components: a CNN backbone for extracting compact feature representations, a CNN backbone, an encoder-decoder converter, and a simple feed-forward network (FFN) that performs the final detection prediction.

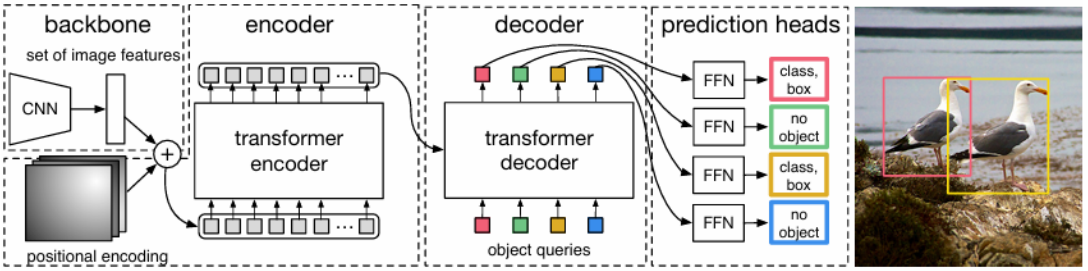


Figure 9: DETR model structure

In the DETR model, the image features are first extracted by a Convolutional Neural Network (CNN) backbone network, and then they are passed to the Transformer decoder. Each image feature is combined with spatial location coding in the encoder's multi-headed self-attentive layer, which helps the model to understand the relative positional relationships of the parts in the image. The decoder processes a series of predefined object queries that interact with the encoder's output to capture specific target information in the scene. In the decoder, a multi-head self-attention layer always pays attention to the query output of the previous layer, in addition to a multi-head cross-attention layer that allows the query to interact with the output of the decoder. Each query category and bounding box prediction of the decoder output is finally computed by two independent feed-forward networks (FNNs), which determines the target category and bounding box location.

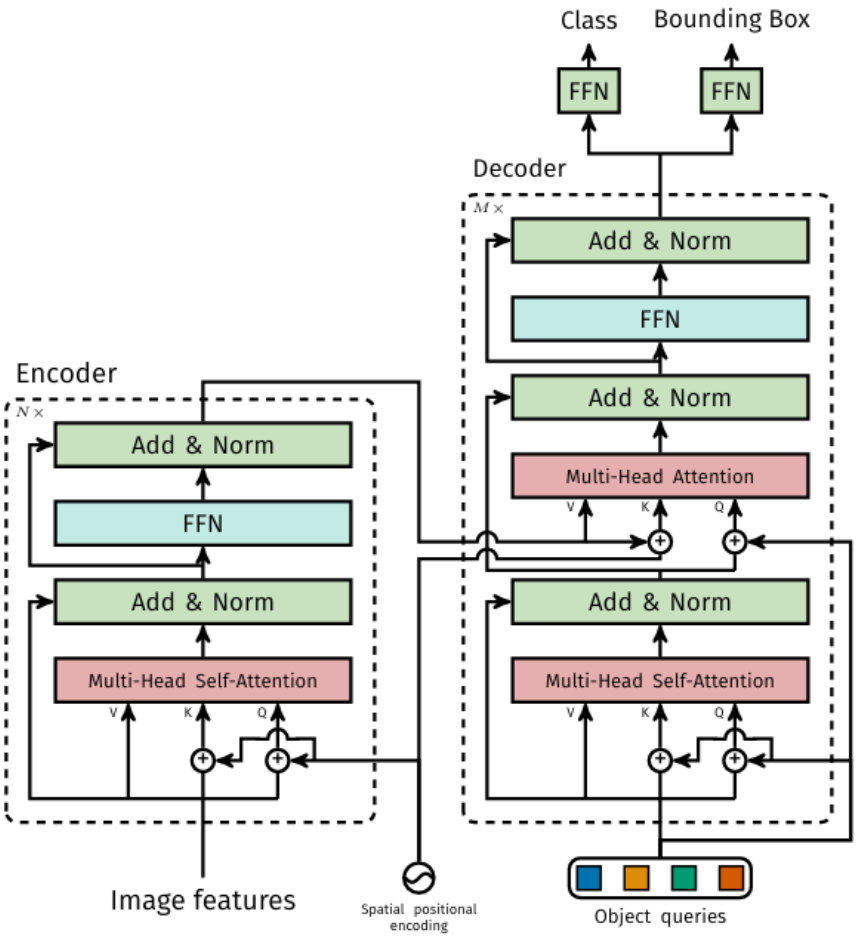


Figure 10: Transformer structure

Meanwhile, DETR employs an innovative loss function design, the Hungarian algorithm. Since traditional target detection usually generates multiple overlapping prediction frames, redundant predictions are removed by non-maximal suppression (NMS). However, DETR uses the Hungarian algorithm to ensure that each prediction frame uniquely matches a real target, thus avoiding the problems of duplicate detection and missed detection. Moreover, since DETR directly outputs non-overlapping target bounding boxes, Vedaldi, Bischof, Brox, and Frahm (2020) mentioned that DETR eliminates the complex post-processing step of NMS and anchor box design, which improves the efficiency of the detection and reduces the model complexity. However, the Hungarian algorithm is very sensitive to small variations in the calculation of the cost matrix, which may lead to inconsistent matches between predicted results and real labels (Senthivel, Vu, & Borzic, 2023). Therefore, this is something that needs to be considered in the experiments.

$$\mathcal{L}_{\text{Hungarian}}(y, \hat{y}) = \sum_{i=1}^{N} \left[-\log \hat{p}_{\hat{\sigma}(i)}(c_i) + \mathbb{1}_{\{c_i \neq \varnothing\}} \mathcal{L}_{\text{box}}(b_i, \hat{b}_{\hat{\sigma}}(i)) \right]$$

To train the DETR model, the standard coco dataset, which is a widely used dataset for target detection, was first used. The data is derived from the client's videos, which the team outputs by taking screenshots and labelling them using cvat and finally outputs the corresponding images and labelled files in JSON format. These files are further divided into training and validation sets to support the training and evaluation of the model.

4.2 Data Collection

4.2.1 MSCOCO Dataset

The MS COCO (Microsoft Common Objects in Context) dataset is a widely used computer vision dataset, especially suitable for object detection tasks. In this project, we use the train2017 and val2017 folders to store training and validation set images respectively. Use instances_train2017.json and instances_val2017.json as annotation information containing object detection and instance segmentation.

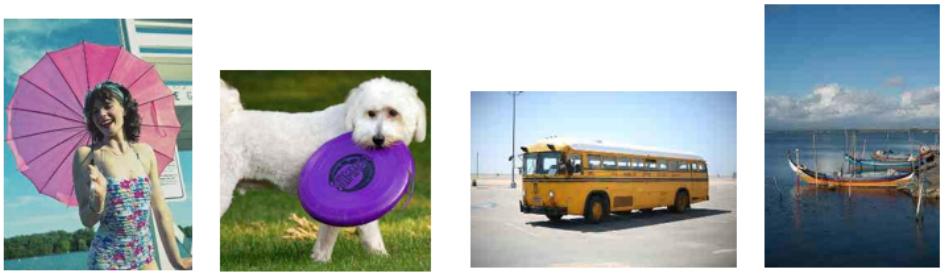


Figure 11: Example of COCO Data

4.2.2 Project Dataset

The project data mainly studies the video data of the camera on the masthead. The video was recorded for a total of 6 days, with one video every five minutes. The video format is .265, and the video title consists of time, for example: 2021-10-08 20-15-00.265.

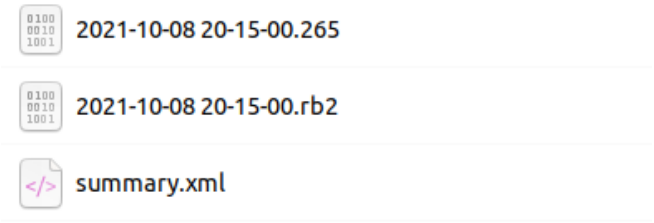


Figure 12: Example of Raw Project Data

After meeting with the client, our team decided to use code to capture pictures from different videos to create a picture data set. We select 100 of them as the training set and 100 images as the verification set. Each data set contains 50 daytime images and 50 nighttime images, and these images are labeled. We also retain the video naming format for each picture, so that the pictures we select can represent the corresponding time period.

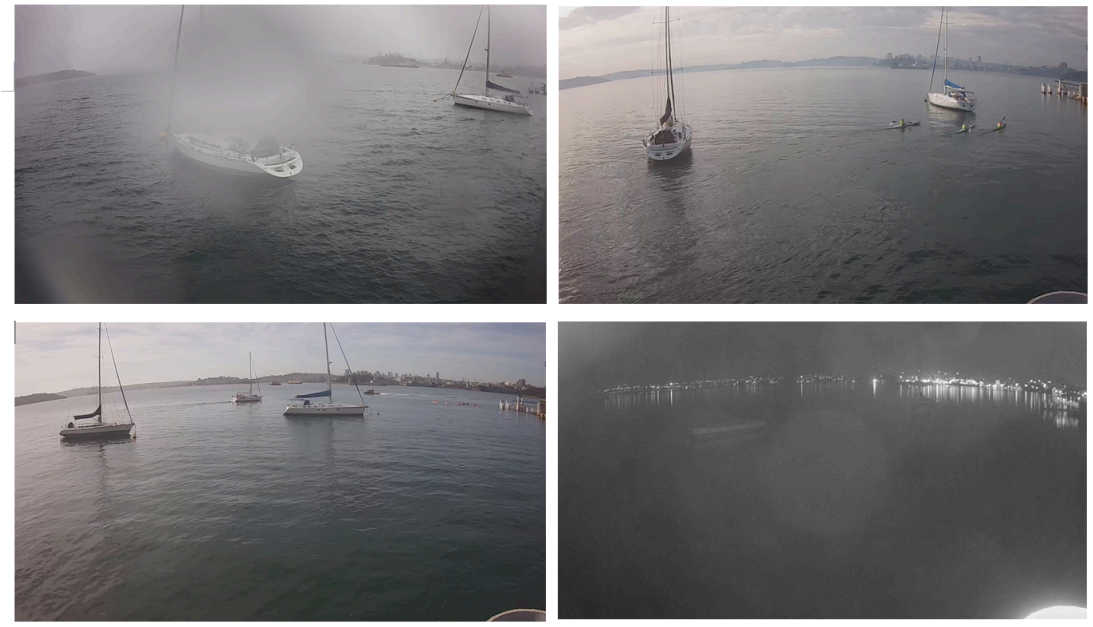


Figure 13: Project Image Examples

4.3 Data Analysis

The first one is the MS COCO dataset. We analyze its image size, resolution, and diversity. This is beneficial to our subsequent data preprocessing, such as image resizing, normalization, and data augmentation. The second one is the customer's dataset. We previously extracted 200 pictures of different weather and time periods

from the video dataset. Among them, 100 pictures were taken during the day and 100 pictures were taken at night. We took 50 pictures each as the training set and the validation set. There is also a set of datasets where we selected an additional 100 pictures, 80 of which were used as training sets and 20 as validation sets. To do small target detection training and validation. We will visualize the distribution of the data. If the categories of the data are unbalanced, we will adopt strategies such as oversampling or undersampling to improve the generalization ability of the model. For the training data, we will uniformly scale the images to the same size and normalize the pixel values to a certain range to reduce the computational complexity. At the same time, we use Cvat to label the images, only labeling all the boats in the image.

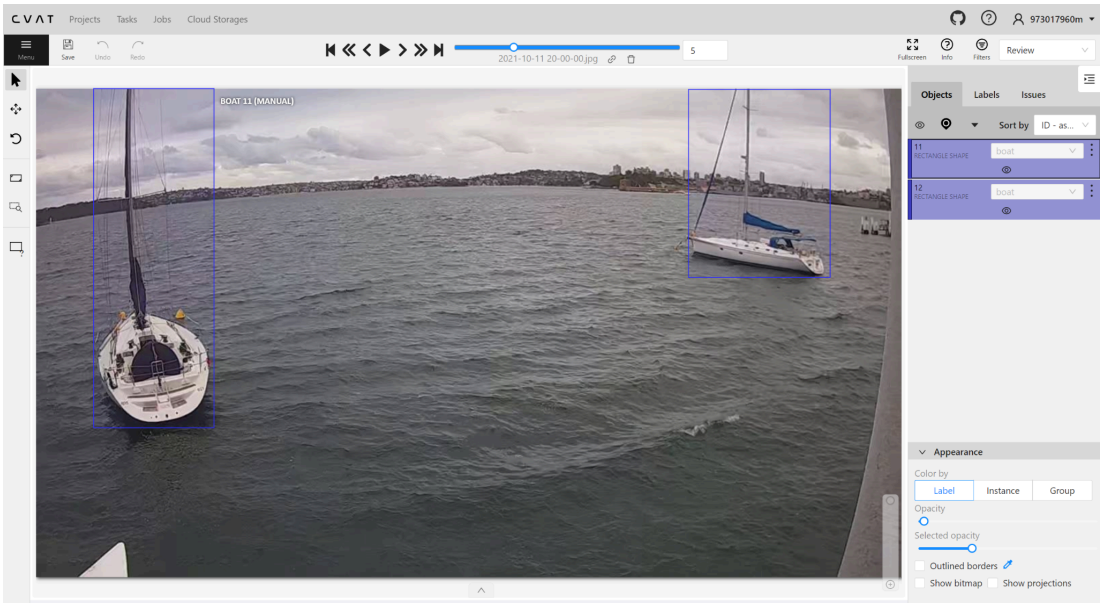


Figure 14 : Cvat Interface

Since the confidential condition of the client's dataset ,we have no access to the annotation of image .After searching and comparing the different annotation tools such as LabelImg , LabelMe and Cvat , we decided to utilize the Cvat tool because compared other tools Cvat support the multi-user collaboration task which means our group member have access to all the images and annotate together , which is suitable for the group task like our project .

4.4 Deployment

To deploy, train, and evaluate DETR, YOLO, Mask R-CNN, and Faster R-CNN models, we used the Ubuntu operating system provided by the client. We used a computer with an NVIDIA GPU to speed up the training process, and installed CUDA that matches the GPU. We installed Anaconda to manage the Python environment and dependent libraries, and created a new virtual environment. In this environment, we installed the deep learning framework PyTorch and related tool libraries. DETR needs to install the transformers library, YOLO needs to download the official code and install the dependencies in requirements.txt, and Mask R-CNN and Faster R-CNN use Detectron2. After completing the above environment deployment, we can load data, train models, and evaluate performance.

4.5 Evaluation

4.5.1 Intersection over Union

Intersection over Union (IoU) is an important indicator to measure the degree of overlap between the predicted box and the actual box of the target detection model. In our project, the target detection algorithm introduces IoU in the loss function to improve the detection accuracy of the model. IoU is the ratio of the intersection area and the union area between the predicted box and the actual box, which is used to evaluate their overlap. Its value range is 0 to 1. The higher the value, the higher the overlap between the predicted box and the actual box.

The calculation formula of IoU is:

$$IoU = \frac{Intersection\ Area}{Union\ Area}$$

Intersection Area: The area of the overlap between the predicted box and the actual box.

Union Area: The total area of the predicted box and the actual box, minus the overlap.

Through this evaluation method, the overlap between the predicted box and the actual box can be quantified, thereby evaluating the detection performance of the model.

4.5.2 Average Precision

Average Precision (AP) is one of the most commonly used evaluation indicators in target detection tasks, which is used to measure the accuracy performance of the model under different thresholds. The higher the AP value, the better the detection performance of the model. The precision calculation formula is as follows:

$$Precision = \frac{TP}{TP + FP}$$

TP is a true positive example and FP is a false positive example.

In order to evaluate the model performance more carefully, the model in our project uses AP values calculated under different IoU thresholds. They are ap(IoU = 0.5:0.95), ap(IoU =0.5) and ap(IoU =0.75). These evaluation indicators can fully reflect the performance of the target detection model.

4.5.3 Average Recall

The average recall (AR) rate is an important indicator for measuring the average recall rate of the model at different detection thresholds in the target detection task. It reflects the proportion of positive samples that the model can detect at various confidence thresholds. The higher the AR value, the stronger the detection ability of the model at different thresholds. The specific definition is:

$$Recall = \frac{TP}{TP + FN}$$

Among them, TP is a true positive example and FN is a false negative example.

AR is obtained by averaging the Recall values under different confidence thresholds. It provides a comprehensive indicator to evaluate the detection ability of the model under all thresholds.

5. RESOURCES

5.1 Hardware & Software

In order to ensure efficient project development and collaboration among members, we chose Jupyter Notebook and Visual Studio Code as the main Integrated Development Environment (IDE). The following is a detailed description of these two tools and the advantages of applying them in the project.

Jupyter Notebook is an interactive computing environment that allows users to create and share documents containing code, narrative text, and more. Its particularly well suited for tasks such as data cleansing, statistical modelling and machine learning.Jupyter Notebook supports a wide range of operating systems, including:

- Windows 10 and windows 11 (32-bit and 64-bit)
- Mac OS
- Various linux distributions

Jupyter Notebook allows Python code to be written and run in a visual environment. Its interactive features help test and verify code (e.g., easily add breakpoints) and provide a more intuitive and efficient medium for team communication.

Visual Studio Code is a free, open source code compiler developed by Microsoft that supports a wide range of programming languages. Visual Studio Code was chosen as the primary code debugging tool for the following reasons:

Cross-platform compatibility: Visual Studio Code is able to run on windows 10 and windows 11 (32-bit and 64-bit), Mac OS and linux systems.

Extension Support: With the extension marketplace, Visual Studio Code can integrate various programming languages in an easy way.

Debugging features: The built-in debugging features make it easy to debug code.

The reason for using Python as a programming language is that python has a clean syntax and is supported by powerful libraries that allow machine learning tasks to progress more efficiently. By using python, the project was able to be developed quickly.

In conclusion, the choice of Visual Studio Code and Jupyter Notebook as the main development tools can greatly improve the efficiency of the project development and the team collaboration rate, so that each team member can smoothly carry out their own code tasks.

5.2 Materials

Each team member wrote the coding portion of the project by using his or her own laptop. However, in order to ensure efficient execution of the project, the actual running of the programme was done on a PC provided by the client. Given that the goal of the project was image detection, the processing of image tasks required a high level of graphics card computing power. Therefore, the following configuration was chosen to run the project:

- Operating system: ubuntu (Linux)
- Memory: 32GiB
- CPU: AMD Ryzen 5 5600 6-core processor x 12
- Graphics card: NVIDIA GeForce RTX 4060Ti (16Gib)

5.3 Roles & Responsibilities

For the whole semester, the roles and responsibilities of each CS07-1 team member are shown in the table below:

Name	Roles	Responsibilities
Weibin Liu	1. Data processing and model training specialist. 2. Group document manager 3. Report writer	1.Perform data processing, including cleaning, preprocessing, etc. for model training.Responsible for model training, tuning and optimization. 2.Create weekly group shared documents to ensure team members are aware of task plans and delivery times.

		3. Weibin Liu was responsible for editing multiple parts of the final report and completed all parts of Masks RCNN and most of the Methodologies.
Yinglong Chen	1. Project Coordinator 2. Assist in building project models and evaluating them. 3. Project communicator	1. Yinglong was responsible for coordinating related matters in the project, such as task allocation, project progress record, project management, key milestones, team coordination, team communication, etc. 2. Yinglong was responsible for the establishment of the DETR model and the pre-evaluation on the COCO dataset, during which the model was investigated in literature and related documentation and code implementation. 3. Yinglong was responsible for reporting to the client at the client meeting as a representative spokesperson and talking with the client about the weekly project progress, and writing the minutes of each meeting and the project status checking report. In this project, Yinglong also acted as a communicator, sending emails to tutors and clients to communicate the progress and achievements of the project or team.

Simiao Yu	1. Labeling dataminer 2. Performance evaluators for Yolo V5 3. report writer.	1. Responsible for the labeling of images to provide labeled datasets for model training. 2. Perform Yolo V5 model training using labeled datasets and the COCO dataset and evaluate based on average precision and average recall metrics. 3. Write the final project report to summarize the Yolo V5 model structure, performance and discuss the experimental results. And write the literature review part of the final report.
Kaiyue Wang	1. Data annotator 2.Performance evaluators for YOLO-V8.	1. Kaiyue Wang is responsible for learning the architecture of the YOLO-V8 model ,training and evaluating the performance on the COCO and our self-labeled dataset then comparing it with the outer model of group members. Participate in labeling data to provide support for model training. 2. Kaiyue Wang took the role to find the suitable annotation tools for the project which is the Cvat tool and annotated 200 images which helped my team to train and evaluate their model on the annotated dataset. 3. Write the final report and take response for the YOLO-V8

		architecture ,performance evaluation and the future work discussion.
Bozhao Zhang	1. Performance Evaluator for Faster R-CNN 2. Experiment Manager 3. Report Writer	1. Develop and test the Faster R-CNN model. Evaluate its performance in our project. Optimise the model to achieve project expectation and requirements. 2. Develop an experimental plan and record experimental data. Analyze the advantages and disadvantages of each model according to the test results and improve the experimental plan. Finally unify the optimal training version of each model. 3. Responsible for routine team report writing, as well as information gathering and summarization during the course of the project.
Shikun Tong	1. Labeling dataminer 2. Performance evaluators for DETR 3. report writer.	1. Shikun Tong is responsible for the labeling of images to provide labeled datasets for model training. 2.Shikun Tong is responsible for learning the DETR model architecture and deploying the model. Regularly conduct performance tests and analyse the results to identify areas for improvement, compare the performance of the DETR model with other models, and produce comparative evaluations. And, after

		evaluation, discuss with the team to make training decisions. 3.Shikun Tong is responsible for editing multiple sections of the final report and visualising and analysing the final results. Ensure that the report is clear, accurate and meets the needs of the stakeholders.

6. Milestones / Schedule

Milestone	Tasks	Reporting	Date
Week-1	Communicate with clients to understand the basic situation of the project. Obtain and observe the data set.	Client meeting to review the project. Weekly tutorial presentation.	10-03-2024
Week-2	Look for and investigate relevant publications. Study and try different tools for labelling pictures. Project Status Checking 1 Due.	Client meeting to review the work plan Weekly tutorial presentation.	17-03-2024
Week-3	Proposal Report Due. Complete project proposal report. Final review of report layout, formatting, naming and other details.	Weekly tutorial presentation.	24-03-2024

Week-4	Select or design a suitable object detection model. The project data is labelled for future model training and evaluation. Labelling may include drawing bounding bounds for target items, adding category labels, and so forth.	Weekly tutorial presentation. Client meeting to review the project.	31-03-2024
Week-5	Prepare and preprocess data sets: 1. Data format conversion - Assess and convert audio data into the required format for further processing. - Evaluate and convert video data into the desired format for subsequent processing. - Inspect and convert radar image data into the desired format for further processing. 2. Data preprocessing - Audio data: Elimination of noise, reduction of noise through processing, adjustment of audio gain, etc. - Video data: Editing of videos, removal of irrelevant sections, adjustment of video quality, etc.	Weekly tutorial presentation. Client meeting to review the project.	14-04-2024
Week-6	Train the chosen or developed object detection model using the COCO dataset. Assess the model's effectiveness.	Weekly tutorial presentation. Client meeting to review the project.	21-04-2024

Week-7	Group progress report Due. Using the project's supplied data set, the chosen or created object detection model is trained. Evaluate the performance of the model.	Weekly tutorial presentation. Client meeting to review the project.	28-04-2024
Week-8	The relevant target detection indexes are examined in relation to the COCO dataset training results. Examples include IoU, mAP, and so on. Optimise training procedures (such as data enhancement, random cropping/scaling/rotation, and so on) to increase model generalisation.	Weekly tutorial presentation. Client meeting to review the project.	05-05-2024
Week-9	The model is trained using the project datasets so that it may pick up knowledge from the given data. It is evaluated using the relevant metrics. Tune hyperparameters (like batch size and learning rate) precisely to accelerate training convergence. And compared with other models.	Weekly tutorial presentation. Client meeting to review the project.	12-05-2024
Week-10	Model performance assessment List the model's advantages and disadvantages over other methods and offer suggestions for future enhancements.	Weekly tutorial presentation. Client meeting to review the project.	19-05-2024

Week-11	Project Final Status and Demo Checking Due. Start writing the final report	Weekly tutorial presentation. Client meeting to review the project.	26-05-2024
Week-12	Final Presentation Due Complete the final presentation slides. Record the final presentation and upload it to Canvas. Continue writing the final report.	Final Presentation.	31-05-2024
Week-13	Final Report Due. Finish writing the final report. Final review of report layout, formatting, naming and other details.	None	07-06-2024

7. RESULTS

The focus of this project was to evaluate the performance of various machine learning models in target detection tasks for different datasets and under different conditions:

1.MS COCO dataset: this dataset used for the open course helped us to evaluate the general suitability of various models in diverse environments with various object types and sizes.

2.Project 20 image dataset: Using 100 images obtained from the client, of which 80 were used for training and 20 for validation, the model focuses on detecting the ships that the client needs to detect.

3. Project 100 image dataset: consists of 100 images for training (50 day, 50 night), and another set of 100 images for validation (due to the blurriness of the images, small targets are ignored to focus on more impactful detections).

Models	AP(IoU = 0.5:0.95)	AP(IoU = 0.5)	AP(IoU = 0.75)	AR
Yolo v5	0.406	0.637	0.423	0.471
Yolo v8	**0.545**	**0.709**	**0.625**	**0.763**
Faster-RCNN	0.402	0.610	0.438	0.541
Mask-RCNN	0.424	0.628	0.461	0.569
DETR	0.420	0.624	0.442	0.574

Table 1: Model Performance on MS COCO Dataset

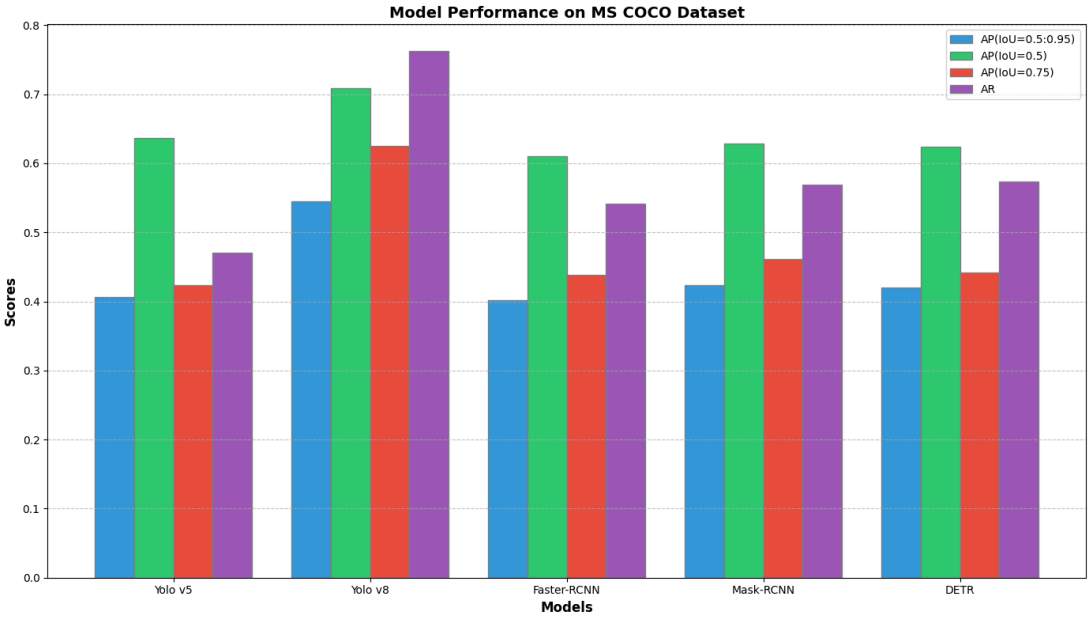


Figure 15: Model Performance on MS COCO Dataset

MS COCO dataset:

Based on the multi-dimensional comparison, Yolo v8 performs well under all evaluation metrics, especially on the more relaxed IoU = 0.5, which indicates that Yolo v8 can locate small objects well and roughly. Moreover, at higher IoU thresholds, Yolo v8 also shows better localisation of small objects compared to the other models.

The other models do not have much difference in their performance in all aspects, especially in AP(IoU=0.75) which is concentrated between 0.423 and 0.461. This indicates that at higher IoU thresholds, most models are able to maintain the same level of accuracy in small object detection, but Yolo v8 still performs better.

Models	AP(IoU = 0.5:0.95)	AP(IoU = 0.5)	AP(IoU = 0.75)	AR
Yolo v5	0.459	0.605	0.517	0.517
Yolo v8	**0.841**	0.889	**0.889**	0.778
Faster-RCNN	0.777	**0.969**	0.850	**0.821**
Mask-RCNN	0.778	0.962	0.864	0.812
DETR	0.677	0.896	0.797	0.733

Table 2: Model Performance on Project 20 image Dataset(Train:80)

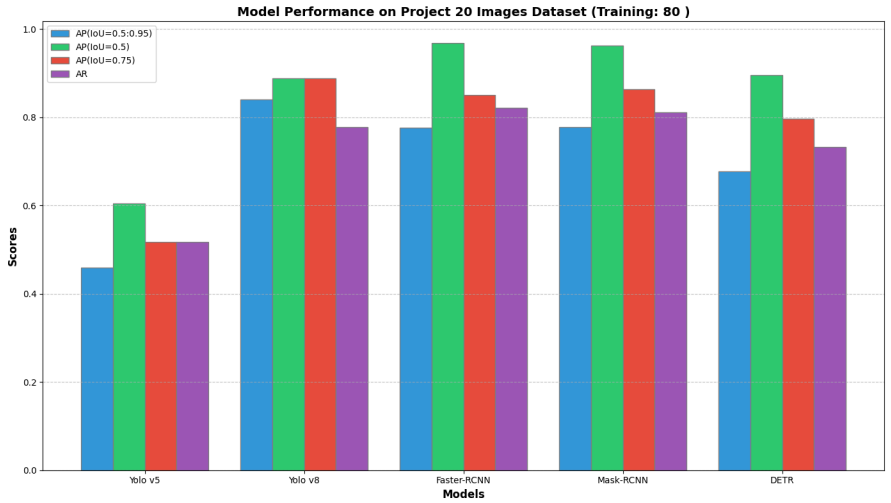


Figure 16: Model Performance on Project 20 image Dataset(Train:80)

Project 20 image dataset:

Most of the performance comparisons in this customer-focused dataset show significant performance improvements in most metrics. This dataset allows the project team to tailor the model to the customer's interests and objects of concern in order to achieve a solution to a specific problem. In this case, Yolo v8 is a good choice if good performance is needed with high recall and lower precision thresholds. Whereas, if more attention needs to be paid to precise detection, Mask-RCNN and Faster-RCNN are more appropriate choices. Finally, DETR on the other hand shows the need for more tuning and optimisation on small datasets.

Models	AP(IoU = 0.5:0.95)	AP(IoU = 0.5)	AP(IoU = 0.75)	AR
Yolo v5	0.494	0.662	0.602	0.602
Yolo v8	**0.824**	**0.971**	**0.931**	**0.938**
Faster-RCNN	0.795	0.930	0.895	0.836
Mask-RCN	0.800	0.960	0.929	0.847

N				
DETR	0.814	0.949	0.925	0.886

Table 3: Model Performance on Project 100 image Dataset

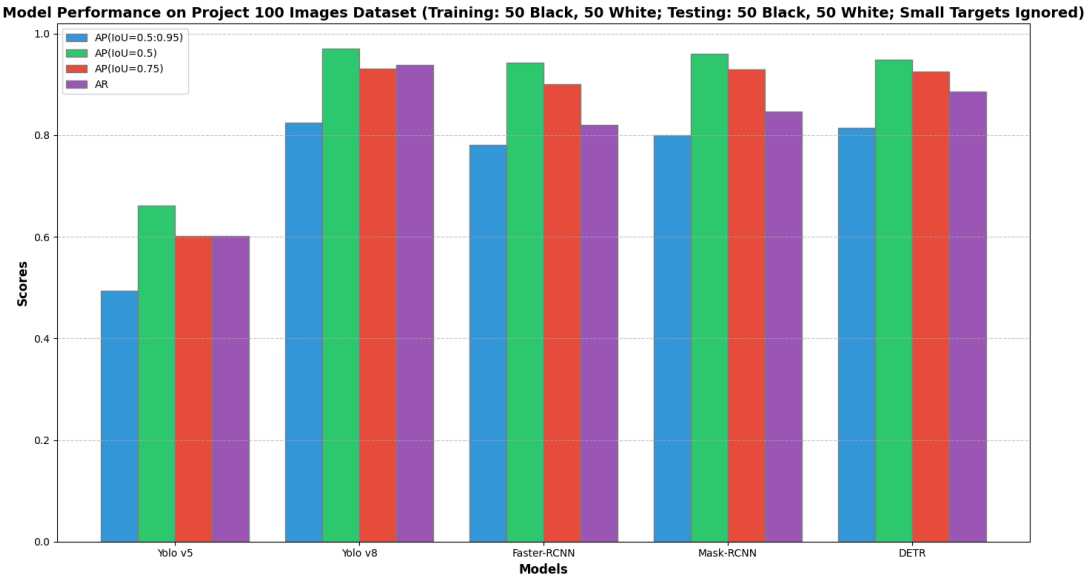


Figure 17: Model Performance on Project 100 image Dataset

Project 100 image dataset:

This figure reflects the contrast between the performance of the different models when dealing with datasets with significant contrast variation (daytime, black sky) and ignoring small targets. And, the removal of small targets is intended to focus the evaluation on more salient objects, which tend to be more critical in real-world applications. The performance of Yolo v8 is the most outstanding among them, which is a good choice if high precision and high recall are required.DETR, Faster-RCNN and Mask-RCNN also demonstrate their effectiveness, especially in ensuring high recall while maintaining target detection precision.

In small object detection, Yolo v8 is the best choice when accuracy is required. However, if the computational resources are limited or in a scenario where response time is required, perhaps Fast RCNN with the shortest inference time of 0.03s per image can be chosen.

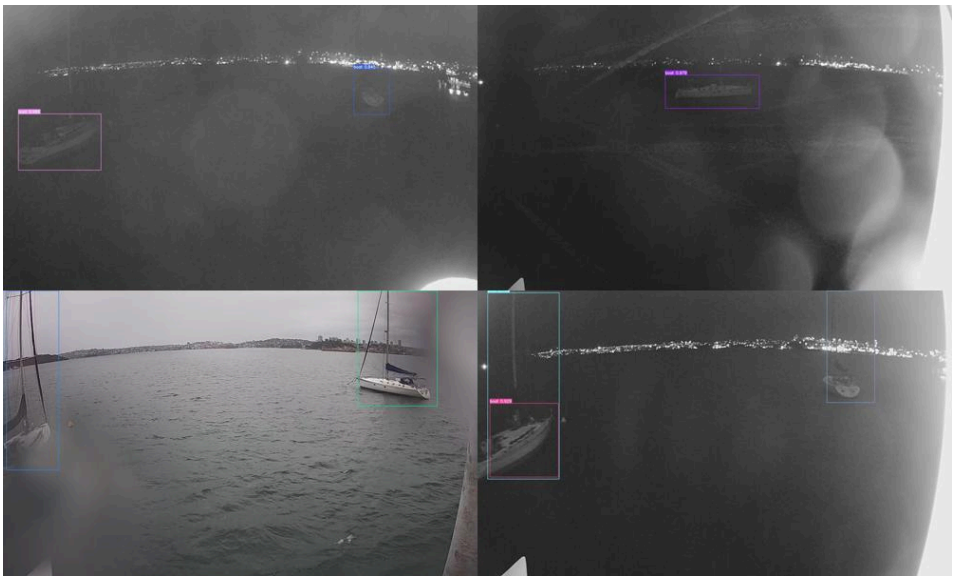


Figure 18: The detection of DETR

As shown in the images provided, the DETR model achieves the ability to detect objects in different marine environments.DETR can still accurately recognise boats in clear and with water droplets obscuring them, demonstrating its usefulness in situations where high accuracy is required. However, while it performs better in daytime, it is still partially unrecognisable or poorly recognised in dark conditions.

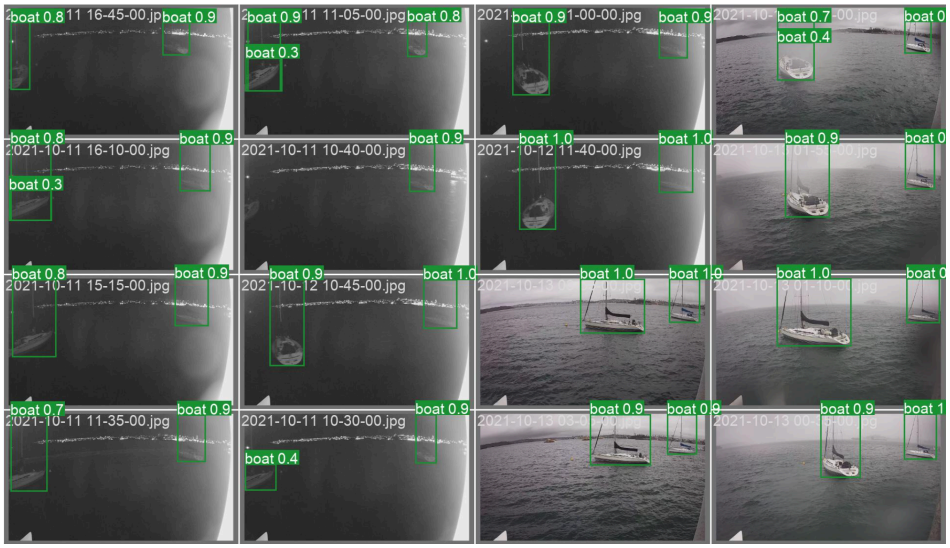


Figure 19: The detection of YOLO-V8

We have achieved the final detection of YOLO-V8 and detect on the sample images extracted from the client's dataset . From the Figure above , YOLO-V8 detects the object with the bounding box with the classification and the confidence score above each box , the confidence score is high up to 0.9 with the label boat

which means the object in the bounding box is more likely to be the boat . There are several detections at night which have a confidence score of 0.3 which means the model is not sure that the detected object is the boat because of the blurry effectiveness and less brightness condition on the image .

Figure 20: Detection on the blurry images of YOLO-V8

To test the ability of the model to detect on the blurry images , we apply the Gaussian Blur effect on several sample images and detect the object by YOLO-V8. Referring to the Figure above which is a sample detection from experiment, the confidence score is generally smaller than the detection on the normal object with the score around 0.6 which means the model is less likely to ensure the classification of the detected object is the boat.

Figure 21: The detection of Yolo V5

This image shows the output of an object detection task using the Yolo V5 model (specifically the ‘Yolo V5x’ variant). The model accurately identifies multiple objects in the maritime scene, draws bounding boxes around them, and labels each object with its object type (e.g., boat) and its detection confidence score. The results highlight the model's ability to detect objects of different sizes and at different distances, with scores indicating the model's confidence in each recognition, ranging from very high confidence (e.g., 0.93 for larger vessels) to medium confidence (e.g., 0.78 for birds). This performance illustrates the effectiveness of using advanced neural networks like Yolo V5 for real-time object detection in different environments.

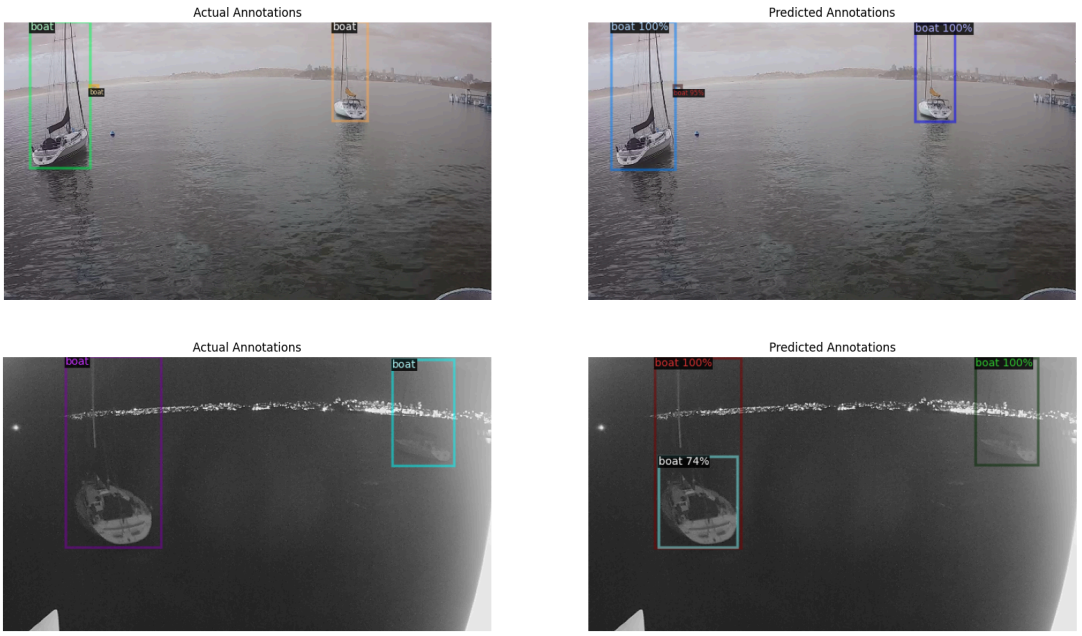


Figure 22: Comparison of Actual and Predicted Annotaions of Faster R-CNN

Above image shows the prediction results in an 80/20 experiment, i.e., using 80 images containing small objects as the training set and 20 images as the test set. The prediction results are randomly selected samples, which can be determined to be generalizable after several extractions. After 2000 rounds of training, the threshold is set to 70%, which means that only the prediction results with a confidence level higher than 70% are displayed. It can be seen that the confidence level of the large ship in the figure is 100% and the confidence level of the small ship is 95%. Secondly, the results are still good during the night scene and have the same

confidence level of 100%. At the same time, there is a bounding box with only the hull of the boat labeled and a relatively low confidence level, which means that there is a relatively small percentage of weights that include only the hull of the boat without the oars.

Model	Train Set	Test Set	AP	AP50	AP75	APs	APm	APl	Ars	Arm	Arl	AR
Faster-RCNN	Pre-trained	coco_val	40.22	61.018	43.815	24.157	43.53	51.978				
Faster-RCNN	Pre-trained	coco_val boat	27.209									
Faster-RCNN	Pre-trained	CS7-1	6.358	29.596	0.572	0.827	4.538	7.371	10	37.5	23.8	24
Faster-RCNN	coco_train boat	coco_val boat	23.59	46.881	21.652	15.754	29.431	43.219	23	40.4	58.3	34
Faster-RCNN	coco_train boat	CS7-1	35.272	51.661	40.65	4.37	15.361	51.58	11.6	24.1	66.9	47.5
Faster-RCNN	CS7-1	CS7-3	0.776	3.971	0.269	12.5	5.2	0.6	46	20	5.4	6.8
Faster-RCNN	80% CS7-1	20% CS7-1	77.67	96.85	84.973	19.1	71.749	83.473	50	80	86.1	82.1
Faster-RCNN	200 Labelled	100 Labelled	49.226	59.267	58.292	0	3.564	77.608	0	4.4	80.8	51.1
Faster-RCNN	100 Labelled	100 Labelled	78.064	94.332	90.113	nan	nan	78.071	nan	nan	82	82

Figure 23: Representative Experiments Result of Faster R-CNN

The figure above shows the results of each metric exhibited by the Faster R-CNN model over multiple experiments, with the metrics documenting in more detail the detection accuracy of objects of different sizes including large, medium and small. You can see that the experimental records in the penultimate and last rows are the optimal versions of the model selected for this project. These two versions have the highest mAP, and it can be seen that the 80/20 version has an accuracy of 50% on small objects, while the 100/100 version has no small objects and is therefore NA. The other versions are not perfectly adapted in the project due to the data complexity and the model weights etc.

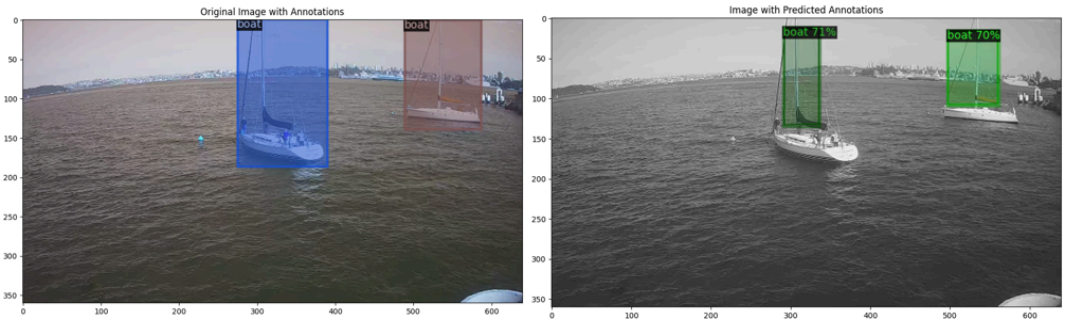


Figure 24: Comparison of Original and Predicted Annotations using Mask R-CNN

As shown in the figure below, this is the target detection result of the Mask R-CNN model. We extracted sample images from the client's dataset for detection. The left figure shows the original image with annotations, and the right figure shows the annotation results predicted by the model. As can be seen from the figure, the

Mask R-CNN model can detect targets in different marine environments. Even when there are water droplets in the image, the model can still accurately identify the ship, which shows that the model is very useful when high accuracy is required. In the image on the right of the figure, the model adds a bounding box to each detected target object and annotates the object type (such as ship) and confidence score above each box. The confidence score is as high as 0.71, indicating that the object in the bounding box is likely to be a ship. These results demonstrate the effectiveness of Mask R-CNN in real-time target detection, especially its ability to detect objects of different sizes and distances in different environments.

8. Discussion

	YOLO V5	YOLO V8	DETR	FASTER RCNN	MASK RCNN
COCO AP	0.406	**0.545**	0.420	0.402	0.424
Excl. small AP	0.494	**0.824**	0.814	0.781	0.800
Incl. small AP	0.459	**0.841**	0.677	0.777	0.778
Inference Time per Image	0.060s	0.051 s	0.077 s	**0.03 s**	0.039s

Table 4: Summary of Performance of Each Model

In our maritime surveillance project, we evaluated all models based on accuracy, computational efficiency, and performance on small and blurry objects.

From the results of the pre-trained models on COCO, YOLOv8 stood out with the highest accuracy of 0.545, proving its effectiveness in general object detection. After training with real application data, YOLOv8 maintained high

performance with an average precision (AP) of 0.824. In the challenging small-target experiment using manually labeled images, YOLOv8 excelled with an AP score of 0.841, demonstrating its robustness. Architecturally, YOLO models improve speed by predicting all boxes and classes in a single evaluation. YOLOv8 showed both high accuracy and efficiency, with an inference time of 0.051 seconds. These characteristics make YOLOv8 a strong candidate for versatile detection tasks.

While YOLOv5 is also known for its high-speed detection capabilities, it did not perform as well with smaller, less obvious targets in our tests. Although it is exceptionally fast due to its design that predicts bounding boxes and class probabilities directly from full images in a single evaluation, YOLOv5 may lack some precision in complex detection tasks, especially with small and blurry objects.

DETR (Detection Transformer) introduces a transformer-based approach to object detection, which excels in precision and handling complex scenes due to its attention mechanisms. However, it is computationally intensive, resulting in slower inference times. This was evident in our results, where DETR performed well in accuracy but lagged in speed, making it less suitable for real-time applications in our energy-constrained environment.

Faster R-CNN, with its Region Proposal Network (RPN) and Feature Pyramid Network (FPN), excelled in balancing accuracy and efficiency. The RPN generated candidate object regions quickly, while the FPN enhanced multi-scale feature extraction, crucial for detecting objects of varying sizes. Our experiments demonstrated that Faster R-CNN had the fastest inference time of 0.03 seconds per image, highlighting its suitability for real-time monitoring on our solar-powered vessels. Its performance in detecting small and blurred objects was also strong, reflecting its robust architecture.

Mask R-CNN, built on the Faster R-CNN architecture, includes a mask prediction branch, allowing it to detect and segment objects simultaneously. This capability made it highly effective for detailed and small object detection. Our results showed that Mask R-CNN performed well with manually annotated images featuring small targets. Its slightly slower inference time compared to YOLOv8 indicated that the added complexity of mask prediction impacted speed, but the trade-off was worthwhile for detailed detection tasks.

In summary, YOLOv8 emerged as a top performer in both accuracy and speed, making it a strong candidate for versatile detection tasks. DETR's precision was beneficial for complex scenes but was limited by slower processing. Faster R-CNN provided the best balance of speed and accuracy, making it ideal for our specific application where efficiency and real-time processing are critical. Mask R-CNN offered detailed detection capabilities at a moderate speed. Each model's strengths and weaknesses were evident in our results, guiding us in selecting the most appropriate model for our maritime monitoring needs.

9. Limitations and Future Works

9.1 Limitations

1. Limitations in data quality and quantity: the project based on a limited dataset for model training and validation because our client told us that we can use 100 images to evaluate the model performance. Therefore, using only a small fraction of labelled images to train and test the model may not be sufficient to cover all real-world scenarios. This may affect the generalisation ability of the model and its performance on unseen data.

2. Issues of model applicability and scalability: the currently employed models are mainly optimised for small target detection. While the models have shown to be efficient and highly accurate on specific tasks, they may require further adaptation and optimisation for different or broader application scenarios.

3. Limitations of real-time processing capability: Although the models used in the project such as YOLOv8 and Faster R-CNN perform well in terms of inference time, continuous real-time video stream processing may encounter resource constraints when actually deployed, especially on unmanned vessel platforms with limited resources.

4. Impact of environmental factors: the complexity of the maritime environment, such as weather variations, light conditions and waves, may adversely affect the effectiveness of the target detection algorithms. Models need to be stable

across a wide range of environmental conditions, which is a challenge in practical applications.

5. Optimisation of algorithms: Despite the use of advanced target detection algorithms, there is still room for optimisation, such as improving the algorithms to reduce false and missed detections, or developing more efficient algorithms to handle large-scale data.

9.2 Future Works

In the future, we are going to improve and optimize the accuracy and inference speed of our models for the application in object detection in boats for surveillance purposes. Considering the limitation of the resource on the boat only powered by solar power, we are going to try multiple energy-efficient optimization techniques such as model compression and quantization to ensure that the object detection model can be light-weight and run efficiently for a long time in the ocean. In our project ,the input data is only from the images taken by the camera , to make the model have better ability or accuracy to predict the bounding box of the object , the Multimodal model can be applied by adding the audio input to the model and combine the audio and video feature as the input for the model to go through such as the AVENet which works by making consistency of audio and visual data through temporal synchronization and feature alignment .Thus in the future we may contact the client to discuss if there is the audio capture device on the boat to collect the sound such as the boat’s horn or the sound made by the plane or other object.

Since the confidential condition of the client dataset means we have no access to the annotation of images, we trained each model based on the image we annotated which mainly contain images during day and at night half and half. Considering the complexity of the environment in the ocean such as storms, rain and motion blur which may influence the performance of model , in the future we are going to expand the self-labeled dataset with more diversity and more labeled samples such as blurry Images ,images in various lighting or weather conditions ,image with different resolutions and images with partial occlusion for further training the YOLO-V8 to improve the model's generalization ability at different condition in the sea.

References

Abdulla, W. (2020, March 29). matterport/Mask_RCNN. Retrieved May 9, 2024, from GitHub website: https://github.com/matterport/Mask_RCNN

Alexander, K., Ross B, G., Kaiming, H., & Piotr, D. (2020, May 26). facebookresearch/detectron2. Retrieved May 15, 2024, from GitHub website: https://github.com/facebookresearch/detectron2

Bochkovskiy, A., Wang, C.-Y., & Liao, H.-Y. M. (2020). YOLOv4: Optimal Speed and Accuracy of Object Detection. *ArXiv*. Retrieved from https://arxiv.org/abs/2004.10934

Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., & Zagoruyko, S. (2020). End-to-End Object Detection with Transformers. *Computer Vision – ECCV 2020*, *12346*, 213–229. https://doi.org/10.1007/978-3-030-58452-8_13

Cheng, B., Yunchao, W., 1⋆, H., Shi, Feris, R., Xiong, J., & Huang, T. (n.d.). *Revisiting RCNN: On Awakening the Classification Power of Faster RCNN*.

Cheng, L., Deng, B., Yang, Y., Lyu, J., Zhao, J., Zhou, K., … He, Y. (2022). Water Target Recognition Method and Application for Unmanned Surface Vessels. *IEEE Access*, *10*, 421–434. https://doi.org/10.1109/access.2021.3138983

He, K., Gkioxari, G., Dollar, P., & Girshick, R. (2017). Mask R-CNN. *2017 IEEE International Conference on Computer Vision (ICCV)*, *pp. 2961-2969*. https://doi.org/10.1109/iccv.2017.322

Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., … Adam, H. (2017). MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. Retrieved May 5, 2024, from arXiv.org website: https://arxiv.org/abs/1704.04861

Hu, B., & Wang, J. (2020). Detection of PCB Surface Defects with Improved Faster-RCNN and Feature Pyramid Network. *IEEE Access*, 1–1. https://doi.org/10.1109/access.2020.3001349

Hu, C., Shi, Z., Wei, H., Hu, X., Xie, Y., & Li, P. (2022). Automatic detection of pecan fruits based on Faster RCNN with FPN in orchard. *International Journal of Agricultural and Biological Engineering*, *15*(6), 189–196. https://doi.org/10.25165/j.ijabe.20221506.7241

Jocher, G., Chaurasia, A., & Qiu, J. (2023, January 1). YOLOv8 by Ultralytics. Retrieved from GitHub website: https://github.com/ultralytics/ultralytics

King, K. (2023, January 10). Brief summary of YOLOv8 model structure · Issue #189 · ultralytics/ultralytics. Retrieved June 1, 2024, from GitHub website: https://github.com/ultralytics/ultralytics/issues/189

Lei, F., Tang, F., & Li, S. (2022). Underwater Target Detection Algorithm Based on Improved YOLOv5. *Journal of Marine Science and Engineering*, *10*(3), 310. https://doi.org/10.3390/jmse10030310

Mehra, A. (2023, April 17). Understanding YOLOv8 Architecture, Applications & Features. Retrieved May 16, 2024, from Labellerr website: https://www.labellerr.com/blog/understanding-yolov8-architecture-applications-features/

Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You Only Look Once: Unified, Real-Time Object Detection. *Cv-Foundation.org*, 779–788. Retrieved from https://www.cv-foundation.org/openaccess/content_cvpr_2016/html/Redmon_You_Only_Look_CVPR_2016_paper.html

Ren, S., He, K., Girshick, R., & Sun, J. (2017). Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *39*(6), 1137–1149. https://doi.org/10.1109/tpami.2016.2577031

Ren, Y., Zhu, C., & Xiao, S. (2018). Object Detection Based on Fast/Faster RCNN Employing Fully Convolutional Architectures. *Mathematical Problems in Engineering*, *2018*, 1–7. https://doi.org/10.1155/2018/3598316

Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L.-C. (2018). MobileNetV2: Inverted Residuals and Linear Bottlenecks. Retrieved May 14, 2024, from openaccess.thecvf.com website: https://openaccess.thecvf.com/content_cvpr_2018/html/Sandler_MobileNetV2_Inverted_Residuals_CVPR_2018_paper.html

Shin, H.-C., Lee, K.-I., & Lee, C.-E. (2020). Data Augmentation Method of Object Detection for Deep Learning in Maritime Image. *2020 IEEE International Conference on Big Data and Smart Computing (BigComp)*, *pp. 463-466*. https://doi.org/10.1109/bigcomp48618.2020.00-25

Tharsan Senthivel, Vu, N.-S., & Borzic, B. (2023). Detection Transformer with Diversified Object Queries. *2023 IEEE International Conference on Image Processing (ICIP)*, *2515–2519*. https://doi.org/10.1109/icip49359.2023.10221970

Wu, Y., Kirillov, A., Massa, F., Lo, W.-Y., & Girshick, R. (2019, October 10). Detectron2: A PyTorch-based modular object detection library. Retrieved May 14, 2024, from ai.meta.com website: https://ai.meta.com/blog/-detectron2-a-pytorch-based-modular-object-detection-library-/

Xie, X., Cheng, G., Wang, J., Yao, X., & Han, J. (n.d.). *Oriented R-CNN for Object Detection*.

Yu, J., Zheng, H., Xie, L., Zhang, L., Yu, M., & Han, J. (2023). Enhanced YOLOv7 integrated with small target enhancement for rapid detection of objects on water surfaces. *Frontiers in Neurorobotics*, *17*. https://doi.org/10.3389/fnbot.2023.1315251

Zhou, J., Jiang, P., Zou, A., Chen, X., & Hu, W. (2021). Ship Target Detection Algorithm Based on Improved YOLOv5. *Journal of Marine Science and Engineering*, *9*(8), 908. https://doi.org/10.3390/jmse9080908

Zhou, Z., Hu, X., Li, Z., Jing, Z., & Qu, C. (2022). A Fusion Algorithm of Object Detection and Tracking for Unmanned Surface Vehicles. *Frontiers in Neurorobotics*, *16*. https://doi.org/10.3389/fnbot.2022.808147