Politecnico di Torino

POLITECNICO DI TORINO

College of Computer Engineering, Cinema and Mechatronics

Master's Degree Thesis

Boat Detection and Classification in Port Areas Using Deep Learning

Supervisors
prof. Renato Ferrero
dr. Antonio Costantino Marceddu

Candidate
Sahand Seyed Mohammadghavami

March 2026

Abstract

This Thesis aims to solve the maritime traffic monitoring problem through vessel detection and classification. Maritime surveillance has a significant role for safety, traffic regulation, and port management. The main goal of this work is to design a vision-based system that where ships can be detected in complex maritime scenes and their type can be classified based on visual information.

The proposed system is based on state-of-the-art object detection models from the YOLO (You Only Look Once) family, selected for their balance between accuracy and real-time performance. Different model variants are trained and evaluated in order to analyze their behavior on maritime scenarios. The SAHI (Slicing Aided Hyper Inference) framework is integrated into the pipeline to verify if it brings an improvement in detection performance on high-resolution images and small objects, which are common challenges in port surveillance environments. A publicly available maritime dataset extracted from video footage is used for training and evaluation, ensuring realistic scene conditions.

Because consecutive frames in video-based datasets often contain very similar visual information, there is a high risk of overfitting. If not properly handled, the model may memorize spatial patterns instead of learning generalizable features. To address this issue, special attention is given to dataset preparation, frame selection, and validation strategy to ensure reliable model generalization. In addition, the SPSCD (Split Port Ship Classification Dataset) is refined by improving the accuracy of the bounding box annotations, correcting imprecise bounding boxes, and removing unnecessary background areas. This refinement increases annotation quality and reduces noise in the training data, which leads to more stable detection performance.

Overall, the study demonstrates that deep learning-based object detection methods can be effectively applied to automated maritime surveillance tasks. At the same time, the results highlight the importance of high-quality annotations, proper dataset structure, temporal considerations, and scene-specific modeling in order to achieve robust and reliable performance in real-world port scenarios.

Keywords: Boat, Deep learning, Object detection, Port, SAHI, Vessel, Vision-based, YOLO

Acknowledgements

I would like to express my deepest gratitude to Professor Renato Ferrero for his expert guidance and Dr. Antonio Marceddu for his insightful co-tutorship. Thank you for all these important and interesting questions that shaped my thesis. I could not have asked for better mentors.

I also owe a deep sense of gratitude to my family and friends for constant encouragement and motivation, whose presence has played a significant role in keeping us motivated during and even post our academic life.

I would, in conclusion, like to express my deepest gratitude to Politecnico di Torino for providing such a rich and enriching experience and for imparting to us a wealth of knowledge during our studies.

Contents

List of Figures

List of Tables

Acronyms

A

AI Artificial Intelligence 1, 14
AIS Automatic Identification System 1, 13, 22, 45
AP Average Precision 16, 17, 34
ASVs Autonomous Surface Vehicles 6

C

CNN Convolutional Neural Network 9, 13, 15, 19, 20, 22
COCO Common Objects in COntext 19
CSPNet Cross Stage Partial Network 18
CV Computer Vision 1, 5, 7, 9, 13, 20, 36

D

DETR DEtection TRansformer 15, 24
DL Deep Learning 1, 7–9, 13, 14
DNN Deep Neural Network 13

E

EO Electro-Optical 1, 2, 13

F

FN False Negative 16
FP False Positive 16
FPN Feature Pyramid Network 18, 21, 24
FPS Frame Per Second 15

H

HOG Histogram of oriented gradients 13
HR High Resolution 2, 20, 21

I

IoU Intersection over Union 16–18, 34

M

mAP mean Average Precision 16–18, 21, 34–38, 41–45

MDA Maritime Domain Awareness 12
ML Machine Learning 5, 7, 13

N

NLP Natural Language Processing 15
NMS Non-Maximum Suppression 15, 21
NMW Non-Maximum Weighted 21

P

PAN Path Aggregation Network 18, 24
PCA Principal Component Analysis 6
PR Precision-Recall 17

R

R-CNN Region Based Convolutional Neural Networks 11, 12, 14
ResNet Deep Residual Learning 10
RL Reinforcement Learning 6
RoI Region of Interest 15
RPN Region Proposal Network 15

S

SAHI Slicing Aided Hyper Inference 2, 21, 22, 24, 26, 31, 39–45
SIFT Scale-Invariant Feature Transform 13
SOD Small Object Detection 19, 20
SPSCD Split Port Ship Classification Dataset 2, 26–28, 33, 44
SSD Single Shot MultiBox Detector 15

T

TP True Positive 16

Y

YOLO You Only Look Once 2, 15, 17, 18, 28, 29, 34–36, 38, 39, 41–44

Chapter 1

Introduction

1.1 Motivation and Problem Statement

Maritime transportation serves as the backbone of global trade, handling over 80% of international goods movement. Ports, as critical hubs in this network, demand efficient monitoring to maintain safe navigation, optimize operational efficiency, and enhance security. As vessel traffic continues to increase, automated surveillance systems have become indispensable components of modern port management.

Traditional monitoring relies heavily on radar and the Automatic Identification System (AIS). These technologies provide essential data on vessel positions and identities, however they face significant limitations in dense port environments. Many small vessels lack AIS transponders, while radar signals often suffer from clutter, reflections from structures and water, and insufficient resolution for precise tracking. These shortcomings have driven interest in vision-based approaches that utilize Electro-Optical (EO) sensors as complementary solutions.

Recent progress in Artificial Intelligence (AI), especially in Computer Vision (CV) and Deep Learning (DL), have enabled the development of powerful visual surveillance systems. However, maritime scenes present unique challenges: dynamic wave patterns, varying illumination, water reflections, and extreme scale differences where vessels may appear as small objects in vast high-resolution images. Accurate detection under these conditions remains a significant challenge.

An additional complication emerges when training models on datasets derived from video footage. Consecutive frames are often highly similar, differing primarily in vessel positions, which introduces strong temporal redundancy. Without appropriate handling, this can result in severe overfitting, producing overly optimistic performance metrics on test splits that fail to generalize effectively to new sequences.

These challenges underscore the need for a robust vision-based system that can reliably detect and classify vessels in complex maritime environments.

1.2 Objectives and Contributions

The primary goal of this thesis is to develop and evaluate a vision-based pipeline for maritime traffic monitoring in port areas, leveraging state-of-the-art deep learning techniques for vessel detection and classification.

The specific objectives are:

- To develop a robust object detection framework capable of handling complex maritime scenes.
- To improve detection accuracy for small vessels in high-resolution imagery.
- To mitigate overfitting caused by temporal redundancy in video-derived datasets.

The main contributions of this thesis are:

- A complete vision-based maritime monitoring pipeline based on You Only Look Once (YOLO) object detection models.
- Integration of the Slicing Aided Hyper Inference (SAHI) [1] framework to enhance small-object detection in High Resolution (HR) images.
- Refinement of the Split Port Ship Classification Dataset (SPSCD) [2], improving annotation quality and overall dataset structure.
- A practical strategy to reduce overfitting due to high temporal similarity in video-extracted datasets.
- A quantitative ablation of input resolution (640 vs 1280) and annotation quality (old vs refined labels), highlighting their impact on localization accuracy.

1.3 Scope and Assumptions

To position the work clearly, this thesis adopts a set of scope boundaries and operational assumptions:

Application scope. The main task addressed is vision-based vessel detection and class-level recognition within port surveillance imagery. The work focuses on frame-level object detection quality and practical deployment-oriented model selection.

Data scope. The experimental analysis uses the SPSCD and its refined annotations. Consequently, the findings are most applicable to port environments with comparable camera geometry, traffic density, and environmental conditions.

System assumptions. The proposed pipeline is based on the assumption of static surveillance viewpoints and the availability of EO imagery with sufficient resolution. Given these conditions, the study prioritizes robust localization of distant vessels and aims to reduce overfitting because of temporal correlation.

1.4 Research Questions

The study is organized around the following research questions:

- **RQ1:** Which detector variant offers the optimal trade-off between localization accuracy and practical efficiency in port surveillance conditions?
- **RQ2:** How strongly does input resolution affect detection quality, especially under strict localization metrics?
- **RQ3:** What is the measurable impact of annotation refinement on model performance?
- **RQ4:** To what extent does sequence-aware validation enhance the reliability of reported results in video-derived maritime datasets?
- **RQ5:** Can sliced high-resolution inference improve practical detection of distant small vessels compared to standard inference?

These research questions guide both the methodological decisions presented in Chapter 4 and the experimental analyses discussed in Chapter 5.

1.5 Contribution Significance

Beyond listing contributions, it is important to clarify their practical significance for maritime monitoring applications.

Integrated pipeline perspective. Many studies treat architecture selection, data quality, and small-object handling as separate topics. This thesis combines these dimensions into one coherent pipeline and evaluates them under a shared protocol, which enables more interpretable conclusions for real-world deployments.

Annotation quality as a first-class variable. This work compares old and refined labels to show that how datasets are curated is more than just a preprocessing step. It is a key factor in localization quality, especially in dense and cluttered scenes.

Validation rigor for video-derived datasets. The sequence-aware fold strategy solves a common problem in surveillance evaluation: temporal leakage between training and validation data. This helps ensure that reported scores reflect true generalization rather than memorization.

Operational relevance. The resulting detector can help with tasks such as traffic awareness, event logging, and boundary movement tracking in port areas. It also keeps its computational needs low enough for real-world use.

1.6 Evaluation Protocol Overview

The evaluation framework aims to provide fair, robust, and reproducible results:

Cross-validation protocol. We use a sequence-aware 5-fold scheme to prevent data leakage from nearby frames in time. Each fold serves once as validation, with the others used for training.

Model comparison. We train several one-stage detector variants using the same augmentation and optimization settings. This way, any differences we see are mainly due to the model architecture and capacity, not changes in the training process.

Ablation studies. We include two controlled ablation studies, one on input resolution (640 vs 1280) and another on annotation quality (old versus refined labels). These tests help us see how spatial detail and label quality affect performance.

Primary metrics. We report results using mAP@50 and mAP@50–95, which together measure both detection success and the precision of object localization.

1.7 Thesis Organization

The thesis is structured as follows:

Chapter 2 presents background information on maritime surveillance, computer vision fundamentals, and deep learning-based object detection.

Chapter 3 reviews related work in maritime vessel detection and classification.

Chapter 4 describes the outlined methodology, including dataset preparation and refinement, model architecture, and training strategies.

Chapter 5 presents the experiments carried out, the results obtained, and the quantitative analysis.

Chapter 6 summarizes the conclusions and describes future research directions.

Chapter 2

Background

2.1 The Evolution of Machine Learning Paradigms

Machine Learning (ML) marks a major change in how computers solve problems, shifting from rule-based 'expert systems' to approaches that learn from data [3]. In expert systems, programmers write out the decision rules. ML systems, on the other hand, learn these rules by analyzing examples and optimizing a goal based on the data. This change is especially important in computer vision, where it is not practical to list every possible way an object might appear.

This change is especially important in maritime monitoring, where conditions such as sea state, lighting, background clutter, and vessel angles are constantly changing. Hard-coded rules do not work well in these situations. ML enables models to adapt to these changes by learning from data and improving over time. The field is usually divided into three main types: supervised learning, unsupervised learning, and reinforcement learning. Each type is defined by the kind of feedback given during training [4]–[6].

2.1.1 Supervised Learning: The Foundation of Object Detection

Supervised learning is the most common approach in CV and is the basis for the vessel detection methods discussed in this thesis. In this approach, the algorithm receives a labeled dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$, where each input image x_i is paired with a ground-truth annotation y_i [4]. For object detection, these labels include class categories such as "vessel" or "buoy" and spatial coordinates in the form of bounding boxes.

The learning process minimizes a loss function that measures the difference between the model's predictions and the actual labels. The model updates its internal parameters using optimization to better match pixel patterns to the correct labels. In most modern detectors, the loss function includes several parts, covering both classification and localization errors.

The main strength of supervised learning is its predictability; when labels are reliable and diverse enough, performance can be rigorously measured and compared.

Its main limitation is the data bottleneck. Large-scale annotation is expensive, and label quality directly affects model behavior. In maritime surveillance, this problem is even more common. Small or distant vessels, partly hidden objects, and crowded port areas make it harder to accurately label data. As a result, noisy labels can degrade localization performance, even when the model still predicts the correct class.

2.1.2 Unsupervised Learning and Data Structure

Unlike supervised methods, unsupervised learning works with unlabeled data to find hidden structures, clusters, or patterns without direct guidance. According to [5], its main goal is to model the underlying probability density of the input space.

Although unsupervised learning is not the main training method in this thesis, it is a useful tool for working with maritime datasets. In practice, there is usually more unlabeled data than labeled data, and unsupervised methods can help find hidden patterns before using supervised training.

Unsupervised learning is often used in maritime surveillance for several purposes:

- **Anomaly Detection:** Identifying "unusual" vessel behavior or unexpected objects in maritime environments by modeling what normal sea conditions look like.
- **Dimensionality Reduction:** Applying methods such as Principal Component Analysis (PCA) autoencoders to compress high-resolution image data into lower-dimensional representations while preserving essential features.
- **Clustering:** Grouping similar visual signatures (such as distinct wave patterns or lighting conditions) to assist in dataset balancing.

These methods help improve supervised learning by making it easier to audit data, split datasets, and find difficult examples.

2.1.3 Reinforcement Learning and Autonomous Navigation

Reinforcement Learning (RL) differs from the previous paradigms by focusing on sequential decision-making. An "agent" interacts with an environment and learns a policy, a mapping from states to actions that maximizes a cumulative reward signal [6]. While supervised learning might tell a system "this is a boat," RL enables a system to decide "how should I move the camera to keep this boat in view?"

While not the primary focus of static object detection, RL is increasingly relevant in the maritime sector for Autonomous Surface Vehicles (ASVs). It enables action-level intelligence such as route planning, collision avoidance, and adaptive sensing. In this sense, supervised detection and RL can be viewed as complementary layers: perception identifies objects, while control policies decide how to react over time.

2.1.4 The Shift to Deep Learning

The move from classical ML pipelines to DL marks one of the most significant advances in CV. Classical systems rely heavily on hand-crafted descriptors chosen by experts. DL replaces this manual stage with representation learning, where multi-layer neural networks discover hierarchical features directly from data [7].

This change is especially important for maritime scenes, where the most useful features can vary quickly with distance, weather, or viewpoint. DL models can adjust to these changes better than fixed descriptors, which is why they perform well in current detection tests and real-world surveillance.

2.2 Fundamentals of Computer Vision

CV is an interdisciplinary field that seeks to develop the theoretical and algorithmic basis for machines to extract, analyze, and understand useful information from digital images or video sequences [8]. At its core, the discipline aims to replicate the sophisticated capabilities of human visual perception, transforming raw pixel data—represented as grids of intensity values—into high-level semantic descriptions of the world [9].

CV uses mathematical models to understand how images are formed and to develop robust methods for analyzing visual information. One of its main goals is to solve common problems like noise, objects blocking each other, and changes in lighting.

These fundamental tasks typically include:

- **Image Formation and Preprocessing:** To understand the geometry and physics of the interaction between light and surfaces, and how it is captured by sensors.
- **Feature Extraction:** Focuses on detecting local structures such as edges, corners, and textures, which constitute the basic elements for advanced visual analysis.
- **High-level Interpretation:** ML techniques are applied for object detection, segmentation, and visual scene analysis, enabling the system to generate appropriate responses to visual inputs [8].

Early CV methods mostly used hand-crafted features and strict geometric models. Today, data-driven techniques are more common. Still, basic principles are important because they help models perform well in real-world applications such as port surveillance and self-driving vehicles.

CV is frequently conceptualized as a layered pipeline. When low-level processing stabilizes visual input and minimizes noise, the mid-level processing extracts structures and spatial relationships. Finally, High-level models understand what the image shows, such as recognizing objects and their categories. The quality

of each layer directly affects subsequent stages; therefore, effective detection depends on network architecture, high-quality data, consistent preprocessing, and strict evaluation.

2.2.1 The Digital Image Representation

At its fundamental level, a computer perceives an image as a numerical matrix. For grayscale imagery, each pixel corresponds to one intensity value (typically 0 to 255). For color imagery, each pixel is represented by three channels in the **RGB** space. This representation is simple and general, but it is also the origin of the small-object problem.

When a distant vessel appears as only a few pixels, there is very little visual information to support classification and localization. Repeated downsampling in deep neural networks makes this problem worse. To keep important details about objects in maritime surveillance, it is important to use high-resolution inputs, combine features from different scales, and apply slicing-based inference.

2.2.2 Low-Level Vision: Image Processing

Before a system can perform high-level reasoning, it first needs to process raw visual data with low-level operators, which are usually linked to **hand-crafted features**. These methods do not classify objects directly. Instead, they improve local structures and reduce issues like noise, blur, and changes in lighting.

In maritime imagery, low-level processing is important because water surfaces introduce repetitive textures and specular reflections that may obscure vessel boundaries. Even when deep models are used downstream, understanding classical preprocessing remains useful for debugging difficult cases and interpreting failure modes.

Traditional image processing techniques include:

- **Filtering:** This method applies mathematical kernels to enhance image sharpness or reduce noise.
- **Edge Detection:** This technique identifies rapid changes in pixel intensity to delineate boundaries. The Canny Edge Detector is a widely used example for structural analysis.
- **Thresholding:** This process segments images by converting grayscale data to binary form, thereby facilitating object extraction from the background.

These methods are computationally efficient and interpretable, but they are sensitive to context variation. A threshold that works in calm daylight can fail in haze or strong glint, and edge responses may become unstable in choppy water. This limitation motivated the transition toward learned representations in modern DL-based pipelines.

2.2.3 High-Level Vision and Deep Learning

Modern CV relies heavily on DL, specifically **Convolutional Neural Network (CNN)**. These architectures, pioneered by LeCun et al. [10], learn a hierarchy of representations from data rather than relying on manually designed descriptors. The early layers capture simple features such as edges, corners, and textures. Middle layers recognize patterns and parts, while the deepest layers capture high-level concepts that match object categories [7].

AlexNet's strong performance in 2012 [11] was a major milestone. It showed that features learned directly from data can outperform hand-designed pipelines on large-scale tests. For maritime surveillance, this is especially relevant because target appearance changes continuously with distance, wake patterns, and climatic conditions. Learned representations adapt to this variation more effectively than fixed feature engineering.

In a computer vision system, high-level modules convert feature maps into useful outputs such as class labels, bounding boxes, masks, or trajectories. The quality of the model depends on more than just its architecture. It also relies on the quality of the training data, the precision of the labels, and the design of the evaluation.

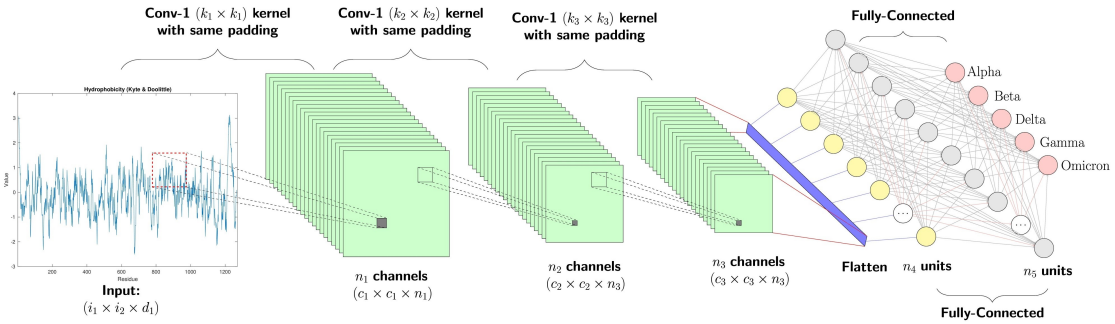

Figure 2.1. Architecture of a CNN showing the hierarchical extraction of features from low-level edges to high-level object shapes.

2.2.4 Key Computer Vision Tasks

Computer vision systems handle perception tasks based on the level of detail required in the output. In maritime monitoring, how you define the task directly affects how the system works. For example, classification can tell you the type of scene, but only detection and segmentation can show where vessels are, which is needed for counting, tracking, or checking safety rules.

In this subsection, the main tasks are presented separately to clarify the information each provides and why object detection is the central task of this thesis. Table 2.1 summarizes the key differences between them.

Table 2.1. Comparison of key computer vision tasks based on their output detail and instance separation capabilities.

Computer Vision Task	**Output Detail**	**Localization**	**Separates Instances**
Image Classification	One label for the whole image	No	No
Object Detection	Labeled bounding boxes for each object	Yes	Yes
Semantic Segmentation	Class labels for all pixels	Yes	No
Instance Segmentation	Per-pixel masks for each individual object	Yes	Yes

Image Classification

Image classification, an example of which is shown in Figure 2.2, assigns one global label to the full image (for example, *open sea*, *port area*, or *foggy scene*). The model does not indicate object locations; it only predicts what is present at the scene level. A major breakthrough in this task was the introduction of Deep Residual Learning (ResNet) by He et al. [12]. ResNet solved the vanishing gradient problem by changing network layers to learn residual functions based on their inputs. This enabled the training of much deeper networks and set a new standard for visual recognition tasks.

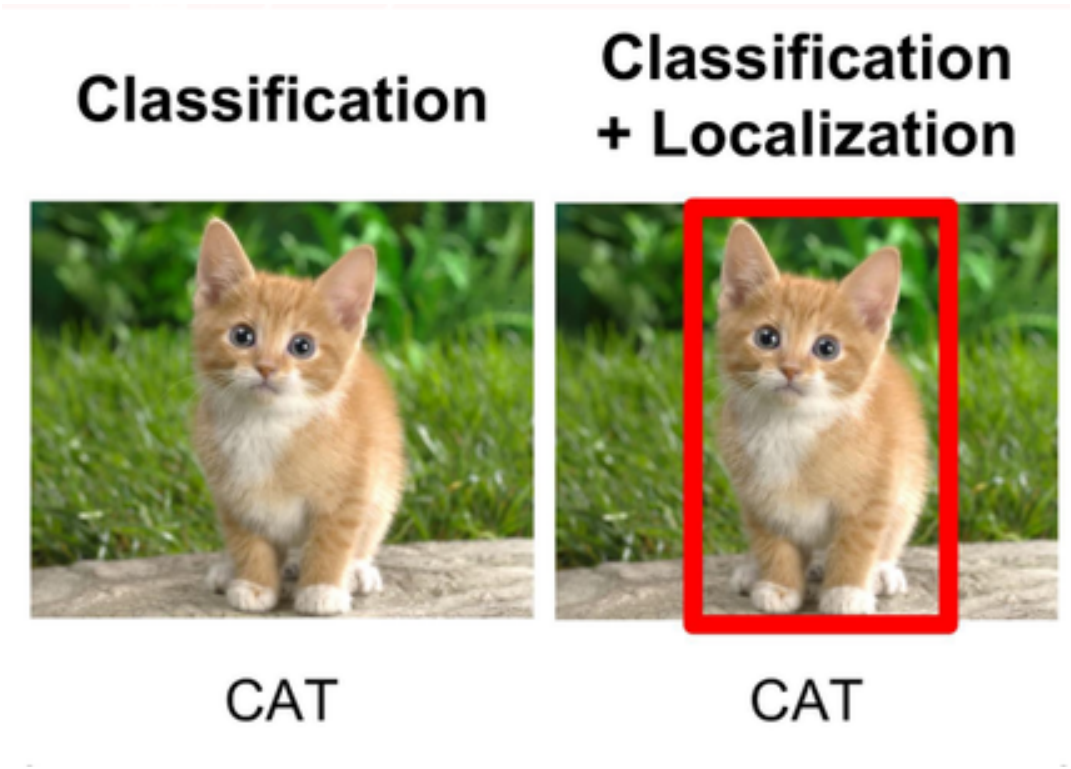

Figure 2.2. Conceptual illustration of image classification, where one label is predicted for the entire frame. [13]

Object Detection

Object detection, an example of which is shown in Figure 2.3, predicts both *what* the objects are and *where* they are by outputting class labels with bounding boxes. This is the central task of this thesis because surveillance systems must localize multiple vessels in each frame while preserving real-time performance. A pivotal advancement in this domain is Faster Region Based Convolutional Neural Networks (R-CNN), introduced by Ren et al. [14]. This architecture introduced a Region Proposal Network (RPN) that shares full-image convolutional features with the detection network, enabling nearly cost-free region proposals. This fundamentally resolved the computational bottleneck of earlier detectors, setting the benchmark for two-stage, real-time object detection systems.

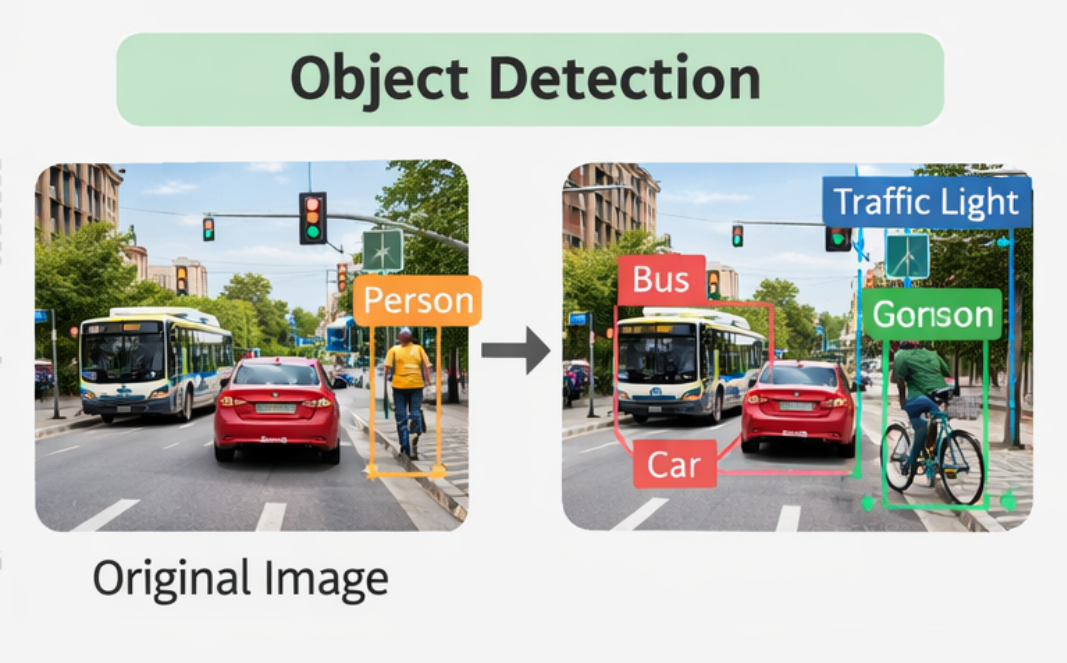

Figure 2.3. Conceptual illustration of object detection with localized objects represented by bounding boxes.

Semantic Segmentation

Semantic segmentation, an example of which is shown in Figure 2.4, performs dense, pixel-level labeling. Every pixel is assigned to a category (e.g., sea, sky, vessel), but different objects of the same class are merged into one region. This task is useful for scene understanding and water-surface analysis. The paradigm for this task was redefined by Fully Convolutional Networks (FCNs), proposed by Long et al. [15]. By adapting contemporary classification networks to process inputs of arbitrary size and produce outputs of corresponding size, FCNs enabled end-to-end, pixels-to-pixels training. This transition drastically improved the efficiency and accuracy of spatially dense predictions.

Figure 2.4. Conceptual illustration of semantic segmentation, where each pixel is assigned a semantic class. [16]

Instance Segmentation

Instance segmentation, an example of which is shown in Figure 2.5, combines detection and segmentation by producing a separate pixel mask for each object instance. Unlike semantic segmentation, two nearby vessels of the same class are still treated as two independent objects. The foundational framework for this task is Mask R-CNN, developed by He et al. Mask R-CNN builds on Faster R-CNN by adding a parallel branch that predicts a high-quality object mask in addition to the branch for bounding box recognition [17]. This method is straightforward, adaptable, and generalizable, and it remains widely used for instance-level recognition tasks.

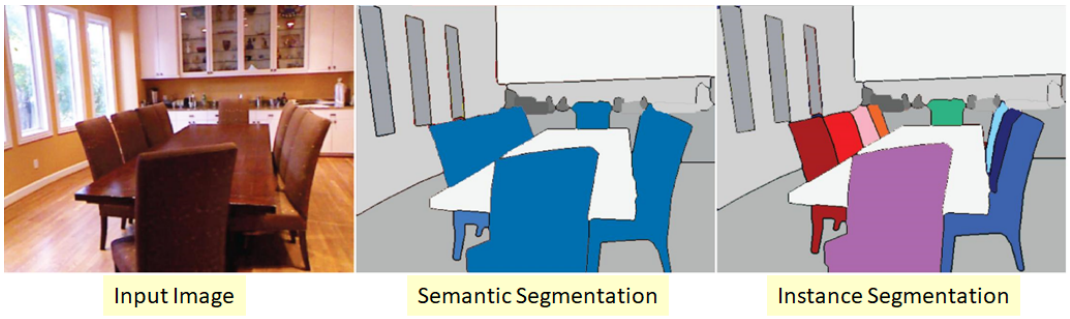

Figure 2.5. Conceptual illustration of instance segmentation with individual object masks. [18]

2.3 Maritime Traffic Monitoring and Port Surveillance

Maritime traffic monitoring plays a key role in today's port management and Maritime Domain Awareness (MDA). Ports are busy, complex places, with many types of vessels moving in tight areas. Good monitoring helps keep navigation safe, manage traffic, and improve port operations.

Traditional maritime surveillance relies on technologies like radar and the AIS. These tools give valuable information about vessel locations and identities, but they do not always work well in ports. For instance, small boats might not have AIS transponders, and radar often struggles with clutter and low resolution. Because of these challenges, more ports are now using vision-based methods with EO sensors to improve monitoring.

2.4 Computer Vision in Maritime Environments

Recent advances in CV and DL have enabled the development of robust vision-based surveillance systems. However, maritime environments introduce unique challenges not encountered in terrestrial environments. As noted by Prasad et al. [19], factors such as water reflections (glint), dynamic background textures (wakes and waves), and varying illumination conditions are significant complications for object detection.

In addition to atmospheric conditions, the fundamental geometry of maritime scenes presents distinct challenges. The sea-sky line (horizon) serves as a critical spatial constraint; however, its detection is frequently impeded by haze or low-contrast conditions in which the sea and sky are radiometrically similar [20]. Furthermore, vessels may appear at widely varying scales depending on their distance from the camera. This scale variability, together with frequent occlusions and low contrast between objects and background, necessitates specialized processing strategies.

Furthermore, as highlighted by [21], unlike static terrestrial surveillance, maritime cameras are often mounted on floating platforms or high-mast coastal stations, leading to **image jitter** and perspective distortion. To address these challenges, robust background-subtraction methods or flexible deep learning models are needed to distinguish between random water movement and intentional ship movement.

2.5 Vision-Based Boat Detection and Classification

Vision-based boat detection aims to localize vessels directly from images or video streams. In addition to detection, classification of vessel types provides higher-level semantic information that is valuable for traffic analysis and decision support systems.

Earlier methods relied on handcrafted features such as Histogram of oriented gradients (HOG) and Scale-Invariant Feature Transform (SIFT) with traditional ML techniques. Although effective in constrained scenarios, these methods lack robustness under real-world conditions. The introduction of CNN-based Deep Neural Network (DNN) architectures has significantly improved performance by enabling end-to-end learning of discriminative features directly from data.

2.6 Deep Learning for Object Detection

Object detection using DL has become a core task in modern AI systems. Unlike image classification, which assigns a single label to an image, object detection involves both localizing objects (via bounding boxes) and classifying them. Modern detectors can be broadly categorized into three architectures, as illustrated in Figure 2.6.

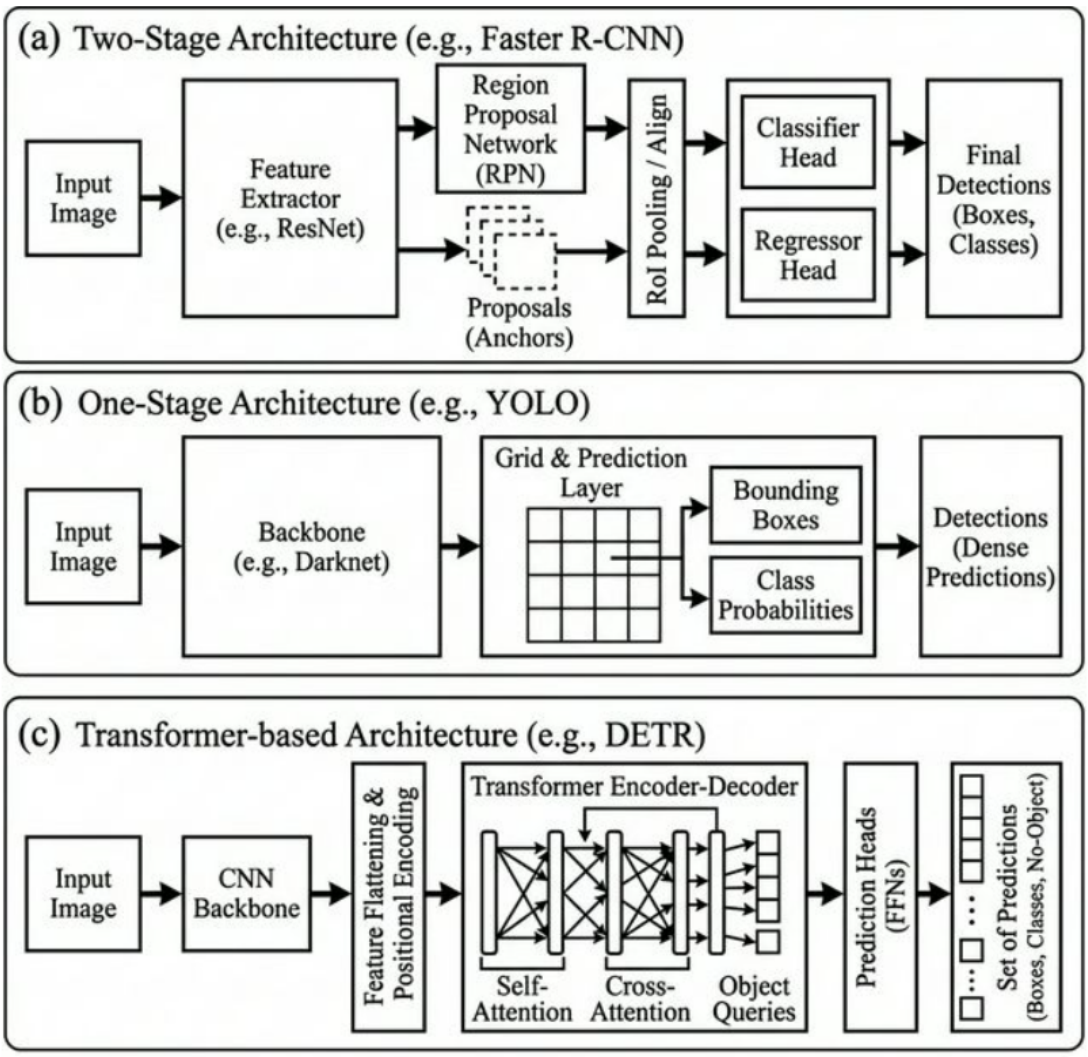

Figure 2.6. Comparison of object detection architectures: (a) Two-Stage Detectors (e.g., Faster R-CNN) use a Region Proposal Network; (b) One-Stage Detectors (e.g., YOLO) predict directly from feature maps; (c) Transformer-Based Detectors (e.g., DETR) use attention mechanisms for set prediction.

2.6.1 Two-Stage Detectors

Two-stage detectors, exemplified by the R-CNN family (R-CNN, Fast R-CNN, Faster R-CNN [22]), operate in a sequential manner.

- **Stage 1 (Region Proposal):** A specialized component, such as the Region Proposal Network (RPN), scans the image to generate a sparse set of Region of Interest (RoI) (candidate regions) that are likely to contain objects.

- **Stage 2 (Classification & Refinement):** The model then takes these candidate regions, predicts their class labels, and adjusts the bounding boxes to better fit..

Although two-stage detectors have often been the most accurate due to the refinement step, they usually require more computing power and are slower, making them unsuitable for real-time use.

2.6.2 One-Stage Detectors

One-stage detectors, such as the YOLO family [23] and Single Shot MultiBox Detector (SSD), discard the region proposal step to optimize for speed. Instead of processing sparse proposals, they treat detection as a "dense prediction" problem. The network spatially divides the image (e.g., into a grid) and directly predicts bounding boxes and class probabilities for every location in the feature maps in a single forward pass. This unified architecture dramatically reduces inference time, processing images at high frame rates, transcending 30 Frame Per Second (FPS), which is essential for dynamic maritime surveillance.

2.6.3 Transformer-Based Detectors

A more recent paradigm shift introduced by DEtection TRansformer (DETR) [24] leverages the Transformer architecture from Natural Language Processing (NLP). Unlike CNN-based approaches that rely on anchors or sliding windows, DETR views detection as a direct set prediction problem.

- **Mechanism:** It uses a CNN backbone to extract features, followed by an encoder-decoder Transformer that uses self-attention mechanisms to model global relationships between object queries and the image context.

- **Bipartite Matching:** During training, it uses bipartite matching loss to uniquely assign predictions to ground truth objects, eliminating the need for Non-Maximum Suppression (NMS) post-processing.

While standard DETR models are computationally intensive and slow to converge, improved variants like Deformable DETR [25] have enhanced performance. However, for edge-based maritime surveillance, one-stage detectors (YOLO) often remain the preferred choice due to their superior efficiency-accuracy trade-off.

2.7 Evaluation Metrics for Object Detection

Performance evaluation of object detection systems relies on a sequence of metric computations that jointly measure *classification correctness*, *localization quality*, and *ranking quality* of predictions. In practice, detections are first matched to ground-truth boxes; then confusion statistics (True Positive (TP), False Positive (FP), False Negative (FN)) are derived; finally, precision-recall based metrics (Average Precision (AP) and mean Average Precision (mAP)) are computed.

2.7.1 Prediction Matching Protocol

Let a detection be defined as (c, b, s), where c is the predicted class, b is the predicted box, and s is the confidence score. For each class, predictions are sorted by decreasing s and matched to ground-truth instances using an Intersection over Union (IoU) threshold τ.

For one prediction, the assignment rule is:

- **TP** class is correct, IoU $\geq \tau$, and the matched ground-truth instance has not been matched by a higher-confidence prediction.
- **FP** class is wrong, or IoU $< \tau$, or duplicate detection of an already matched ground-truth instance.
- **FN** a ground-truth object that remains unmatched.

2.7.2 Intersection over Union (IoU)

IoU quantifies geometric overlap between predicted and ground-truth boxes:

$$IoU = \frac{\text{Area}(B_p \cap B_{gt})}{\text{Area}(B_p \cup B_{gt})} \tag{2.1}$$

where B_p is the predicted box and B_{gt} is the ground-truth box. IoU is in [0,1], where larger values indicate better localization.

2.7.3 Precision, Recall, and F1-Score

Given TP, FP, and FN counts, the basic detection metrics are:

$$Precision = \frac{TP}{TP + FP} \tag{2.2}$$

$$Recall = \frac{TP}{TP + FN} \tag{2.3}$$

$$F1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \tag{2.4}$$

Precision measures prediction reliability (how many predicted objects are correct), while recall measures coverage (how many true objects are found).

2.7.4 Average Precision (AP)

For each class, precision and recall are computed at different confidence thresholds, generating a Precision-Recall (PR) curve. AP is the area under this curve:

$$AP = \int_0^1 p_{interp}(r)\, dr \tag{2.5}$$

where $p_{interp}(r)$ is the interpolated precision at recall r. In COCO-style evaluation, AP is computed from ranked detections and interpolated over discrete recall samples.

2.7.5 Mean Average Precision (mAP)

If there are K classes, mAP is the mean AP across classes:

$$mAP = \frac{1}{K} \sum_{k=1}^{K} AP_k \tag{2.6}$$

Two standard object detection variants are:

- **mAP@50:**APcomputed at a fixed IoU threshold $\tau = 0.50$.
- **mAP@50–95:** AP averaged across IoU thresholds $\tau \in \{0.50, 0.55, \ldots, 0.95\}$, then averaged across classes.

The second metric is stricter because it rewards not only detection but also precise localization at high IoU levels.

2.8 The Architecture of YOLO

The YOLO family of object detectors represents a paradigm shift from traditional two-stage detectors by unifying object localization and classification into a single regression problem. Since its inception by Redmon et al. [23], YOLO has evolved through multiple iterations (v1–v11, YOLO26), consistently establishing the state-of-the-art for real-time object detection.

2.8.1 Grid-Based Detection and Single-Stage Inference

At the core of the YOLO architecture is the concept of dividing the input image into an $S \times S$ grid. If the center of an object falls into a grid cell, that cell is responsible for detecting the object. Unlike region-based approaches that require a separate proposal step, YOLO predicts bounding box coordinates (x, y, w, h), objectness confidence scores, and class probabilities simultaneously for all grid cells in a single forward pass. This unified approach allows the network to reason globally about the full image and all objects within it, leading to faster inference speeds and reduced background errors compared to sliding-window methods.

2.8.2 Core Components: Backbone, Neck, and Head

Modern YOLO architectures, including the YOLOv11 and YOLO26 models used in this work, generally consist of three main components:

- **Backbone:** The feature extractor responsible for downsampling the input image and capturing hierarchical features. Recent versions utilize Cross Stage Partial Network (CSPNet) [26] and improved module designs adopted across YOLOv11 and YOLO26 families (e.g., optimized C3-based blocks). These modules employ gradient path aggregation to reduce computational redundancy while maintaining rich gradient flow, enabling deeper networks to converge more effectively.

- **Neck:** The intermediate component that aggregates features from different backbone levels to improve detection across scales. Modern YOLOs employ a combination of Feature Pyramid Network (FPN) [27] and Path Aggregation Network (PAN) [28]. FPN conveys strong semantic features from top to bottom, while PAN creates bottom-up paths to restore localization details. This multi-scale fusion is critical for detecting objects of varying sizes, particularly small vessels in maritime environments.

- **Head:** The final detection layers that map the aggregated features to bounding box predictions and class scores. In recent "anchor-free" versions, the head is decoupled (predicting box regression and class classification in separate branches) to resolve the conflict between localization accuracy and classification confidence [29].

2.8.3 Evolution: From Anchor-Based to Anchor-Free

Early YOLO versions (v2–v7) relied on "anchor boxes" predefined reference boxes with specific aspect ratios—to stabilize training. While effective, anchor-based methods introduce additional hyperparameters and complexity (e.g., matching IoU thresholds).

YOLOv11 and YOLO26 follow an **anchor-free** detection approach. Instead of predicting offsets from predefined anchors, the model directly predicts the distance from the center of the grid cell to the four sides of the bounding box. This simplification makes the model more robust to objects with extreme aspect ratios (which are common in maritime scenarios, such as long, thin ships viewed from the side) and improves generalization capabilities [29].

2.8.4 Performance Metrics

The performance of YOLO-based detectors in this thesis is assessed using the standardized protocol detailed in Section 2.7, with particular emphasis on **mAP@50** and **mAP@50–95** for model comparison [30].

2.9 Transfer Learning and Fine-Tuning

A significant challenge in training deep CNNs for specialized domains, such as maritime surveillance, is the requirement for vast amounts of labeled data to optimize millions of model parameters. **Transfer learning** addresses this by leveraging knowledge gained from a source task—where data is abundant—to improve performance on a target task with limited data [31]. This approach is based on the observation that the initial layers of a CNN learn generic features, such as Gabor filters and color blobs, which are applicable to almost any visual task, regardless of the specific dataset [32].

As a network progresses toward its output layers, the features transition from general to specific, becoming more attuned to the nuances of the original training set (e.g., the ImageNet dataset). In the context of vessel detection, transfer learning allows a model pre-trained on generic objects to maintain its "visual vocabulary" while adapting its deeper layers to the specific radiometric and geometric properties of maritime environments.

The process typically involves two distinct phases:

- **Feature Extraction:** The weights of the backbone network (pre-trained on a large-scale dataset like Common Objects in COntext (COCO) or ImageNet) are frozen, and only the final classification or detection "head" is trained on the maritime data. This preserves the robust feature extractors learned from millions of images.
- **Fine-Tuning:** After the new head has stabilized, the weights of the upper layers of the backbone are unfrozen and updated with a very low learning rate. This allows the model to refine its high-level filters to better distinguish between vessels and sea-surface anomalies (e.g., wakes or glint) without destroying the foundational low-level features [32].

This strategy is particularly effective for Small Object Detection (SOD) in port surveillance. By starting with a model that already "understands" shapes and textures, the training process converges faster and is significantly less prone to overfitting, even when the available labeled maritime dataset is relatively small compared to industry-standard benchmarks.

2.10 K-Fold Cross-Validation

One of the main challenges in developing machine learning models is ensuring they perform well on new data rather than simply memorizing the training set. This issue is called overfitting. To better evaluate how well a model will work in practice, this study uses K-fold cross-validation.

According to [33], the process starts by randomly splitting the dataset into K equal, non-overlapping parts called folds. The model is then trained and tested K times. Each time, one fold is used for validation while the other $K - 1$ folds are

used for training. After all K rounds, the performance results are averaged to give a more reliable estimate of how well the model predicts.

The main advantage of this method is that every data point is used for both training and validation, and each is tested once. This is useful when there is limited data or rare features, like in high-resolution images. It helps prevent bias from splits that are too simple or too difficult. Typically, $K = 5$ or $K = 10$ is chosen to balance testing time and result variation.

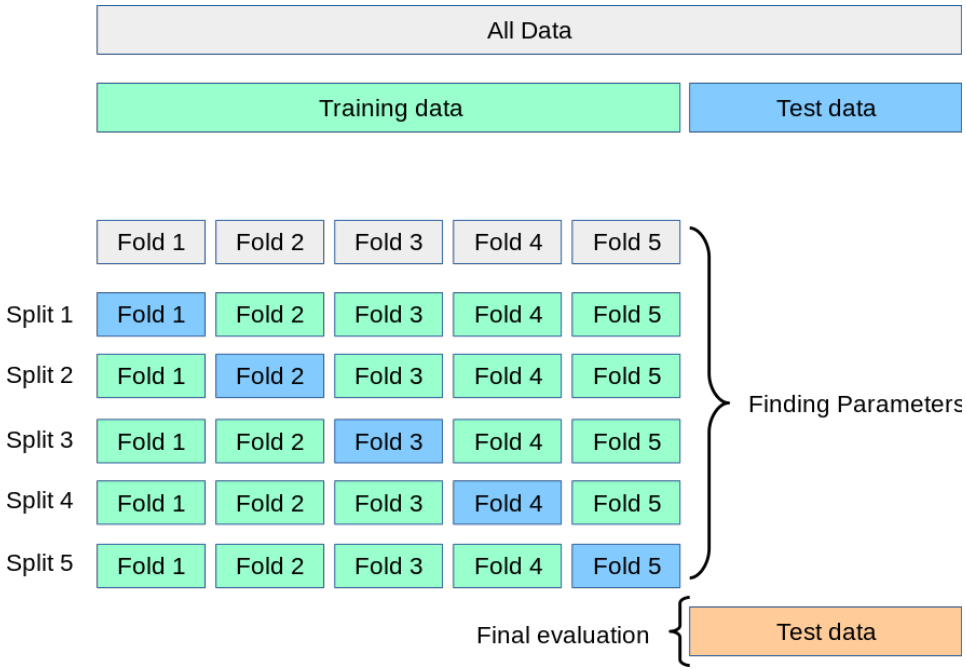

Figure 2.7. Illustration of k-fold cross-validation over a training set, where each fold is used once for validation while the remaining folds are used for training. Adapted from the scikit-learn user guide [34], [35].

2.11 Small Object Detection in High-Resolution Images

SOD remains a significant bottleneck in CV, particularly in maritime surveillance where distant vessels may occupy as little as 0.01% of the total image area. As noted by [36], small objects suffer from limited appearance information, making them highly susceptible to noise and background interference.

One of the main problems in SOD is the resolution trade-off. While HR images keep fine details, most modern CNN models need the input to be downscaled (for example, to 416×416 or 640×640 pixels) to work in real time. This downscaling often causes a permanent loss of spatial information, so small targets may shrink to a single pixel or disappear completely during the pooling layers of a deep network.

To mitigate this, researchers have proposed two main strategies:

- **Multi-Scale Feature Extraction:** Architectures such as the **FPN** [27] leverages lateral connections to combine high-resolution architectural features from early layers with rich semantic information from deeper layers.
- **Sub-patch or Tiling Strategies:** Rather than downscaling the entire HR frame, the image is divided into overlapping patches, allowing the network to process local regions at their original resolution to preserve small-scale features [37].

By utilizing these specialized processing strategies, models can achieve higher sensitivity to small vessels while managing the inherent computational complexity of high-resolution maritime streams.

2.12 Image Slicing Techniques and SAHI

Many object detection systems struggle with high-resolution images, especially due to the "small object problem." In this case, distant vessels appear too coarse on the feature map, making them hard to locate accurately. Image slicing, or tiling, helps solve this by splitting a HR image into a grid of overlapping patches called slices. By processing each patch separately, the detector keeps the original pixel density and improves the signal-to-noise ratio for small targets [37].

The SAHI framework, developed by Akyon et al. [1], provides a standard, model-independent way to slice images. SAHI splits the input image into slices of size $M \times N$ with a set overlap. This overlap makes sure that objects on the edge of a slice are fully shown in at least one neighboring window.

During inference, the detection model runs on each slice, and the local predictions are mapped back to the original image. To handle duplicate detections from overlapping areas, SAHI uses strong post-processing methods like NMS or Non-Maximum Weighted (NMW) merging. According to [1], this approach greatly improves small-object detection and often increases Mean Average Precision (mAP) without requiring retraining the detection model.

2.13 Overfitting and Data Leakage in Video-Derived Datasets

A significant portion of maritime datasets is curated from high-resolution video footage through temporal sampling. While this provides a vast amount of labeled data, it introduces the risk of **temporal redundancy**, where consecutive frames contain near-identical pixel information. As noted by [38], if these frames are distributed across both training and validation sets via simple random shuffling, it leads to **data leakage**. In this scenario, the model "memorizes" specific background noise or lighting conditions rather than learning the generalizable semantic features of the vessels.

This issue can make performance metrics look better than they really are, giving a false impression of how strong the model is. Research by [39] shows that models trained on temporally similar data often do not perform well in new maritime settings with different wave patterns or weather conditions.

To address this, it is important to use a sequence-based splitting strategy. Rather than shuffling single frames, whole video sequences should be assigned to the training, validation, or test sets. This way, the validation set includes footage that is different in time from what the model saw during training, providing a more accurate test of whether the model can detect objects based on what it has learned, not just on familiar scenes.

2.14 Summary

This chapter has established the theoretical and technical framework for addressing the challenges of vessel detection in complex environments. It started by covering the **fundamentals of computer vision**, tracing the field's evolution from **digital image representation** and **hand-crafted feature extraction** (e.g., Canny edge detection) to the modern paradigm of **CNN**. By transitioning from manual filtering to hierarchical feature learning, these architectures have significantly advanced the core tasks of **object detection** and **instance segmentation**.

The chapter also placed these technologies in the context of **maritime traffic monitoring** and **port surveillance**. Traditional systems like **AIS and Radar** are important, but they struggle to detect small or non-cooperative vessels, underscoring the need for robust vision-based solutions. To address the challenges of object segmentation in high-resolution maritime images, the chapter highlighted advanced methods, including the **SAHI framework** and **multi-scale feature extraction**. It also stressed the importance of using **K-fold cross-validation** and **sequence-based splitting** to ensure the models perform well across different situations and do not overfit to similar video data.

These foundational concepts provide the necessary justification for the experimental methodology and system architecture described in the following chapter.

Chapter 3

Related Works

3.1 Introduction

This chapter reviews prior work most relevant to vessel detection and classification in maritime monitoring, with a focus on port scenarios. The objective is not to reintroduce background concepts from Chapter 2, but to critically compare existing approaches, identify methodological limitations, and position the contribution of this thesis.

3.2 Maritime Vision Datasets and Problem Setting

Progress in maritime detection is strongly conditioned by dataset quality and scene realism. Public datasets often emphasize open-sea imagery, while port environments include denser traffic, stronger occlusion, and higher background clutter. The SPSCD dataset introduced by Petkovic et al. [2] is particularly relevant because it provides high-resolution port imagery across seasons, weather, and traffic conditions, with multi-class vessel annotations.

Dataset design choices directly affect model reliability. Studies on maritime and video-based perception report that correlated frames can inflate evaluation metrics when splitting is not sequence-aware [38], [39]. For this reason, recent works increasingly emphasize robust splitting protocols and careful annotation quality control rather than relying only on aggregate accuracy numbers [2], [40].

3.3 Detection Paradigms in Related Literature

3.3.1 From Two-Stage to Real-Time One-Stage Detectors

Two-stage detectors (R-CNN family) established strong localization performance in generic detection benchmarks [22], but their computational cost often limits real-time deployment in surveillance systems. In maritime monitoring, where continuous processing is required, one-stage detectors have therefore become dominant, especially the YOLO family [23], [40].

Recent Ultralytics implementations continue this trend by improving efficiency and training usability for practical deployments [29]. In practice, maritime papers typically prefer one-stage models when the target application requires low latency and stable throughput, even if heavier architectures may provide marginal gains in ideal offline settings [40].

3.3.2 Transformer-Based Detectors and Deployment Constraints

Transformer-based detectors such as DETR and Deformable DETR are influential in generic object detection due to their end-to-end formulation and strong global reasoning [24], [25]. However, maritime surveillance studies still report a gap between benchmark performance and deployable inference speed, especially for edge-oriented systems and long-duration monitoring [40]. As a result, current operational pipelines in this domain still favor optimized one-stage detectors.

3.4 Small Object Detection and High-Resolution Inference

Small object detection remains a central unresolved issue in maritime imagery because distant vessels occupy very few pixels. Broad reviews confirm that aggressive downscaling degrades localization and recall for small targets [36]. Multi-scale feature designs such as FPN/PAN alleviate this to some extent [27], [28], but they do not fully remove the loss of fine detail when high-resolution frames are resized.

Image tiling and slicing methods therefore play a major role in recent literature. Unel et al. [37] showed that tiling can significantly improve small-object recall by preserving local resolution. Akyon et al. proposed SAHI as a model-agnostic slicing and merging framework, demonstrating consistent gains across detector families [1]. Maritime-focused surveys also report SAHI-style patch inference as one of the most practical strategies for distant-vessel detection in wide-area scenes [40].

3.5 Domain-Specific Challenges in Port Surveillance

Compared with open-sea settings, port monitoring introduces stronger domain complexity: structured background clutter (piers, cranes, buildings), high vessel density, frequent occlusion, and reflections. Maritime video studies describe these effects as major causes of false positives and unstable tracking-by-detection pipelines [19], [21]. In addition, sea-sky ambiguity and varying atmospheric conditions reduce contrast and can degrade detection confidence, especially for small or partially occluded vessels [20].

These observations indicate that model quality alone is insufficient; evaluation and training procedures must explicitly account for domain shift, temporal correlation, and annotation precision in realistic port scenes [2], [39].

3.6 Comparative Synthesis of Prior Works

Table 3.1 summarizes the most relevant literature dimensions for this thesis.

Table 3.1. Summary of representative related works and limitations relevant to this thesis.

Ref	Primary Focus	Main Strength	Limitation for This Thesis Scope
[2]	Port dataset construction (SPSCD)	Realistic high-resolution multi-class port data	Does not provide a full integrated optimization pipeline for slicing, resolution ablation, and sequence-aware validation
[40]	Maritime detection survey/benchmarking	Broad comparison of deep detectors in maritime imagery	Limited port-specific methodological integration across annotation quality, leakage control, and movement-oriented objectives
[1]	SAHI slicing framework	Strong model-agnostic gains for small objects	Not specialized for port-domain temporal redundancy and vessel-class-specific protocol design
[37]	Tiling for small object detection	Demonstrates benefit of patch-based inference	Generic setting; does not directly address maritime port constraints
[38], [39]	Data leakage in correlated data	Highlights split-induced overestimation risks	Not maritime-pipeline specific; requires adaptation to sequence-based vessel data

3.7 Research Gaps and Thesis Positioning

Based on the reviewed literature, three gaps are especially relevant:

- **Integrated design gap:** Existing studies often evaluate architecture choice, slicing, and dataset quality separately rather than as a single coherent pipeline [1], [40].
- **Evaluation rigor gap:** Sequence correlation and data leakage are recognized problems, but many maritime experiments still underreport split protocols and robustness checks [38], [39].
- **Port-specific applicability gap:** Results from generic benchmarks do not always transfer to dense, cluttered port scenes with strong scale variation [19]–[21].

This thesis addresses these gaps by combining: (i) sequence-aware cross-validation on SPSCD, (ii) annotation refinement analysis, (iii) detector-family comparison under fixed training conditions, and (iv) high-resolution slicing-based inference with SAHI for improved small-vessel sensitivity [1], [2], [29].

3.8 Summary

The reviewed literature confirms that maritime vessel detection has advanced significantly with deep one-stage detectors, yet robust performance in real port environments still depends on data curation, leakage-aware validation, and high-resolution small-object handling. These findings motivate the methodological choices presented in Chapter 4 and the experimental protocol analyzed in Chapter 5.

Chapter 4

Methods

4.1 Methodological Overview

This chapter presents the complete methodology used to build the proposed maritime monitoring pipeline. The design choices are driven by four constraints observed in port surveillance: (i) dense traffic and occlusion, (ii) small-object visibility at long range, (iii) temporal correlation in video-derived data, and (iv) the need for practical inference speed in real deployments [19], [21], [36].

To address these constraints, the workflow is organized into modular stages:

1. dataset curation and annotation refinement;
2. sequence-aware fold construction for robust validation;
3. detector training under a fixed and reproducible recipe;
4. high-resolution sliced inference for small-vessel recovery.

This structure allows each stage to be analyzed independently while preserving end-to-end consistency.

4.2 Dataset Preparation and Quality Control

4.2.1 Source Dataset and Acquisition Context

The experiments are based on the SPSCD dataset [2], collected in the Port of Split from real surveillance video streams. It contains 19,337 images at 1920×1080 resolution, acquired across different seasons, illumination regimes, and sea states. These characteristics make SPSCD suitable for stress-testing detector robustness in realistic maritime conditions.

4.2.2 Class Taxonomy and Label Space

The task is formulated as 12-class object detection, following the SPSCD taxonomy [2]. Let $\mathcal{C} = \{1, \ldots, 12\}$ be the class set. For each image I, ground truth is represented as:

$$\mathcal{Y}(I) = \{(c_i, b_i)\}_{i=1}^{N_I}, \quad c_i \in \mathcal{C} \tag{4.1}$$

where $b_i = [x_c, y_c, w, h]$ are YOLO-normalized box parameters.

Table 4.1. Vessel-class taxonomy used in this thesis (SPSCD protocol). [2]

Cat. ID[1]	Category name	Description
0	Small Craft	Small training sailboat with a length $\leq 6\ m$, dinghy, canoe, jet skis, windsurf boards, stand-up pedal board
1	Small Fishing Boat	Private boats and fishing boats with a length $\leq 7\ m$
2	Small Passenger Ship	Passenger ships and excursion ships up to 500 GT and length $\leq 50\ m$
3	Fishing Trawler	Fishing boats with length a $> 7\ m$
4	Large Passenger Ship	Large ship, used exclusively for passenger transport, with a length $\geq 130\ m$
5	Sailing Boat	Mono-hull and multi-hull sailing boats with a length $> 6\ m$
6	Speed Craft	Speedboats of all types (made of rubber, plastic, aluminium, etc.) and a length $> 2\ m$
7	Motorboat	Motorboats, also called powerboats with a length of $7\ m$ to $12\ m$
8	Pleasure Yacht	Private or chartered pleasure yachts (or motorboats) with a length $> 12\ m$
9	Medium Ferry	Passenger ferry or RO-RO ferry with a length $\leq 90\ m$, ≤ 1000 passengers and/or ≤ 150 vehicles
10	Large Ferry	Passenger ferry or RO-RO ferry with a length $> 90\ m$, > 1000 passengers and/or > 150 vehicles
11	High speed craft	High-speed passenger catamarans with a length of up to $50\ m$

4.2.3 Annotation Refinement Protocol

Starting from the original labels [2], a second manual review pass was performed to improve localization precision in cluttered scenes. Each bounding box was checked with three criteria:

- **Tightness:** box boundaries should minimally include background around the hull;
- **Completeness:** visible vessel extent should be fully covered when not heavily occluded;
- **Class consistency:** class identity should remain stable across similar viewpoints.

Boxes violating these criteria were corrected. This operation primarily targets the stricter localization metric (mAP@50–95), where coarse or noisy boxes are penalized more strongly [30].

4.3 Sequence-Aware Split Strategy

4.3.1 Motivation

Video-derived datasets contain near-duplicate frames. If random frame-level splitting is used, train and validation partitions may share highly correlated samples, creating optimistic results due to leakage [38], [39]. To avoid this issue, splitting is performed at the sequence level.

4.3.2 Formal Definition

Let $\mathcal{S} = \{s_1, \ldots, s_M\}$ be the set of temporal sequences and let each image belong to exactly one sequence. A valid fold partition must satisfy:

$$\forall a \neq b,\ \mathcal{S}_{train}^{(a)} \cap \mathcal{S}_{val}^{(a)} = \emptyset, \tag{4.2}$$

with each sequence assigned to exactly one validation fold across the 5-fold cycle.

This ensures no temporal overlap between training and validation data within each fold.

4.3.3 Cross-Validation Procedure

A 5-fold sequence-aware scheme is adopted:

- in fold k, four sequence groups are used for training and one for validation;
- the process is repeated for $k \in \{1, \ldots, 5\}$;
- final scores are reported as fold-wise averages and variability statistics.

Compared with a single split, this protocol reduces dependence on one specific partition and provides a more robust estimate of expected performance.

4.4 Model Selection and Training Design

4.4.1 Detector Families and Variants

Four one-stage models are evaluated under identical training conditions:

- YOLOv11n,
- YOLOv11s,
- YOLOv26n,
- YOLOv26s.

The comparison isolates the effect of architecture generation (v11 vs v26) and capacity scale (n vs s), while preserving protocol consistency [29].

4.4.2 Training Infrastructure and Hyperparameters

Training is executed on dual NVIDIA T4 GPUs (2 × 16GB VRAM) with mixed precision enabled. Table 4.2 reports core settings.

Table 4.2. Main training hyperparameters used in the experimental campaign.

Parameter	**Value**
Epochs	150
Batch size	64
Input size	640 × 640 (baseline), 1280 × 1280 (ablation)
Patience (early stopping)	30
Device	Dual NVIDIA T4 (2×16GB VRAM)
Workers	8
Precision	AMP enabled
Optimizer	Auto (Ultralytics default schedule)
Initial learning rate (lr_0)	0.01
Final learning rate factor (lrf)	0.01
Momentum	0.937
Weight decay	0.0005
Warmup epochs	3.0
Warmup momentum	0.8
Warmup bias LR	0.1
IoU threshold (validation)	0.7
Max detections per image	300

Data augmentation policy is reported in Table 4.3. To ensure fair comparison between models, the same policy is applied to all.

Table 4.3. Data augmentation policy applied during training.

Augmentation	**Value**
HSV-H / HSV-S / HSV-V	0.015 / 0.7 / 0.4
Translation	0.1
Scale	0.5
Horizontal flip	0.5
Mosaic	1.0
Close mosaic (last epochs)	10
Random erasing	0.4
Auto-augment policy	RandAugment
Disabled transforms	Rotation, shear, perspective, vertical flip, mixup, copy-paste

4.4.3 Experimental Matrix

The study combines architecture and data factors in a controlled matrix. Table 4.4 summarizes the main experiment blocks.

Table 4.4. Experimental matrix used in this thesis.

Block	Models	Input Size	Labels
Baseline comparison	v11n, v11s, v26n, v26s	640×640	Refined
Resolution ablation	v11n, v11s, v26n, v26s	640 vs 1280	Old vs Refined
Label-quality ablation	v11n, v11s, v26n, v26s	640 and 1280	Old vs Refined
SAHI inference study	v11n (sliced setup)	full-resolution sliced	Refined

4.5 High-Resolution Inference with SAHI

4.5.1 Sliced Inference Configuration

For small-vessel recovery, SAHI is applied at validation/test time [1]. Based on the implemented validation script, the slicing configuration is:

- slice size: 640×640,
- overlap ratio: 0.25,
- confidence threshold: 0.25,
- IoU evaluation thresholds: 0.50 to 0.95 (step 0.05).

This configuration preserves local detail while keeping overlap sufficient to reduce border truncation effects.

4.5.2 Prediction Merging and Duplicate Suppression

Each slice is inferred independently, then detections are mapped to global image coordinates. Duplicate predictions caused by overlap are resolved by non-maximum suppression (NMS). Let $\mathcal{B} = \{b_i\}$ be merged boxes and s_i their confidence scores. NMS iteratively keeps the highest-scoring box and removes boxes with IoU above threshold τ:

$$\mathcal{B}_{kept} = \mathrm{NMS}(\mathcal{B}, s, \tau). \tag{4.3}$$

4.6 Implementation and Reproducibility

The pipeline is implemented in Python with the Ultralytics framework (PyTorch backend) [29]. Experiments were executed on Kaggle GPU infrastructure. Reproducibility is supported through:

- fixed fold definitions,
- fixed augmentation recipe,
- explicit hyperparameter reporting,

- per-fold checkpoint retention and post-hoc validation scripts.

These decisions facilitate consistent replication and fair comparison across architectures and ablations.

4.7 Summary

This chapter defined a leakage-aware and reproducible methodology for maritime vessel detection in complex port scenes. The approach combines sequence-aware cross-validation, annotation refinement, controlled model comparison, and SAHI-based high-resolution inference. Together, these components form the basis for the experimental analysis presented in Chapter 5.

Chapter 5

Experiments and Results

5.1 Experimental Setup

This chapter reports the quantitative and qualitative evaluation of the proposed detection pipeline on SPSCD. The analysis is organized to answer four questions:

- Which detector family and model scale provides the best performance in this port scenario?
- How stable are the results across sequence-aware folds?
- How do annotation version (old vs refined labels) and input resolution (640×640 vs 1280×1280) affect localization quality?
- Does sliced high-resolution inference (SAHI) provide practical gains for small-vessel visibility?

5.1.1 Data Split and Validation Protocol

A sequence-aware 5-fold cross-validation protocol is used to avoid temporal leakage from near-duplicate video frames [38], [39]. This protocol relies on a manually created sequence index file. The file lists every continuous video segment in SPSCD along with its metadata, including the sequence ID, start and end frame numbers, recording date, time of day (day part), and sky condition (such as clear, foggy, or overcast). This structured record enabled splitting the data in a principled, reproducible way, rather than using a random image split.

Using the sequence index, each fold boundary was placed at sequence transitions rather than at individual frame boundaries. This guarantees that temporally adjacent frames – which are visually near-duplicate due to low inter-frame motion in port footage – are never split across training and validation subsets, directly preventing information leakage. For each fold, training uses four sequence groups, and validation uses the held-out group.

In addition to preventing leakage, we used the sequence metadata to ensure condition balance. We assigned folds so that training, validation, and the fixed

independent test set each have a representative mix of environmental conditions, such as foggy, sunny, daytime, and nighttime scenes. This balance helps prevent overfitting to specific lighting or weather conditions that may appear only in one partition. We also evaluate the best fold checkpoints on a fixed, independent test set to compare the final model's performance.

5.1.2 Experiment Matrix and Chapter Organization

The training campaign follows a full-factor structure: four detector variants (YOLO v11n, YOLOv11s, YOLOv26n, YOLOv26s), two annotation settings (old labels and refined labels), and two input resolutions (640×640 and 1280×1280). Accordingly, this chapter reports results as a condition-wise comparison rather than a baseline-versus-ablation narrative. The presentation flow is: (i) cross-validation ranking for one fully populated condition (refined labels, 640×640), (ii) independent test-set transfer, (iii) cross-condition comparisons for resolution and annotation quality, and (iv) SAHI-specific analysis.

5.1.3 Evaluation Metrics and Reporting Style

Primary metrics follow COCO-style detection evaluation [30]:

- **mAP@50**: detection quality at IoU = 0.50;
- **mAP@50–95**: average AP over IoU thresholds 0.50:0.05:0.95.

To increase interpretability beyond raw fold tables, this chapter also reports:

- fold mean and sample standard deviation,
- coefficient of variation ($CV = \sigma/\mu$) as a stability indicator,
- absolute deltas for ablation studies.

5.2 Condition-Wise Cross-Validation Results: Refined Labels at 640 X 640

5.2.1 Per-Fold Scores

Tables 5.1 and 5.2 present fold-level validation scores for one fully available condition (refined labels in 640×640), which is used here as the reference point for architecture-comparison before cross-condition analysis.

Table 5.1. 5-fold cross-validation scores for **mAP@50** (trained on 640 × 640 images with refined labels).

Model	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5
YOLOv11n	0.703	**0.790**	0.742	0.779	0.726
YOLOv11s	0.745	**0.826**	0.793	0.787	0.763
YOLOv26n	0.664	**0.767**	0.741	0.753	0.691
YOLOv26s	0.732	**0.790**	0.764	0.783	0.759

Table 5.2. 5-fold cross-validation scores for **mAP@50–95** (trained on 640 × 640 images with refined labels).

Model	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5
YOLOv11n	0.540	0.575	0.568	**0.587**	0.578
YOLOv11s	0.602	**0.640**	0.614	0.634	0.618
YOLOv26n	0.517	0.585	0.575	**0.594**	0.548
YOLOv26s	0.578	**0.625**	0.596	0.620	0.617

5.3 Condition-Wise Cross-Validation Results: Refined Labels at 1280 × 1280

5.3.1 Per-Fold Scores

Tables 5.3 and 5.4 present fold-level validation scores for all four models trained at 1280 × 1280 with refined labels. This condition provides a direct parallel to the 640 × 640 cross-validation results and enables a fold-aligned resolution comparison. All five folds are available for every model at this resolution.

Table 5.3. 5-fold cross-validation scores for **mAP@50** (trained on 1280 × 1280 images with refined labels). Bold indicates the best fold score per model.

Model	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5
YOLOv11n	0.808	**0.866**	0.846	0.775	0.856
YOLOv11s	0.787	0.853	**0.869**	0.833	0.850
YOLOv26n	0.796	0.824	**0.845**	0.768	0.819
YOLOv26s	0.809	0.835	**0.853**	0.794	0.826

5.3.2 Discussion

At 1280 × 1280, the architecture ranking broadly mirrors the 640 × 640 condition, but with elevated absolute scores across all models. YOLOv11s achieves the highest mean mAP@50 (0.838), while YOLOv26s leads on mean mAP@50–95 (0.709), reflecting a slight advantage in strict localization under high-resolution supervision.

Table 5.4. 5-fold cross-validation scores for **mAP@50–95** (trained on 1280×1280 images with refined labels). Bold indicates the best fold score per model.

Model	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5
YOLOv11n	0.675	**0.715**	0.697	0.663	**0.715**
YOLOv11s	0.657	0.706	0.725	0.720	**0.729**
YOLOv26n	**0.687**	0.685	0.684	0.666	**0.688**
YOLOv26s	0.688	0.713	0.723	0.699	**0.724**

YOLOv26n exhibits the lowest CV on mAP@50–95 (1.4%), suggesting particularly stable localization behaviour across folds despite using the smallest YOLOv26 variant. The lowest overall CV on mAP@50 belongs to YOLOv26s (2.8%), confirming consistent detection quality across sequence-aware splits.

Comparing fold-level ranges between the two resolution conditions highlights the resolution benefit at the fold level: the per-fold mAP@50–95 values at 1280×1280 uniformly exceed those at 640×640 for corresponding folds and models, consistent with the averaged test-set results presented in Section 5.5.

5.4 Independent Test-Set Performance

5.4.1 Final Model Comparison

Best checkpoints from each fold were evaluated on a fixed independent test set. Results are summarized in Table 5.5.

Table 5.5. Average performance on the independent test set.

Model	Variant	Avg mAP@50	Avg mAP@50–95
YOLOv11	Nano (n)	0.874	0.700
YOLOv11	Small (s)	**0.889**	**0.732**
YOLOv26	Nano (n)	0.851	0.686
YOLOv26	Small (s)	0.886	0.729

5.4.2 Cross-Validation to Test Transfer

All models show higher scores on the independent test set than on fold validation averages. Table 5.6 reports absolute gains against the cross-validation means.

This behavior is coherent with strict sequence-aware splits: fold validation scenes can be harder and more heterogeneous, while the fixed test set may be comparatively less adverse for some classes.

Table 5.6. Difference between test-set average and CV mean (Test - CV).

Model	Δ mAP@50	Δ mAP@50–95
YOLOv11n	+0.1260	+0.1304
YOLOv11s	+0.1062	+0.1104
YOLOv26n	+0.1278	+0.1222
YOLOv26s	+0.1204	+0.1218

5.5 Cross-Condition Comparison I: Input Resolution Effect (640 vs 1280)

5.5.1 Numerical Comparison

To isolate the role of spatial resolution within matched training settings, all four detector variants were trained at both 640 × 640 and 1280 × 1280 using refined labels. Table 5.7 reports the average test-set scores across the best checkpoint from each fold, while Figure 5.1 visualizes the model-wise comparison between 640 and 1280 settings for both mAP@50 and mAP@50–95.

Table 5.7. Resolution comparison for all models: average test-set scores across best checkpoints per fold (refined labels).

Model	Input Size	Avg mAP@50	Avg mAP@50–95	Δ mAP@50–95 vs 640
YOLOv11n	640 × 640	0.873	0.699	–
	1280 × 1280	0.932	0.811	+0.112
YOLOv11s	640 × 640	0.889	0.732	–
	1280 × 1280	0.931	0.823	+0.091
YOLOv26n	640 × 640	0.851	0.686	–
	1280 × 1280	0.935	0.819	+0.133
YOLOv26s	640 × 640	0.885	0.729	–
	1280 × 1280	0.931	0.832	+0.103

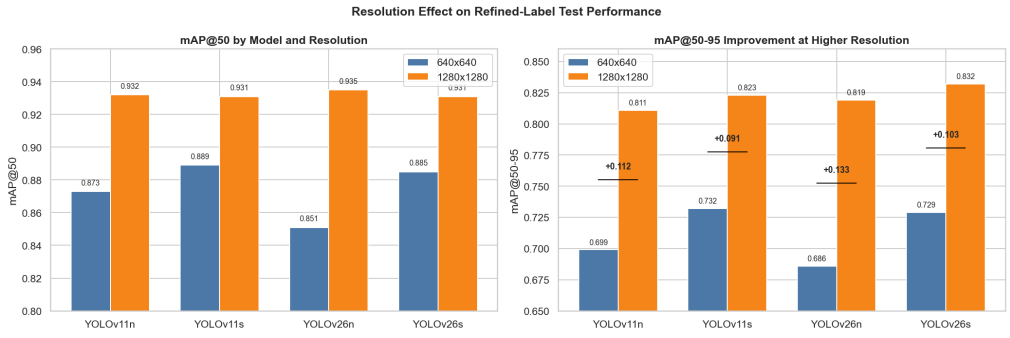

Figure 5.1. Resolution effect on refined-label test performance across all models: left panel shows mAP@50 and right panel shows mAP@50–95, comparing 640×640 and 1280 × 1280 with per-model improvement annotations.

5.5.2 Discussion

Across all four models, increasing the input resolution from 640 × 640 to 1280 × 1280 consistently improves both mAP@50 and mAP@50–95. The gain in strict localization quality (mAP@50–95) ranges from +0.091 (YOLOv11s) to +0.133 (YOLOv26n), while mAP@50 improves by +0.042–+0.084. The benefit is most pronounced on mAP@50–95, indicating that high resolution primarily improves localization precision rather than only coarse detection. This is expected in maritime scenes, where distant vessels occupy a small number of pixels and are sensitive to downscaling artifacts [36], [37].

5.6 Cross-Condition Comparison II: Annotation Quality Effect (Old vs Refined Labels)

5.6.1 Numerical Comparison

To evaluate annotation quality in a unified way, Table 5.8 summarizes old-vs-refined label performance for all four models at both input resolutions. Results are reported as average test-set mAP@50–95 across best checkpoints per fold, with explicit deltas where paired old/refined results are available. Figure 5.2 visualizes the same comparison in bar-chart form for faster cross-model interpretation.

Table 5.8. Annotation quality effect across all models: old vs refined labels (average test-set mAP@50–95).

Model	Old @ 640	Refined @ 640	Δ @ 640	Old @ 1280	Refined @ 1280	Δ @ 1280
YOLOv11n	0.663	0.699	+0.036	0.738	0.811	+0.073
YOLOv11s	0.699	0.732	+0.033	0.737	0.823	+0.086
YOLOv26n	0.660	0.686	+0.026	0.735	0.819	+0.084
YOLOv26s	0.701	0.729	+0.028	0.750	0.832	+0.082

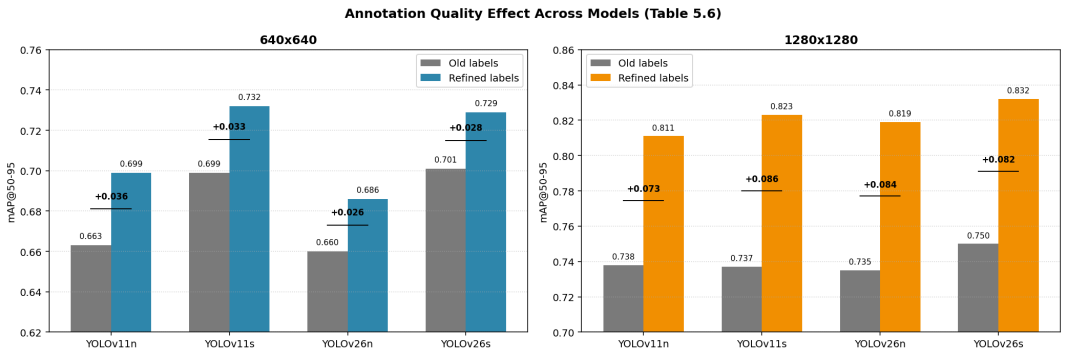

Figure 5.2. Visualization of Table 5.8: old vs refined label performance at 640 × 640 and 1280 × 1280 across all models, including per-model improvement annotations where paired results are available.

5.6.2 Qualitative Evidence

Figures 5.3 and 5.4 provide representative old-vs-refined examples (YOLOv26s) to illustrate the annotation-level effect. In these matched samples, refined labels produce tighter and more complete object extents, especially around vessel masts and elongated hull boundaries. This visual behavior is consistent with the model-wise results in Table 5.8, where all models improve under refined labels and the largest gains appear at 1280×1280.

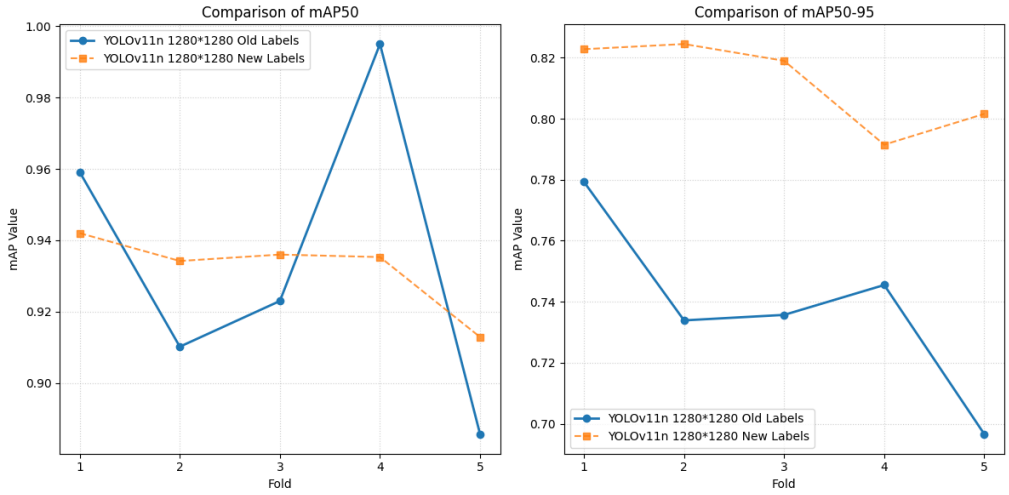

Figure 5.3. Representative qualitative comparison at 1280×1280 with old vs refined labels.

5.6.3 Discussion

Viewed jointly across models, annotation refinement consistently improves strict localization wherever paired results are available. At 640×640, gains range from $+0.026$ to $+0.036$, while at 1280×1280 gains are larger ($+0.073$ to $+0.084$). This indicates a complementary effect between annotation quality and spatial detail: high-resolution inputs are exploited more effectively when supervision geometry is accurate.

5.7 SAHI Inference Study

5.7.1 Protocol and Available Evidence

SAHI was applied at inference time using overlap-aware slicing and prediction merging [1]. To obtain robust validation behavior with SAHI, the original high-resolution frames (1920×1080) were first sliced into 640×640 patches for training. This design intentionally matches the detector's training scale and the SAHI slice scale used during inference, reducing train–inference pixel-scale mismatch that can otherwise degrade localization consistency.

For transparency, this chapter therefore reports:

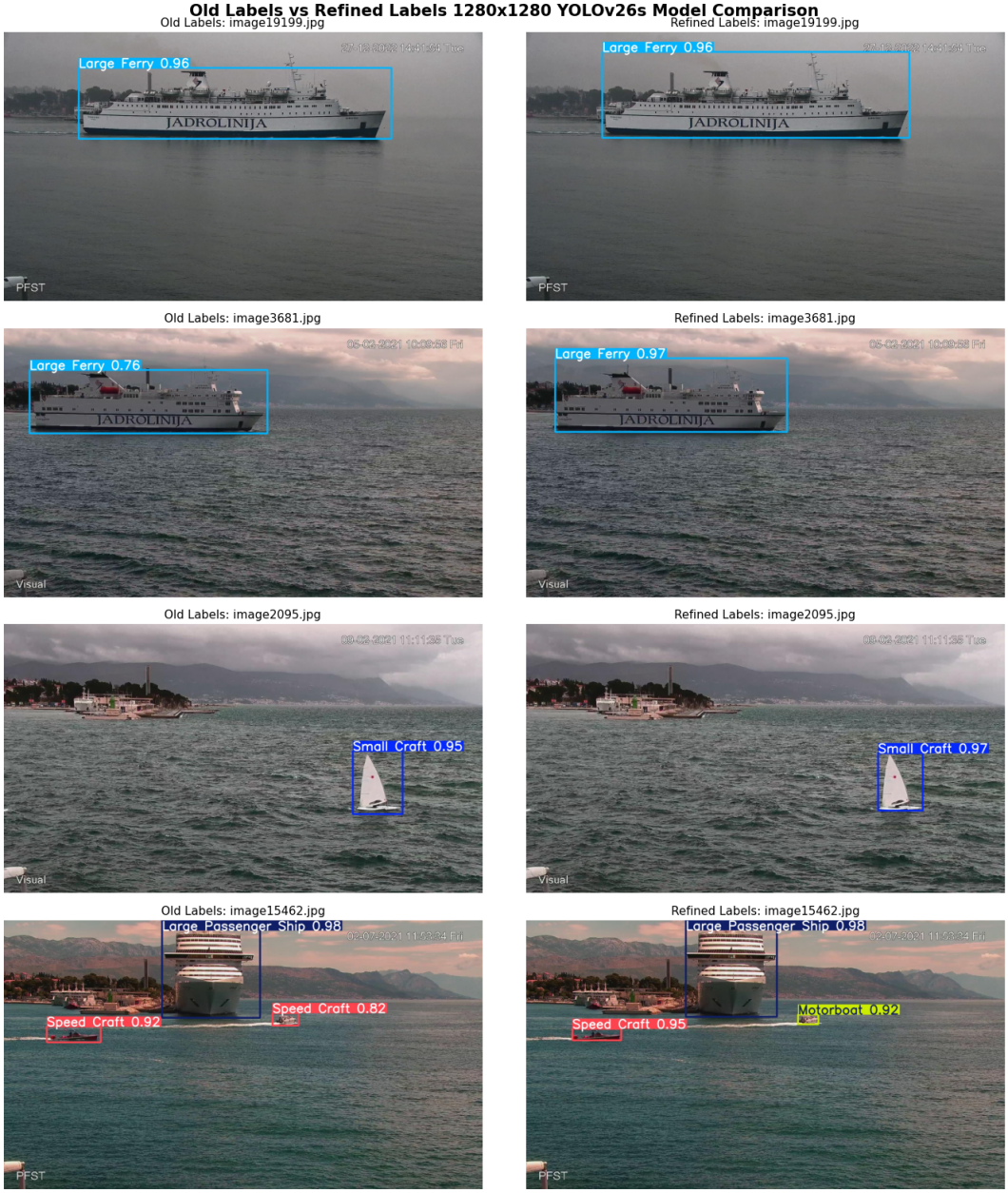

Figure 5.4. Representative YOLOv26s side-by-side predictions on identical samples using old-label and refined-label best checkpoints; refined labels produce tighter boxes and fewer mislocalized detections.

- the exact SAHI evaluation setup from the validation script,
- the rationale for resolution alignment between training and inference,
- qualitative comparisons from generated outputs,
- the practical interpretation for deployment.

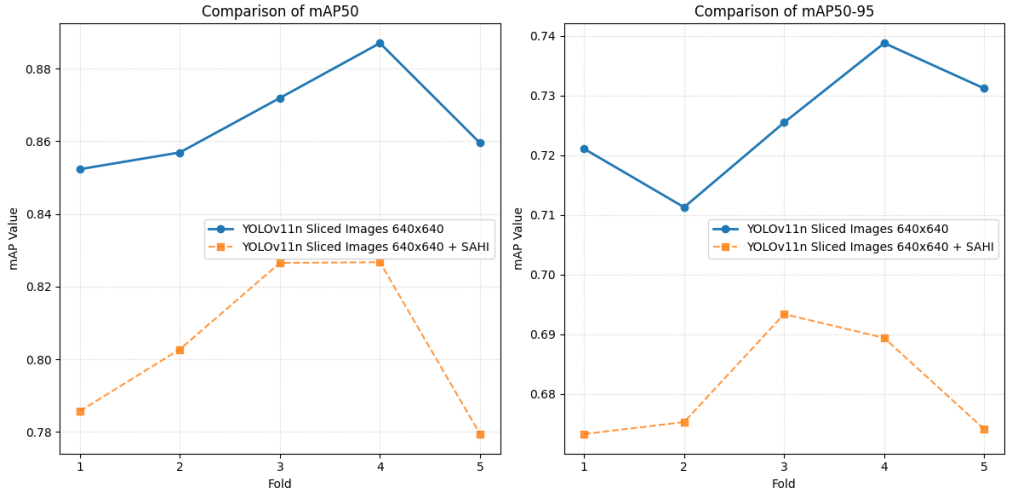

Figure 5.5. Qualitative comparison of standard YOLO inference and SAHI-assisted inference.

5.7.2 Qualitative Findings

Visual inspection across folds indicates a recurrent pattern when comparing standard YOLO inference and SAHI-assisted inference:

- large and medium vessels are reliably detected in both inference modes;
- some distant small vessels that are weak or missed in standard inference are recovered with SAHI;
- qualitative gains are most evident in crowded long-range regions where object size is near the detector's native feature-map limit.

These observations are consistent with previous SAHI literature on small-object recovery [1], [37]; however, in this study, they are local and scene-dependent and did not translate into higher aggregate validation mAP.

5.7.3 Quantitative Comparison on the Sliced Validation Setup

Table 5.9 summarizes fold-wise validation performance for YOLOv11n trained on sliced 640×640 images, with and without SAHI at inference.

Table 5.9. Validation comparison on sliced data (640×640 training patches).

Method	**Avg mAP@50**	**Avg mAP@50–95**	**Δ mAP@50–95**
YOLOv11n sliced (no SAHI)	0.866	0.726	–
YOLOv11n sliced + SAHI	0.804	0.681	-0.044

Although SAHI is theoretically beneficial for small-object recall, in this specific validation setup it did not improve aggregate mAP. A likely explanation is that

resolution mismatch had already been minimized by the 640×640 slicing strategy during training, so additional tiled inference provided limited net benefit while introducing extra merge sensitivity in dense maritime scenes. Another plausible factor is slice-induced object fragmentation: when high-resolution frames are partitioned into fixed tiles, large vessels may be split across tile boundaries and appear as partial objects. This can bias the learned size prior of the detector, causing weaker geometric consistency for full-scale large boats at evaluation time.

5.8 Qualitative Error Analysis

Beyond aggregate mAP, recurring error modes were observed:

- **Class confusion at distance:** visually similar small craft classes are occasionally interchanged;
- **Occlusion fragmentation:** partially visible vessels can produce box truncation or under-sized predictions;
- **Background-induced false positives:** port structures and wave highlights may trigger low-confidence detections.

These failure cases suggest that future improvement should combine detector tuning with temporal aggregation and class-aware post-processing.

5.9 Threats to Validity

The following factors should be considered when interpreting results:

- **Dataset scope:** experiments are centered on one port dataset; cross-port transfer is not yet quantified.
- **SAHI reporting granularity:** available repository artifacts for SAHI are visual-first; future runs should export machine-readable metric summaries per fold.

Explicitly reporting these limitations improves the reliability and reproducibility of conclusions.

5.10 Summary

The experiments show that model rankings depend on the specific conditions, rather than there being a single best model overall. With refined labels at 640×640, YOLOv11s performs best on the independent test set (mAP@50–95 = 0.732). At higher resolution and with refined labels, YOLOv26s achieves the top strict-localization result (mAP@50–95 = 0.832), followed closely by YOLOv11s (0.823)

and YOLOv26n (0.819). For old-label conditions, the highest mAP@50–95 at 640×640 comes from YOLOv11s (0.699), while at 1280×1280 it is YOLOv26s (0.750).

Across paired old-vs-refined comparisons, annotation refinement consistently improves strict localization for all evaluated models at both resolutions. At 640×640, the gains range from $+0.026$ to $+0.036$, while at 1280×1280 they range from $+0.073$ to $+0.086$, confirming that the impact of annotation quality becomes stronger when higher spatial detail is available. These results reinforce two main conclusions: increasing resolution improves localization quality, and higher-quality annotations allow the models to better exploit high-resolution detail. For SAHI, the study confirms that matching training and inference scale through 640×640 slicing is an important design choice for reliable evaluation; however, under this aligned setup, SAHI did not improve aggregate validation mAP. Together, these findings validate the methodological choices of Chapter 4 and show that final model selection should be made per operating condition (label quality and input resolution) rather than by a single global ranking.

Chapter 6

Conclusions and Future Works

6.1 Conclusions

This thesis introduced a deep learning pipeline for detecting and classifying maritime vessels in port environments. Using the latest YOLO architecture and a full-factor experimental design that varied detector scale, input resolution, annotation quality, and inference strategy, the system addresses key challenges, such as detecting small objects and handling high-resolution images in port surveillance.

The experimental evaluation on the SPSCD dataset through a rigorous sequence-aware 5-fold cross-validation reveals a condition-dependent model ranking rather than a single universal winner. In the primary condition (refined labels at 640×640), **YOLOv11s (Small)** achieved the best independent test-set performance (mAP@50-95 of 0.732, mAP@50 of 0.889), offering a strong balance between accuracy and efficiency. At higher resolution (1280×1280) with refined labels, **YOLOv26s** recorded the best reported test-set strict-localization result (mAP@50-95 of 0.832), with YOLOv11s (0.823) and YOLOv26n (0.819) close behind, demonstrating that larger-scale YOLOv26 variants become increasingly competitive as spatial detail improves.

Three cross-condition comparisons provided further insight into the key factors driving performance. First, increasing training resolution from 640×640 to 1280×1280 consistently improved mAP@50-95 across all four models, with gains ranging from +0.091 (YOLOv11s) to +0.133 (YOLOv26n). Second, annotation refinement consistently improved localization quality across all tested models and resolutions, with gains ranging from +0.026 to +0.036 at 640×640 and from +0.073 to +0.086 at 1280×1280, confirming that high-resolution inputs and precise supervision are complementary. Third, SAHI-assisted inference at 640×640 tile size did not improve aggregate mAP@50-95 on this dataset ($\Delta = -0.044$), indicating that the benefit of sliced inference is scene-dependent and that matched training–inference scale is a non-trivial design choice.

In summary, the findings demonstrate that model ranking in maritime vessel detection is influenced by input resolution, annotation quality, and inference strategy. These results provide a validated, condition-aware framework for port monitoring and identify clear directions for future work, particularly through higher-resolution

SAHI evaluation and improved annotation practices to enhance operational effectiveness.

6.2 Future Works

While the proposed system demonstrates strong performance, several avenues for future research and optimization remain:

6.2.1 Edge Deployment and Optimization

To facilitate real-time processing on resource-constrained harbor cameras, future work should focus on model compression techniques. Methods such as **quantization** (reducing precision to FP16 or INT8) and **TensorRT** optimization could significantly reduce inference latency without compromising accuracy, enabling deployment on edge devices like the NVIDIA Jetson series.

6.2.2 Multispectral Data Integration

The current system relies solely on visible spectrum (RGB) imagery. Maritime environments, however, are subject to adverse weather conditions like fog, rain, and low light. Integrating **thermal (infrared) imagery** would create a more robust multi-modal system capable of 24/7 surveillance, independent of lighting conditions.

6.2.3 Sensor Fusion with AIS and Radar

Visual detection can be corroborated with data from AIS and Radar systems. A sensor fusion approach would allow for the correlation of visual tracks with broadcasted vessel identities, enabling the detection of "dark" (non-cooperative) vessels that do not transmit AIS signals—a key capability for security applications.

6.2.4 SAHI Scale Sensitivity and Larger Slicing Windows

An important future direction is to evaluate SAHI with larger slicing windows, especially 1280×1280, instead of only 640×640. A larger tile size may preserve more global context for large vessels, reducing the risk that long hulls are split across tile borders and learned as fragmented objects. This could improve size consistency and localization quality when SAHI is enabled. A controlled ablation should therefore compare SAHI at 640 and 1280 slice sizes (with matched overlap and confidence thresholds) and report fold-wise mAP@50/mAP@50–95 together with error analysis on large-vessel scenes.

Bibliography

[1] F. C. Akyon, S. O. Altinuc, and A. Temizel, «Slicing aided hyper inference and fine-tuning for small object detection», in *2022 IEEE International Conference on Image Processing (ICIP)*, IEEE, 2022, pp. 966–970. DOI: 10.1109/ICIP46576.2022.9897990.

[2] M. Petković, I. Vujović, Z. Lušić, and J. Šoda, «Image dataset for neural network performance estimation with application to maritime ports», *Journal of Marine Science and Engineering*, vol. 11, no. 3, 2023, ISSN: 2077-1312. DOI: 10.3390/jmse11030578. [Online]. Available: https://www.mdpi.com/2077-1312/11/3/578.

[3] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer, 2006.

[4] S. J. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, 3rd. Prentice Hall, 2010.

[5] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*, 2nd. Springer, 2009.

[6] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 2nd. MIT Press, 2018.

[7] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[8] R. Szeliski, *Computer Vision: Algorithms and Applications*, 2nd. Springer Nature, 2022.

[9] A. Torralba, P. Isola, and W. T. Freeman, *Foundations of Computer Vision*. MIT Press, 2024.

[10] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, «Gradient-based learning applied to document recognition», *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[11] A. Krizhevsky, I. Sutskever, and G. E. Hinton, «Imagenet classification with deep convolutional neural networks», in *Advances in neural information processing systems*, vol. 25, 2012.

[12] 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.

[13] GeeksforGeeks, *What is image classification?*, https://www.geeksforgeeks.org/computer-vision/what-is-image-classification/, Accessed: 2026-03-16, 2024.

[14] S. Ren, K. He, R. Girshick, and J. Sun, «Faster r-cnn: Towards real-time object detection with region proposal networks», *Advances in neural information processing systems*, vol. 28, 2015.

[15] J. Long, E. Shelhamer, and T. Darrell, «Fully convolutional networks for semantic segmentation», in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3431–3440.

[16] S. Mohiuddin, *Street-level-semantic-segmentation*, `https://github.com/smohiudd/street-level-semantic-segmentation`, Accessed: 2026-03-16, 2019.

[17] 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.

[18] J. Solawetz, *What is instance segmentation? a guide. [2025]*, `https://blog.roboflow.com/instance-segmentation/`, Accessed: 2026-03-16, Nov. 2024.

[19] D. K. Prasad, D. Rajan, L. Rachmawati, E. Rajabally, and C. QuekJ, «Video processing in the maritime domain», in *Visual Object Tracking with Deep Learning*, Springer, 2017.

[20] Z. Shao and et al., «Sea-sky line detection in maritime video surveillance: A review», *IEEE Access*, vol. 8, 2020.

[21] D. D. Bloisi, L. Iocchi, A. Pennisi, and L. Tombolini, «Argos-venice: A video surveillance dataset for marine traffic monitoring», *IEEE Transactions on Intelligent Transportation Systems*, 2017.

[22] R. Girshick, J. Donahue, T. Darrell, and J. Malik, «Rich feature hierarchies for accurate object detection and semantic segmentation», in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 580–587.

[23] 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.

[24] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, «End-to-end object detection with transformers», in *European Conference on Computer Vision*, Springer, 2020, pp. 213–229.

[25] X. Zhu, W. Su, L. Lu, B. Li, X. Wang, and J. Dai, «Deformable detr: Deformable transformers for end-to-end object detection», in *International Conference on Learning Representations*, 2020.

[26] C.-Y. Wang, H.-Y. M. Liao, Y.-H. Wu, P.-Y. Chen, J.-W. Hsieh, and I.-H. Yeh, «Cspnet: A new backbone that can enhance learning capability of cnn», in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, 2020, pp. 390–391.

[27] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, «Feature pyramid networks for object detection», in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2117–2125.

[28] S. Liu, L. Qi, H. Qin, J. Shi, and J. Jia, «Path aggregation network for instance segmentation», in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 8759–8768.

[29] G. Jocher, A. Chaurasia, and J. Qiu, *Ultralytics YOLO*, version 8.0.0, Accessed: 2024-01-01, 2023. [Online]. Available: `https://github.com/ultralytics/ultralytics`.

[30] R. Padilla, S. L. Netto, and E. A. B. Da Silva, «A survey on performance metrics for object-detection algorithms», *International Conference on Systems, Signals and Image Processing (IWSSIP)*, pp. 237–242, 2020.

[31] S. J. Pan and Q. Yang, «A survey on transfer learning», *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, 2010.

[32] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, «How transferable are features in deep neural networks?», in *Advances in Neural Information Processing Systems (NeurIPS)*, 2014.

[33] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2nd. Springer Science & Business Media, 2009.

[34] scikit-learn developers, *Cross-validation: Evaluating estimator performance*, `https://scikit-learn.org/stable/modules/cross_validation.html`, Accessed: 2026-03-07, 2025.

[35] F. Pedregosa, G. Varoquaux, A. Gramfort, *et al.*, «Scikit-learn: Machine learning in python», *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[36] K. Tong, Y. Wu, and F. Zhou, «Recent advances in small object detection based on deep learning: A review», *The Visual Computer*, vol. 36, 2020.

[37] F. O. Unel, B. O. Ozkalayci, and C. Cigla, «The power of tiling for small object detection», in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, 2019.

[38] A. Medelyan, *Data leakage in machine learning: What it is and how to avoid it*, The Gradient, 2018.

[39] T.-H. Hsu and et al., «Data leakage and generalization in deep learning for video-based surveillance», *Journal of Real-Time Image Processing*, 2021.

[40] Q. Yin, X. Li, L. Sha, and Q. Li, «Deep learning-based object detection in maritime unmanned aerial vehicle imagery: Review and experimental comparisons», *IEEE Geoscience and Remote Sensing Magazine*, 2023.