BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ KỸ THUẬT

THÀNH PHỐ HỒ CHÍ MINH

NGÔ ĐỨC ĐẠT

NHẬN DẠNG VÀ PHÂN LOẠI TÀU TRONG CẢNH GIỚI BỜ BIỂN SỬ DỤNG TRÍ TUỆ NHÂN TẠO

LUẬN ÁN TIẾN SĨ

NGÀNH: KỸ THUẬT ĐIỆN TỬ

Tp. Hồ Chí Minh, tháng 4 năm 2026

BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ KỸ THUẬT

THÀNH PHỐ HỒ CHÍ MINH

NHẬN DẠNG VÀ PHÂN LOẠI TÀU TRONG CẢNH GIỚI BỜ BIỂN SỬ DỤNG TRÍ TUỆ NHÂN TẠO

LUẬN ÁN TIẾN SĨ

NGÀNH: KỸ THUẬT ĐIỆN TỬ

Người hướng dẫn khoa học 1: PGS.TS LÊ MỸ HÀ

Người hướng dẫn khoa học 2: PGS.TS NGUYỄN MẠNH HÙNG

Phản biện 1:

Phản biện 2:

Tp. Hồ Chí Minh, tháng 4 năm 2026

DANH MỤC CÔNG TRÌNH CỦA NGHIÊN CỨU SINH LIÊN QUAN ĐẾN ĐỀ TÀI

1. Image-Based Ship Detection Using Deep Variational Information Bottleneck. Sensors, 23(19), 8093.

https://doi.org/10.3390/s23198093

2. Transformer based ship detector: An improvement on feature map and tiny training set. EAI Endorsed Transactions on

Industrial Networks and Intelligent Systems, 12(1).

https://doi.org/10.4108/eetinis.v12i1.6794

3. Clustering based ship classification using radar signal and neuron network. In Proceedings of the 2021 International

Conference on System Science and Engineering (ICSSE) (pp. 122-127). IEEE.

https://doi.org/10.1109/ICSSE52999.2021.9538475.

4. A Vision-based Container-Code Checking System: Case Study at International Terminal *(*IWIS*)*

https://doi.org/10.1109/IWIS58789.2023.10284525

SENSORS Impact factor 2025

The **latest impact factor** of **SENSORS** and all the other Web of Science journals was released on 20th June 2024 by Clarivate. Through this web page, researchers can check the impact factor, total citation, journal quartile, and journal aim & scope.

SENSORS: Aim & Scope

SENSORS is a Web of Science indexed journal tha publishes research in the area: **INSTRUMENTS & INSTRUMENTATION|Q2|24/76**.
The **P-ISSN** of this journal is **N/A.**

Impact factor of SENSORS

Journal Title	SENSORS
Impact Factor	3.4
Journal Quartile	INSTRUMENTS & INSTRUMENTATION|Q2|24/76
E-ISSN	1424-8220
P-ISSN	N/A

Impact Factor from year 2018-2024

Year	Impact Factor
2018	2.475
2019	3.031
2020	3.275
2021	3.576
2022	
2023	3.9

Journals

- Agricultural and Biological Sciences
- Arts and Humanities
- Biochemistry, Genetics and Molecular Biology
- Business, Management and Accounting
- Chemical Engineering
- Chemistry
- Computer Science
- Decision Sciences
- Dentistry
- Earth and Planetary Sciences
- Economics, Econometrics and Finance
- Energy
- Engineering
- Environmental Science
- Health Professions
- Immunology and Microbiology
- Materials Science
- Mathematics
- Medicine

2024	3.4

The **Journal Impact Factor** is defined as citations to the journal in the JCR year to items published in the previous two years, divided by the total number of scholarly items, also known as citable items, (these comprise articles and reviews) published in the journal in the previous two years.

Impact factor data has a strong influence on the scientific community, affecting decisions on where to publish, whom to promote or hire, the success of grant applications, and even salary bonuses.

Only journals listed in the Science Citation Index Expanded (SCIE) and Social Sciences Citation Index (SSCI) receive an Impact Factor.

About SCI

SCI is a multidisciplinary citation index. Science Citation Index covers 9,000+ journals across 177 scientific disciplines. SCI was established in 1900 to the current. SCI-indexed journals are indexed by SCI and SSCI. SCI is described as the world's leading journal.

All journals covered in this database are reviewed for sufficiently high quality each year.

How to publish in SCI journals?

1. Identify an SCI indexed Journal.
2. Compose Your Research Paper As Per The Guidelines Of The Journal.
3. Visit the journal's website to submit your research paper.
4. Notification Of Acceptance From The Publisher.
5. Confirmation Of Your Paper Being Published

- Multidisciplinary
- Neuroscience
- Nursing
- Pharmacology, Toxicology and Pharmaceutics
- Physics and Astronomy
- Psychology
- Social Sciences
- Veterinary

Scopus Preview

Sources

ISSN | Enter ISSN or ISSNs | Find sources

ISSN: 1424-8220 ×

CiteScore 2024 has been released. View CiteScore methodology >

x

Filter refine list

Apply Clear filters

Display options

- [ ] Display only Open Access journals

Counts for 4-year timeframe

- ( ) No minimum selected
- ( ) Minimum citations
- ( ) Minimum documents

Citescore highest quartile

- [ ] Show only titles in top 10 percent
- [ ] 1st quartile
- [ ] 2nd quartile
- [ ] 3rd quartile
- [ ] 4th quartile

Source type

- [ ] Journals
- [ ] Book Series
- [ ] Conference Proceedings
- [ ] Trade Publications

Apply Clear filters

1 result

Download Scopus Source List | Learn more about Scopus Source List

All | Export to Excel | Save to source list

View metrics for year: 2023

	Source title	CiteScore	Highest percentile	Citations 2020-23	Documents 2020-23	% Cited
[ ] 1	Sensors *Open Access*	7.3	83% 24/141 Instrumentation	256,060	35,041	80

Top of page

Artificial Intelligence in Imaging Sensing and Processing

- **Print Special Issue Flyer**
- **Special Issue Editors**
- **Special Issue Information**
- **Keywords**
- **Benefits of Publishing in a Special Issue**
- **Published Papers**

A special issue of ***Sensors*** (ISSN 1424-8220). This special issue belongs to the section "**Intelligent Sensors**".

Deadline for manuscript submissions: **closed (31 July 2023)** | Viewed by 19907

Share This Special Issue

Special Issue Editor

Dr. Ching-Chun Huang **E-Mail** **Website**
Guest Editor
Department of Computer Science, National Yang Ming Chiao Tung University, Hsinchu, Taiwan
Interests: computer vision and multimedia system; intelligent control system; machine learning for signal processing; human–machine interface
Special Issues, Collections and Topics in MDPI journals

Special Issue Information

Dear Colleagues,

Many modern intelligent systems and applications have embedded deep-learning-based image sensing and processing. For instance, smartphones, surveillance cameras, UAVs, autonomous

original image, while the other did not. We also calculated the transformation matrix to obtain the real positions **[...] Read more.**
(This article belongs to the Special Issue **Artificial Intelligence in Imaging Sensing and Processing**)

► **Show Figures**

Open Access Article 16 pages, 17630 KiB

Progressively Unsupervised Generative Attentional Networks with Adaptive Layer-Instance Normalization for Image-to-Image Translation

by **Hong-Yu Lee**, **Yung-Hui Li**, **Ting-Hsuan Lee** and **Muhammad Saqlain Aslam**
Sensors **2023**, *23*(15), 6858; **https://doi.org/10.3390/s23156858** - 1 Aug 2023
Cited by 29 | Viewed by 3401

Abstract Unsupervised image-to-image translation has received considerable attention due to the recent remarkable advancements in generative adversarial networks (GANs). In image-to-image translation, state-of-the-art methods use unpaired image data to learn mappings between the source and target domains. However, despite their promising results, existing approaches **[...] Read more.**
(This article belongs to the Special Issue **Artificial Intelligence in Imaging Sensing and Processing**)

► **Show Figures**

Open Access Article 16 pages, 2632 KiB

Rethinking Feature Generalization in Vacant Space Detection

by **Hung-Nguyen Manh**
Sensors **2023**, *23*(10), 4776; **https://doi.org/10.3390/s23104776** - 15 May 2023
Cited by 1 | Viewed by 1916

Abstract Vacant space detection is critical in modern parking lots. However, deploying a detection model as a service is not an easy task. As the camera in a new parking is set up at different heights or viewing angles from the original parking lot **[...] Read more.**
(This article belongs to the Special Issue **Artificial Intelligence in Imaging Sensing and Processing**)

► **Show Figures**

Show export options

Displaying articles 1-7

sensors

MDPI

Article

Image-Based Ship Detection Using Deep Variational Information Bottleneck

Duc-Dat Ngo [1], Van-Linh Vo [1], Tri Nguyen [2], Manh-Hung Nguyen [1,*] and My-Ha Le [1,*]

[1] Faculty of Electrical and Electronics Engineering, University of Technology and Education, Ho Chi Minh City 7000, Vietnam; datnd.ncs@hcmute.edu.vn (D.-D.N.); 20139080@student.hcmute.edu.vn (V.-L.V.)
[2] Faculty of Information Technology, Industrial University of Ho Chi Minh City, Ho Chi Minh City 7000, Vietnam; tringuyen21072002@gmail.com
* Correspondence: hungnm@hcmute.edu.vn (M.-H.N.); halm@hcmute.edu.vn (M.-H.L.)

Abstract: Image-based ship detection is a critical function in maritime security. However, lacking high-quality training datasets makes it challenging to train a robust supervision deep learning model. Conventional methods use data augmentation to increase training samples. This approach is not robust because the data augmentation may not present a complex background or occlusion well. This paper proposes to use an information bottleneck and a reparameterization trick to address the challenge. The information bottleneck learns features that focus only on the object and eliminate all backgrounds. It helps to avoid background variance. In addition, the reparameterization introduces uncertainty during the training phase. It helps to learn more robust detectors. Comprehensive experiments show that the proposed method outperforms conventional methods on Seaship datasets, especially when the number of training samples is small. In addition, this paper discusses how to integrate the information bottleneck and the reparameterization into well-known object detection frameworks efficiently.

Keywords: ship detection; maritime security; information bottleneck

Citation: Ngo, D.-D.; Vo, V.-L.; Nguyen, T.; Nguyen, M.-H.; Le, M.-H. Image-Based Ship Detection Using Deep Variational Information Bottleneck. *Sensors* **2023**, *23*, 8093. https://doi.org/10.3390/s23198093

Academic Editor: Marco Leo

Received: 9 August 2023
Revised: 19 September 2023
Accepted: 20 September 2023
Published: 26 September 2023

1. Introduction

Detecting and tracking vessels are routine and pressing tasks due to considerations related to security, safety, and environmental management. The regulation V/19-1 of the 1974 SOLAS convention requires specific ships must be equipped with a Long-Range Identification and Tracking (LRIT) system [1]. However, the information is only transmitted automatically every six hours from the LRIT equipment installed on the ship via the Inmarsat satellite. Another solution for tracking and identifying ships is the automatic identification system (AIS) [2]. An automatic identification system (AIS) is an automated tracking system that displays other vessels in the vicinity. The broadcast transponder system operates in the VHF mobile maritime band; hence, the transceiver range is limited and depends on weather conditions. Moreover, the system can be turned off manually and can be easily tampered with. For the aforementioned reasons, each country has established a radar-based monitoring system [3] for overseeing its maritime regions. This system is robust in various environments and can detect multiple objects at far distances. However, the system can provide only the distance and bearing of the object, not information about each vessel category. Modern naval observation stations have recently been equipped with high optical zoom cameras to supplement the traditional radar system. These systems can support day and night vision with outstanding image quality. This allows using image information for maritime border security.

Detecting and categorizing ships from images are applications of object detection [4] in computer vision. Recently, deep learning is the most successful solution for these applications by training a detector. The early deep-learning-based detectors RCNN [5]

or FastRCNN [6] have a better accuracy compared with hand-crafted detectors [7], but they are not efficient enough to work in real-time. Later, one-stage detectors SDD [8] or YOLO [9] have been introduced to speed up the inference. These methods need anchor boxes and post-processing to detect objects. Recently, feature pyramid networks [10] and decoupled head [11] have been introduced to detect small objects with higher performance. In addition, the success of transformer [12] on natural language processing (NLP) also opens a new approach for object detection when transformer-based detectors [13,14] could obtain a comparable result with CNN-based detectors [8,11]. Transformer-based detectors do not require anchor boxes and post-processing for detection. Hence, it can work as an end-to-end training process.

Deep-learning-based detectors have reported many promising results. However, training a ship detector for maritime security is still an open question with several challenges. First, high-quality datasets may not be available for research. Satellite datasets such as [15–17] can provide many images for ship detection; however, cameras at naval observation stations are front-view cameras. The PASCAL VOC2012 database [18] provides front-view ship images, but the number of training samples is very limited. In the dataset, few ship objects are available, and all of them are categorized as a single "boat" class. Recently, a large-scale dataset [19] has been introduced. This dataset includes 31,455 images with six classes. However, only 7000 images are published for research purposes. The second challenge in ship detection is the complexity of environmental factors. The large-scale dataset [19] had reported several factors such as background selection, lighting environment, visible proportion, and occlusion have been reported. These above challenges raise a research question: Could we learn generalization features that focus on a ship and eliminate environmental factors based on small datasets?

Given a limited dataset, conventional methods use data argumentation to address these challenges by enriching the training dataset. In classification tasks, well-known data argumentation are listed as flipping, rotation, scaling, cropping, translation, and adding Gaussian noise have been integrated into training frameworks. In object detection tasks, Mosaic is a successful solution that has been introduced in YOLOv4 [20] and reused in later YOLO versions. However, these data argumentations may not model environmental factors such as background selection and visible proportion in an ocean scenario. Recently, Light_SDNet [21] enriched a large-scale ship dataset [19] by adding haze and rain to the original ship images.

In this paper, we address the challenge by co-designing two factors. First, we aim to learn features that focus only on ships and skip all background information. This will help to address the background selection challenges. Second, a reparameterization is used to enrich the dataset in feature space. Unlike conventional methods where the training data are increased in the image domain, we aim to add some uncertainty in the feature domain. Hence, the classifier can work with noise from the environment. The variational information bottleneck (VIB) [22] is used to select features that focus only on objects and eliminate background information. Later, these features add some uncertainty before feeding to a classification head. This approach could be integrated into any well-known detectors; however, this paper uses YOLOX [11] as our baseline due to its outperforming on testing datasets. As shown in Figure 1, the proposed method help to have a better performance, especially on small datasets. Also, heatmaps in Figure 1 prove that learned features focus on objects. It means the model may work better in a practical environment.

Conventionally, a detector includes a backbone, a neck, and a decoupled head that addresses regression and classification tasks. Given an input image x, the backbone extracts the feature $F^i(x)$ at different scales ith. For each feature $F^i(x)$, the neck module connects the feature to the corresponding classification head and regression head as in YOLOX [11]. We found that box regression is a critical task to ensure the success of a detector in training; hence, the regression head in [11] is reused to make the network converge smoothly. The modification is on the classification head. On this head, an encoder extracts the mean $\mu^i \in \mathbb{R}^{dW^iH^i}$, and the corresponding variance $\sigma^i \in \mathbb{R}^{dW^iH^i}$ of features $F^i(x)$. As the

classifier must work well with variant features, we sample latent features $z^i \sim \mathcal{N}(\mu^i, \sigma^i)$ from the mean and variance. Then, the classification result $y^i \in \mathbb{R}^{KW^iH^i}$ is predicted from sampled feature z^i by $y^i = cls(z^i; \theta^i)$. A good feature z must represent fine-grained features of ship categories. Hence, the mutual information [23] $I(y; z)$ should be maximized. Additionally, the background information from input x should be eliminated on feature space z. Hence, the mutual information $I(x; z)$ should be minimized. Two constraints are optimized together in a variational information bottleneck (VIB) loss [22]. Since this loss could be integrated into any supervised learning framework, it could be accompanied by regression and object losses to train a detector.

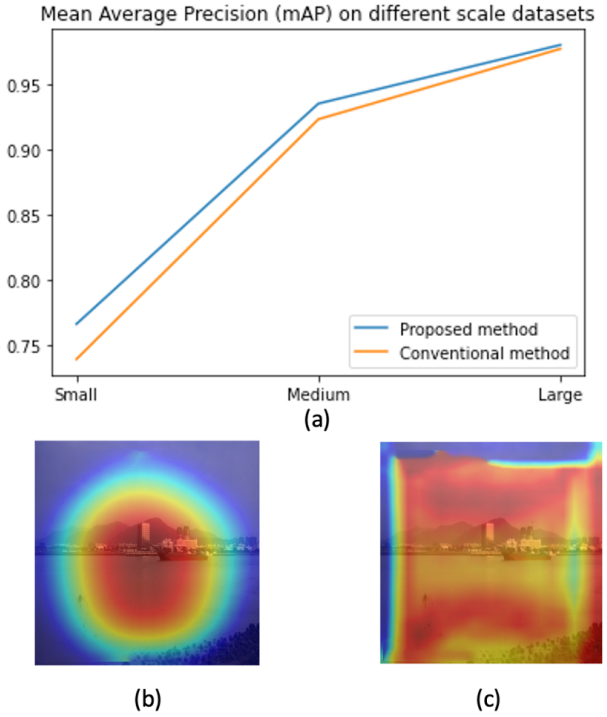


Figure 1. The contribution of the proposed method. (**a**) Performance on different scale datasets (**b**) A heat map by the proposed method (**c**) A heat map by a baseline method.

In summary, the paper's contributions are listed below:

1. Regularly, VIB and the parameterization trick are used in classification tasks. However, this paper discusses integrating these techniques into object-detection frameworks. The method outperforms SoTA in detecting ship objects, especially in small-scale datasets.
2. We carefully test the effect of VIB and parameterization at different positions in decoupled heads. The result shows that VIB can only work on the classification head and should not allow the VIB loss effect on the regression head.
3. A feature analysis proves that the proposed method could learn feature focus on objects rather than the background.

2. Related Works

2.1. Object Detection

Deep learning-based object detection has been developed rapidly in recent years. Early works [5,6,24–26] are considered as two-stage detectors because two processes are needed in an inference. First, a region that may involve an object is selected. The region is considered the location of an object on the image. Second, the region is cropped and fed to a classifier to estimate its categories. To detect small objects, the feature pyramid [10] had been used to extract features in multiple levels. The FPN has two pathways: a bottom-up

pathway which is a ConvNet computing feature hierarchy at several scales, and a top-down pathway that upsamples coarse feature maps from higher levels into high-resolution features. FPN is a region proposal network (RPN) in Faster R-CNN [25].

Another approach to solving the detection is single-state detectors [8,9,27–31]. These single-stage detectors directly predict the image pixels as objects and their bounding box attributes. YOLO [9] is the first representation of a single-stage detector. It can work very fast, but the accuracy is not high. Single Shot MultiBox Detector (SSD) [8] was the first method of single-stage detectors that matched the accuracy of contemporary two-stage detectors like Faster R-CNN [25]. RetinaNet [30] proposed a focal loss as the means to remedy the imbalance between background and foreground objects. The focal loss parameter reduces the loss contribution from easy examples. The authors demonstrate its efficacy with the help of a simple, single-stage detector. Later, CenterNet [29] models objects as points. As the predictions are points but not bounding boxes, non-maximum suppression (NMS) [9] is not required for post-processing. EfficientDet [31] introduces efficient multi-scale features (BiFPN) and model scaling. BiFPN is a bi-directional feature pyramid network with learnable weights for cross-connection input features at different scales. In addition, it jointly scales up all dimensions of the backbone network, BiFPN network, class/box network, and resolution. Therefore, this method achieves better efficiency and accuracy than previous detectors while being smaller and computationally cheaper.

The next generation of the YOLO family, such as YOLOv4 [20], and YOLOv5 [32], incorporated many exciting ideas to design a fast object detector that could work in existing production systems. Recently, YOLOX [11] has introduced the decouple head to separate the classification and regression tasks. It allows the detector to convert easily. Also, data arguments like Mosaic and Mixup have been introduced to increase accuracy.

Transformer [12] had been very successful in NLP [33]. Therefore, many works [13,14,34] have tried to apply the transformer concept to object detection. Transformers present a paradigm shift from CNN-based neural networks. While its application in vision is still nascent, its potential to replace convolution from these tasks is very real. The state-of-the-art transformer-based detectors have promising results on the COCO dataset [35], but utilize comparatively higher parameters than convolutional models.

2.2. Ship Detection

Several modifications of well-known object detection methods have been introduced to improve the performance of ship detectors. Liu_2022 [36] based on the SSD [8] framework and VGG backbone to detect a ship on small scales. The author [36] used a local attention network to fuse cross-features; also, a merge module combines features from different scales to improve detection results. The YOLO family is also used by many works to enhance the detection of ship datasets. Based on the YOLO framework, Biaohua_2022 [37] introduced a "Cross-level Attention and Ratio Consistency Network" (CARC) for ship detection. In this paper, the backbone was Resnet-34; the neck was a cross-level-attention module that used channel attention and spatial attention to extract features at different scales. The features were concatenated and fed to a head. Cui_2019 [38], Liu_2020 [39], and Li_2021 [40] based on YOLOV3 to detect ships. Cui_2019 [38] introduced YOLOv3-ship consisting of dimension clusters, network improvement, and Squeeze-and-Excitation(SE) module embedding. Liu_2020 [39] introduced two new anchor-setting methods and cross-feature fusion to enhance the performance of YOLOV3. Instead of using the FPN [10] to connect the backbone to a head, the method used a Cross PANet, which can combine the location information of the low-level feature maps with the semantic information of the high-level feature maps. Li_2021 [40] is based on YOLOV3 Tiny [28] to develop a two-training process. Here, CBAM attention [41] is used to detect large targets; later, a fine-tuning is made to detect small targets.

Recently, advanced versions of the YOLO framework were introduced for ship detection. Zhang_2021 [42] used YOLOV4 with a Reverse Depthwise Separable Convolution (RDSC) to detect ships. The proposed RDSC replaced the Depthwise Separable Convo-

lution (DSC) [43] in the ResUnit of the YOLOV4 backbone. With the help of RDSC, the complexity of the network model is reduced while ensuring accuracy. Han_2021 [44] also uses the YOLOV4 backbone with an attention mechanism to improve performance. Light_SDNet [21] modified the YOLO5 backbone by a Gost Unit [45] and DepthWise Convolution (DWConv) [46] to reduce the number of parameters; also, data augmentations like haze generation and rain generation have been introduced to enrich the training set. Recently, YOLOX has been considered a robust and powerful method for object detection; Zhang_2022 [47] used the YOLOX framework to design a lightweight method. Instead of using a PANnet [48] for feature fusion, the paper used a Lightweight Adaptive Channel Feature Fusion (LACFF) to overcome the inconsistent scale of feature maps. The features of all other layers are adjusted to the same shape. Afterward, the channels are fused according to the learned weights. Similar to Zhang_2022 [47], our work is also based on YOLOX; however, we do not focus on feature fusion but introduce a loss that selects suitable features on the classification head.

Transformer-based methods [13] are also a possible solution for ship detection. Yani_2022 [49] used distillation learning to train a DETR-based ship detector. A teacher model was trained based on a large-scale CoCo Dataset, and the student model was fine-tuned based on the Seaship dataset [19]. The method helps to reduce the FLOPs and number of parameters. However, its mAP is not improved compared with the conventional DETR framework.

3. Proposed Method

3.1. Overview System

Table 1 summarizes mathematic notations in the paper, and Figure 2 introduces the concept of the proposed method. Given a backbone, features at different scales are extracted. Here, we use the Darknet53 backbone [28] to extract features at multiple scales. The PAFPN [48] serves as a neck that connects these features to a decoupled head. Detail of the backbone and the neck are introduced in Section 3.3. The decouple head includes a classification head and a regression head. While the classification head aims to classify a ship category, the regression head estimates a relative bounding box and the object ability for each cell. On the classification branch, we use $1 * 1$ kernels to extract the $\mu_j \in \mathbb{R}^d$ and $\sigma_j \in \mathbb{R}^d$ at the j^{th} position on a feature map. Using these kernels, tensors $\mu \in \mathbb{R}^{dxHxW}$ and $\sigma \in \mathbb{R}^{dxHxW}$ are obtained. These tensors are used to estimate the VIB loss [22]; additionally, a reparameterization process samples a new latent $z_j \in \mathbb{R}^d$ for the j^{th} position. A classifier takes the latent $z \in \mathbb{R}^{dHW}$ and predicts the vessel category $\hat{y}_{cls} \in \mathbb{R}^{KHW}$. The detail of the VIB module is described in Table 2. The kernel size is $(1,1)$ means that the $Encoder_\mu$ extracts cross exchange-feature but does not change the size of feature maps. It allows us to reuse the original classification head.

Table 1. Mathematical Notation..

Notation	Description
x, y, z	The input, the output, and the latent feature of the network.
i	The index of scale level.
j	The index of position on a feature map.
d	The dimension of latent vectors in VIB module
$\mu^i \in \mathbb{R}^{dH^iW^i}, \sigma^i \in \mathbb{R}^{dH^iW^i}$	The latent feature and its corresponding variance at the ith scale.
$\mu_j \in \mathbb{R}^d, \sigma_j \in \mathbb{R}^d$	The feature and its corresponding variance at the position jth in a map
$y_{cls}, \hat{y}_{cls}$	The classification ground truth and output.
$y_{reg}, \hat{y}_{reg}$	The box ground truth and output.
$y_{object}, \hat{y}_{object}$	The object ground truth and output.

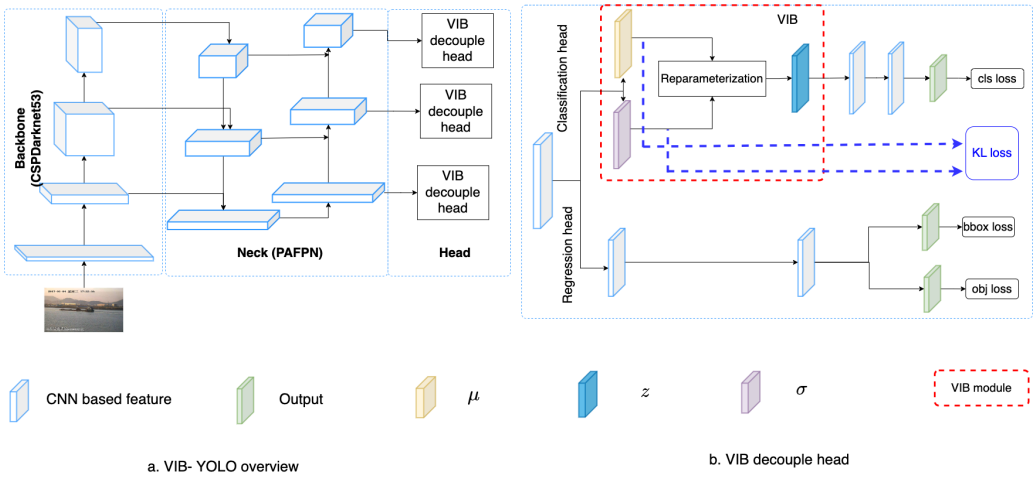


Figure 2. Network structure. (**a**) The overview of proposed VIB-based object detection. (**b**) The proposed VIB-based classification head.

Table 2. Network detail. Here, $Encoder_{\mu}$ means the encoder to extract μ, $Encoder_{\sigma}$ means the encoder to extract σ, i is the index of the scale level, and C^i is the number of channels in the input of the ith level.

Block	Layer	Parameters
$Encoder^i_{\mu}$	nn.conv	Size=(1,1), In = C^i, Out = C^i
$Encoder^i_{\sigma}$	nn.conv	Size=(1,1), In = C^i, Out = C^i
Re-parametrize	$z = \mu + \epsilon * \sigma$	$\epsilon \sim \mathcal{N}(0, \sigma^2)$

A YOLO detector [11] addresses the bounding box regression, object classification, and category classification at the same time. In our work, the IoU loss ($L_{box}(\hat{y}_{box}, y_{box})$) helps to train the bounding box regression, and the cross-entropy loss helps to train the object classification and the category classification. In addition, the VIB loss also helps to select features by introducing a feature selection loss $L_{KL}(\mu, \sigma)$. α_{box}, α_{obj}, α_{cls}, α_{KL} are hyper-parameters that control the contribution of $L_{box}(\hat{y}_{box}, y_{box})$, $L_{obj}(\hat{y}_{obj}, y_{obj})$), $L_{cls}(\hat{y}_{cls}, y_{cls})$, and $L_{KL}(\mu, \sigma)$, respectively, the loss function in Equation (1) is used to train the detector.

$$L(\hat{y}, y) = \alpha_{box} L_{box}(\hat{y}_{box}, y_{box}) + \alpha_{obj} L_{obj}(\hat{y}_{obj}, y_{obj}) + \alpha_{cls} L_{cls}(\hat{y}_{cls}, y_{cls}) + \alpha_{KL} L_{KL}(\mu, \sigma). \quad (1)$$

Recently, the IoU loss (L_{box}) has been recognized by many researchers as a good solution to evaluate a predicted bounding box. The IoU loss helps the model to improve the quality of its bounding box predictions by penalizing boxes that do not closely match the ground truth in terms of overlap. It is crucial for achieving accurate object localization in object detection tasks. Equation (2) explains the concept of L_{IoU}.

$$L_{\text{IoU}}(\hat{y}_{box}, y_{box}) = \sum_{j=1}^{WH} w_j^{box} (1 - IoU_j) \quad (2)$$

where:

- W, H is the width and height of the output.
- IoU_j is intersection over union between the predicted box $\hat{y}_{box}$ and the ground-truth box y_{box} at the position j^{th}.
- w_j^{box} is a mask that decides which locations will be used to compute the loss.

Object classification loss ($L_{obj}(\hat{y}_{obj}, y_{obj})$) is concerned with identifying whether there is any object present within a bounding box. It is a binary classification as in Equation (3), where the model predicts a probability score indicating whether an object is present or not in each bounding box. Category classification loss ($L_{cls}(\hat{y}_{cls}, y_{cls})$) is focused on determining

the specific class or category of the object if one is found. Category classification is a multi-class classification problem, where the model predicts the probability distribution over different object categories for each bounding box as in Equation (4).

$$L_{obj}(\hat{y}_{obj}, y_{obj}) = \sum_{j=1}^{WH} w_j^{obj} \left(t_j \cdot \log(p_j) + (1 - t_j) \cdot \log(1 - p_j)\right) \tag{3}$$

where:

- p_j is the predicted objectness probability.
- $t_j$is the ground-truth objectness label.
- w_j^{obj} is a mask that decides which locations will be used to compute the loss.

$$L_{cls}(\hat{y}_{cls}, y_{cls}) = \sum_{j=1}^{WH} w_j^{cls} \sum_{k=1}^{C} t_{jk} \cdot \log(p_{jk}) \tag{4}$$

where:

- C is the number of object classes.
- p_{jc} is the predicted class probability (usually obtained through softmax activation).
- t_{jc} is the ground-truth class label for the j-th location and the c-th class.
- w_j^{cls} is a mask that decides which locations will be used to compute the loss.

Finally, the feature selection loss is shown in Equation (5). A detailed explanation of the loss will be introduced in Section 3.2.

$$L_{KL}(\mu, \sigma) = KL\Big(p(z \mid x)||q(z)\Big) = \frac{1}{d} \sum_{j=1}^{WH} \sum_{k=1}^{d} (\mu_{j,k}^2 + \sigma_{j,k}^2 - 2log(\sigma_{j,k}) - 1). \tag{5}$$

where:

- d is the dimension of latent features.

3.2. Feature Section Loss

Feature selection involves the process of choosing pertinent features tailored to a particular task. Drawing from the principles of information bottleneck theory [50], optimal features are concise representations that contain precisely the necessary information to address the task without redundancy. The necessity for this can be elucidated through the following two constraints:

- The latent z must help to well predict the output y (vessel categories);
- Given the latent z, we cannot infer input x very well.

In the realms of probability theory and information theory, the interrelation between two variables finds measurement through mutual information ($I(.)$) [23]. Consequently, these dual constraints are formulated by maximizing the mutual information $I(y; z)$ while minimizing the mutual information $I(x; z)$. The former constraint signifies that z aids in predicting vessel categories y, while the latter constraint signifies that z does not possess the capability to deduce the input image x.

Let β represent a Lagrange multiplier; the optimization problem is depicted in Equation (6). A better solution makes L_{IB} have a greater value.

$$L_{IB} = I(y; z) - \beta I(x; z). \tag{6}$$

The mutual information $I(.)$ [23] can gauge the information of one variable in relation to another variable by using Equation (7).

$$\begin{aligned} I(X;Y) &= I(Y;X) \\ &= H(X,Y) - H(X|Y) - H(Y|X) \\ &= H(X,Y) - H(X|Y) - H(Y|X) \\ &= KL\Big(p(x,y)||p(x)p(y)\Big)_{x\in X, y\in Y} \\ &= \mathbb{E}_{(x,y)\sim p(x,y)}\left[\log\frac{p(x,y)}{p(x)p(y)}\right] \\ &= \int p(x,y)\log\frac{p(x,y)}{p(x)p(y)}dxdy. \end{aligned} \tag{7}$$

Thus, $I(y;z)$ and $I(x;z)$ are expressed in Equation (8) and Equation (9), respectively.

$$\begin{aligned} I(y;z) &= \int p(y,z)\log\frac{p(y,z)}{p(y)p(z)}dydz = \int p(y,z)\log\frac{p(y\mid z)}{p(y)}dydz \\ &= \int p(y,z)\log p(y\mid z)dydz - \int p(y)\log p(y)dy. \end{aligned} \tag{8}$$

$$I(x;z) = \int p(x,z)\log\frac{p(x,z)}{p(x)p(z)}dxdz = \int p(x,z)\log\frac{p(z\mid x)}{p(z)}dxdz. \tag{9}$$

To maximize the mutual information $I(y;z)$, we approximate this term by a lower bound. When the lower bound obtains a greater value, the $I(y;z)$ has a greater value. $q(y\mid z)$ is a variational approximation of $p(y\mid z)$; the lower bound is founded by Kullback–Leibler divergence as in Equation (10).

$$KL\Big(p(y\mid z)||q(y\mid z)\Big) \geq 0 \Longrightarrow \int p(y\mid z)\log p(y\mid z)dy \geq \int p(y\mid z)\log q(y\mid z)dy. \tag{10}$$

By incorporating the lower bound from Equation (10), the expression for $I(y;z)$ in Equation (8) can be reformulated as Equation (11).

$$I(y;z) \geq \int p(y,z)\log q(y\mid z)dydz - \int p(y)\log p(y)dy \tag{11}$$

In this context, the entropy of the labels $H(y) = -\int p(y)\log p(y)dy$ is considered independent and can be disregarded. As a result, the maximum value of $I(y;z)$ is approximated as shown in Equation (12).

$$\begin{aligned} I(y;z) &\approx \int p(z\mid x)p(y\mid x)p(x)\log q(y\mid z)dxdydz \\ &\approx \int p(z\mid x)p(y,x)\log q(y\mid z)dxdydz. \end{aligned} \tag{12}$$

To minimize the mutual information $I(x;z)$, we approximate this term (Equation (9)) by an upper bound. When the upper bound obtains a smaller value, the $I(x;z)$ has a smaller value. We denote $q(z)$ as a variational approximation to the marginal $p(z)$. Using the KL divergence, the upper bound of $I(x;z)$ is introduced as Equation (13).

$$KL\Big(p(z)||q(z)\Big) \geq 0 \Longrightarrow \int p(z)\log p(z)dz \geq \int p(z)\log q(z)dz. \tag{13}$$

Utilizing the upper bound in Equation (9), we can re-express $I(x;z)$ as presented in Equation (14).

$$\begin{aligned} I(x;z) &= \int p(x,z)\log p(z \mid x)dxdz - \int p(z)\log p(z)dz \\ &\leq \int p(x,z)\log p(z \mid x)dxdz - \int p(z)\log q(z)dz \\ &= \int p(x)p(z \mid x)\log \frac{p(z \mid x)}{q(z)}dxdz \\ &= \int p(x,y)p(z \mid x)\log \frac{p(z \mid x)}{q(z)}dxdzdy. \end{aligned} \tag{14}$$

Through the utilization of the lower bound for $I(y;z)$ and the upper bound for $I(x;z)$, the Lagrangian function in Equation (6) can be approximated as represented in Equation (15).

$$\begin{aligned} L_{IB} &= I(y;z) - \beta I(x;z) \\ &\approx \int p(z \mid x)p(y,x)\log q(y \mid z)dxdydz - \beta \int p(z \mid x)p(x,y)KL\Big(p(z \mid x)||q(z)\Big)dxdydz \\ &= \mathbb{E}_{(x,y)\sim p(x,y),z\sim p(z|x)}\Big[\log q(y \mid z) - \beta KL\Big(p(z \mid x)||q(z)\Big)\Big]. \end{aligned} \tag{15}$$

In our application, the term $q(y \mid z)$ is modeled by a classifier; and $\log q(y \mid z)$ is a classification loss $L_{cls}(\hat{y}_{cls}, y_{cls})$. In addition, the latent z could be sampled from a reparameterization trick $g(\epsilon, x)$ where $\epsilon \sim p(\epsilon) = \mathcal{N}(0, I)$. Hence, z is estimated by Equation (16).

$$z = \mu + \epsilon * \sigma. \tag{16}$$

Using Equation (16), the term $p(z \mid x)$ is estimated by Equation (17). $q(z) = \mathcal{N}(0, I)$, and the term $KL\Big(p(z \mid x)||q(z)\Big)$ is estimated by Equation (5). In addition, the term $KL\Big(p(z \mid x)||q(z)\Big)$ could serve as a feature selection loss $L_{KL}(\mu, \sigma)$ in Equation (1). Hence, the parameter β is replaced by the parameter α_{KL}. Equation (5) represents $L_{KL}(\mu, \sigma)$ and it is applied at every scale level with classification loss, box loss, and object loss.

$$p(z \mid x) = \mathcal{N}(\mu, \sigma^2), \tag{17}$$

3.3. Backbone and Neck Module

The proposed method could be integrated with any backbone. However, a modification is needed on the neck to match the selected backbone and the decoupled head. We have tried several backbones in Section 4.5 and pointed out that the Darknet backbone and PAFPN neck can perform better than others.

Detail of the Darknet and PAFPN are correspondingly in Figures 3 and 4. Here, the Darknet backbone used CSPLayer to extract features. The features at 2nd, 3rd, and 4th CSPlayer are used in the PAFPN neck. Finally, output features are used with decoupled heads at different scales.

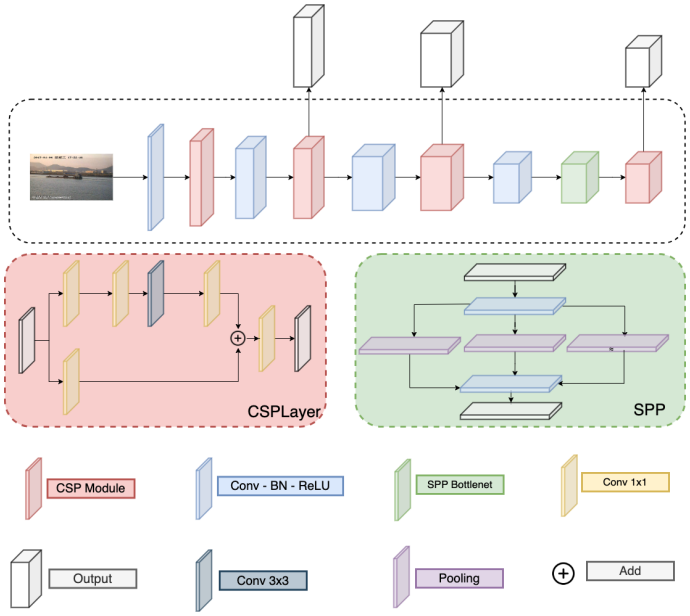


Figure 3. Darknet structure.

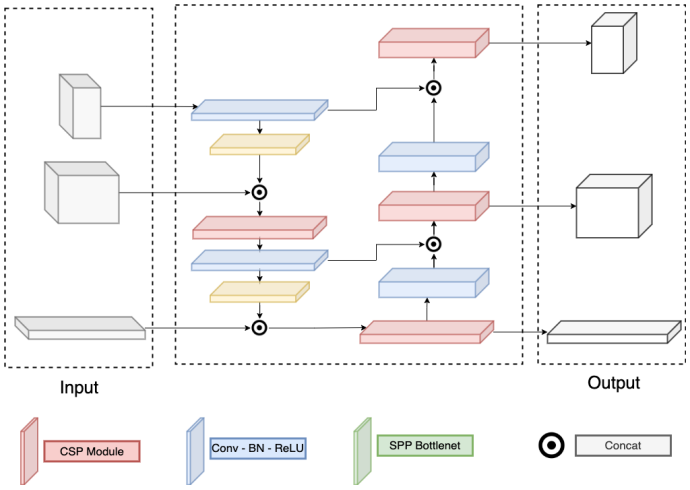


Figure 4. PAFPN structure.

4. Experimental Results

4.1. Datasets and Experiment Setting

This paper evaluates our proposed method using the SeaShips dataset [19]. The SeaShips dataset is built based on the images captured by an in-field video monitoring system deployed around Hengqin Island, Zhuhai City, China. Each camera records the scene from 6:00 a.m. to 8:00 p.m. every day. The images are obtained from one-minute video clips, resulting in 60 clips per hour. Each clip spans 60 s and contains approximately 1500 frames. For every 50 frames (approximately every two seconds), one image is extracted for inclusion in the dataset. This dataset encompasses six distinct ship types, namely bulk carriers, ore carriers, ordinary cargo ships, container ships, passenger ships, and fishing boats. It features a diverse background environment, characterized by varying light intensities to ensure dataset diversity. Additionally, the dataset incorporates numerous special scenarios, including instances of target occlusions. The SeaShips dataset is highly regarded within the research community for its professionalism, public availability, large-scale nature, and accuracy in ship labeling. It has earned a strong reputation for facilitating effective ship inspection procedures. While dataset [19] has 31,455 images, only 7000 images are published for research at [21,38]. Hence, we use the published dataset for our experiments.

Literature reviews show that many research works use 80% of the published data for training/validating and 20% for testing. Hence, we select D_1^{Train}, which includes 5600 images for training, and D_1^{Test}, which includes 1400 images for testing. In addition, recent works [37,49] also use a more challenging setting where 50% of the data are the training set, and the rest is the testing set. We also follow this setting to prepare D_2^{Train} and D_2^{Test} for comparison. To evaluate the performance on a very small dataset, we randomly select several subsets S_1, S_2, S_3 that include 30%, 70%, and 100% of samples from D_2^{Train} for training in later experiments.

Our experiment uses SGD optimizer, learning rate = 0.01, weight decay = 0.001, n_epoch = 200, and batch_size = 8. We use the reduce-mean operator on batch_size, and the reduce-sum operator on prediction output. The mAP is used to select the best model. The loss function in Equation (1) use $\alpha_{box} = 10.0, \alpha_{obj=}1.0, \alpha_{cls} = 1.0$ and the $\alpha_{KL} = 0.125$.

4.2. Select the Hyper-Parameter

This section aims to select suitable hyperparameters for our training process. The major contribution of our work is introducing the feature selection loss $L_{KL}(\mu, \sigma)$ to the YoloX framework. Therefore, our first experiment is selecting a suitable hyper-parameter α_{KL} in Equation (1). A small α_{KL} may not help to learn better features, whereas a large α_{KL} may focus too much on feature learning and forget the main task. In this experiment, the S_2 dataset is used for training, and theD_2^{Test} dataset is used for testing. The mAP metric on six classes is used for comparison. Figure 5 shows how the KL loss affects the result. Without the feature selection, the mAP is only 0.923; when α_{KL} is 0.05, the mAP increases to 0.928. The higher the α_{KL}, the higher our mAP. However, when $\alpha_{KL} = 0.15$, the mAP begins to be reduced; and if $\alpha_{KL} = 0.2$ the mAP is 0.914. The mAP based on $\alpha_{KL} = 0.2$ is smaller than when the mAP without feature loss. This phenomenon is because the feature loss reduces the features selected for the main task. When the feature is reduced too much, the classifier may not have enough information for the classification task. In the next experiments, we select the $\alpha_{KL} = 0.125$ for our proposed method.

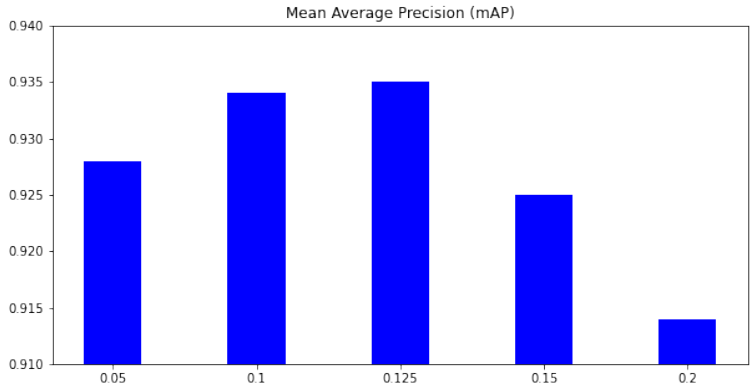


Figure 5. The mean average precision (mAP) on different α_{VIB}. x-axis means the α_{KL} parameter, and y-axis is the mean average precision (mAP) over all classes.

The next experiment aims to evaluate the contribution of other hyperparameters on the performance. In Equation (1), a hyper-parameter controls the contribution of its corresponding loss to the training process. Therefore, a greater hyperparameter will force the algorithm to learn this task first. By adjusting the hyper-parameter settings, we can control the steps of the learning process. For instance, if we want to train the category classification task before other tasks, the hyper-parameter setting should be $\alpha_{cls} = 10$ and $\alpha_{box} = \alpha_{obj} = 1$. Table 3 shows all settings in detail. The α_{KL} hyperparameter is not included in this table because the feature selection task is an auxiliary task and must be learned lastly after other tasks. By default, we set $\alpha_{KL} = 0.125$.

Table 3. Hyperparameter settings that control the algorithm steps.

Scenario	α_{box}	α_{obj}	α_{cls}
L_{box} first	10	1	1
L_{obj} first	1	10	1
L_{cls} first	1	1	10

A comparison between the three scenarios is presented in Table 4. Here, the D^1_{Train} dataset is used for training, and the D^1_{Test} dataset is used for testing. The results show that changing steps of the learning process may not affect the performance too much. Because all component losses are combined into a unique loss by Equation (1), the algorithm will automatically focus on the task that may not work well and ensure all tasks can be learned at the end of a training process. However, in object detection, category classification is conditional on a predicted bounding box. Therefore, from literature reviews, a greater α_{box} can provide better performance. As shown in Table 4, the mAP is 0.989 if we focus on L_{box} first. This value is slightly greater than the performance by learning L_{obj} or L_{cls} first.

Table 4. Performance comparisons of hyperparameter settings. The best results are marked in bold.

Scenario	Fishing Boat	Container Ship	Ore Carrier	Bulk Cargo Carrier	Passenger Ship	General Cargo Ship	mAP
L_{box} first	**0.977**	**0.999**	**0.994**	**0.994**	**0.982**	**0.987**	**0.989**
L_{obj} first	0.967	0.987	0.992	0.992	0.942	0.987	0.978
L_{cls} first	0.966	0.988	0.984	0.990	0.959	0.990	0.979

4.3. Compare with SoTA

This section compares our proposed method with SoTA on the mAP metric. There are several experiment settings from different works. Given 70,000 published images from SeaShip [19], Zhang_2022 [47], and Zhang_2021 [42] used 90% of data for training and validating; the other 10% of data are the testing dataset. Liu_2020 [39], Liu_2022 [36], Han_2021 [44], and Light_SDNet [21] use 80% of data for training and validating dataset, the other 20% is the testing dataset. To compare with these works, we use D^{Train}_1 for training and D^{Test}_1 for testing. The result in Table 5 shows that our method is better than other methods in terms of average precision. There are two reasons for this improvement. First, our method is based on the YOLOX framework, which is the recent SoTA frame for object detection. Light_SDNet is based on YOLO5, and its result is also promising. Liu_2020 [39] and Liu_2022 [36] are based on older versions in the YOLO family; hence, the performance is smaller than the SoTA framework as Light_SDNet and our method. Second, the parameterization adds some uncertainty to the training process. It allows the model to work with more data during the training phase. In comparison, Light_SDNet [21] also adds more haze and rain to the original image and gets a very good result (mAP = 0.988%). The major difference between our method and Light_SDNet [21] is how we add noise to training data. While Light_SDNet [21] adds noise to the image domain, our method adds noise to the feature domain. Last but not least, the VIB Loss learns features that focus on the object and remove redundant features in the background. Feature analysis in Section 4.4 will visualize feature maps in detail.

In addition, Biaohua_2022 [37] and Yani_2022 [49] used 50% of published images for training and the rest 50% for testing. Hence, we use D^{Train}_2 and D^{Test}_2 for training and testing correspondingly. As shown in Table 5, the mAP on previous works is up to 0.965%. Our proposed method can achieve significantly better performance than previous works. It proves the benefit of our method when the number of training samples is reduced. The primary driving force behind this improvement is the adoption of our method, which builds upon the robust YoloX framework for object detection. It is worth noting that ship

detection research typically leverages an object detection framework as its foundation, often with some custom modifications. Therefore, inheriting the capabilities of such a novel and powerful framework naturally leads to improved results.

Table 5. Performance comparisons of SoTA methods. The best results are marked in bold.

Method	Train + Val/Test (in %)	Fishing Boat	Container Ship	Ore Carrier	Bulk Cargo Carrier	Passenger Ship	General Cargo Ship	mAP
Zhang_2022 [47]	90/10	0.824	0.940	0.859	0.915	0.787	0.914	0.873
Zhang_2021 [42]	90/10	-	-	-	-	-	-	0.946
Liu_2020 [39]	80/20	-	-	-	-	-	-	0.908
Liu_2022 [36]	80/20	-	-	-	-	-	-	0.964
Han_2021 [44]	80/20	-	-	-	-	-	-	0.906
Light_SDNet [21]	80/20	**0.986**	0.995	**0.989**	0.990	0.982	0.989	0.988
Proposed method	80/20	0.979	**1**	0.987	**0.994**	**0.994**	**0.993**	**0.991**
Yani_2022 (ESDT) [49]	50/50	-	-	-	-	-	-	0.593
Yani_2022 (DETR) [49]	50/50	-	-	-	-	-	-	0.965
Biaohua_2022 [37]	50/50	0.940	0.987	0.966	0.978	0.937	0.972	0.963
Proposed method	50/50	0.970	0.986	0.984	0.991	0.964	0.989	**0.98**

4.4. Contribution of the VIB Loss on Small Datasets

In this section, we discuss the contribution of VIB loss on different scale datasets. The S_1 (525 images), S_2 (1225 images), and S_3 (3500 images) datasets are used for training. The testing dataset D_2^{Test} contains 3500 images. The results in Table 6 show that the VIB loss helps improve performance significantly on small datasets. If the training dataset is S_1, the mAP improves 3%. When the number of training samples is increased, the improvement is reduced. The enhancement on mAP is 1.2% if S_2 is used for training, and if the training dataset is S_3, the mAPs from both settings are quite equivalent. This phenomenon is reasonable because an unsupervised loss may help avoid overfitting on small datasets, and a reparameterization allows a classifier to be robust.

To clearly explain the benefit of the proposed method on feature learning, we compare the feature learned by our method (with VIB) and the baseline method (without VIB) using the S_3 dataset. Features before the last layer of necks and heads are extracted and visualized. We select 20 feature maps with the highest response scores for each scale level. We denote j as a pixel on a feature map F, which has size (W, H); the score of the feature map is $\frac{1}{WH}\sum_{j=1}^{WH} F_j$. These feature maps are accumulated together to formulate a unique response map. The result could represent important pixels on input images.

Figure 6 shows the heat map corresponding to an input image. The first row represents feature maps with VIB; the second row represents those without VIB. Since features are extracted at three scale levels, three responses are provided for head modules. The results show that VIB loss can learn features that focus on the object. Without VIB, the response likes a uniform distribution. With VIB, feature responses focus around the object but not all pixels. The phenomenon is also repeated at the neck module. It means the VIB loss can be backpropagated to the neck level and learn a better feature.

In addition, we also evaluate the sparse level and the discriminate level of feature maps. A sparse feature map means many values in the feature map are close to zeros. If a feature map is more sparse, it means the learned filters are not responding to patterns that do not contribute to the prediction process. Also, a sparse feature map means we only select a few features. A discriminate level is the difference between the maximum value and the minimum value in a feature map. A greater discrimination level means some positions have a high response, whereas others have a low response. Hence, the learned filters can strongly respond to useful patterns rather than other patterns. Given a feature map $F \in \mathbb{R}^{WH}$ where j is a position on the map, the sparse level is estimated by $\sum_{j=0}^{WH} |F_j < thre|$, and the discriminate level is $max(F) - min(F)$.

Table 6. Performance on small datasets. S_1 means 30% training samples, S_2 means 70% training samples, S_3 means 100% training samples from D_2^{Train}. The best results are marked in bold.

Method	Metrics	Fishing Boat	Container Ship	Ore Carrier	Bulk Cargo Carrier	Passenger Ship	General Cargo Ship	mAP
	dets	2849	1272	5374	3581	774	3598	
S_1 with VIB	recall	0.878	**0.941**	**0.924**	**0.913**	**0.657**	**0.946**	
	AP	0.796	0.884	**0.831**	**0.765**	**0.524**	**0.794**	**0.766**
	dets	3201	884	4654	2851	696	2944	
S_1 without VIB	recall	**0.892**	0.920	0.923	0.876	0.637	0.940	
	AP	**0.805**	**0.890**	0.822	0.710	0.466	0.743	0.739
	dets	1863	584	2012	1653	354	1129	
S_2 with VIB	recall	**0.940**	**0.984**	0.962	**0.971**	0.891	0.964	
	AP	**0.922**	**0.980**	**0.935**	**0.953**	0.873	**0.946**	**0.935**
	dets	2324	674	2848	1923	570	1585	
S_2 without VIB	recall	0.936	0.975	0.964	**0.963**	**0.899**	**0.978**	
	AP	0.903	0.970	0.932	0.928	0.862	0.945	0.923
	dets	1544	466	1562	1341	297	948	
S_3 with VIB	recall	**0.978**	0.986	0.990	**0.995**	**0.972**	**0.993**	
	AP	**0.970**	0.986	0.984	**0.991**	**0.964**	**0.989**	**0.98**
	dets	1574	470	1561	1360	294	883	
S_3 without VIB	recall	0.965	**0.989**	**0.995**	**0.993**	0.960	**0.993**	
	AP	0.957	**0.988**	0.990	0.987	0.953	**0.989**	0.977

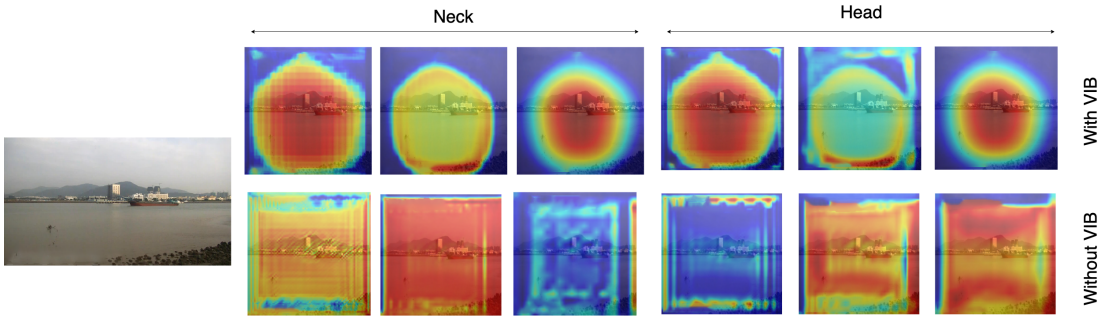


Figure 6. Heatmaps on the classifier head and the neck.

Utilizing the input image depicted in Figure 6, we extract feature maps from the final layer of the classification head. The statistics regarding the sparsity and discriminative characteristics of these feature maps are presented in Table 7. The findings indicate that VIB generates feature maps with increased sparsity. This outcome can be attributed to the influence of the object function $L_{KL}(\mu, \sigma)$, which enforces a zero mean on the features. Furthermore, the higher discriminative level observed in the VIB-based results underscores the robustness of the learned features.

Behind the reasoning experiments, the computational cost has been introduced. Table 8 represents the frame per second (FPS), the Giga-Floating-Point Operations Per Second (GFlops), and the number of parameters of the model. Among them, FPS measures the speed of the model, GFlops evaluates the performance of hardware when running deep learning workloads, and the number of parameters represents the size of the model. The result in Table 8 shows that adding VIB to the network does not increase the computational cost too much. The number of parameters is slightly increased because the VIB module has been added to the network. However, the FPS and GFlops are quite similar in both cases. This means the additional VIB module did not increase the computational cost.

Table 7. Statistics of sparse level and discriminate level on feature maps.

	With VIB		Without VIB	
	Sparse ↑	**Discriminate ↑**	**Sparse ↑**	**Discriminate ↑**
mean	194.44	32.45	0.183	0.587
std	136	59.557	0.486	0.599
min	0	0.277	0	0.203
percentiles (25%)	0	0.6698	0	0.203
percentiles (50%)	270	0.6698	0	0.3648
percentiles (75%)	304	34.011	0	0.5318
max	335	281.06	3	4.2183

Table 8. A comparison between computation cost with and without using VIB.

	With VIB	**Without VIB**
FPS	12.39	12.45
GFlops	9.92	9.91
# parameters (M)	55.33	54.15

4.5. Effect of Backbone

This section discusses how the proposed method works with different backbones. ResNet, MobileNetv2, and DarkNet are used as backbones for comparison. The input channel of the necked is adapted to meet the output of these backbones. The average precisions (AP) for six classes are shown in Table 9. In this experiment, S_2 serves as a training dataset. The result shows that DarkNet is the best backbone among these pre-trained models. This is reasonable because DarkNet had been recognized as the best backbone in the YOLO family. In addition, the increment given by VIB loss on ResNet [51] is 5.9 %. It means VIB loss can help a lot with some particular backbone.

Table 9. A mAP comparison with and without VIB on different backbones.

Backbone	**With VIB**	**Fishing Boat**	**Container Ship**	**Ore Carrier**	**Bulk Cargo Carrier**	**Passenger Ship**	**General Cargo Ship**	**mAP**
ResNet-18	Yes	0.837	0.978	0.861	0.886	0.741	0.931	**0.873**
	No	0.817	0.961	0.824	0.735	0.706	0.841	0.814
DarkNet	Yes	0.922	0.980	0.935	0.953	0.873	0.946	**0.935**
	No	0.903	0.970	0.932	0.928	0.862	0.945	0.923
MobileNetv2	Yes	0.895	0.975	0.895	0.880	0.794	0.908	**0.891**
	No	0.872	0.965	0.918	0.902	0.772	0.914	0.890

4.6. Effect of Pre-Processing Methods

Our approach enhances the classifier head by introducing a degree of uncertainty to the extracted features. However, it is worth noting that introducing uncertainty into the image domain has been explored in previous works [52–54]. For instance, in the Seg-based method [52], researchers trained their model using segmentation images. We have incorporated a similar approach into our ship dataset, generating segmentation images to create a new dataset for training our ship detector. Additionally, the NoiBased approach [53,54] method enriches datasets by introducing noise to input images and employing denoising techniques to bolster system robustness. Drawing inspiration from this observation, we

conducted training sessions for our ship dataset both with and without the introduction of noise.

In this context, the S_2 dataset serves as the training dataset, while D_2^{Test} is designated as the testing dataset. The results presented in Table 10 reveal that applying a thresholding method for preprocessing the ship dataset may not yield optimal outcomes. The mean Average Precision (mAP) generated by this method falls short of the results achieved by alternative approaches. This disparity can be attributed to the inherent complexity of cluttered backgrounds within the dataset, making it challenging to identify a single segmentation method suitable for all images. Furthermore, in some instances, portions of ships may inadvertently be excluded, thus compromising the model's overall performance.

The NoiBased method [53,54] offers a potential remedy by augmenting the dataset through the introduction of noise into the input images. This augmentation leads to a modest performance improvement. Specifically, the mAP registers at 0.923 without the addition of noise, and it increases marginally to 0.925 when noise is incorporated. However, the incremental improvement is relatively slight, possibly because the feature extractor has already learned to filter out noise from the input images, resulting in similarities between the extracted features in both scenarios.

Our approach stands out as the most effective due to the deliberate introduction of uncertainty at the classifier head. In this configuration, the feature extractor is unable to eliminate the introduced uncertainty, placing a greater onus on the classifier to exhibit robustness in handling this uncertainty. This rationale has motivated the incorporation of uncertainty into the feature domain, a practice widely adopted in numerous research studies to enhance model performance.

Table 10. Performance comparisons among pre-processing methods. The best results are marked in bold.

Method	Fishing Boat	Container Ship	Ore Carrier	Bulk Cargo Carrier	Passenger Ship	General Cargo Ship	mAP
Seg-based [52]	0.873	0.952	0.896	0.859	0.805	0.894	0.880
NoiBased [53,54]	0.912	0.973	0.923	0.935	0.871	0.935	0.925
No noise	0.903	0.970	0.932	0.928	0.862	0.945	0.923
Proposed method	**0.922**	**0.980**	**0.935**	**0.953**	**0.873**	**0.946**	**0.935**

4.7. The Position of VIB Network

In the proposed method, we have inserted the VIB module at the beginning of the classification head. However, the VIB model could be inserted at any position of the network structure. Hence, in this section, we have tried several setups to evaluate how to use a VIB in an object detection task. In YOLOX, the classification head has two sequence convolution blocks. The proposed method inserts the VIB block at the beginning of the classification head, as in Figure 2. However, we can also set up the VIB module at the middle of the classification head as in Figure 7 or at the beginning of the decouple head as in Figure 8. Here, the S_2 dataset is used for training as in Section 4.5. The result in Table 11 shows that the VIB module is only suitable to be inserted on the classification branch. If the VIB module affects the regression branch, the network cannot converge.

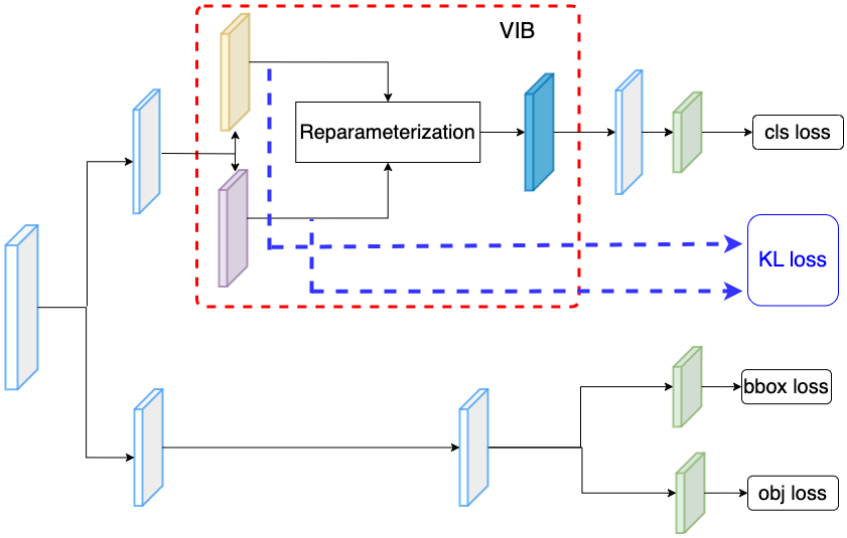


Figure 7. VIB on the mid of classification head.

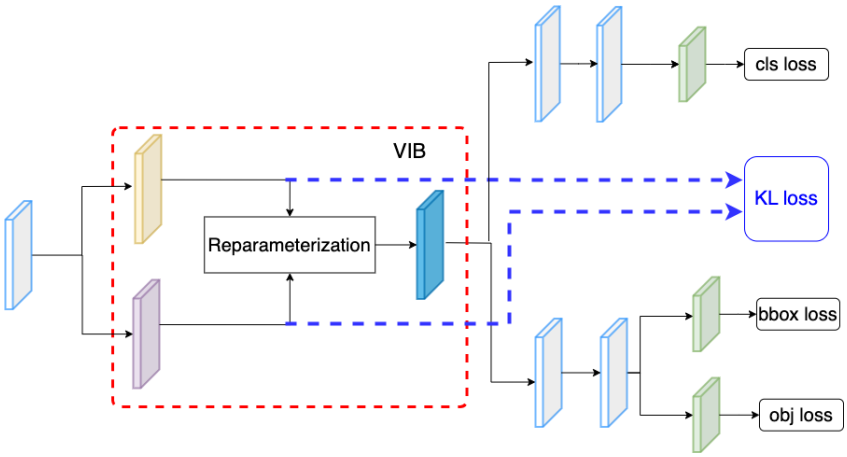


Figure 8. VIB at the beginning of the decouple head.

Table 11. Performance when the VIB module is inserted at different positions on the YOLOX framework.

Method	Metrics	Fishing Boat	Container Ship	Ore Carrier	Bulk Cargo Carrier	Passenger Ship	General Cargo Ship	mAP
VIB at the middle of classification head	dets	1863	584	2012	1653	354	1129	
	recall	0.940	0.984	0.962	0.971	0.891	0.964	
	AP	0.922	0.980	0.935	0.953	0.873	0.946	0.935
VIB at the beginning of decouple head	dets	-	-	-	-	-	-	
	recall	-	-	-	-	-	-	
	AP	-	-	-	-	-	-	

To explain the phenomenon, the L_{box}, L_{cls}, and L_{KL} over a training phase are shown in Figure 9.

The model can converge smoothly if the VIB module is on the classification head (the second row of Figure 9). The L_{box} quickly degrades to range [4–5] with only 3000 iterations. Box prediction's success is a critical requirement to train the classification head. At the beginning of the training process, L_{cls} increases when L_{box} is large; then, it degrades smoothly when the L_{box} is smaller. The L_{KL} should contribute later in the training process because it is an auxiliary loss but not a major task.

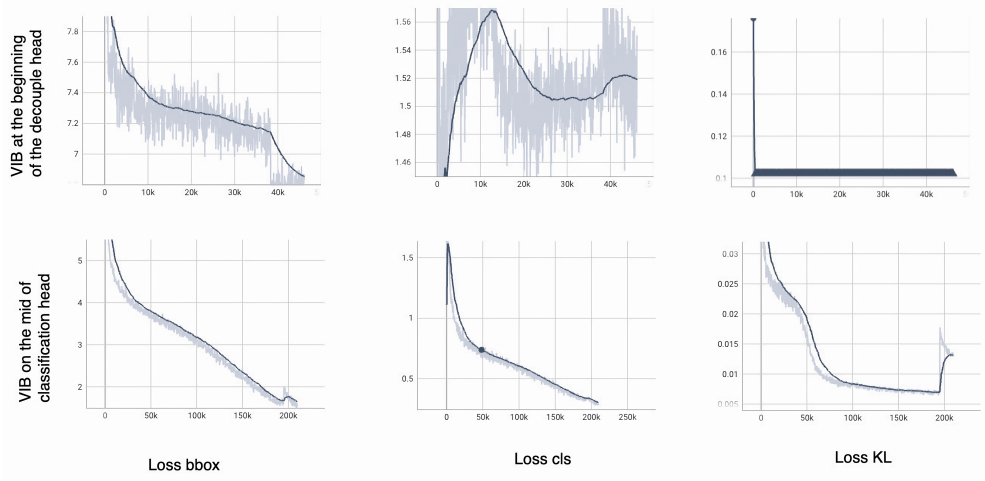


Figure 9. The loss over a training process. The bold line is smooth values over iterations. The light line is the actual value in an iteration.

The model cannot converge if the VIB module is at the beginning of the decoupled head (the first row of Figure 9). The L_{box} degrades but is still higher than 7 after 40,000 iterations. While the box prediction is unsuccessful, the classification head may be unable to learn. The L_{cls} increases and reduces over a training process. The classification head cannot be learned if L_{box} is still large. This phenomenon shows that the KL loss and the reparameterization make the regression more challenging. Consequently, the classification head cannot be learned, and the model fails to converge.

5. Conclusions

In this paper, we proposed a novel method for ship detection. Based on the YOLOX framework, we introduce a VIB module on the classification head of the network. Comprehensive experiments prove that our method is beneficial on small training datasets. The learned features will focus on the object rather than distribute uniformly over images. Our method also provides promising results in comparison with SoTA ship detection.

Author Contributions: Conceptualization, resources, investigation, writing-original draft, analysis, D.-D.N.; software, V.-L.V.; validation, T.N.; methodology, M.-H.N.; writing—review and editing, M.-H.L. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by HCMC University of Technology and Education, VietNam.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are openly available in [Zhang] at [https://doi.org/10.1109/ACCESS.2022.3199352], reference number [21].

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

References

1. Szeto, A.; Pelot, R. The use of long range identification and tracking (LRIT) for modelling the risk of ship-based oil spills. In Proceedings of the AMOP Technical Seminar on Environmental Contamination and Response 2011, Banff, AB, Canada, 4–6 October 2011.
2. Mao, S.; Tu, E.; Zhang, G.; Rachmawati, L.; Rajabally, E.; Huang, G. An Automatic Identification System (AIS) Database for Maritime Trajectory Prediction and Data Mining. *arXiv* **2016**, arXiv:1607.03306.
3. Paterniani, G.; Sgreccia, D.; Davoli, A.; Guerzoni, G.; Di Viesti, P.; Valenti, A.C.; Vitolo, M.; Vitetta, G.M.; Boriani, G. Radar-Based Monitoring of Vital Signs: A Tutorial Overview. *Proc. IEEE* **2023**, *111*, 277–317. [CrossRef]
4. Zhou, X.; Gong, W.; Fu, W.; Du, F. Application of deep learning in object detection. In Proceedings of the 2017 IEEE/ACIS 16th International Conference on Computer and Information Science (ICIS), Wuhan, China, 24–26 May 2017; pp. 631–634. [CrossRef]

5. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Rich feature hierarchies for accurate object detection and semantic segmentation. *arXiv* **2013**, arXiv:1311.2524.
6. Girshick, R. Fast R-CNN. In Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, 7–13 December 2015; pp. 1440–1448. [CrossRef]
7. Girshick, R.; Iandola, F.; Darrell, T.; Malik, J. Deformable Part Models are Convolutional Neural Networks. In Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 July 2015. [CrossRef]
8. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.Y.; Berg, A.C. SSD: Single Shot MultiBox Detector. In Proceedings of the Computer Vision—ECCV 2016, Amsterdam, The Netherlands, 11–14 October 2016; Springer International Publishing: Cham, Switzerland, 2016; pp. 21–37. [CrossRef]
9. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You Only Look Once: Unified, Real-Time Object Detection. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 26 June–1 July 2016; pp. 779–788. [CrossRef]
10. Lin, T.Y.; Dollár, P.; Girshick, R.; He, K.; Hariharan, B.; Belongie, S. Feature Pyramid Networks for Object Detection. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 936–944. [CrossRef]
11. Ge, Z.; Liu, S.; Wang, F.; Li, Z.; Sun, J. YOLOX: Exceeding YOLO Series in 2021. *arXiv* **2021**, arXiv:2107.08430.
12. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.; Polosukhin, I. Attention Is All You Need. *arXiv* **2017**, arXiv:1706.03762.
13. Carion, N.; Massa, F.; Synnaeve, G.; Usunier, N.; Kirillov, A.; Zagoruyko, S. End-to-End Object Detection with Transformers. In Proceedings of the European Conference on Computer Vision, Glasgow, UK, 23–28 August 2020; Springer: Cham, Switzerland, 2020; pp. 213–229. [CrossRef]
14. Liu, Z.; Lin, Y.; Cao, Y.; Hu, H.; Wei, Y.; Zhang, Z.; Lin, S.; Guo, B. Swin Transformer: Hierarchical Vision Transformer using Shifted Windows. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Montreal, BC, Canada, 11–17 October 2021. Available online: https://www.computer.org/csdl/proceedings-article/iccv/2021/281200j992/1BmGKZoEzug (accessed on 10 July 2023)
15. Lee, S.H.; Park, H.G.; Kwon, K.H.; Kim, B.H.; Kim, M.Y.; Jeong, S.H. Accurate Ship Detection Using Electro-Optical Image-Based Satellite on Enhanced Feature and Land Awareness. *Sensors* **2022**, *22*, 9491. [CrossRef]
16. Patel, K.; Bhatt, C.; Mazzeo, P.L. Deep Learning-Based Automatic Detection of Ships: An Experimental Study Using Satellite Images. *J. Imaging* **2022**, *8*, 182. [CrossRef]
17. Stofa, M.M.; Zulkifley, M.A.; Zaki, S.Z.M. A deep learning approach to ship detection using satellite imagery. *IOP Conf. Ser. Earth Environ. Sci.* **2020**, *540*, 012049. [CrossRef]
18. Everingham, M.; Van Gool, L.; Williams, C.K.I.; Winn, J.; Zisserman, A. The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. 2012. Available online: http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html (accessed on 10 July 2023).
19. Zhang, Z.; Zhang, L.; Wang, Y.; Feng, P.; He, R. ShipRSImageNet: A Large-Scale Fine-Grained Dataset for Ship Detection in High-Resolution Optical Remote Sensing Images. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2021**, *14*, 8458–8472. [CrossRef]
20. Bochkovskiy, A.; Wang, C.Y.; Liao, H.Y.M. YOLOv4: Optimal Speed and Accuracy of Object Detection. *arXiv* **2020**, arXiv:2004.10934.
21. Zhang, M.; Rong, X.; Yu, X. Light-SDNet: A Lightweight CNN Architecture for Ship Detection. *IEEE Access* **2022**, *10*, 86647–86662. [CrossRef]
22. Alemi, A.A.; Fischer, I.; Dillon, J.V.; Murphy, K. Deep Variational Information Bottleneck. *arXiv* **2016**, arXiv:1612.00410.
23. Shannon, C.E. A Mathematical Theory of Communication. *Bell Syst. Tech. J.* **1948**, *27*, 379–423. [CrossRef]
24. He, K.; Zhang, X.; Ren, S.; Sun, J. Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **2015**, *37*, 1904–1916. [CrossRef]
25. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 1137–1149. [CrossRef] [PubMed]
26. He, K.; Gkioxari, G.; Dollár, P.; Girshick, R. Mask R-CNN. In Proceedings of the 2017 IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017; pp. 2980–2988. [CrossRef]
27. Redmon, J.; Farhadi, A. YOLO9000: Better, Faster, Stronger. *arXiv* **2016**, arXiv:1612.08242.
28. Redmon, J.; Farhadi, A. YOLOv3: An Incremental Improvement. *arXiv* **2018**, arXiv:1804.02767.
29. Duan, K.; Bai, S.; Xie, L.; Qi, H.; Huang, Q.; Tian, Q. CenterNet: Keypoint Triplets for Object Detection. *arXiv* **2019**, arXiv:1904.08189.
30. Chen, W.; Shah, T. Exploring Low-light Object Detection Techniques. *arXiv* **2021**, arXiv:cs.CV/2107.14382.
31. Tan, M.; Pang, R.; Le, Q.V. EfficientDet: Scalable and Efficient Object Detection. *arXiv* **2020**, arXiv:cs.CV/1911.09070.
32. Grekov, A.N.; Shishkin, Y.E.; Peliushenko, S.S.; Mavrin, A.S. Application of the YOLOv5 Model for the Detection of Microobjects in the Marine Environment. *arXiv* **2022**, arXiv:cs.CV/2211.15218.
33. Katz, D.M.; Hartung, D.; Gerlach, L.; Jana, A.; Bommarito, M.J., II. Natural Language Processing in the Legal Domain. *arXiv* **2023**, arXiv:cs.CL/2302.12039.

34. Zhu, X.; Su, W.; Lu, L.; Li, B.; Wang, X.; Dai, J. Deformable DETR: Deformable Transformers for End-to-End Object Detection. *arXiv* **2020**, arXiv:2010.04159.
35. Lin, T.; Maire, M.; Belongie, S.J.; Bourdev, L.D.; Girshick, R.B.; Hays, J.; Perona, P.; Ramanan, D.; Doll'a r, P.; Zitnick, C.L. Microsoft COCO: Common Objects in Context. In *ECCV*; Springer International Publishing: Cham, Switzerland, 2014; Volume 8693. [CrossRef]
36. Zheng, J.; Liu, Y. A Study on Small-Scale Ship Detection Based on Attention Mechanism. *IEEE Access* **2022**, *10*, 77940–77949. [CrossRef]
37. Ye, B.; Qin, T.; Zhou, H.; Lai, J.; Xie, X. Cross-level Attention and Ratio Consistency Network for Ship Detection. In Proceedings of the 2022 26th International Conference on Pattern Recognition (ICPR), Montreal, QC, Canada, 21–25 August 2022; pp. 4644–4650. [CrossRef]
38. Cui, H.; Yang, Y.; Liu, M.; Shi, T.; Qi, Q. Ship Detection: An Improved YOLOv3 Method. In Proceedings of the OCEANS 2019, Marseille, France, 17–20 June 2019; pp. 1–4. [CrossRef]
39. Liu, T.; Pang, B.; Ai, S.; Sun, X. Study on Visual Detection Algorithm of Sea Surface Targets Based on Improved YOLOv3. *Sensors* **2020**, *20*, 7263. [CrossRef]
40. Li, H.; Deng, L.; Yang, C.; Liu, J.; Gu, Z. Enhanced YOLO v3 Tiny Network for Real-Time Ship Detection From Visual Image. *IEEE Access* **2021**, *9*, 16692–16706. [CrossRef]
41. Woo, S.; Park, J.; Lee, J.Y.; Kweon, I. CBAM: Convolutional Block Attention Module. In Proceedings of the 15th European Conference, Munich, Germany, 8–14 September 2018; pp. 3–19. [CrossRef]
42. Liu, T.; Pang, B.; Zhang, L.; Yang, W.; Sun, X. Sea Surface Object Detection Algorithm Based on YOLO v4 Fused with Reverse Depthwise Separable Convolution (RDSC) for USV. *J. Mar. Sci. Eng.* **2021**, *9*, 753. [CrossRef]
43. Guo, J.; Li, Y.; Lin, W.; Chen, Y.; Li, J. Network Decoupling: From Regular to Depthwise Separable Convolutions. *arXiv* **2018**, arXiv:cs.CV/1808.05517.
44. Han, X.; Zhao, L.; Ning, Y.; Hu, J. ShipYOLO: An Enhanced Model for Ship Detection. *J. Adv. Transp.* **2021**, *2021*, 1060182. [CrossRef]
45. Han, K.; Wang, Y.; Tian, Q.; Guo, J.; Xu, C.; Xu, C. GhostNet: More Features from Cheap Operations. *arXiv* **2020**, arXiv:cs.CV/1911.11907.
46. Ye, R.; Liu, F.; Zhang, L. 3D Depthwise Convolution: Reducing Model Parameters in 3D Vision Tasks. *arXiv* **2018**, arXiv:cs.CV/1808.01556.
47. Zhang, Q.; Huang, Y.; Song, R. A Ship Detection Model Based on YOLOX with Lightweight Adaptive Channel Feature Fusion and Sparse Data Augmentation. In Proceedings of the 2022 18th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), Madrid, Spain, 29 November–2 December 2022; pp. 1–8. [CrossRef]
48. Liu, S.; Qi, L.; Qin, H.; Shi, J.; Jia, J. Path Aggregation Network for Instance Segmentation. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 8759–8768. [CrossRef]
49. Zhang, Y.; Er, M.J.; Gao, W.; Wu, J. High Performance Ship Detection via Transformer and Feature Distillation. In Proceedings of the 2022 5th International Conference on Intelligent Autonomous Systems (ICoIAS), Dalian, China, 23–25 September 2022; pp. 31–36. [CrossRef]
50. Tishby, N.; Pereira, F.C.; Bialek, W. The information bottleneck method. In Proceedings of the 37-th Annual Allerton Conference on Communication, Control and Computing, Monticello, IL, USA, 22-24 September 1999; pp. 368–377.
51. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 26 June–1 July 2016; pp. 770–778. [CrossRef]
52. Ashwani Kumar Aggarwal, P.J. Segmentation of Crop Images for Crop Yield Prediction. *Int. J. Biol. Biomed.* **2022**, *7*, 40–44.
53. Thukral, R.; Arora, A.; Kumar, A.; Kumar, G. *Denoising of Thermal Images Using Deep Neural Network*; Springer: Singapore, 2022. pp. 827–833. [CrossRef]
54. Thukral, R.; Kumar, A.; Arora, A.; Gulshan. Effect of Different Thresholding Techniques for Denoising of EMG Signals by using Different Wavelets. In Proceedings of the 2019 2nd International Conference on Intelligent Communication and Computational Techniques (ICCT), Jaipur, India, 28–29 September 2019; pp. 161–165. [CrossRef]

Last Updated on June 22, 2023

EAI Endorsed Transactions on Industrial Networks and Intelligent Systems- Impact Score, Ranking, SJR, h-index, Citescore, Rating, Publisher, ISSN, and Other Important Details

Published By: European Alliance for Innovation

Impact Score

2.08

SJR

0.357

h-Index

10

Rank

13509

EAI Endorsed Transactions on Industrial Networks and Intelligent Systems Impact Score (IS) Trend

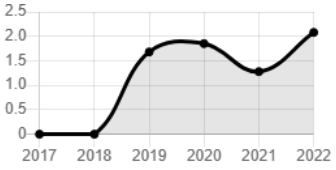


Table: Impact Score

Year	Impact Score (IS)
2023/2024	Updated Soon
2022	2.08
2021	1.28
2020	1.85
2019	1.68
2018	0.00
2017	0.00

Important Details

Type of Publication	Journal
Discipline	Computer Networks and Communications (Q3); Computer Science Applications (Q3); Control and Systems Engineering (Q3); Information Systems (Q3)
Impact Score	2.08
SCImago Journal Rank (SJR)	0.357
h-index	10
Overall Rank	13509
Publisher Name	European Alliance for Innovation
Publication Country	Belgium
International Standard Serial Number (ISSN)	24100218
Coverage and History	2014-2022
Best Quartile	Q3
Total Citations Received (Last 3 Year)	106

About EAI Endorsed Transactions on Industrial Networks and Intelligent Systems

EAI Endorsed Transactions on Industrial Networks and Intelligent Systems is a **journal** published by **European Alliance for Innovation**. This journal covers the area[s] related to **Computer Networks and Communications, Computer Science Applications, Control and Systems Engineering, Information Systems, etc**. The coverage history of this journal is as follows: 2014-2022. The rank of this journal is **13509**. This journal's impact score, h-index, and SJR are 2.08, 10, and 0.357, respectively. The ISSN of this journal is/are as follows: **24100218**.

The **best quartile** of **EAI Endorsed Transactions on Industrial Networks and Intelligent Systems** is **Q3**. This **journal** has received a total of **106** citations during the last three years (Preceding 2022).

EAI Endorsed Transactions on Industrial Networks and Intelligent Systems Impact

Top Journals/Conferences in Computer Networks and Communications

IEEE Journal on Selected Areas in Communications
Institute of Electrical and Electronics Engineers Inc. | United States

Nature Machine Intelligence
Springer Nature Switzerland AG | Switzerland

IEEE Communications Magazine
Institute of Electrical and Electronics Engineers Inc. | United States

International Journal of Information Management
Elsevier Ltd. | United Kingdom

Synthesis Lectures on Communication Networks
Morgan and Claypool Publishers | United States

IEEE Network
Institute of Electrical and Electronics Engineers Inc. | United States

IEEE Internet of Things Journal
Institute of Electrical and Electronics Engineers Inc. | United States

IEEE Transactions on Neural Networks and Learning Systems
IEEE Computational Intelligence Society | United States

IEEE Transactions on Cognitive Communications and Networking
Institute of Electrical and Electronics Engineers Inc. | United States

Foundations and Trends in Communications and Information Theory
Now Publishers Inc | United States

AVS Quantum Science
American Institute of Physics | United States

Internet and Higher Education
Elsevier BV | United Kingdom

npj Quantum Information
Nature Partner Journals | United Kingdom

Information Systems Research
INFORMS Institute for Operations Research and the Management Sciences | United States

Scopus Preview

Sources

Title | Enter title | Find sources

Title: EAI Endorsed Transactions On Industrial Networks And Intelligent Systems ×

CiteScore 2024 has been released. View CiteScore methodology >

x

Filter refine list

Apply Clear filters

Display options

- [ ] Display only Open Access journals

Counts for 4-year timeframe

- ( ) No minimum selected
- ( ) Minimum citations
- ( ) Minimum documents

Citescore highest quartile

- [ ] Show only titles in top 10 percent
- [ ] 1st quartile
- [ ] 2nd quartile
- [ ] 3rd quartile
- [ ] 4th quartile

Source type

- [ ] Journals
- [ ] Book Series
- [ ] Conference Proceedings
- [ ] Trade Publications

Apply Clear filters

1 result

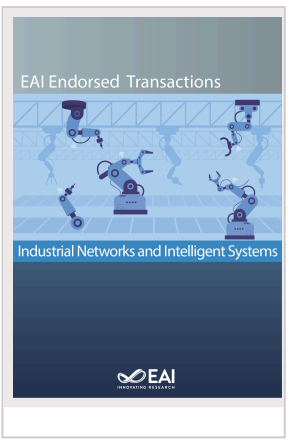
Download Scopus Source List | Learn more about Scopus Source List

All | Export to Excel | Save to source list

View metrics for year: 2024

	Source title	CiteScore	Highest percentile	Citations 2021-24	Documents 2021-24	% Cited
1	EAI Endorsed Transactions on Industrial Networks and Intelligent Systems *Open Access*	4.8	68% 120/375 Control and Systems Engineering	367	77	70

Top of page



EAI Endorsed Transactions on Industrial Networks and Intelligent Systems

Issue 1, 2025

Editor(s)-in-Chief: Trung Q. Duong, Le Nguyen Bao and Nguyen-Son Vo

Articles Information

Transformer Based Ship Detector: An Improvement on Feature Map and Tiny Training Set
Appears in: inis **25**(1):
Authors: Duc-Dat Ngo, Van-Linh Vo, My-Ha Le , Hoc-Phan , Manh Hung Nguyen
Abstract: The exponential increment of commodity exchange has raised the need for maritime border security in recent years. One of the most critical tasks for naval border security is ship detection inside and ... more >>

An Efficient Method for BLE Indoor Localization Using Signal Fingerprint
Appears in: inis **25**(1):
Authors: Trong-Thanh Han, Phuc Nguyen Dinh, Toan Nguyen Duc, Vu Nguyen Long, Hung Dinh Tan
Abstract: The rise of Bluetooth Low Energy (BLE) technology has opened new possibilities for indoor localization systems. However, extracting fingerprint features from the Received Signal Strength Indicator (RS... more >>

Joint Adaptive Modulation and Power Control Scheme for Energy Efficient FSO-based Non-Terrestrial Networks
Appears in: inis **25**(1):
Authors: Thang V. Nguyen, Hien T. T. Pham, Ngoc T. Dang
Abstract: Free-space optics (FSO)-based non-terrestrial networks (NTN) have garnered significant attention as a potential technology for forthcoming 6G wireless communications due to their exceptional data rate... more >>

Drug classification system based on drug composition and usage instructions
Appears in: inis **25**(1):
Authors: Hoang-Dieu Vu, Vu-Hien Pham, Quang-Dung Le
Abstract: This study presents a natural language processing (NLP) approach to

ISSN: 2410-0218

Submit Article

Submission Instructions

Ethics and Malpractice Statement

Back to Journal Page

2025
Issue 3
Issue 2
Issue 1

2024
Issue 4
Issue 3
Issue 2
Issue 1

2023
Issue 4
Issue 3
Issue 2
Issue 1
2022
Issue 4
Issue 32
Issue 31
Issue 30
2021
Issue 29
Issue 28
Issue 27
Issue 26
2020
Issue 25
Issue 24
Issue 23
Issue 22
2019
Issue 21
Issue 20
Issue 19
Issue 18
2018
Issue 17
Issue 16
Issue 15
Issue 14
2017
Issue 13
Issue 12
Issue 11
Issue 10
2016
Issue 9
Issue 8
Issue 7
Issue 6
2015

classify drugs based on compositional and usage descriptions. NLP techniques including text preprocessing, word embedding, and deep ... more >>

Predicting the Severity of COVID-19 Pneumonia from Chest X-Ray Images: A Convolutional Neural Network Approach

Appears in: inis **25**(1):
Authors: Thien B. Nguyen-Tat, Viet-Trinh Tran-Thi, Vuong M. Ngo
Abstract: This study addresses significant limitations of previous works based on the Brixia and COVIDGR datasets, which primarily provided qualitative lung injury scores and focused mainly on detecting mild an... more >>

Publisher EAI ISSN 2410-0218 Volume 12
Published 2025-01-01

Issue 5
Issue 4
Issue 3
Issue 2

2014

Issue 1

EAI Endorsed Transactions
on Industrial Networks and Intelligent Systems
Research Article EAI.EU

Transformer Based Ship Detector: An Improvement on Feature Map and Tiny Training Set

Duc-Dat Ngo[1], Van-Linh Vo[1], My-Ha Le[1], Hoc-Phan [1], Manh-Hung Nguyen[1,*]

[1]HCMC University of Technology and Education- Faculty of Electrical and Electronics Engineering- Ho Chi Minh City (7000)-VietNam.

Abstract

The exponential increment of commodity exchange has raised the need for maritime border security in recent years. One of the most critical tasks for naval border security is ship detection inside and outside the territorial sea. Conventionally, the task requires a substantial human workload. Fortunately, with the rapid growth of the digital camera and deep-learning technique, computer programs can handle object detection tasks well enough to replace human labor. Therefore, this paper studies how to apply recent state-of-the-art deep-learning networks to the ship detection task. We found that with a suitable number of object queries, the Deformable-DETR method will improve the performance compared to the state-of-the-art ship detector. Moreover, comprehensive experiments on different scale datasets prove that the technique can significantly improve the results when the training sample is limited. Last but not least, feature maps given by the method will focus well on key objects in the image.

Received on 29 07 2024; accepted on 18 10 2024; published on 06 11 2024

Keywords: Maritime border security; Deformable DETR; Object detection; Hyper-parameter

doi:10.4108/eetinis.v12i1.6794

1. Introduction

The global commodity exchange has surged in recent years, leading to an increase in waterway transport services. While this growth benefits the economy, it also presents significant challenges in protecting national water borders. As a result, ship detection has become a crucial aspect of national security. Accurately detecting ships approaching the shore and thoroughly verifying their legality are essential tasks. Typically, data from coastal surveillance cameras is used to monitor passing ships, and human inspectors review the images. However, this manual process requires considerable human effort and is prone to errors due to various distractions. Therefore, developing an automated ship detection system can reduce costs and improve monitoring efficiency, particularly for developing countries. Nevertheless, there are still practical challenges that the ship detection model must overcome to achieve optimal performance, such as data imbalance, lighting conditions, occlusion, missing object parts, and size diversity, as shown in Figure 1.

Ship detection and classification from images are well-known applications in computer vision, traditionally addressed by object detection techniques. While earlier deep learning models like Region-based Convolutional Neural Network (R-CNN) [1] and FastRCNN [2] surpassed hand-crafted methods, their sluggish performance hindered real-world applications. One-stage detectors such as Single Shot Setector (SSD) [3] and You Only Look Once [4], though faster, rely on cumbersome anchor boxes and post-processing steps. Recent advancements like feature pyramid networks (FPN) [5] and decoupled heads [6] have partially mitigated these issues, but not all. To improve the performance of ship detectors, many researchers try to customize these detectors, such as network architecture [7–11], feature fusion [12, 13], or feature selection loss [14].

While many promising results have been reported, training a good detector is still an open question. Several hyper-parameters, such as anchor-box generation or non-maximum suppression procedure, should be

*Corresponding author. Email: hungnm@hcmute.edu.vn

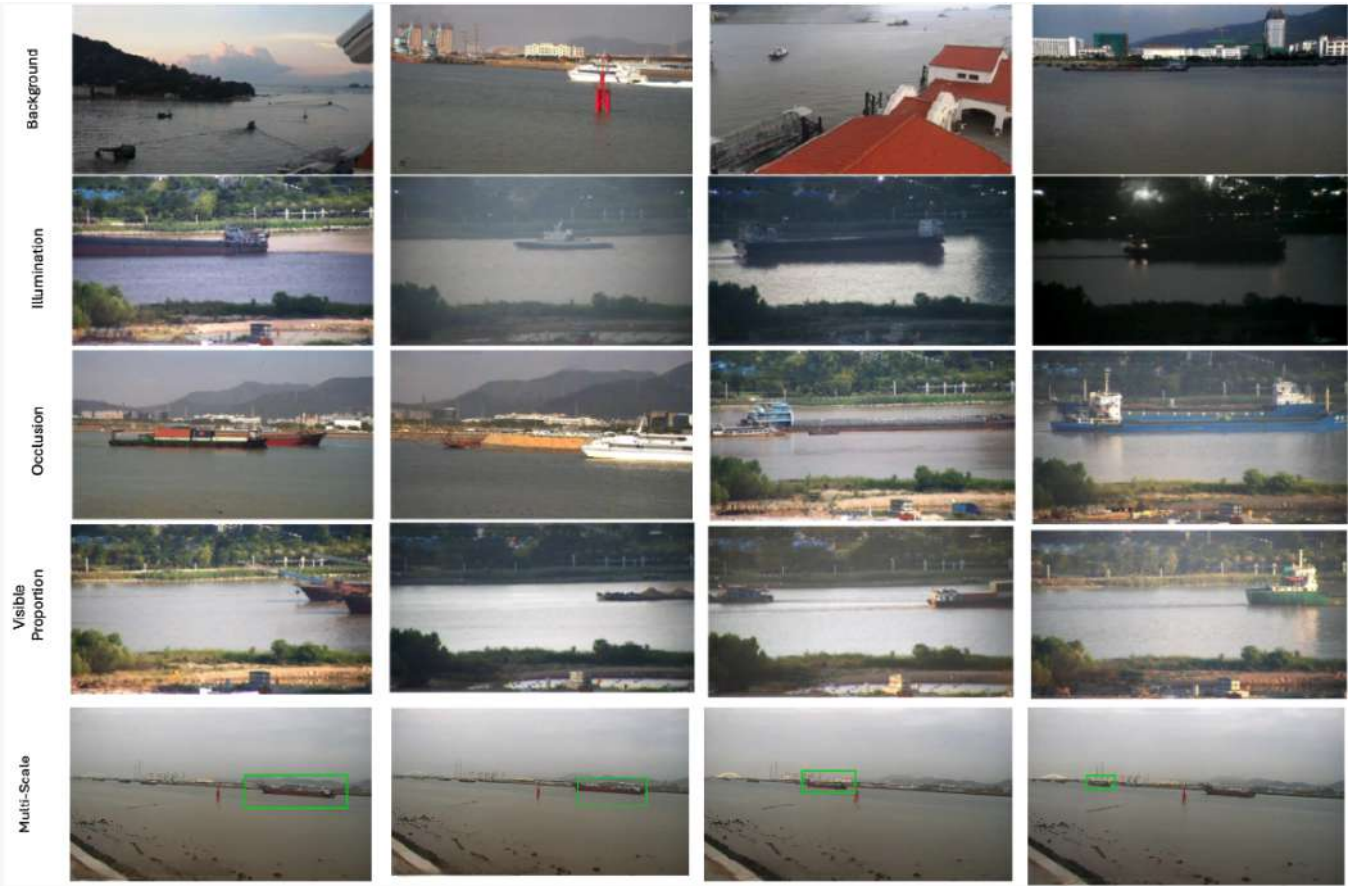


Figure 1. Challenges of ship detection.

carefully designed to ensure the success of training processes. Motivated by the observation, DETR [15, 16] provides end-to-end object detection with transformers. This method forces unique predictions via bipartite matching; hence a fixed small set of learned object queries may directly output the final set of predictions in parallel. Yani_2022 [17] had try to applied DETR to ship detection via distillation learning. Here, a teacher provides a prediction to train a student model. However, the performance is not comparable to CNN-based methods [12, 13] in the student model.

In our work, we revisit how to use DETR to train a ship-detector. As shown in Figure 2, each learnable query will provide one object detection result on the input image. Hence, too many queries will provide false detection, especially when the number of objects is sparse, like ships. Hence, a suitable number of object queries may directly affect the detection result.

We prove this viewpoint by comprehensive experiments on a well-known large-scale ship dataset [18]. First, we follow guidance from [12–14, 18] to prepare training and testing datasets and train our model. The experiment aims to compare our method with state-of-the-art methods. Second, we re-used the testing set but reduced the training set as in [14, 17, 19]. This experiment helps to evaluate the performance of our method when the training samples are limited. In addition, an ablation study is applied to evaluate the effect of the number of object queries. Last but not least, feature visualization explains why our method can work more robustly than CNN-based methods. In summary, our contributions are as bellowed:

(i) Our detector demonstrates performance that is on par with state-of-the-art (SoTA) methods on well-known benchmarks. It has shown comparable accuracy, precision, and robustness through rigorous testing and evaluation, effectively matching the results of leading models in the field.

(ii) When the number of training samples is limited, our method demonstrates a significant performance improvement. By leveraging advanced techniques and efficient learning algorithms, our approach maximizes the utility of available data, ensuring robust and accurate results even with constrained training sets. This capability highlights the strength and adaptability of our method, making it particularly valuable in scenarios where data acquisition is challenging or

costly. Consequently, our method is reliable for achieving high performance in data-scarce environments.

(iii) Feature analysis provides insights into why our method operates robustly compared to traditional CNN-based approaches. By examining the features extracted by our model, we can see how it effectively captures and utilizes relevant patterns in the data, leading to improved performance.

2. Releated Work

2.1. Object Detection

Recently, deep learning has been considered an advantageous solution for computer vision tasks. Deep learning-based object detection can be categorized into two branches: region proposal-based methods and regression/classification-based methods. Region proposal-based methods usually have two steps. The first step is finding areas where an object is likely to exist. The second step is to perform the classification for that object. Therefore, this method is called the two-stage object detection method. Some well-known methods included R-CNN [1], Fast R-CNN [20], Faster R-CNN [21], FPN [5], etc.

On the other hand, regression/classification-based object detection will predict an object's location through regression and the object's label through classification. Instead of splitting the object detection into two steps, this technique uses a convolutional neural network to predict location information (regression results) and label information (classification results). Since only one CNN network is used, these are considered single-stage methods. Well-known methods of this approach are YOLO [4], SSD [3], etc. The single-stage methods have a much faster processing speed than the two-stage methods. For example, Fast R-CNN [20] can handle 0.5 frames per second, whereas the first version of YOLO can handle 45 frames per second, or SSD can handle 58 frames per second. Therefore, single-stage methods are often used in practice.

There is a trade-off between accuracy and inference time in object detection. While Fast R-CNN [20] achieved an accuracy of 0.7 mAP, it can process very slowly. The first version of YOLO called YOLOv1 [4], only achieved an accuracy of 0.63 mAP, but it is much faster than FastRCNN. For this reason, many researchers are trying to increase the accuracy of the YOLO structure. Recently, later versions of the YOLO have been discussed to overcome the disadvantages of the original YOLO model. For example, in YOLOv1, each cell can only predict one object at most. For cases where many objects are in the same cell, YOLOv1 may not have a good result. Moreover, YOLOv1 predicts the position of objects as a bounding box directly, and the objective functions of YOLOv1 [4] do not have a separate evaluation between the error caused by bounding box widths and heights.

Instead of predicting the absolute bounding box, YOLOv2 [22] considers anchor boxes of the main component to predict each relative bounding box. In detail, instead of indicating the position of a box on an image, the CNN network predicts the offset between the bounding box and predefined anchor boxes. Predicting the offset is much easier than predicting the box coordinates. If more or more anchor contours surround the object, it is possible to define the anchor contours that overlap with the labeled object contour.

One of the key challenges in object detection is handling the scale issue. Objects appear smaller when they are far from the camera and larger when they are close. The Feature Pyramid Network (FPN) [5] addresses this scale challenge by introducing pyramid features, where the model extracts features at different scales, similar to the layers of a pyramid. This approach allows the detection of objects at varying distances. However, FPN's drawbacks include high memory consumption, reduced detection speed, and increased model complexity. To extract multi-scale features without these burdens, the YOLOF model [23] was developed. Instead of employing the multi-input, multi-output architecture of FPN, YOLOF utilizes a single-input, single-output encoding architecture based on Dilated CNN [24]. This design enables YOLOF to consume less memory while maintaining accuracy comparable to FPN.

One of the challenges in object detection is determining the number and size of anchor boxes, especially due to varying distances between the object and the camera. Anchor boxes are usually determined using the k-means clustering algorithm, but this process still requires human intervention and is not fully automated for end-to-end training. YOLOX, a notable model, has made modifications to eliminate the anchor box selection process. It does this by replacing the coupled head with a decoupled head and predicting only one feature box for each location on the feature map. This effectively removes the need for anchors. Additionally, by separating the regression and classification tasks into two distinct branches, the model can converge more effectively.

Inspired by the success of transformers in natural language processing (NLP), researchers have begun applying transformer concepts to object detection. Transformers represent a significant departure from convolutional neural networks (CNNs). Although their use in vision tasks is still in its early stages, transformers have shown promising potential to replace convolutions. State-of-the-art transformer-based detectors have achieved impressive results on various object

detection datasets, though they typically require more parameters than convolutional models. Recently, several transformer-based methods have been developed for object detection, including Vision-Transformer-Detection (ViTDET) [25], DETR [15, 16, 26], and Shift-Window-Transformer (Swin) [27]. Among these, DETR [15] stands out as one of the early transformer-based methods to provide competitive results compared to CNN-based detectors. Additionally, DETR operates more closely as an end-to-end training method than other CNN-based approaches.

2.2. Ship Detection

In the field of ship detection, various customizations have been made to popular detection frameworks like SSD (Single Shot Multibox Detector) [3] and YOLO (You Only Look Once) [4] to improve their performance. For example, Liu et al. [7] enhanced the SSD framework by adding a VGG backbone, which improved the detection of small objects. They also introduced a local attention network and a merge module to integrate features from different scales, leading to a significant improvement in accuracy. On the other hand, the "Cross-level Attention and Ratio Consistency Network" (CARC) [19] uses YOLO with a ResNet-34 backbone and incorporates cross-level attention modules to extract multi-scale features for enhanced detection capabilities. However, traditional frameworks may struggle with effectively handling scale variations, which can impact the accuracy across different object sizes.

A better backbone can significantly enhance accuracy. Consequently, researchers such as Cui [9], Liu [12], and Li [8] have based their ship detection models on YOLOv3. Their modifications focus on model customization, incorporating attention modules to detect targets at different scales. Despite these advancements, challenges persist in accurately localizing and classifying small or occluded objects.

Recent advancements continue to refine ship detection capabilities, incorporating models such as YOLOv4 (Zhang_2021 [10], Han_2021 [28]) and YOLOv5 (SDNet_2022 [11]) with simplified networks and attention mechanisms. Subsequently, Zhang_2022 [13] and VIB_2023 [14] have refined YOLOX [29], a lightweight method for feature fusion to address inconsistencies in feature map scales. While methods like those proposed by Zhang_2021 [10], Han_2021 [28], SDNet_2022 [11], and Zhang_2022 [13] focusing on feature fusion modules to enhance feature learning, VIB_2023 [14] introduces a feature selection loss to enforce the model to learn sparse and discriminative features, helping the feature map focus more on the detected objects.

Not only CNN-based frameworks but also DETR can be used to detect ships. Yani et al. [17] utilize DETR [15] with distillation learning for ship detection. First, a teacher model is trained using the DETR method. Then, distillation loss and Hungarian loss are applied to train a lightweight DETR student model. Unlike Han_2021 [28] or SDNet_2022 [11], which use a default setting on a large-scale dataset [30] to train a detector, Yani [17] defined a new setting where 50% of the data is used for testing. In this scenario, DETR outperforms CNN-based detectors. However, the distilled model does not maintain the same high accuracy as the original DETR model.

3. Methodology

3.1. DETR Architecture

Model Description. The DETR model comprises a feature extractor, an encoder, and a decoder, as depicted in Figure 2. The feature extractor is a CNN backbone that extracts high-level information from an image; a 2D sinusoidal positional encoding also helps encode position information for each pixel. Image features and positional features are concatenated and fed into the transformer-based encoder. The encoder consists of multiple stacked multi-head self-attention layers. Features from the encoder are then passed to the transformer-based decoder. The decoder also takes several learnable object queries as input. These queries serve as a latent of suitable positions on the image. Given one query, the decoder will use a feature from the encoder to predict if there is any object at the positional query.

In the feature extractor, given an input image with a $H_0 \times W_0 \times 3$ tensor, the CNN backbone generates a feature map of size $H \times W \times C$. According to [15], $C = 2048$, $H = H_0/32$, and $W = W_0/32$. A 1x1 convolution layer reduces the feature channels from C to d ($d < C$), creating a new feature map of size $H \times W \times d$. Because The transformer-based encoder requires a 2D input, the feature map is resized to a $d \times HW$ matrix. Rows of the matrix are named tokens, each token being a d-dimensional vector.

The encoder includes many stacked multi-head self-attention layers. Each multi-head self-attention block consists of several self-attention modules. In a multi-head self-attention module, features extracted from self-attention modules are concatenated and projected to the output via a linear projection. Three individual linear projections extract a tuple $(key, value, query)$ in a self-attention module. The similarities between key and $query$ serve as attention factors among $value$, fusing $value$ into self-attention features.

The decoder takes new features from the encoder and learnable object queries. Features from the encoder represent image information at every position, while queries serve as learnable positional encodings, questioning whether there is an object at a specific location. The decoder uses the image information

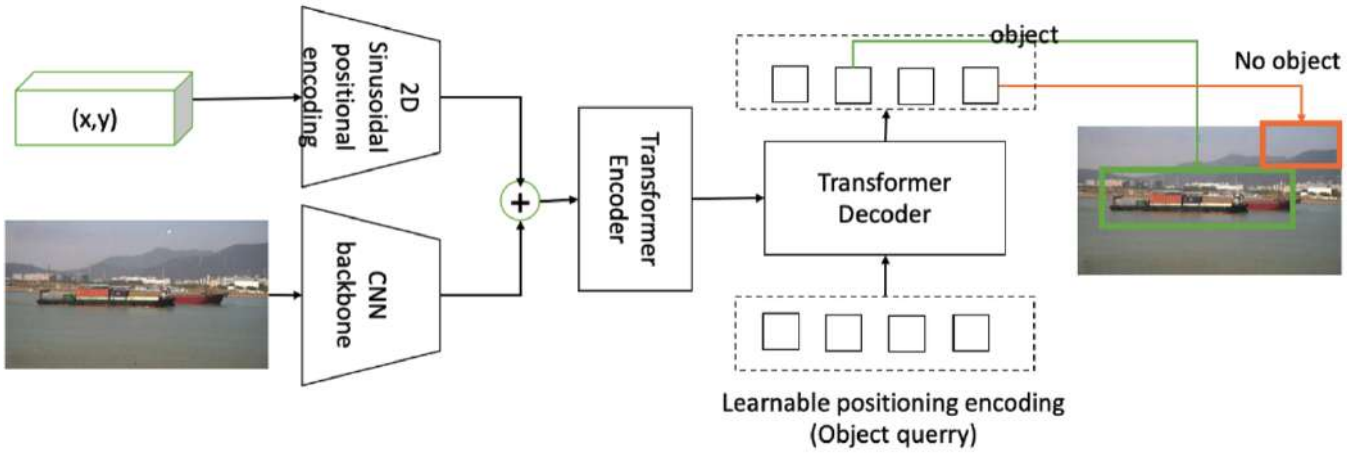


Figure 2. DETR Architecture

to determine if the encoded location contains any class. These queries are learnable from the dataset. The architecture of the encoder and decoder modules is shown in Figure 2. The outputs of the decoder module are fed into a feedforward neural network to predict the object's position and category. These outputs correspond to each object query learned in the decoder block. If an object query encodes no object at a position, the classification head result is "No Object".

Loss Function. The network returns a set of N detection results corresponding to N object queries. Each result is a tuple that includes $(class, box)$ and represents one and only one object without duplication. Therefore, a matching process is needed to map one prediction to one ground truth. Because the number of ground truth objects in an image is smaller than the number of object query N, a set $\varnothing$ of patches cropped randomly from the background is used as extra ground truth. Therefore, "background" objects with an arbitrary position and the label "No Object" help to balance the numbers of actual labeled locations and predictions. Denote σ is a matching solution; the optimal matching $\hat{\sigma}$ is the solution of an optimization process as in Equation 1. Where y_i is a ground truth of a box that includes class label and bounding box label (c_i, b_i); and $\hat{y}_i$ is a prediction result. This optimal assignment is solved by the Hungarian algorithm [31]. Note that this cost is not calculated on each object but as a collective combination of N objects generated for each image.

$$\hat{\sigma} = \underset{\sigma \in \Xi_N}{\operatorname{argmin}} \sum_{i}^{N} L_{\text{match}} \left(y_i, \hat{y}_{\sigma(i)}\right) \tag{1}$$

A good model can accurately predict objects and bounding boxes with higher overlapping contours. Therefore, the objective function $L_m\left(y_i, \hat{y}_{\sigma(i)}\right)$ should include these criterions as Equation 2.

$$L_m\left(y_i, \hat{y}_{\sigma(i)}\right) = -1_{\{c_i \neq \varnothing\}} \log \hat{p}_{\sigma(i)}(c_i) + 1_{\{c_i \neq \varnothing\}} L_{box}\left(b_i, \hat{b}_{\sigma(i)}\right) \tag{2}$$

The first term of Equation 2 guides the model to predict the object's category accurately. The second term helps to predict better bounding boxes. Here $c_i \neq \varnothing$ refers to detecting objects that are not dummy objects. Because each ground truth object is detected once, the equation 2 is applied to all ground truth objects in the image.

The optimal matching $\hat{\sigma}$ is estimated on a matching that includes ground truth objects and dumpy objects (defined by $\varnothing$). Because the boxes of dumpy objects are meaningless, the box loss is only valid if the box is not in the $\varnothing$ set. In contrast, the dumpy objects should be predicted as "No Object" correctly. Therefore, the classification loss should include the $\varnothing$ set. Motivated by the observation, the Hungary loss (L_H) [31] in Equation 3 is used to train the model.

$$L_H(y, \hat{y}) = \sum_{i=1}^{N} \left[-\log \hat{p}_{\sigma(i)}(c_i) + 1_{\{c_i \neq \varnothing\}} L_{\text{box}}\left(b_i, \hat{b}_{\sigma(i)}\right) \right]. \tag{3}$$

According to the literature reviews, influence scaling is critical in bounding box estimation. For example, a bounding box of a large object would have a width of 0.2, while a bounding box of a small object would have a width of 0.02. If the conventional Euler distance is used to measure the bounding box loss, the model may be too biased towards large objects and ignore small objects. Therefore, the Generalized Intersection over Union (GIOU) [32] loss function is introduced to calculate the bounding box loss together with the L1-norm [33] loss function. Denote $\lambda_{iou}, \lambda_{L1} \in R$ are hyper-parameters that control a learning process for GIOU

loss and L1-norm loss; the formula of the boundary loss is in Equation 4.

$$L_{\text{box}}\left(b_i, \hat{b}_{\sigma(i)}\right) = \lambda_{\text{iou}} L_{\text{iou}}\left(b_i, \hat{b}_{\sigma(i)}\right) + \lambda_{L1}\left\|b_i - \hat{b}_{\sigma(i)}\right\|_1 \tag{4}$$

3.2. Deformable DETR

Deformable Attention block. In the DETR model, the attention block transmits almost uniform attention weights to all pixels in the feature map. Thus, training requires a longer time. Using a deformable mechanism, the custom attention block only samples a small set of crucial points around a reference point, regardless of the feature map size in the spatial domain. This design allows for a reduction in training time. Besides, the computational complexity and needed memory are reduced. Figure 3 illustrates how deformable convolution is integrated into a multi-head self-attention layer, and the equation 5 is the model representation of the layer.

$$\text{DeformAttn}\left(z_q, p_q, x\right) = \sum_{m=1}^{M} W_m \left[\sum_{k=1}^{K} A_{mqk}.W'_m x\left(p_q + \Delta p_{mqk}\right)\right] \tag{5}$$

where,

- $x \in R^{C\times H\times W}$ is an input feature map.
- M is the sum of interactions, where $m \in [1, M]$.
- K is the total number of standard keys sampled (K < HW), index $k \in [1, K]$.
- $W'_m x$ is a linear projection of the input feature map. W'_m extract $value$ of a self-attention layer.
- Δp_{mqk} represents the sampling deviation of the k^{th} sampling point in the m^{th} attention head. Given a query z_q, a linear projection extracts a $2 \times (M * K)$ to represent K offset vectors for M attention head.
- A_{mqk} is the attention map of the k^{th} sampling point in the m^{th} attention output. Given a query z_q, a linear projection extracts a $M \times K$ tensor as an attention map A_{mqk}. The softmax function is applied on every head to ensure $\sum_{k=1}^{K} A_{mqk} = 1$.
- The term $\left[\sum_{k=1}^{K} A_{mqk}.W'_m x\left(p_q + \Delta p_{mqk}\right)\right]$ represents the interaction among sampling values to encode a new feature. The new feature is the output of a self-attention layer. Concatenating several self-attention features and then projecting the feature to output, we have a multi-head self-attention.

Multi-scale Deformable Attention Module. The multi-scale technique is commonly used to detect objects of different sizes. Feature maps at different scale levels detect objects from different distances. Multi-scaling techniques are integrated into a Deformable Attention block as Equation 6.

$$\text{MSDeformAttn}\left(z_q, \hat{p}_q, \left\{x^l\right\}_{l=1}^{L}\right) = \sum_{m=1}^{M} W_m \left[\sum_{l=1}^{L}\sum_{k=1}^{K} A_{mlqk} \cdot W'_m x^l\left(\varnothing_l\left(\hat{p}_q\right) + \Delta p_{mlqk}\right)\right] \tag{6}$$

where,

- L is the input feature level, index $l \in [1, L]$
- $\{x^l\}_{l=1}^{L}$ is the feature map at different scales; with $x^l \in R^{(C\times H_l\times W_l)}$
- Δp_{mlqk} represents the sampling deviation of the k^{th} sampling point at the l^{th} feature level and the m^{th} attention head.
- $\hat{p}_q \in [0;1]^2$ is the normalized coordinates. The upper left point is (0;0); and the lower right point is (1;1).
- The function $\varnothing_l(\hat{p}_q)$ will re-scale the normalized coordinates $\hat{p}_q$ to the input feature map of level l.

4. Experiment

4.1. Dataset Description

In order to demonstrate the effectiveness of the Deformable DETR framework for ship detection, we utilize a widely recognized large-scale dataset referenced in the work of [18]. This dataset originates from a comprehensive video monitoring system situated around Hengqin Island in Zhuhai City, China. It comprises a diverse range of ship images captured under different sea conditions, collected from 6:00 am to 8:00 pm daily.

For a fair comparison, we follow the setup outlined in [18] for preparing the training and testing sets. This approach is widely used in various research works such as Liu (2020) [12], Liu (2022) [7], Han (2021) [28], SDNet (2022) [11], and VIB (2023) [14], and has become a well-known benchmark. Under this setup, 80% of the dataset is used for training, and the remaining 20% is used for testing. The training and testing datasets are denoted as D_1^{Train} and D_1^{Test} respectively.

Some works (Biaohua_2022 [19] and Yani_2022 [17]) prepare a challenging scenario where 50% dataset is used for training and 50% dataset is used for testing. In this scenario, the training and testing datasets are denoted as D_2^{Train} and D_2^{Test} in our paper. Furthermore,

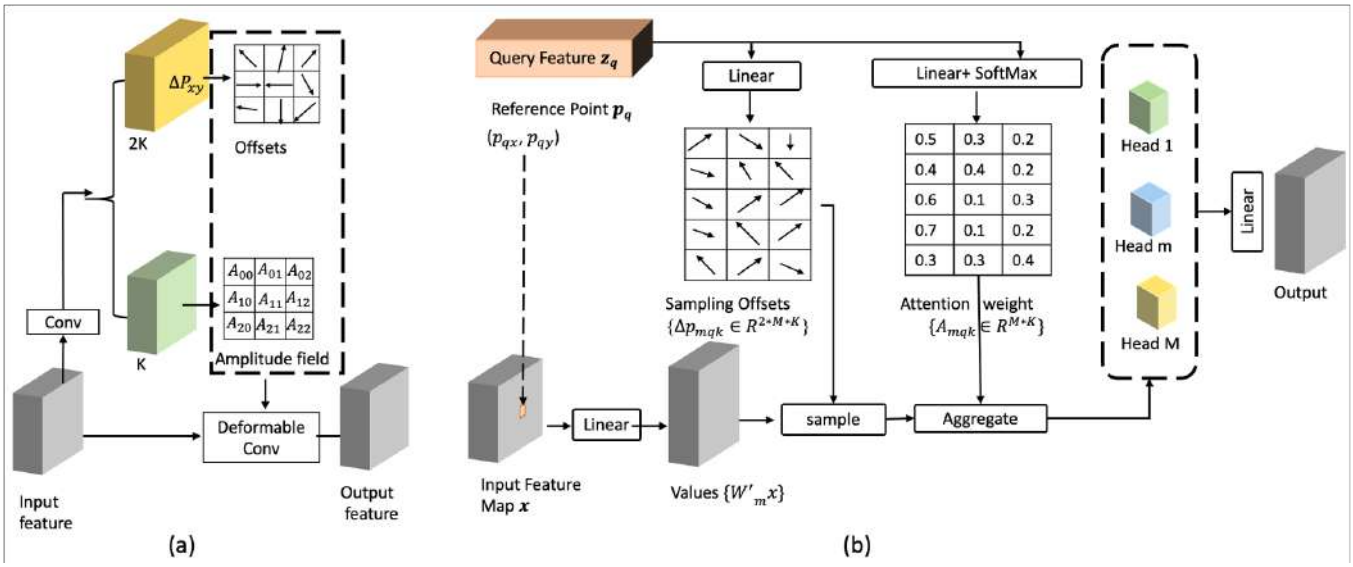


Figure 3. Illustration of the Deformable Attention module [16]. (a) The concept of deformable convolution. (b) Deformable convolution in multi-head attention.

Table 1. Performance comparisons of Deformable DETR with the number of queries. The best results are marked in **bold**.

Class	Deformable DETR											
	300 query			200 query			100 query			50 query		
	dets	recall	AP	dets	recall	AP	dets	recall	AP	dets	recall	AP
fishing boat	204449	0,913	0,798	208817	0,985	0,970	119774	**0,986**	**0,969**	96577	0,972	0,949
container ship	6300	0,989	0,894	11460	0,995	0,995	31083	0,995	0,989	18949	**0,998**	**0,997**
ore carrier	53520	0,987	0,923	49362	**0,997**	**0,992**	62122	0,996	0,989	18949	0,993	0,987
bulk cargo carrier	45541	0,985	0,912	47754	0,996	**0,992**	46490	**0,997**	0,986	76230	0,996	**0,989**
passenger ship	22498	0,968	0,610	16199	0,970	**0,957**	14866	0,968	0,936	20043	**0,972**	0,926
general cargo ship	17692	0,987	0,930	16408	0,995	**0,992**	75665	0,995	0,991	98537	**0,996**	**0,990**
mAP			0.844			**0.982**			0.977			0.973

to evaluate the performance on limited datasets, VIB_2023 [14] introduced S_1, S_2, and S_3 subsets which are randomly selected from D_2^{Train}. These datasets contain 30%, 70%, and 100% of D_2^{Train} , respectively. These subsets enable thorough investigation into the scalability and adaptability of proposed methods across varying dataset sizes and training conditions.

Our experiment uses AdamW optimizer, learning rate = 0.0001, weight decay = 0.0001, n_epoch=200, batch_size=8, $\lambda_{iou} = 2.0$, and $\lambda_{L1} = 5.0$. We use the reduce-mean operator on batch data and the reduce-sum operator on prediction output. The mAP is used to select the best model. These experiments are implemented based on the mmdetection library [34].

4.2. Hyper-parameter selection

As discussed in Section 3, the number of queries can affect the number of outputs of a DETR-based detector. Therefore, this section tries to select the most suitable parameter to control the model. We compare the performance when $n_{queries}$ receives a vault in the $[300, 200, 100, 50]$ list. The D_2^{Train} and D_2^{Test} are selected as the training and testing dataset to ensure a challenging setting. It means 50% of the dataset is used for testing.

The findings are presented in Table 1. In mmdetection, the default value for the number of queries ($n_{queries}$) is 300. It is observed that using the default setting leads to an increased number of detections (dets). For example, the number of detections for fishing boats is 204449. This tendency results in a decrease in average precision (AP) to 0.798, while the recall increases to 0.913. Additionally, the higher number of detections for fishing boats can be attributed to the higher frequency of the corresponding label in the dataset. By reducing $n_{queries}$, the bias in detections is mitigated. Specifically, when $n_{queries}$ are set to 200, 100, and 50, the detections for fishing boats decrease to 208817, 119774, and 96577 respectively. Furthermore, this reduction in queries minimizes the bias in detections across different ship categories. With $n_{queries} = 300$, the lowest number

Table 2. Performance comparisons of Deformable DETR given by various learning rates. The best results are marked in **bold**.

	Lr=10^{-3}			Lr=10^{-4}			Lr=10^{-5}		
	Dets	recall	AP	Dets	recall	AP	Dets	recall	AP
fishing boat	75600	0.122	0.001	24874	**0.971**	**0.912**	15869	0.986	0.877
container ship	12600	0.223	0.007	9193	**0.993**	**0.986**	17028	0.988	0.968
ore carrier	12600	0.254	0.005	33544	**0.994**	0.965	24892	0.988	0.919
bulk cargo carrier	12600	0.401	0.016	35317	**0.995**	**0.949**	21334	0.997	0.906
passenger ship	14000	0.211	0.001	14543	**0.952**	0.861	42789	0.982	0.760
general cargo ship	12600	0.361	0.009	22529	**0.960**	**0.969**	18088	0.994	0.916
mAP			0.006			**0.941**			0.891

of detections is for container ships (6300). However, when $n_{queries}$ = 50, the lowest number of detections is around 18949, and the detections for container ships, ore carriers, and passenger ships are quite similar.

Besides the $n_{queries}$, the learning rate (Lr) is another important factor that affects the system's performance. Default, Lr is set as 10e-4. In the revised version, we have modified the Lr to 10e-5 and 10e-3. The small set (S_1 dataset) is used for training, and the D_2^{Test} dataset is used for testing.

The results presented in Table 2 provide a quantitative evaluation of different learning rates. When a high learning rate (Lr=10e-3) is used, the model exhibits poor generalization, reflected in low recall and AP values. This is likely due to excessively large parameter updates during training, leading to instability or failure to properly converge. On the other hand, with a low learning rate (Lr=10e-5), while the model maintains high recall, the slight decrease in AP indicates that the learning rate may be too low, resulting in slower convergence and insufficient fine-tuning of parameters. The optimal learning rate is found to be (Lr=10e-4), where the model achieves the best balance between parameter updates and stability. At this rate, the model attains high precision and recall across all ship classes, making it the most effective choice in this scenario.

4.3. Compare with SoTA

This section compares our proposed method with SoTA on the mAP metric. For each method, we use a corresponding training and testing dataset. For instance, we use D_1^{Train} and D_1^{Test} to train our model and compare with Zhang_2022 [13], and Zhang_2021 [10], Liu_2020 [12], Liu_2022 [7], Han_2021 [28], and SDNet_2022 [11], and VIB_2023 [14]. Also, we use D_2^{Train} and D_2^{Test} to train another model and compare to Biaohua_2022 [19], Yani_2022 [17], and VIB_2023 [14].

Because the mAP is maximum when $n_{queries}$ = 200 for our method, as shown in Table 1, we select the setting in this experiment. The comparison among SoTA is reported in Table 3, and some conclusions can be drawn as below:

- Baseline framework is the key factor to have a better result. Cui_2019 [7] and Liu_2020 [12] base on YoloV3. Therefore, its performance is not as good as Liu_2022 [7], which is based on the SSD framework. Han_2021 [28] is based on YoloV4, and its performance does not show an improvement compared to Liu_2020 [12] base on YoloV3. Using the advantage YoloV5, SDNet_2022 [11] significantly improved compared with Liu_2020. The YoloV5 framework helps mAP increases up to 8% compared to the YoloV3 framework. VIB_2023 [14] is based on YoLoX, and our method is based on the DETR backbone. These frameworks have recently been advantageous methods for object detection. Hence, the results are better than others. It is worth noting that ship detection research typically leverages an object detection framework as its foundation, often with some custom modifications. Therefore, inheriting the capabilities of such a novel and powerful framework naturally leads to improved results.
- When the number of samples in the training set is reduced, DETR-based methods tend to work better than CNN-based methods. In the table, Biaohua_2022 [19] is a CNN-based detector, and Yani_2022 [17] is a DETR-based detector. The mAP given by Yani_2022 and Biaohua_2022 are 0.965 and 0.9963, respectively. However, the performance could be improved if we select a suitable hyperparameter. In our work, by setting $n_{queries}$ = 200 the mAP could improve to 0.981.
- Our method is comparable to the best CNN-based method for ship detection. If the training dataset has more samples than the testing dataset, our method is comparable to the Yolo-based method like VIB_2023 [14]. In detail, when D_1^{Train} and D_1^{Test} are used for training and testing, both mAPs for our method and VIB_2023 [14] are similar. However, when the number of samples in the training set is equal to that in the testing set, our method is slightly better than the VIB_2023

Table 3. Performance comparisons of various methods. The best results are marked in **bold**.

Method	Train+Val / Test (in %)	fishing boat	container ship	ore carrier	bulk cargo carrier	passenger ship	general cargo ship	mAP
Zhang_2022 [13]	90/10	0.824	0.940	0.859	0.915	0.787	0.914	0.873
Zhang_2021 [10]	90/10	-	-	-	-	-	-	0.946
Cui_2019 [9]	80/20	0.900	0.940	0.90	0.910	0.910	0.900	0.910
Liu_2020 [12]	80/20	-	-	-	-	-	-	0.908
Han_2021[28]	80/20	-	-	-	-	-	-	0.906
Liu_2022 [7]	80/20	-	-	-	-	-	-	0.964
SDNet_2022 [11]	80/20	**0.986**	0.995	**0.989**	0.990	0.982	0.989	0.988
VIB_2023 [14]	80/20	0.979	**1**	0.987	**0.994**	0.994	**0.993**	**0.991**
Ours (n_{query} = 200)	80/20	0.982	**1**	**0.989**	0.991	**0.995**	0.990	**0.991**
Yani_2022 (ESDT) [17]	50/50	-	-	-	-	-	-	0.593
Yani_2022 (DETR) [17]	50/50	-	-	-	-	-	-	0.965
Biaohua_2022 [19]	50/50	0.940	0.987	0.966	0.978	0.937	0.972	0.963
VIB_2023 [14]	50/50	0.970	0.986	0.984	**0.991**	**0.964**	0.989	0.98
Ours (n_{query} = 200)	50/50	0.970	**0.995**	**0.992**	0.992	0.957	**0.992**	**0.982**

[14]. In detail, when D_2^{Train} and D_2^{Test} are used for training and testing, our method is slightly better than the SoTA in VIB_2023 [14].

In Table 3, the results indicate that the DETR method generally outperforms the YoLoX-based method [14] when the number of training samples is reduced. Therefore, we designed an experiment using D_2^{Test} as the testing set and a subset of D_2^{Train} as the training set. These subsets are categorized into $Small$, $Medium$, and $Large$ levels, corresponding to S_1, S_2, and S_3 subsets. The results in Figure 4 compare our method with the state-of-the-art (SOTA) method proposed in VIB_2023 [14].

If the training dataset is S_1, the mAP given by our method improves 17.5% compared to VIB_2023 [14]. When the number of training samples is increased, the improvement is reduced. The enhancement on mAP is 3.3% if S_2 subset is used for training and if the training dataset is S_3, the mAPs from both settings are pretty equivalent. The observation clearly points out that the DETR-based method could help if the number of training samples is limited. Table 4 reports the detailed detection result for each ship category.

While DETR-based detectors can achieve higher accuracy on smaller datasets, it is important to also assess their computational complexity compared to CNN-based detectors. Table 5 presents a comparison of the computational performance between two object detection models: VIB_2023 [14] and Deformable DETR. Three key metrics are compared: Frames Per Second (FPS), GFlops, and the number of parameters (in millions). Both models show similar FPS, with DETR slightly outperforming VIB_2023 [14] at 12.6 FPS versus 12.39 FPS on a TiTanRTX GPU. However, DETR exhibits higher computational complexity, requiring 11.01 GFlops, compared to the more efficient 9.92 GFlops of VIB_2023 [14]. Despite its higher computational demands, DETR uses fewer parameters, with 39.82 million compared to VIB_2023's 55.33 million. This suggests that DETR maintains competitive speed with fewer parameters but at the cost of increased computational complexity.

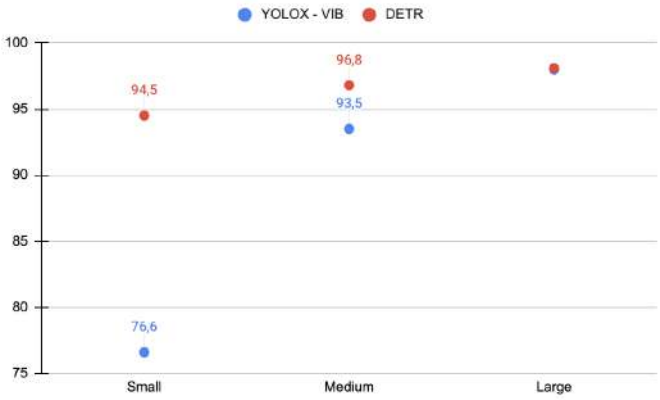


Figure 4. Performances when the number of training samples is limited. "Small" means 30% training samples, "Medium" means 70% training samples, and "Large" means 100% training samples from D_2^{Train}.

Table 4. Performance on small datasets. S_1 means 30% training samples, S_2 means 70% training samples, S_3 means 100% training samples from D_2^{Train}.

Scenario	Metrics	fishing boat	container ship	ore carrier	bulk cargo carrier	passenger ship	general cargo ship	mAP
S_1 by VIB	recall	0.878	0.941	0.924	0.913	0.657	0.946	
	AP	0.796	0.884	0.831	0.765	0.524	0.794	0.766
S_1 by DETR	recall	0.971	0.993	0.994	0.995	0.952	0.96	
	AP	0.912	0.986	0.965	0.949	0.861	0.969	0.941
S_2 by VIB	recall	0.940	0.984	0.962	0.971	0.891	0.964	
	AP	0.922	0.980	0.935	0.953	0.873	0.946	0.935
S_2 by DETR	recall	0.977	1	0.992	0.995	0.968	0.997	
	AP	0.958	0.995	0.967	0.978	0.922	0.986	0.968
S_3 by VIB	recall	0.978	0.986	0.990	0.995	0.972	0.993	
	AP	0.970	0.986	0.984	0.991	0.964	0.989	0.98
S_3 by DETR	recall	0.985	0.995	0.999	0.998	0.980	0.997	
	AP	0.970	0.995	0.992	0.992	0.957	0.992	0.982

Table 5. Complexity comparison between VIB-detector and DETR-detector.

	VIB_2023 [14]	DETR
FPS	12.39	12.6
GFlops	9.92	11.01
#parameters (M)	55.33	39.82

4.4. Ablation study of loss functions.

This section discuss an ablation study on training losses. Deformable DETR employs multiple loss functions for training, including focal loss, GIoU loss, and L1 loss, each serving a distinct purpose: focal loss for classification, GIoU for bounding box regression, and L1 for object detection. The combination of these losses ensures the success of the training process. While all loss functions are crucial, their contributions can be adjusted. By default, the weights are set at 2.0 for focal loss, 2.0 for GIoU loss, and 5.0 for L1 loss. To evaluate the impact of each, we reduced the weight of these losses by a factor of ten, one at a time, and compared the results to the default setting. The results in Table 6 demonstrate that L_{cls} (classification loss) is the most significant; reducing its weight leads to a significant performance drop. Conversely, reducing the object loss has less impact, with performance remaining close to the original setting. Localization is slightly affected by GIoU loss, as mAP decreases to 0.931 compared to 0.941 in the original setting.

Table 6. mAP comparisons of Deformable DETR when reducing training losses.

	L_{GIoU}	L_1	L_{cls}
fishing boat	0.899	0.899	0.227
container ship	0.976	0.985	0.246
ore carrier	0.953	0.947	0.212
bulk cargo carrier	0.951	0.966	0.151
passenger ship	0.845	0.849	0.0173
general cargo ship	0.964	0.969	0.160
mAP	0.931	0.936	0.178

4.5. Feature analysis

Experimental results in Section 4.3 point out that the DETR method is better than the CNN methods if the number of samples in the training set is limited. However, it is worth explaining why the DETR method can have a better result in the special scenario. The major difference between the two methods is the attention module in the DETR encoder. This module allows non-local interaction to learn a better feature. Therefore, we visualize the features given by both methods after a model's backbone, neck, and head. Given one input image, feature maps are extracted after one module. The sum of one feature map represents whether this feature map is important or not. Therefore, we selected 20 important feature maps for each backbone, neck, and head to create a heat map. The map is averaged from all important feature maps and represents key points on an image.

Figure 5 illustrates examples of heat maps generated from an input image. The first row displays feature maps from DETR, the second row shows heat maps from the VIB_2023 [14] method, which employs a feature selection loss for learning features, and the third row presents heat maps from YOLOX [29], which relies purely on CNN networks. The results indicate that DETR, with its attention mechanism, can better

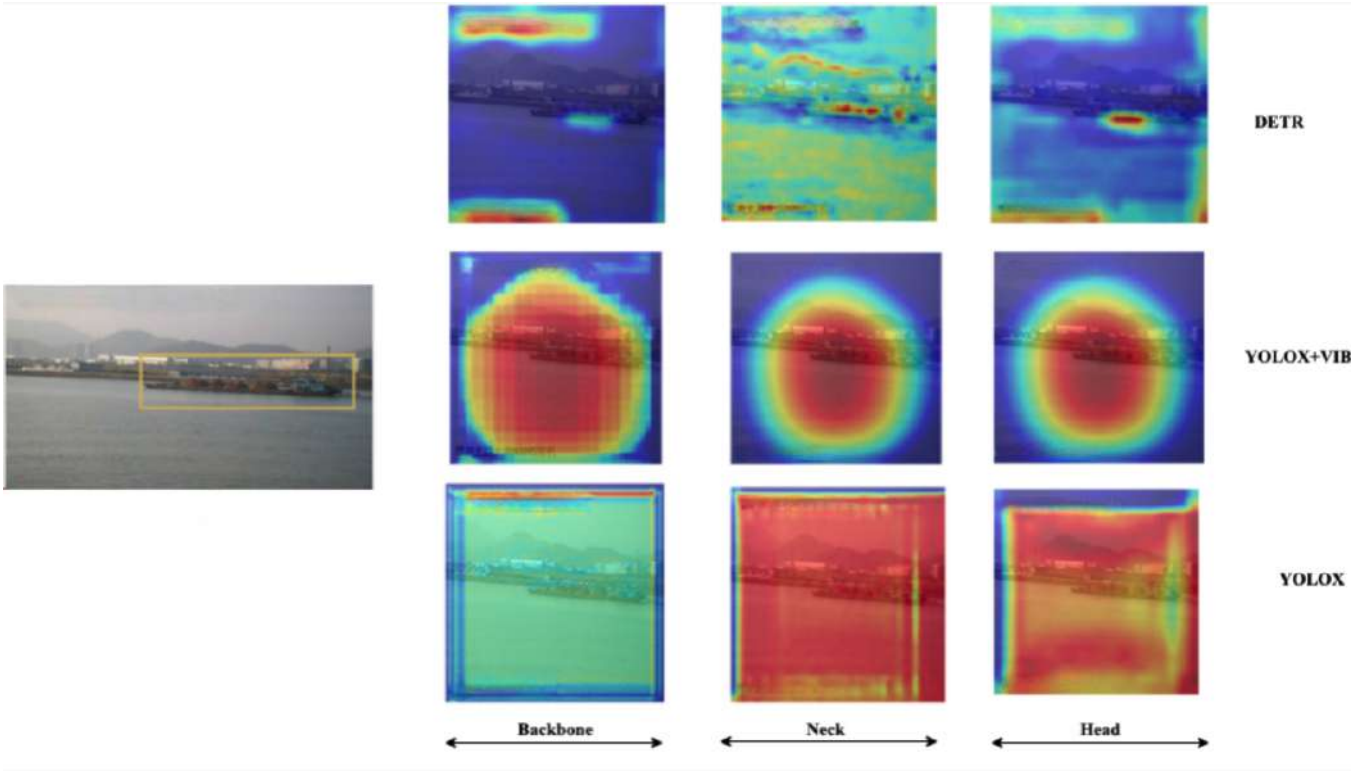


Figure 5. Feature maps on the classifier head, the neck and the backbone. (Text on original image had been removed.)

focus on non-background objects. For instance, after the head, the feature map highlights the text with a score on the survival system, and the ship is also highlighted, though not as prominently as the text. As the features are processed through the neck, higher semantic features are learned, causing the ship to become more prominent while the focus on the text diminishes. Key points are concentrated on the ship at the head, with reduced attention on the text.

In contrast, VIB_2023 [14] produces sparse heat maps where many pixels at the image's edge do not respond. However, these maps do not precisely focus on the object. This occurs because VIB_2023 [14] uses a feature selection loss to identify important features, leading to sparse and highly discriminated feature maps that do not exactly center on the object. The map distributions are quite similar overall, with slight improvements from the backbone to the head. Without the feature selection loss, the distribution of heat maps would likely be more uniform as shown in the third row.

5. Conclusion

This paper discusses a simple but efficient method for ship detection. By adjusting the number of object queries in the DETR model, we can have a comparable detector for ship detection on a large-scale dataset. In addition, the method shows a significant improvement if we do not have enough data in a training set. Heat map visualization explains why our method could be better than CNN-based methods. The features are learned to focus on non-background information, and the discriminated level of the feature map is better.

References

[1] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," 2013.

[2] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 6, pp. 1137–1149, 2017.

[3] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "SSD: Single shot MultiBox detector," in *Computer Vision – ECCV 2016*, pp. 21–37, Springer International Publishing, 2016.

[4] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, (Las Vegas Blvd), pp. 779–788, 2016.

[5] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," 2016.

[6] Q. Li, D. Xiao, and F. Shi, "A decoupled head and coordinate attention detection method for ship targets in sar images," *IEEE Access*, vol. 10, pp. 128562–128578, 2022.

[7] J. Zheng and Y. Liu, "A study on small-scale ship detection based on attention mechanism," *IEEE Access*, vol. 10, pp. 77940–77949, 2022.
[8] H. Li, L. Deng, C. Yang, J. Liu, and Z. Gu, "Enhanced yolo v3 tiny network for real-time ship detection from visual image," *IEEE Access*, vol. 9, pp. 16692–16706, 2021.
[9] H. Cui, Y. Yang, M. Liu, T. Shi, and Q. Qi, "Ship detection: An improved yolov3 method," in *OCEANS 2019 - Marseille*, pp. 1–4, 2019.
[10] T. Liu, B. Pang, L. Zhang, W. Yang, and X. Sun, "Sea surface object detection algorithm based on yolo v4 fused with reverse depthwise separable convolution (rdsc) for usv," *Journal of Marine Science and Engineering*, vol. 9, no. 7, 2021.
[11] M. Zhang, X. Rong, and X. Yu, "Light-sdnet: A lightweight cnn architecture for ship detection," *IEEE Access*, vol. 10, pp. 86647–86662, 2022.
[12] T. Liu, B. Pang, S. Ai, and X. Sun, "Study on visual detection algorithm of sea surface targets based on improved yolov3," *Sensors*, vol. 20, no. 24, 2020.
[13] Q. Zhang, Y. Huang, and R. Song, "A ship detection model based on yolox with lightweight adaptive channel feature fusion and sparse data augmentation," in *2022 18th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pp. 1–8, 2022.
[14] D.-D. Ngo, V.-L. Vo, T. Nguyen, M.-H. Nguyen, and M.-H. Le, "Image-based ship detection using deep variational information bottleneck," *Sensors*, vol. 23, no. 19, 2023.
[15] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, *End-to-End Object Detection with Transformers*, pp. 213–229. 11 2020.
[16] X. Zhu, W. Su, L. Lu, B. Li, X. Wang, and J. Dai, "Deformable detr: Deformable transformers for end-to-end object detection," arXiv, 2020.
[17] Y. Zhang, M. J. Er, W. Gao, and J. Wu, "High performance ship detection via transformer and feature distillation," in *2022 5th International Conference on Intelligent Autonomous Systems (ICoIAS)*, pp. 31–36, 2022.
[18] Z. Shao, W. Wu, Z. Wang, W. Du, and C. Li, "Seaships: A large-scale precisely annotated dataset for ship detection," *IEEE Transactions on Multimedia*, vol. 20, no. 10, pp. 2593–2604, 2018.
[19] B. Ye, T. Qin, H. Zhou, J. Lai, and X. Xie, "Cross-level attention and ratio consistency network for ship detection," in *2022 26th International Conference on Pattern Recognition (ICPR)*, pp. 4644–4650, 2022.
[20] R. Girshick, "Fast r-cnn," in *2015 IEEE International Conference on Computer Vision (ICCV)*, (Santiago, Chile), pp. 1440–1448, 2015.
[21] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," 2015.
[22] J. Redmon and A. Farhadi, "Yolo9000: Better, faster, stronger," 2016.
[23] Q. Chen, Y. Wang, T. Yang, X. Zhang, J. Cheng, and J. Sun, "You only look one-level feature," in *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 13034–13043, 2021.
[24] X. Lei, H. Pan, and X. Huang, "A dilated cnn model for image classification," *IEEE Access*, vol. 7, pp. 124087–124095, 2019.
[25] Y. Li, H. Mao, R. Girshick, and K. He, "Exploring plain vision transformer backbones for object detection," *arXiv preprint arXiv:2203.16527*, 2022.
[26] Z. Zong, G. Song, and Y. Liu, "Detrs with collaborative hybrid assignments training," *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 6725–6735, 2022.
[27] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 9992–10002, 2021.
[28] X. Han, L. Zhao, Y. Ning, and J. Hu, "Shipyolo: An enhanced model for ship detection," *Journal of Advanced Transportation*, vol. 2021, pp. 1–11, 06 2021.
[29] Z. Ge, S. Liu, F. Wang, Z. Li, and J. Sun, "Yolox: Exceeding yolo series in 2021," 2021.
[30] Z. Zhang, L. Zhang, Y. Wang, P. Feng, and R. He, "Shiprsimagenet: A large-scale fine-grained dataset for ship detection in high-resolution optical remote sensing images," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 14, pp. 8458–8472, 2021.
[31] H. W. Kuhn, "The Hungarian Method for the Assignment Problem," *Naval Research Logistics Quarterly*, vol. 2, pp. 83–97, March 1955.
[32] H. Rezatofighi, N. Tsoi, J. Gwak, A. Sadeghian, I. Reid, and S. Savarese, "Generalized intersection over union: A metric and a loss for bounding box regression," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, (Long Beach, CA, USA), pp. 658–666, 2019.
[33] P. Hinz, "The layer-wise l1 loss landscape of neural nets is more complex around local minima," 2021.
[34] K. Chen, J. Wang, J. Pang, Y. Cao, Y. Xiong, X. Li, S. Sun, W. Feng, Z. Liu, J. Xu, Z. Zhang, D. Cheng, C. Zhu, T. Cheng, Q. Zhao, B. Li, X. Lu, R. Zhu, Y. Wu, J. Dai, J. Wang, J. Shi, W. Ouyang, C. C. Loy, and D. Lin, "MMDetection: Open mmlab detection toolbox and benchmark," *arXiv preprint arXiv:1906.07155*, 2019.

IEEE Industrial Electronics Society

ASIA&PACIFIC-Ulsan Chapter

IEEE

THE 3rd IEEE INTERNATIONAL WORKSHOP ON INTELLIGENT SYSTEMS

IWIS 2023, Ulsan, Republic of Korea

Aug. 9-11, 2023

PROGRAM

울산대학교 UNIVERSITY OF ULSAN

Break		**14:30–15:00**	**Break**	
Oral Session WB(8) Machine Learning and Deep Learning Systems I Chairs: Prof. Kang-Hyun Jo, Co-Chairs: Prof. Van-Thanh Hoang	**Grand Room**	**15:00–15:15**	**Hierarchical Vision Transformers with Shuffled Local Self-Attentions**	Xuan-Thuy Vo, University of Ulsan, Republic of Korea
		15:15–15:30	**Multi-Band Feature Fusion in Satellite Images for Landscape Classification**	Russo Ashraf, University of Ulsan, Republic of Korea
		15:30–15:45	**Gender Recognizer based on Human Face using CNN and Bottleneck Transformer Encoder**	Adri Priadana, Muhamad Dwisnanto Putro, University of Ulsan, Republic of Korea
		15:45–16:00	**Energy Marketplace Platform using Blockchain Technology**	Ameni Boumaiza, Scientist, Qatar.
		16:00–16:15	**Gesture Recognition in Indonesian Sign Language Using Hybrid Deep Learning Models**	Muhammad Yusuf Daffa Izzalhaqqi, Wahyono Wahyono, Universitas Gadjah Mada, Indonesia
		16:15–16:30	**A Vision-based Container-Code Checking System: Case Study at International Terminal**	Hoang Anh Phan Van, Vietnam
		16:30–16:45	**Enhancing brain tumor classification through customization of the Vision Transformer Learning**	Xuan-Khoa Thai-Hoang, Van-Dung Hoang, HCMC University of Technology and Education, Vietnam
		16:45–17:00	**A Modified UNet for Skin Lesion Segmentation using Transfer Learning**	Afroza Akter, Kaushik Deb, Sharmistha Chanda Tista, Chittagong University of Engineering and Technology, Bangladesh; Kang-Hyun Jo, University of Ulsan, Republic of Korea
Break		**17:00–17:20**	**Break**	
Social Active Event		**17:20–18:00**	**University of Ulsan → Welcome Reception at the Shilla Stay Ulsan**	
		18:00–20:00	**Welcome Reception at the Shilla Stay Ulsan (Dinner and Drinks)**	

A Vision-based Container-Code Checking System: Case Study at International Terminal

Duc-Dat Ngo
Ho Chi Minh City University of Technology and Education
Ho Chi Minh City, Vietnam
datnd.ncs@hcmute.edu.vn

Van-Hoang-Anh Phan
Intelligent Systems Laboratory
Ho Chi Minh City University of Technology and Education
Ho Chi Minh City, Vietnam
21151070@student.hcmute.edu.vn

Huynh-The Pham
Faculty of Electrical and Electronics Engineering
FPT University
Ho Chi Minh City, Vietnam
theph@fe.edu.vn

Tien-Tan Be
Tan Cang Technical Services JSC
Ho Chi Minh City, Vietnam
betientandn@gmail.com

Van-Binh Nguyen
Institute of Engineering-Technology
Thu Dau Mot University
Ho Chi Minh City, Vietnam
binhnv@tdmu.edu.vn

My-Ha Le*
Faculty of Electrical and Electronics Engineering
Ho Chi Minh City University of Technology and Education
Ho Chi Minh City, Vietnam
halm@hcmute.edu.vn

***Abstract*—Checking the container code at an international terminal is necessary to ensure safety, security, and compliance in the global trade ecosystem. By implementing such a system, ports can mitigate risks, prevent illegal activities, and facilitate the smooth flow of legitimate cargo while protecting the interests of all stakeholders involved. This paper introduces a vision-based container-code-checking system built on a versatile and reasonably priced hardware platform with promising performance. More concretely, the proposed system includes two different stages executed continuously. To begin with, the CRAFT is packaged in EasyOCR to detect the container-code area. Subsequently, a Convolutional Neural Network combines with Spatial Transformer Network to classify each character of the detected container-code area. From that, our system facilitates data collecting and analysis related to container movements, inspections, and compliance. The demo video can be watched here: https://www.youtube.com/watch?v=JsRZE0k_lfM**

***Index Terms*—classification, container-code-checking system, deep learning, detection, and international terminal.**

I. Introduction

Container-checking code systems in security gates are becoming increasingly important as global trade and commerce have become more complex. These systems are designed to help monitor the movement of goods in and out of ports, airports, and other transport hubs to ensure that they comply with regulations and do not threaten security.

One of the primary benefits of container-checking code systems is that they help to prevent the smuggling of illegal and dangerous goods. These systems can detect hidden compartments and false bottoms in containers and identify containers containing certain types of goods prohibited or restricted by international trade regulations. Another critical benefit of container-checking code systems in security gates is that they help to ensure that goods are correctly declared and accounted for, which can help to prevent fraudulent activity, such as mislabeling or undervaluing goods to avoid paying fees or tariffs.

*Corresponding author

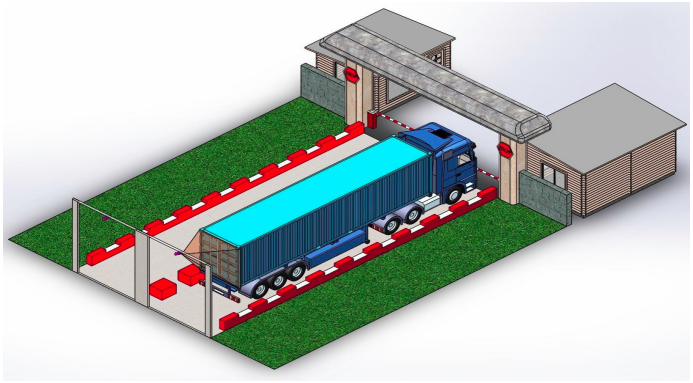


Fig. 1: A setup to collect data in the terminal gate.

Overall, container-checking code systems play an important role in maintaining the security and integrity of international trade, which is essential for ensuring the safety of people and communities worldwide. These systems help to prevent the movement of illicit goods while also helping to promote transparency and accountability in the global supply chain.

The container code is considered in this study, including seven digits and four letters. The two most popular solutions are radio frequency identification (RFID) and optical character recognition (OCR) [1]. Recognition of container code using radio frequency identification (RFID) technology in the protocol [2] to improve container terminal efficiency. The RFID-based solution has near-perfect accuracy but requires high installation and maintenance costs [1]. For this reason, OCR-based container code recognition was used in our article. OCR technology in natural scene images has been extensively studied for license plates, road signs, traffic signs,

and especially container code recognition. In the article [3], the authors used OCR and a new deep learning-based approach for automatic number plate recognition. In the paper [4], the author proposes a three-stage recognition of the license plates based on the OpenCV Engine and Tesseract OCR, consisting of license plate recognition, character segmentation, and character recognition.

Many studies have used edge statistics because characters have more edges than other parts of the image [1]. Based on the container's code area and surface color, a container code localization algorithm using the analysis of the Maximally Stable Extremal Region (MSER) and its associated domain has been proposed in the paper [5]. The article [6] presents an adaptive deep-learning platform for container code localization and recognition, noisy text fields are removed by the Adaptive Result Aggregation (ASA) algorithm. A real-time boundary-based container code text segmentation network has been proposed in the paper [7] that can accurately localize text in real-time. Although these approaches accomplish outstanding results, they are prone to be disturbed by various noises and cannot identify the best regions for code detection.

Given mentioned above, this paper introduces a vision-based container-code-checking system built on a versatile and reasonably priced hardware platform with promising performance. More concretely, the proposed system includes two different stages executed continuously. To begin with, the CRAFT is packaged in EasyOCR to detect the container-code area. Subsequently, a Convolutional Neural Network combines with Spatial Transformer Network to classify each character of the detected container-code area. From that, our system facilitates data collecting and analysis related to container movements, inspections, and compliance. To sum up, our main contributions are recapped as follows:

- Our system was built on a modular and cost-effective hardware platform.
- The proposed system can operate well with actual data collected from Tan Cang Company. Also, this system is currently being considered to scale up and deploy Tan Cang Technical Services JSC.

II. System Overview

A. Hardware Platform

We have devised a two-part system to facilitate remote monitoring as illustrated in Fig. 2. The first part is a slave component installed on the harbor gate. At the same time, a second is a master unit that processes deep learning algorithms and sends control signals to operate the actuators. The master unit serves as a Processing Unit (PU), which utilizes a communication technique called LoRa (Long Range) to analyze all the signals. The remaining branch utilizes the Arduino Uno as the primary microcontroller. Its responsibilities include receiving maneuvering information, managing barriers via relays, and indicating warnings through LEDs.

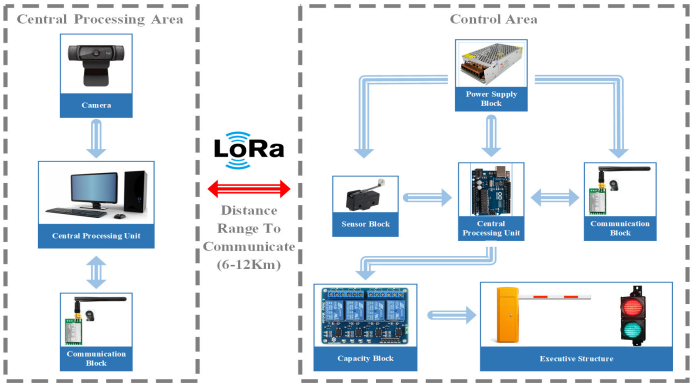


Fig. 2: Hardware overview.

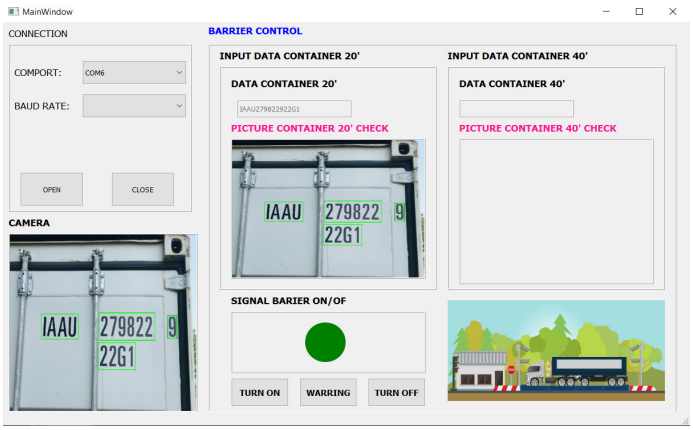


Fig. 3: Graphic user interface designed for monitoring.

B. User Interface

An interface is created to enhance the user experience and provide a means to monitor container code, as shown in Fig. 3. It assists users in verifying the correctness of the code should the main algorithm fail. This UI includes three main components: show types of containers, image-captured frame, and barrier manual control.

C. Container-Code Configuration

According to Fig. 4, the five components of the container identification system consist of the Owner code, Product Group Code, Serial Number, Checking digit, Type, and Size of the container with the ISO standard.

III. Methodology

The proposed system includes two different stages executed continuously. To begin with, the CRAFT is packaged in Easy-

Fig. 4: Code Configuration.

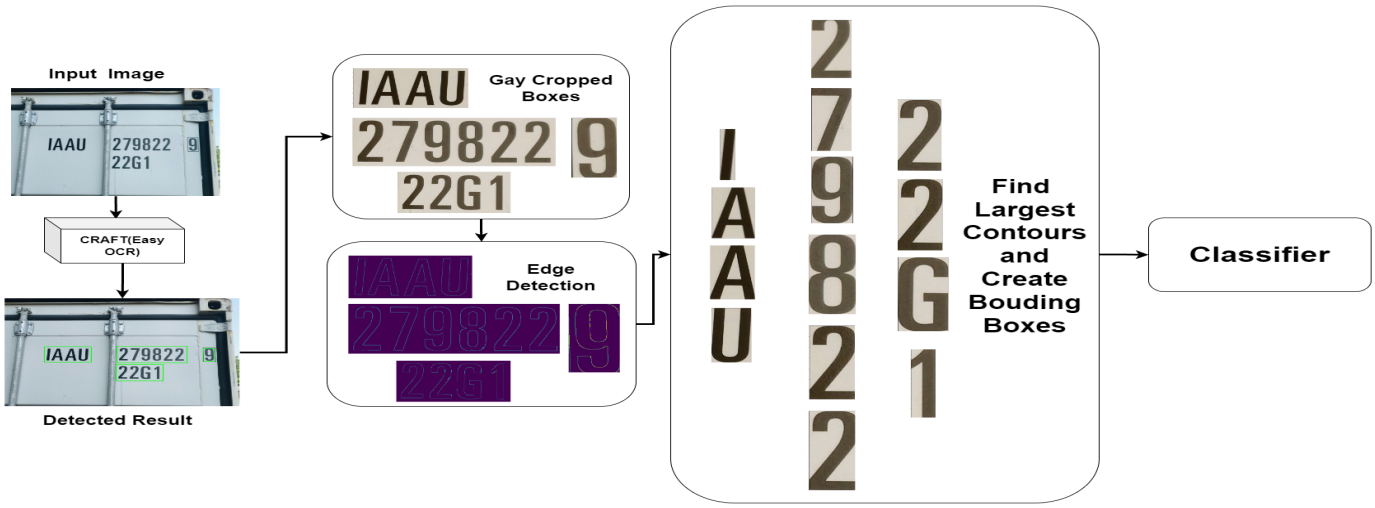


Fig. 5: The pipeline of character parsing process.

OCR to detect the container-code area. Subsequently, a Convolutional Neural Network combines with Spatial Transformer Network to classify each character of the detected container-code area. From that, our system facilitates data collecting and analysis related to container movements, inspections, and compliance. The overall architecture is depicted in Fig. 5.

A. EasyOCR Framework

EasyOCR [8] is a plug-in framework allowing computer vision engineers to conduct Optical Character Recognition efficiently. It integrates several OCR engines and deep learning models that can accurately recognize text in various languages and formats, including printed and handwritten text, machine-printed text, and digits. According to the explanation [9], it consists of localization and recognition stages. As for text localization, the framework utilizes CRAFT [9], which emerged as a promising model for text detection recently. Recently, the latest version is None-VGG-BiLSTM-CTC integrated into the API. It supports over 70 languages, including Arabic, Chinese, English, French, Hebrew, Hindi, Japanese, Korean, Russian, and Spanish. Despite its data variation, the classifier is sometimes unsatisfied in real-time environments. In our cases, the model constantly misclassifies some characters, such as 6 and G.

B. Classifier

In this part, we suggest a lightweight recognizer to reinforce the reliability of the whole system. In many cases, due to the differences between experimental domains, the final predictions may not be optimized if we used one-stage detection within data from just a single dataset. Thus, we diversified the learning domains using another dataset called Char74k [10]. In detail, having been extracted by the detection model of the EasyOCR framework, the bounding boxes are then resized and fed as input for ST-CNN.

1) Spatial Transformer Network: There is the fact that the cropped bounding boxes do not appear ideally in an image frame. Some problems that can be considered here are spatial effects on the foreground, such as scale, rotation, etc. In order to solve this problem, the paper [11] proposed an idea to perform the transformation to feature maps, including cropping, scaling, and rotating automatically. This work facilitates the classifiers to reduce the burden and boost their accuracy. Hence, we applied the STN to learn transformation parameters, and STN is divided into three components: Localisation network, Grid generator, and Sampler.

Localisation network (LN) is a simple local CNN or MLP (Multiple Layers Perceptron) with the inputs being feature maps. In our case, we placed this module at the beginning of the whole model so that the input image would be directly fed as input for LN. The output of LN is a matrix containing six parameters θ for 2D affine transformation calculation τ_θ. The spatial transformation has a form $A_\theta = \begin{bmatrix} \theta_{11} & \theta_{12} & \theta_{13} \\ \theta_{21} & \theta_{22} & \theta_{23} \end{bmatrix}$, which means the output layer of LN must contain six nodes.

Grid Generator would generate a sampling grid depending upon generated parameters θ. In this step, the Grid Generator takes the responsibility of irritating over the regular grid G of the target image. Then it makes an inverse transformation T_θ to find out the corresponding positions in the input image. Specifically, individual points in the input image are transformed using a pointwise transformation illustrated in the (1) to their corresponding positions in the output image using the previously learned transformation parameters.

$$\tau(\begin{bmatrix} x_{input} \\ y_{input} \end{bmatrix}) = A_\theta \times \begin{bmatrix} x_{output} \\ y_{output} \\ 1 \end{bmatrix} = \begin{bmatrix} \theta_{11} & \theta_{12} & \theta_{13} \\ \theta_{21} & \theta_{22} & \theta_{23} \end{bmatrix} \times \begin{bmatrix} x_{output} \\ y_{output} \\ 1 \end{bmatrix}, \quad (1)$$

where the element 1 is added to the homogenous vector $\begin{bmatrix} x_{output} \\ y_{output} \end{bmatrix}$ to represent a point in a homogenous coordinate

system and become multipliable with the transformation matrix.

Sampler plays a crucial role in warping the input picture following the learned transformation parameters. In the original paper, the authors used the bilinear interpolation technique to produce smoother outputs that outperformed other techniques in practical experiments. Mathematically, the bilinear interpolation can be represented by the following equation:

$$f(x,y) = (1-\alpha)(1-\beta)f(x_1,y_1) + \alpha(1-\beta)f(x_2,y_1) + (1-\alpha)\beta f(x_1,y_2) + \alpha\beta f(x_2,y_2), \quad (2)$$

where:

- The values of $f(x_1, y_1)$, $f(x_1, y_2)$, $f(x_2, y_1)$, $f(x_2, y_2)$ indicate the four nearest neighbors pixels.
- The α value reflects the relative distance between the x-coordinates of the output pixel and its left neighbor pixel.
- The β value corresponds to the variance in y-coordinate between the output pixel and its top neighbor pixel.

2) The Integrated Classifier: A legendary architecture was employed as a VGG16 classifier consisting of 16 layers of convolutional and fully connected neural network layers. Additionally, the VGG16 design was incredibly distinctive in allowing deeper neural networks to perform tasks like recognizing objects more effectively. By integrating the Spatial Transform (ST) module, the feature maps would be better enhanced when fed to the VGG16 model. The architecture of the whole classifier is depicted in Fig. 6.

3) The Pipeline of System: The Algorithm 1 processes an input image with container code by iterating through a list of bounding boxes (B_f). It crops the image (I) based on each bounding box (b_i), then converts the cropped image to grayscale(I_G), applies thresholding to create a binary image (I_T), find contours in the thresholded image (I_T) using the FindContour function and store in the variable C. Next, sorting the contours in the variable C from top-left to bottom-right is based on the *SortContours* function. Subsequently, the *FilterContour* function was leveraged to select contours according to their area, ensuring only relevant contours are considered. In the next stage, the image is further cropped based on the refined contour coordinates (x, y, w, h), and the cropped image is assigned in the variable I_c'. After obtaining the character-cropped images, the ST-VGG16 model processes these images to extract information. This process is repeated for each bounding box, and the processed results are outputted and stored in the variable r. Afterward, the individual predicted character (r) was append to the list R. Overall, the code utilizes various techniques and models to extract information from the container code in the input image.

Algorithm 1

Input: RGB image I
Outputs: The predicted characters
Begin
$B_f \longleftarrow [b_1, b_2, ..., b_n]$ /* B_f is the list bounding box.
$R \longleftarrow []$ /* R is the list of predicted character
For b_i **in** B_f
I_c = CropImages (b_i) /*Crop the input image based on the bounding box coordinates
I_G = ConvertColortoGray(I_c) /* Convert the cropped image to grayscale
I_T = Threshold(I_G) /* Apply image thresholding
C = FindContour(I_T) /* Find contours in the thresholded image
C = SortContours (C) /* Sorting contours top-left to bottom-right.
For C_i **in** C
$x, y, w, h \longleftarrow C_i$
FilterContour(x, y, w, h)/* Choose the contours relied on area of them
$I_{c'}$ = CropImages(x, y, w, h)
r = STN-VGG16($I_{c'}$)
R.append(r)
Return *R*
End

TABLE I: Trainning Parameters.

Parameters	Value
Batch Size	256
Learning rate	$4e^{-4}$
Momentum	0.9
Epoch	150

IV. Experimental Results

A. Hardware Setup

To build up a hardware platform for this study, we used reasonably priced components illustrated in Fig. 7. In detail, we employed two microcontrollers Arduino Uno to handle input and output signals. The first microcontroller receives signals from a computer and utilizes a LoRa SX1278 433Mhz transceiver module to transmit signals to the second microcontroller. On the other hand, the second microcontroller is responsible for processing the received signals from the first microcontroller and controlling various devices in the container terminal, such as signal LED lights through a Relay module and a DC motor for managing the opening and closing of the barrier. These components are affordable, priced totally at around 30 dollars, making them highly suitable for product research and development purposes. And in this system, we tried to design modules with functions and programming methods similar to facilities in the harbor. This work would make the scaling period in the future more convenient. Furthermore, the image-parsing process was executed on Laptop with a graphic card NVIDIA GTX 1650, and module 5 was plugged into the Laptop to receive a signal through UART protocol.

Regarding training configuration, the classifier was built on the Pytorch framework with Ubuntu 18.04 and CUDA 10.2. As for the training process, the parameters were set up as Table. I. The Adam optimizer was used to improve the training phase's performance. The whole model was also trained on a server with the graphic card being NVIDIA Tesla T4. This server is used for estimating the execution time of the model and the accuracy of the models.

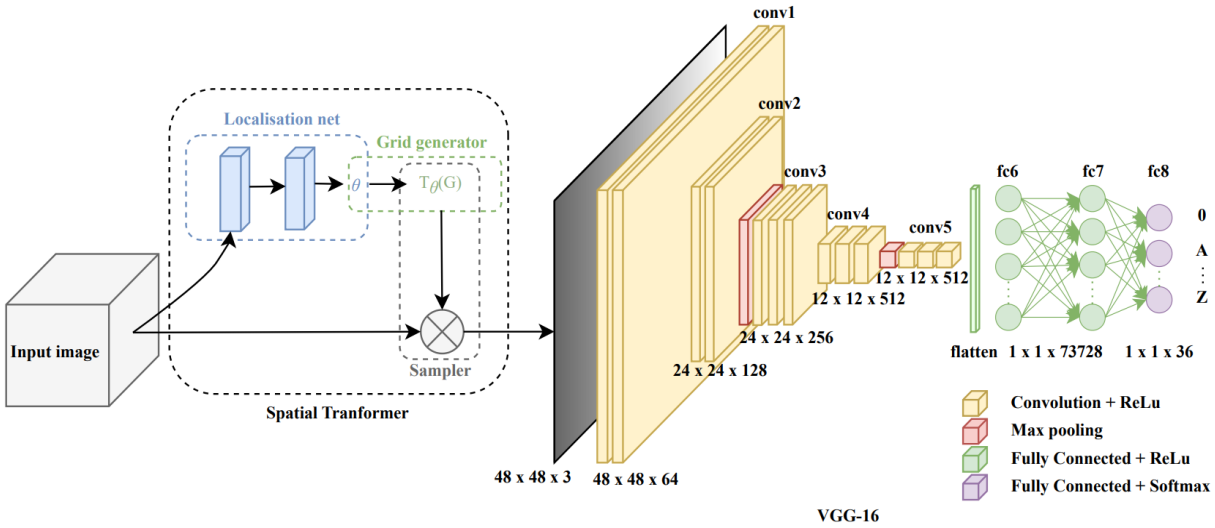


Fig. 6: The Spatial Transform VGG-16 classifier.

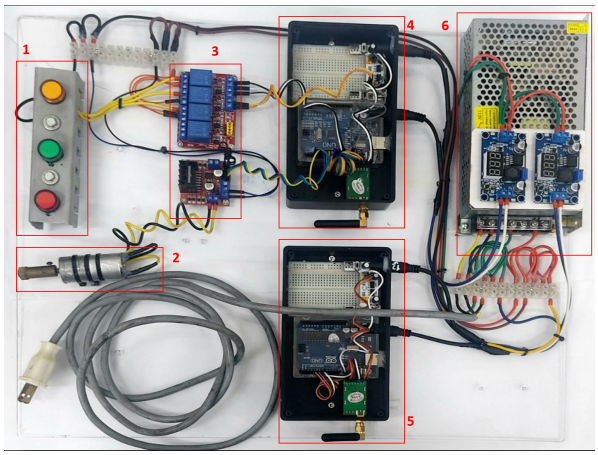


Fig. 7: Hardware platform. 1) signal LED, 2) DC Motor represents a barrier in practice, 3) Drivers including relays and bridge circuit, 4) micro-controller receives signals from LoRA and maneuvers actuators, 5) micro-controller is used for transmitting a signal from computer to receiver, 6) power unit.

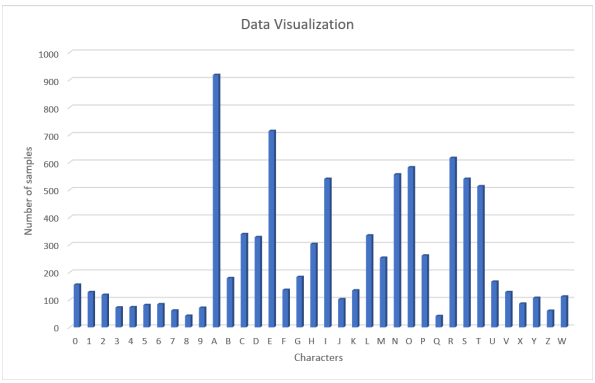


Fig. 8: Data visualization of the number of samples per class.

B. Data Description

This section describes the dataset (Chars74K) used to recognize the individual cropped character. The standard data was built from Google Street View images [10]. Because the letter on the container door is capitalized, we just leveraged characters, including digits and capital letters. Furthermore, we merged two data sets GoodImg [10] and BadImg [10] for the training process and split the data with a 9:1 ratio. Notably, a total dataset comprises 9116 images from different classes. Due to the imbalance among classes in the dataset depicted in Fig. 8, we applied multiclass Focal Loss [12] to mitigate the imbalance constraints. The Focal Loss is illustrated as follows:

$$FL(p_t) = -\alpha_t(1-p_t)^{\gamma}\log(p_t), \tag{3}$$

in that:

- $\alpha_t = 1 - \frac{N_t}{N}$ and N_t are the number of samples of class t and N is the total number of samples in the dataset.
- p_t is the predicted probability of true class.
- γ is the focusing hyperparameter, we select $\gamma = 2$.

C. Results and Evaluation

1) Detection Performance: There is a fact that the results of this phase have a profound impact on the subsequent steps within the processing pipeline. Hence, in this work, we tried to set up a camera angle to directly capture the Region of Interest (RoI). Detection results were depicted in Fig. 9.

2) Classification Performance: The image-processing algorithm separated single characters after detecting individual boxes from container door surfaces. This step would stably operate as long as the illuminating condition in RoI is guaranteed and the cropped bounding boxes are precise. Ultimately, to perform and prove the effectiveness of the enhanced classifier, the quantitative results are illustrated in Table. II were drew

Fig. 9: Detection results with various environments.

(a)

(b)

Fig. 10: Cropping samples with its binary images. (a) is the result of one container with the code being $'CCLU778420445G1'$, (b) is the result of another container with the code being $'WHSU514790045G1'$.

after comprehensive experiments, while the qualitative cropped images used to feed the STN-VGG16 model were depicted in Fig. 10. These numbers show the classifier's performance is quite good in our experiments. The whole system achieved promising results with the processing time being around 2.25s on average.

V. Conclusion

To sum up, this paper presents a container-code checking system utilizing computer-based methods. Concretely, the system first detects and crops bounding boxes from the original images. Next, several image-processing techniques, including grayscale thresholding or finding contours, and so on, were applied to separate each individual character. Lastly, these provided characters act as input for a classifier called STN-VGG16. The comprehensive results with reliable accuracy reveal that our proposed system could be scaled up and developed in the future. Furthermore, this system is currently considered to be applied and tested in an international harbor.

TABLE II: The comparison of different kinds of models in text classification. (C) is using Cross-Entropy Loss, and (F) is using Focal loss. PR is the number of parameters.

Metrics / Models	Accuracy Train	Accuracy Test	FPS	FLOPs ($\times 10^9$)	PR (M)
VGG16(C)	72.84	70.72	300	5.911	52.49
STN-VGG16(C)	90.79	89.74	260	5.915	52.51
STN-VGG16(F)	98.88	94.09	260	5.915	52.51
HOG [13]	-	76.8	-	-	-

References

[1] Y. Yoon, K.-D. Ban, H.-S. Yoon, and D. Kim, "Automatic container code recognition from multiple views," *ETRI Journal*, vol. 38, 05 2016.

[2] K. Sung-Soo, L. Myoun-Soo, S. Yong-Seok, N. Ki-Chan, and K. Kyu-Suk, "A study on the application of rfid to container terminals," *Journal of Korean navigation and port research*, vol. 29, 12 2005.

[3] Y. Shambharkar, S. Salagrama, K. Sharma, O. Mishra, and D. Parashar, "An automatic framework for number plate detection using ocr and deep learning approach," *International Journal of Advanced Computer Science and Applications*, vol. 14, no. 4, 2023.

[4] A. Agbemenu, J. Yankey, and E. O., "An automatic number plate recognition system using opencv and tesseract ocr engine," *International Journal of Computer Applications*, vol. 180, pp. 1–5, 05 2018.

[5] M. Weng, Q. Liu, and J. Guo, "Mser and connected domain analysis based algorithm for container code locating process," in *2017 International Conference on Industrial Informatics - Computing Technology, Intelligent Technology, Industrial Information Integration (ICIICII)*, 2017, pp. 83–86.

[6] R. Zhang, Z. Bahrami, T. Wang, and Z. Liu, "An adaptive deep learning framework for shipping container code localization and recognition," *IEEE Transactions on Instrumentation and Measurement*, vol. 70, pp. 1–13, 2021.

[7] K. Liu, C. Sun, and H. Chi, "Boundary-based real-time text detection on container code," in *2021 International Symposium on Computer Science and Intelligent Controls (ISCSIC)*, 2021, pp. 78–81.

[8] D. Vedhaviyassh, R. Sudhan, G. Saranya, M. Safa, and D. Arun, "Comparative analysis of easyocr and tesseractocr for automatic license plate recognition using deep learning algorithm," in *2022 6th International Conference on Electronics, Communication and Aerospace Technology*. IEEE, 2022, pp. 966–971.

[9] Y. Baek, B. Lee, D. Han, S. Yun, and H. Lee, "Character region awareness for text detection," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 9365–9374.

[10] T. E. de Campos, B. R. Babu, and M. Varma, "Character recognition in natural images," in *International conference on computer vision theory and applications*, vol. 1. SCITEPRESS, 2009, pp. 273–280.

[11] M. Jaderberg, K. Simonyan, A. Zisserman *et al.*, "Spatial transformer networks," *Advances in neural information processing systems*, vol. 28, 2015.

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

[13] L. H. Thanh, "A novel approach for shipping container code recognition," in *Dalat university journal of science*, 06 2017, p. 165–174.

2021 International Conference on System Science and Engineering (ICSSE) | 978-1-6654-4848-2/21/$31.00 ©2021 IEEE | DOI: 10.1109/ICSSE52999.2021.9538466

PROCEEDINGS OF

2021 INTERNATIONAL CONFERENCE ON SYSTEM SCIENCE AND ENGINEERING (ICSSE)

26-28 August 2021
Ho Chi Minh City, Vietnam

Electronic ISBN
978-1-6654-4848-2

IEEE Xplore® Digital Library

Research the Anti-dazzle Headlight System by Leds Matrix with Image Processing Method **77**
Nguyen Van Long Giang and Nguyen Thien Dinh

A Deep Reinforcement Learning Model using Long Contexts for Chatbots **83**
Quoc-Dai Luong Tran and Anh-Cuong Le

Multi-view Transformation in Recommender Systems **88**
Thi-Linh Ho and Anh-Cuong Le

A Smart Direct Controller for a 3-DOF Robot **92**
Pham Tan Phat, Bui Manh Huy and Dang Xuan Ba

A High-Performance Speech-Recognition Method Based on a Nonlinear Neural Network **96**
Phung Hung Binh, Pham Viet Hoang and Dang Xuan Ba

Multi-Oriented License Plate Detection Based On Convolutional Neural Networks **101**
Lam Mai, Xiu-Zhi Chen and Yen-Lin Chen

An Efficient Data Collecting Method for Enhanced Real-Time Drowsiness Detection Systems **105**
Minh-Thien Duong, Truong-Dong Do, Manh Cuong Le, Van-Binh Nguyen and My-Ha Le

The Novel Method of Pedestrian Fall Detection Based on PSO and RF using Accelerometer Data **111**
Hong-Lam Le, Duc-Nhan Nguyen and Ha-Nam Nguyen

An Intelligent Control Method for Redundant Robotic Manipulators with Output Constraints **116**
Dinh Manh Hung, Dao Tung Linh and Dang Xuan Ba

Clustering based Ship Classification using radar signal and Neuron Network **122**
Duc-Dat Ngo, Manh-Hung Nguyen, Quang-Thai-Dan Nguyen and My-Ha Le

A Scalable Virtual Try-on System based on Cloud Computing **128**
Duong Van Ngoc and Cao Xuan Canh

Efficient 3D Face Reconstruction Model Based on Dense Mesh Solution Using Rendering and Partial Search **133**
Tran Duc Long and Ngo Hai Linh

Kinematics, Dynamics and Control Design for a 4-DOF Robotic Manipulator **138**
Thien-Quang Nguyen, Van-Truong Phan, Duy-Thien Vo, Van-Hoang Trinh, Hoang-Viet Nguyen, Manh-Son Tran and Duc-Thien Tran

A Shortest Smooth-path Motion Planning for a Mobile Robot with Nonholonomic Constraints **145**
Hung Hoang, Anh Khoa Tran, Lam Nhat Thai Tran, My-Ha Le and Duc-Thien Tran

Kinematics and Dynamics for a 4-DOF Parallel Robot **151**
Kien Cuong Dinh, Ngoc Sang Dao, Hai Dang Le, Hoang Lam Le, Tu Duong Thi Cam and Duc Thien Tran

Enhancement of Robustness in Object Detection Module for Advanced Driver Assistance Systems **158**
Le-Anh Tran, Truong-Dong Do, Dong-Chul Park and My-Ha Le

Adaptive Optimal Control of Four-Wheel Omni Robot using Reinforcement Learning **164**
Tuan Nguyen Khac, Nguyen Thai Huu, Minh Nguyen Van and Tuyen Bui Trung

MiniRos: an Autonomous UGV Robot for Education and Research **170**
Tri Bien Minh, Hua Thanh Luan, Do Xuan Phu, Tran Quang Nhu and Bui Minh Duong

Clustering based Ship Classification using Radar Signal and Neuron Network

Duc-Dat Ngo
Ho Chi Minh City University of Technology and Education
Ho Chi Minh City, Viet Nam
datnd.ncs@hcmute.edu.vn

Manh-Hung Nguyen
Ho Chi Minh City University of Technology and Education
Ho Chi Minh City, Viet Nam
hungnm@hcmute.edu.vn

Quang-Thai-Dan Nguyen
Ho Chi Minh City Electricity Information Technology Company
Ho Chi Minh City, Viet Nam
dannqt@hcmpc.com.vn

My-Ha Le✉
Ho Chi Minh City University of Technology and Education
Ho Chi Minh City, Viet Nam
halm@hcmute.edu.vn
(Corresponding author)

2021 International Conference on System Science and Engineering (ICSSE) | 978-1-6654-4848-2/21/$31.00 ©2021 IEEE | DOI: 10.1109/ICSSE52999.2021.9538475

***Abstract*— In this paper, we propose a ship classification method based on surveillance radar signals. To get a robust system, our method needs to handle challenges from the high variance given by the poses of a ship. The reflecting radar signals are highly relevant to the poses rather than the ship categories. Therefore, in this paper, we apply a clustering technique to separate the data into suitable domains. Later, the data is classified by corresponding classifiers. Several neural network configurations have been tested to understand the contribution of the proposed preprocessing method. Comprehensive experiments point out that our method helps to improve the relatively good accuracy of a three-category classification.**

Keywords*— *Clustering*; *Signal Recognition*; *Neural Network*; *Radar System

I. INTRODUCTION

Surveillance on the sea [1,2,3] is a critical task to ensure national security, maritime safety, over-sea protection of sovereignty, combating piracy, terrorism, and other external threats. Typically, a pulse-radar system could be used to monitor an ocean region. As shown in Fig.1, a surveillance pulse radar system includes a transmitter, a receiver, a transceiver switch, a synchronization device, a display device, and an emitting/receiving antenna. The synchronization signal from the sync block is sent to the generator to trigger the modulation block in the transmitter. The active modulating block produces a square pulse. This pulse modulates the ultra-high-frequency oscillation, which helps the generator to generate an ultra-high-frequency pulse. The transmitter generates an ultra-high frequency pulse to the transceiver switch. Later, the signal is passed out into space by the antenna. Then, its reflected signal is received by the antenna through the transceiver switch to the receiver. At the receiver, the received signal is demodulated to get the visualized signal. Finally, the reflecting signal is visualized on a display device. Base on the waveform of the reflecting signal, we may know what type of target.

Based on the direction of the emitting ray and the time of arrival (ToA) of the reflecting signal, the position of a target could be estimated robustly. However, determining the type of ship still is a challenging task. Traditionally, people base on the waveform of reflected signals to determine the category of a target ship. As shown in Fig.2 the signal is affected seriously by the pose of a ship. In Fig.2(a,b,c) the reflecting signals of a fishing ship are quite different when the ship is moving to follow different directions. In contrast, as shown in Fig. 2(a,d,g), three different ship- categories but share the same pose will have similar observations. In a general case, there are several factors that can affect the waveform of a reflecting signal. Weather conditions, the ship's materials, and the load capacity of each ship are typical reasons that change the waveform. For instance, given the same ship, but when the ship has a lot of cargo, the reflecting signal will be different from the train carrying less cargo. Hence, less experienced staff may make a wrong prediction for the ship classification task. Motivated by these reasons, in this paper, we propose to use machine learning-based methods to solve the ship classification.

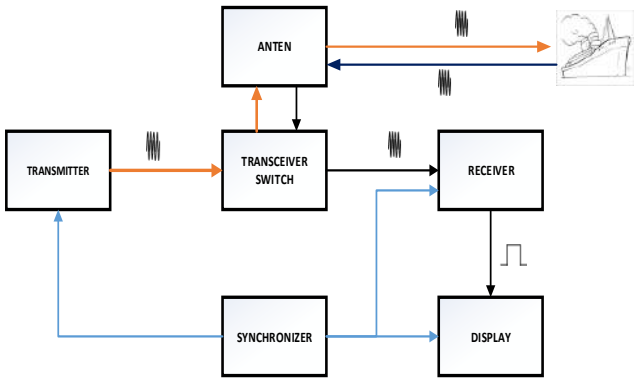


Fig. 1. Block diagram of a surveillance radar system

Unlike traditional methods that directly use a classifier to identify ship categories, we propose to use a clustering algorithm to assign a ship into a suitable cluster before classification. Each cluster represents a specific domain such as poses, weather conditions, or the material of a ship. Given a cluster, we train a classifier to identify ship categories. As shown in Fig. 2, we expect the clustering will separate the training data into domains where samples share a specific latent factor. Hence the classifier will be learned easier. In detail, given the radar-based surveillance system, we separately apply clustering and classification on frequency features [4] or time-domain features to find out better features for machine learning tasks. Base on the experiment, we consider that K-mean clustering and time domain are suitable for our application. Hence, we apply a chain that includes K-mean clustering and Neuron Network classification to recognize three ship-categories.

In summarization, our main contributions are listed as follows:

- We collect a dataset that includes reflecting radar signals from ships at sea. To our best knowledge, this is the first radar dataset collected in Vietnam for surface marine monitoring.

- In a scenario of radar-based marine surveillance, we compare the effects of Fourier features and time-domain features on machine learning algorithms. The experiments help to select a suitable clustering method for our classification module.
- We propose a clustering-based classification for ship recognization. By preprocessing the data based on clustering, we reduce the complexity of the classification task. Consequently, the proposed method gets better results when it was tested with various complexity settings and various training sets.

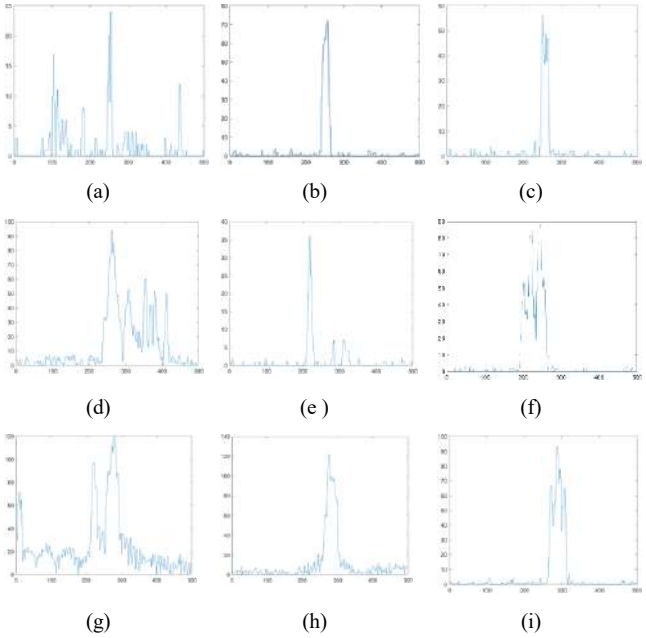


Fig. 2. Reflecting signal given by fishing ship, civil-transport ship and military-transport ship given by various poses.
(a) Signal given by a passing fishing ship
(b) Signal given by a moving-in fishing ship
(c) Signal given by a moving-out fishing ship
(d) Signal given by a passing civil-transport ship
(e) Signal given by a moving-in civil-transport ship
(f) Signal given by a moving-out civil-transport ship
(g) Signal given by a passing military- transport ship
(h) Signal given by a moving-in military- transport ship
(i) Signal given by a moving-out military- transport ship

II. RELATED WORKS

A. Feature extraction

Conventionally, to classify a 1D signal such as our waveform, a segmentation task must be implemented first. Base on the segmented signal, we apply a classifier to recognize its category. Hence the success of the classification task is highly dependent on the result from the segmentation task. However, due to the difference in sampling rate and the shifting effect, a segment may not robust in the time domain. Therefore, frequency features [4] have been widely used in 1D signal-based classification.

Unlike usual cases, our emitting radar signal is modulated by a square pulse. Hence, the receiving signal has the same size as the square pulse. As shown in Fig. 2, the receiving signal is a 500-element array in the time domain. In this case, we do not need to handle the segmentation task. Also, the shifting effect may not an issue in our application. Therefore, using the time domain may be a reasonable solution in this case.

B. Classification

For the classification task, the Support Vector Machine (SVM) [5,6] and Neuron Network (NN) [7,8] are two well-known methods that can address the nonlinear challenge from a dataset. While the Neuron network relies on nonlinear active functions and hidden layers to represent a nonlinear mapping, the nonlinear SVM classifier is based on pre-defined kernels to provide nonlinear modelings. Typically, the quality of a classifier is dependent on the relevance between model complexity and training dataset. If the model is extremely complex but the dataset is simple, we may have an overfitting phenomenon. In contrast, if the model is simple but the dataset is a challenge, we may have an underfitting phenomenon. We expect the model complexity is suitable with the dataset complexity. In this case, the training accuracy may slightly higher than the testing accuracy; and the model is considered that it is generalized enough. While NN relies on the number of classes and the number of nodes in each hidden layer to determine the complexity of a model; SVM relies on kernel bandwidth to control its complexity. Choosing the right control parameters is a critical task to train a classifier successfully.

III. PROPOSED METHOD

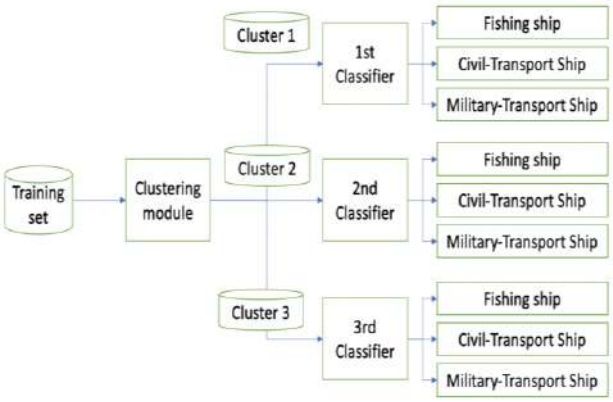


Fig. 3. System overview.

In this chapter, we introduce the proposed method in detail. For a given dataset, we use the K-means algorithm to separate this dataset into three clusters. For each group, a NN is used to classify ship categories. The categories include fishing ship, civil-transport ship, and military-transport ship. During the testing phase, a testing sample is assigned to a specific cluster. Depend on the cluster that the test sample is assigned, a corresponding classifier will be selected for classification. In the next subsection, the K-means clustering and the Neuron network will be introduced in detail.

A. KMeans Clustering

Given a dataset $X=[x_1, x_2, \ldots, x_N] \in R^{d \times N}$ that include N samples which represented by d features, a KMeans algorithm separates this dataset into K clusters. In each cluster, its members should be as similar as possible. Each cluster is represented by a cluster centers as $m_k \in R^{dx1}$ (k=1~K); and for each point in the dataset, a label should be assigned to identify which cluster that the sample is belong in. Typically, the label could be represented by a one-hot vector as $y_{ik} \in \{0,1\}$ and $\sum_{k=1}^{K} y_{ik} = 1$. If $y_{ik} = 1$, it means the i^{th} sample is belong to the k^{th} cluster. Here, we expect that a sample

should close to the cluster center that it's label is belong to. Hence the loss function in Equ. (1) is used to learn the $\{y_i, m_k\}$.

$$J = \frac{1}{N}\sum_{i=1}^{N}\sum_{k=1}^{K} y_{i,j}\,\|x_i - m_k\|_2^2 \tag{1}$$

Denote Y= [y_1 ; y_2 ; …; y_N] and M=[m_1 , m_2, …m_K] are the matrix form of all labels and cluster centers, the solution Y and M could be found by Equ. (2).

$$Y, M = \underbrace{agrmin}_{Y,M}\sum_{i=1}^{N}\sum_{k=1}^{K} y_{ik}\|x_i - m_k\|_2^2 \tag{2}$$

subject to: $y_{ik}\in\{0,1\}$ $\forall$i,k; $\sum_{j=1}^{K} y_{ij} = 1\ \forall i$

Because Y and M are from two different categories, the Expectation–Maximization algorithm [9] are applied to solve the solution in Equ. (2). Here, we randomly initialize M and find Y to minimum J. The solution of Y could be estimated by Equ. (3):

$$y_i = \arg\min_{y_i}\sum_{k=1}^{K} y_{ik}\|x_i - m_k\|_2^2$$
$$\text{subject to: } y_{ik}\in\{0,1\}\ \forall j;\ \sum_{k=1}^{K} y_{ik} = 1 \tag{3}$$

Later, we fix Y and estimate M by Equ. (4):

$$m_j = \frac{\sum_{i=1}^{N} y_{ik}x_i}{\sum_{i=1}^{N} y_{ik}} \tag{4}$$

To find the optimal Y and M, Equ. (3) and (4) are repeated until the parameters are converted.

B. Neural Network

Neural Network (NN) is a powerful tool for classification tasks. As shown in Fig.4, the NN a combination of multi perceptron layers. Typically, there are three types of layers could be found in a NN. The input layer is the leftmost layer of the network and it represents the inputs of the network. The number of nodes in the input layer is the number of features. The output layer is the bottom right layer of the network that represents the outputs of the network. The number of nodes in the output layer is the number of classes that we aim to classify. A hidden layer is a layer between the inlet and the exit layer representing the logical inference of the network. A NN can have many hidden layers; hence the model representation of a NN could be found in Equ. (5):

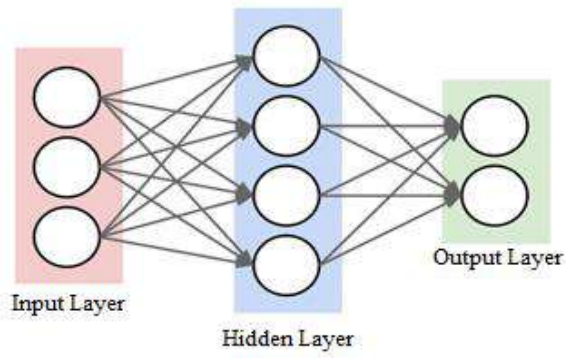


Fig.4. Block diagram of Neural Network

$$y(\boldsymbol{x}, W) = f_L(f_{L-1}\left(f_{L-2}\left(\ldots\left(f_2(f_1(\boldsymbol{x}))\right)\right)\right)) \tag{5}$$

Here, x_i is the i^{th} sample in a dataset, W is the weight of the network, and $f_l(\boldsymbol{x})$ is the feature extracted at the l^{th} layer. Denote σ_l is the activation function and W_l is the weight of the l^{th} layer respectively; the corresponding layer $f_l(.)$ of a neuron network is represented by Equ. (6)

$$f_l(\boldsymbol{x}) = \sigma_l(W_l f_{l-1}(x)) \tag{6}$$

To train a NN, we expect that the prediction of the i^{th} sample $y(\boldsymbol{x_i}, W)$ should be similar to the target $\boldsymbol{t_i}$. In a classification task, the similarity could be modeled by a cross-entropy loss as in Equ. (7). A smaller loss means a higher similarity.

$$J_{class} = \frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{C} -t_{i,j}\log(P(y(\boldsymbol{x_i}, W) = j)) \tag{7}$$

Here t_i is a one-hot label vector of the i^{th} sample and $P(y(\boldsymbol{x_i}, W) = j)$ is the probability that the prediction $y(\boldsymbol{x_i}, W)$ is belong to the j^{th} sample. Usually, the probability is the output of a NN.

When the number of hidden nodes are increased, NN will be more dense and complex. In order to avoid overfitting in a dense network, a regularization loss have been introduced to ignore less important features. Denote L is the number of layers and s_l is the number of nodes in the l^{th} layer, the regularization loss is presented in Equ. (8). The J_{Regu} loss is minimum when all $W_l^{i,j}$ are zero. In this case, the network cannot learn anything. Hence, a λ parameter is used to control the contribution of the regularization term to the prediction process as Equ. (9). To evaluate the performance of NNs of a dataset, we may modify the network configuration from sparse to dense. During the process, λ should be very small to avoid overfitting in a dense network.

$$J_{Regu} = \frac{1}{N}\sum_{l=1}^{L-1}\sum_{i=1}^{s_l}\sum_{j=1}^{s_{l+1}} W_l^{i,j} \tag{8}$$

$$J_{train} = J_{class} + \lambda J_{Regu} \tag{9}$$

To minimize the above loss function J_{train}, we use the Gradient Descent [10]. It is an iteration process to find the optimal solution. Denote τ is the iterations step and α is the learning rate, the parameter will be updated by Equ. (10).

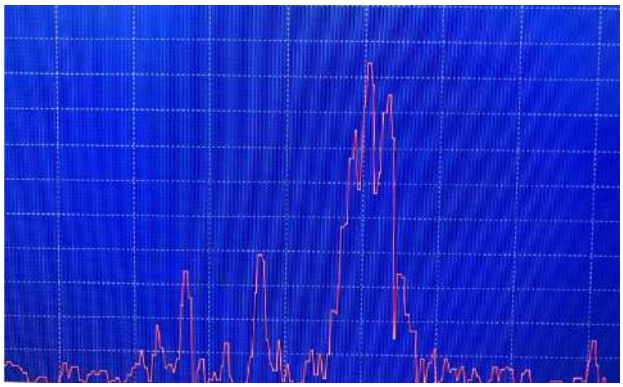
$$W^{\tau+1} = W^{\tau} - \alpha\frac{\partial J_{class}}{\partial W} \tag{10}$$

IV. EXPERIMENTS

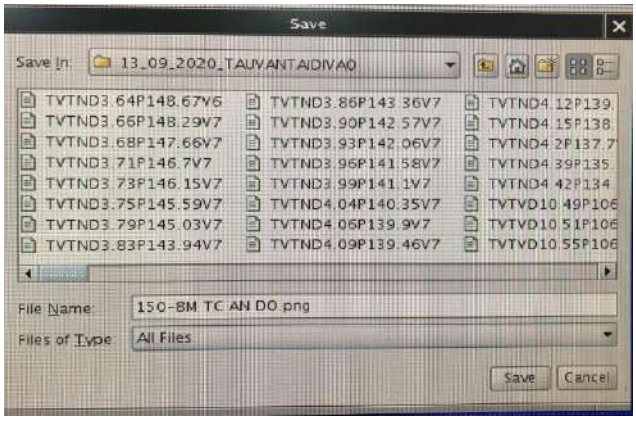
In this section, we present experiments to demonstrate the effectiveness of the proposed method. The dataset is collected by a coastal landscape radar system. This system is owned by the VietNam government. Fig. 5a shows a reflecting waveform on the surveillance screen and Fig.5b is exported excel files given by a software that accompany with the system. Each excel file is a waveform represents a reflecting signal. Overall, we collect 611 samples of fishing ships; 688

samples of civil-transport ships, and 328 samples of military-transport ships.

A. Compare frequency features and time-domain features.



(a) A reflecting signal on a display screen.



(b) Exported excel files of reflecting signals from the radar system

Fig.5. Data Acquisition System

TABLE I. CLUSTERING RESULTS BY VARIOUS FEATURES

Training phase				
		Fishing Ship	Military - Transport Ship	Civil-Transport Ship
FFT	C0	418	14	21
	C1	0	13	205
	C2	1	186	232
FFT+Scaling	C0	0	18	299
	C1	399	33	28
	C2	20	162	201
FFT+Log	C0	367	9	9
	C1	2	45	308
	C2	50	159	141
Time domain	C0	0	0	341
	C1	419	7	28
	C2	0	206	89
Testing phase				
FFT	C0	192	6	10
	C1	0	9	114
	C2	0	100	106
FFT+Scaling	C0	0	16	116
	C1	186	15	12
	C2	6	84	102
FFT+Log	C0	175	2	6
	C1	1	32	160
	C2	16	81	84
Time domain	C0	0	0	180
	C1	192	4	14
	C2	0	111	36

Typically, the 1D signal-based classification [11,12] requires a segmented signal for the classification task . Because the segmentation in time domain is effected seriously by the shifting effect, frequency features had been introduced as robust features for classification. However, as discussed in section II.A, our application may not be effected by the shifting. Motivated by the observation, in this section, we compare Fats Fourier features [13] and time domain feature on fundamental machine learning tasks. The clustering task and the classification task are selected for this comparison. For each task, we use 33% data for testing and 66% data for training. The algorithms implemented based on the sklearn library and random seed are fixed to ensure the same training and testing set are selected through our experiments.

For the clustering task, we use the K-means algorithm to separate the data into 3 clusters. We compare FFT features [13] and time domain feature by using the the diversity in each cluster. If a cluster has only one ship-category, it means the clustering result is good. In contrast, if a cluster has three ship-categories and the amount of each category are quite equal, it means the clusting may not work. Therefore, base on the clustering result we will know which kind of features are suitable for the clustering task. By using these experiments, we may identify the best clustering method to accompany with our classification.

Experimental results in Table 1 show that clustering results given by the time-domain feature are much better than clustering results given by frequency features. In detail, in the training phase, the time domain provides the C0 cluster that contains only civil-transport ship; the C1 cluster has 92.3% population are fishing ships, the C2 cluster only contain transport ships but no fishing ships (69% population are military-transport ships). Moreover, in Tab.1, the clustering results is consistent between training phase and testing phase if the time domain feature are used. During the training phase, if a specific ship-category does not appear in a cluster; then in the testing phase, the ship-category will not appear in the corresponding group. This suggests that clustering will not cause any negative effect to the later classification.

In constrast, if Fats Fourier features are used for clustering, we find that the clustering results is very divert. These clusters do not focus on any ship-category. The reason may caused by the non-linear property in the Fourier transform. The non-linear property could be compensated by a normalization [14] or a logarithm operation. As shown in Table 1, when the nonlinear is compensated, the clustering results are improved and the diversity in each cluster is reduced significantly. However, the improved results still are not comparable to the robust result on the time domain. This proves that the signal in the frequency domain is not really useful for ship clustering.

Beside the clustering task, we also compare these features on a classification task. Based on extracted features in the clustering experience. We use three NN-network configurations (S1, S3, S5) to classify three ship-categories. Among the configuration, the S1 is the highest complexity because it includes more hidden layers and the numbers of nodes per layer are also high. The S5 configure is the lowest complexity because there are only 2 hidden layers and the numbers of nodes per layer is low. Details of these configurations are described in Table 3. Experimental results in Table 2 show that the FFT features are not suitable for ship classification. The very low accuracy given by FFT features indicate that the model cannot converge well. If we apply the

scaling method [14] or the logarithm function to compensate nonlinear factors, the model is converted but the accuracy is not very high. The best accuracy could be achieved is 96% on frequency features. In contrast, the time-domain features can get an accuracy up to 98%. The classification results are consistent to the clustering results. Both of them shows that the signal in the time domain is good enough to process reflecting radar signal.

TABLE II. CLASSIFICATION RESULTS BY VARIOUS FEATURES (TRAINING / TESTING RATIO 66% / 33%)

		ACC	F1-Score
FFT	s0	0.1195	0.2364
	s1	0.5680	0.6312
	s2	0.3910	0.4357
FFT+Scaling	s0	0.9606	0.9608
	s1	0.9606	0.9608
	s2	0.9492	0.9497
FFT+Log	s0	0.9414	0.9422
	s1	0.9243	0.9255
	s2	0.9245	0.9255
Time domain	s0	0.9813	0.9814
	s1	0.9795	0.9795
	s2	0.9795	0.9795

B. *Classification result with and with-out clustering*

In this section, we demonstrate the effectiveness of the proposed method when using clustering to pre-process the reflecting radar signal. Typically, accuracy and the F1- score are used to evaluate the system performance. However, to emphasize the contribution of the clustering process, we use the enhancement rate of the accuracy and the F1-score to evaluate the contribution of the preprocessing step. The enhancement is measured by Equ.10.

$$E = \frac{P_{clus} - P}{P} 100 \qquad (11)$$

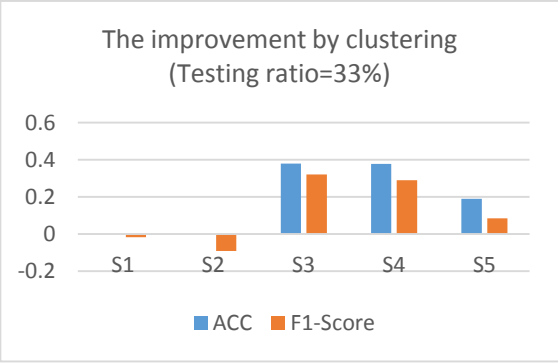


Fig.6 Accuracy and F1-Score under 67% training and 33% testing

Here, P is the system performance without the clustering preprocess; P_{clus} is the system performance with the clustering preprocess. The performance cold is the accuracy or F1-score. If this value is greater than 0, we can conclude that the preprocess by proposed clustering help to improves the system performance. In contrast, if this value is less than 0, it means that clustering does not enhance performance. We conducted experiments with various NN configurations. As shown in Table 3, these configurations are from very dense (S1) to very sparse (S5). In addition, we test the system with two scenarios that have different training/testing ratios. It helps to understand how the number of training samples can affect our proposed method.

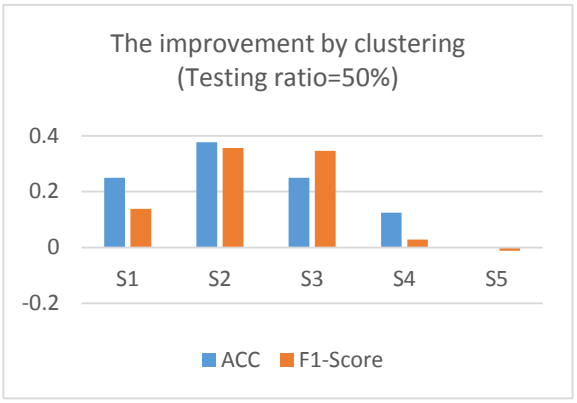


Fig.7. Accuracy and F1-Score boost when training /testing ratio is 66% / 33%

In the first scenario, we select 67% data for training and 33% data for testing. The experimental result is shown in Fig. 5. When the NN network is very dense (S1-S2 configurations), clustering does not help much but even reduces performance. Typically, a dense classifier will be overfitting if the dataset does not have enough training data. When clustering is applied, the training data is separated into three groups. Hence, the training set for each individual classifier is reduced. This phenomenon degrades the system performance. On another hand, when clustering is not used, all training samples are used to train a unique classifier. Because of this reason, the performance without clustering is better than within clustering. We can consider that in a dense-networks, clustering preprocessing does not increase performance. As in Fig.6, in S1 and S2 configurations, the accuracy is kept the same, and F1-Score is slightly reduced. In contrast, with sparse NNs such as S3 and S4 configurations, the clustering helps increase performance significantly in both ACC and F1-Score. When the network configuration becomes sparse, the network does not require a huge dataset to learn all parameters of the method. Therefore, small datasets given by the clustering can help to learn the model well. If the network is too small as the S5 configuration, it tends to underfit both within and without clustering. In all of these configurations, the maximum ACC value when not using clustering was 98% while the maximum ACC value when using clustering was 99%. This demonstrates that clustering preprocessing can improve the overall performance.

TABLE III. DETAIL SETTING OF NN CONFIGURATIONS

	Solver	Regularization Coefficient	Hidden Layer Size	Random State
S1	lbfgs	$1e^{-5}$	(300,200,100,50)	100
S2	lbfgs	$1e^{-5}$	(300,100,50)	100
S3	lbfgs	1e-5	(200,75)	100
S4	lbfgs	1e-5	(100,75)	100
S5	lbfgs	1e-5	(75,50)	100

In the second scenario, we reduced the number of training samples. Here, only 50% of data is used for training; and the rest is used for testing. The enhancement scores are shown in Fig.7. The results point out that clustering helps to increases the system performance in both dense and sparse configurations. In detail, the accuracy increments for S1, S2,

S3, and S4 configurations are 0.25, 0.37, 0.249, and 0.125 respectively. It is worth to discuss why clustering help to improve the system performance in this scenario. Because the amount of training data is reduced, the dense network will not have enough data to train a robust classifier. Consequently, overfitting happens. By using clustering, the number of challenge samples is reduced in each cluster. As shown in Table 1, each cluster focus on some specifict class. Therefore, when fewer data were used for training, the reduced samples may be simple instances. For instance, according to Table 1, the most challenging cluster is the C1 because this cluster has three ship- categories. If 66% of data is used for training, we may have 419 samples are fishing ships, 7 samples are military-transport ships, and 28 samples are civil-transport ships. While 50% of data is selected for training, some samples may be reduced. Most of the reduced samples are fishing ships; the number of transport ships is retained. Therefore, the training results are not affected seriously. On the other hand, reducing the number of training samples for the easier cluster (C0-C2) does not actually affect the quality of the system. The reason is that these samples are already easy and do not cause overfitting. For instance, no matter how much samples are reduced in cluster C0, we always consider that it is a civil-transport ship. Hence, the clustering help to train a robust system when the number of training samples is limited.

V. CONCLUSION

In this paper, we propose a clustering-based classification method that helps to recognize ship categories on the sea. Experiments on the clustering task prove the time-domain feature is better than Fast Fourier features. In the classification task, many NN configurations had been tested. The results showed that the clustering help to increase the accuracy to 1%. Moreover, the clustering help to learn a robust classifier when the number of training samples is reduced.

REFERENCES

[1] E. Vorobev, A. Bezuglov, V. Veremyev and V. Kutuzov, "System for adjustment of angle coordinates for sea surface surveillance radar," *2017 Signal Processing Symposium (SPSympo)*, 2017, pp. 1-5, doi: 10.1109/SPS.2017.8053652.

[2] T. B. Sarikaya, G. Soysal, M. Efe, E. Sobaci and T. Kirubarajan, "Sea-land classification using radar clutter statistics for shore-based surveillance radars," *International Conference on Radar Systems (Radar 2017)*, 2017, pp. 1-4, doi: 10.1049/cp.2017.0488.

[3] P. -L. Shui, X. -Y. Xia and Y. -S. Zhang, "Sea–Land Segmentation in Maritime Surveillance Radars via K-Nearest Neighbor Classifier," in *IEEE Transactions on Aerospace and Electronic Systems*, vol. 56, no. 5, pp. 3854-3867, Oct. 2020, doi: 10.1109/TAES.2020.2981267.

[4] L. Du, L. Li, B. Wang and J. Xiao, "Micro-Doppler Feature Extraction Based on Time-Frequency Spectrogram for Ground Moving Targets Classification With Low-Resolution Radar," in *IEEE Sensors Journal*, vol. 16, no. 10, pp. 3756-3763, May15, 2016, doi: 10.1109/JSEN.2016.2538790.

[5] M. E. Demirhan and Ö. Salor, "Classification of targets in SAR images using SVM and k-NN techniques," *2016 24th Signal Processing and Communication Application Conference (SIU)*, 2016, pp. 1581-1584, doi: 10.1109/SIU.2016.7496056.

[6] Mingqiu Ren, Jinyan Cai, Yuanqing Zhu and Minghao He, "Radar emitter signal classification based on mutual information and fuzzy support vector machines," *2008 9th International Conference on Signal Processing*, Beijing, 2008, pp. 1641-1646

[7] C. Wang, J. Pei, R. Wang, Y. Huang and J. Yang, "A new ship detection and classification method of spaceborne SAR images under complex scene," *2019 6th Asia-Pacific Conference on Synthetic Aperture Radar (APSAR)*, 2019, pp. 1-4, doi: 10.1109/APSAR46974.2019.9048382.

[8] U. Kaydok, "Chaff Discrimination Using Convolutional Neural Networks and Range Profile Data," *2020 IEEE International Radar Conference (RADAR)*, 2020, pp. 373-377, doi: 10.1109/RADAR42522.2020.9114645.

[9] S. Rajkamal, "Selecting Reviewers for Research by Clustering Proposals Using Expectation Maximization Clustering Algorithm," *2017 International Conference on Technical Advancements in Computers and Communications (ICTACC*), 2017, pp. 56-60, doi: 10.1109/ICTACC.2017.24.

[10] Zhongbo Sun, Tianxiao Zhu and Haiyin Gao, "A sufficient descent hybrid conjugate gradient method and its global convergence for unconstrained optimization," *2012 24th Chinese Control and Decision Conference (CCDC)*, 2012, pp. 735-739, doi: 10.1109/CCDC.2012.6244111.

[11] A. E. Vincent and K. Sreekumar, "A survey on approaches for ECG signal analysis with focus to feature extraction and classification," *2017 International Conference on Inventive Communication and Computational Technologies (ICICCT)*, 2017, pp. 140-144, doi: 10.1109/ICICCT.2017.7975175.

[12] Sung-Soo Kim and T. Kasparis, "A modified domain deformation theory on 1-D signal classification," *in IEEE Signal Processing Letters*, vol. 5, no. 5, pp. 118-120, May 1998, doi: 10.1109/97.668949.

[13] H. Chen, C. Wang, T. Chen and X. Zhao, "Feature selecting based on fourier series fitting," *2017 8th IEEE International Conference on Software Engineering and Service Science (ICSESS)*, 2017, pp. 241-244, doi: 10.1109/ICSESS.2017.8342905.

[14] M. S. Azmi, N. A. Arbain, A. K. Muda, Z. A. Abas and Z. Muslim, "Data normalization for triangle features by adapting triangle nature for better classification," *2015 IEEE Jordan Conference on Applied Electrical Engineering and Computing Technologies (AEECT)*, 2015, pp. 1-6, doi: 10.1109/AEECT.2015.7360572.