AUTOMATED DETECTION OF VESSELS IN REMOTE SENSING IMAGES USING DEEP LEARNING MODELS

Hércules Carlos Dos Santos Pereira[1], *Caio Eduardo Dias*[1], *Valdivino Alexandre de Santiago Júnior*[1], *Elcio Hideiti Shiguemori*[1,2], *Matheus Corrêa Domingos*[1], *André Estevam Costa Oliveira*[1] *and Dener César Mendes*[1]

[1]Laboratório de Inteligência ARtificial para Aplicações AeroEspaciais e Ambientais (LIAREA), Instituto Nacional de Pesquisas Espaciais (INPE)
[2]Instituto de Estudos Avançados (IEAv)
hercules.pereira@inpe.br, caio.dias@inpe.br, valdivino.santiago@inpe.br, elcio@ieav.cta.br, matheus.domingos@inpe.br, andre.costa@inpe.br and dener.mendes@inpe.br

ABSTRACT

This study presents an approach for automated detection of vessels using convolutional neural networks (CNNs) and visual transformers (ViTs). The approach involves comparing different deep learning models including RoboFlow, You Look Only Once v8 (YOLOv8), RetinaNet, Detection Transformer (DETR), and Real-Time Detection Transformer (RT-DETR). The comparison includes a general accuracy metric, mean Average Precision (mAP), and time performance metrics, training time, frames per second (FPS), and latency. The dataset selected for the study is HRSC2016-MS, which contains 1,680 optical remote sensing images, covering 7,655 labeled instances of vessels and presenting a variety of conditions, such as lighting, weather, and different scales. Considering all the metrics, YOLOv8 stands as the best approach confirming the current popularity of such a family of object detectors.

Keywords – *convolutional neural networks, deep learning, remote sensing, vessels, visual transformers.*

1. INTRODUCTION

In 2022, more than 2 million ships were used to dominate 80% of international trade, transporting a wide range of cargo that ranges from bulk products, like grains and fuels, to containerized cargoes, like clothing and electronics. In addition to their primary function in transporting goods, ships play an essential role in the exploration of oceanic resources, such as fishing and mining, in the installation of communication cables between continents, in the transport of people as in the Amazon region, and are fundamental for the tourism industry [1]. By the end of 2024, it is estimated that 360 cruise ships will transport 30 million passengers, marking a 9.2% increase compared to 2019 figures, prior to the pandemic [2].

Vessel monitoring is essential for a wide range of purposes such as maritime safety, traffic monitoring, enforcement of the National Security Law, environmental management, search and rescue, research and development, and route planning. In a way, vessel monitoring not only promotes safety, but also plays an important role in environmental protection and law enforcement [3]. Detecting vessels is not a trivial problem, given that the context in which the object is inserted must be taken into account, that is, the variation in light, weather and scale of the vessel can influence detection.

In the context of computer vision, object detection is a task which consists of the ability to identify the location of an object in an image and classify it. In this study, object detection consists of finding the relevant classes, such as a vessel present in the image. Therefore, a machine/deep learning model can be used to perform detection using a bounding box in the area of the image presented.

Some approaches have been presented to solve this problem. In [4], the authors analyzed an approach for real-time ship detection using synthetic aperture radar (SAR) imagery. In the experiment, the authors compared the obtained results with other models. The fastest and most accurate detection was obtained by the You Look Only Once (YOLO) model.

Another study also addressed ship detection using convolutional neural networks (CNNs) [5]. In this work, the authors proposed the Rotate YOLO (RYOLO) models, designed to detect ships accurately and quickly, taking into account the vessels' orientation angle. RYOLO delivered a mean Average Precision (mAP) of 96.7% and a processing speed of 45.6 frames per second (FPS), standing out as an effective approach for ship detection in maritime videos.

Despite these previous studies, it is important that more detailed investigations can be conducted considering different types of models, in addition to CNNs, and various metrics, addressing not only performance in terms of accuracy but also in temporal terms. This type of evaluation can be highly valuable in suggesting more suitable solutions for researchers and practitioners in the field of remote sensing (RS).

Therefore, this study presents an approach for automated detection of vessels using not only CNNs but also the most recent visual transformers (ViTs). The approach involves comparing different deep learning models including RoboFlow [6], YOLOv8 [7], RetinaNet [8], Detection Transformer (DETR) [9], and Real-Time Detection Transformer (RT-DETR) [10]. The comparison includes an overall accuracy metric, mAP, and time-performance metrics, such as training time, FPS, and latency. The dataset selected for the study was HRSC2016-MS [11] which contains 1,680 optical remote sensing images, covering 7,655 labeled instances of vessels and presenting a variety of conditions, such as lighting, weather, and different scales. The specific goals of the study are: (i) to automatically detect vessels; (ii) to identify the model, among CNNs and ViTs, with the best generalization ability; and (iii) to evaluate execution time, FPS, and latency. Thus, it is possible to offer more appropriate suggestions for this problem, according to various analytical perspectives.

2. MATERIAL AND METHODS

2.1. Dataset

HRSC2016-MS is an expanded version of the HRSC2016 dataset, focused on vessel detection in optical remote sensing images [11]. It combines original HRSC2016 data, captured in ports in the United States, with new Google Earth data, mainly from the port of Murmansk, Russia. With a total of 1,680 images (1,070 from HRSC2016 and 610 from Google Earth), the dataset covers different situations (day/night, clear/cloudy sky) and different shipping schedules. New annotations have been included, mainly for small vessels, increasing the number and diversity of sizes and proportions of the instances. Figure 1 shows four examples of images from the dataset.

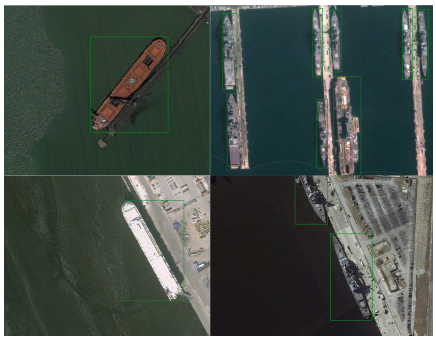

Figure 1: Examples of images contained in the dataset.

Other relevant information of the dataset are: (i) "Image Resilience" between 361 x 339 $\sim$ 1329 x 830; (ii) "Instances": 7,655; (iii) "Bounding Box Sizes": 5 x 10 $\sim$ 489 x 739; (iv) and "Instance Proportions": $0.092\ to 11{,}692$. It is important to highlight that the dataset contains a single instance “ship” which means ship.

2.2. Preprocessing

Two techniques were applied to preprocess the data to guarantee consistency and efficiency in model training. An auto-orientation step was applied to the images, ensuring that they were all aligned more consistently, regardless of their original orientation. This technique ensures that features of interest are correctly positioned in all images.

Furthermore, all images with different dimensions (361 x 339, 1329 x 830 pixels) were resized to a resolution of 640 x 640 pixels using the stretching technique. This method adjusts the original image dimension to the new size, maintaining the overall visual integrity, even though it may distort the aspect ratio. Resizing is essential to standardize the model input, contributing to optimized processing and computational efficiency.

2.3. Data Augmentation

To increase the diversity of the dataset, several transformations were applied to the images, aiming to introduce variations in the samples. The transformations applied include: (i) Horizontal and Vertical Flipping: the images were flipped horizontally and vertically; (ii) 90° Rotation: the images were rotated both clockwise and counterclockwise; (iii) Grayscale: conversion to grayscale applied to 19% of the images; (iv) Blur: blur applied with an intensity of up to 1 pixel; and (v) Noise: noise added to up to 1.05% of the pixels.

3. SELECTED MODELS

3.1. Roboflow

Roboflow is a computer vision platform that enables the management and annotation of datasets for tasks such as object detection, classification, and instance segmentation. Users can upload their own data to train models such as CogVLM, YOLO-NAS, YOLOv9, and YOLOv8, as well as the Roboflow Instance Segmentation Model, Yolo World Model, and OCR Model [6].

3.2. YOLOv8

The YOLO model family is one of the most widely recognized object detection approaches, balancing speed and accuracy by reframing the task as a single regression problem, directly mapping from image space to bounding box coordinates and class probabilities [7]. Its architecture comprises three main components: the backbone, which extracts meaningful patterns at multiple scales; the neck, which performs feature fusion and integrates contextual information; and the head, responsible for generating bounding boxes and confidence scores [12]. In the YOLOv8 model, 50 training epochs were used, with a batch size of 16, learning rate of 0.001, momentum set to 0.937, and optimization with stochastic gradient descent (SGD). Intersection over Union (IoU) and Non-Maximum Suppression (NMS) thresholds were set to 0.2 and 0.6, respectively, along with early stopping with a patience of 50 epochs and input normalization.

3.3. RetinaNet

RetinaNet [8] is an object detection algorithm known for its use of focal Loss, a loss function designed to handle class imbalance common in object detection tasks. This approach reduces the weight of easy examples and increases the focus on hard examples, improving the detection of small and rare objects. The RetinaNet architecture employs a backbone network, such as ResNet or Feature Pyramid Network (FPN), to extract features, and a detection head that predicts bounding boxes and class labels. The use of anchors at multiple scales and aspect ratios enables the identification of objects of different sizes and shapes [8].

As for training, hyperparameters include the total number of iterations, defined to span 50 epochs, the number of data loading workers to parallelize data handling, the per-image batch size adjusted to optimize memory, and a constant learning rate of 0.001.

3.4. DETR

DETR simplifies object detection by removing traditional components such as anchors and NMS [13]. While traditional

models use anchors to predict object locations, DETR employs a transformer to directly detect objects [9]. Detection in DETR begins with feature extraction using CNNs like ResNet. These features are then processed by a transformer encoder, applying self-attention to capture the global context of the image [14]. The transformer decoder generates predictions with a fixed set of queries, making the process more efficient [15]. DETR training used a learning rate of 1×10^{-4}, weight decay of 1×10^{-4}, and batch size of 8, for 50 epochs.

3.5. RT-DETR

RT-DETR is an optimized version of the DETR architecture, designed for speed applications without sacrificing accuracy. While DETR simplifies object detection, its real-time performance is limited by the computational demands of transformers, which involve many self-attention operations [16]. According to the authors, RT-DETR reduces this complexity by decreasing the number of self-attention heads and replacing some operations with more efficient convolutional layers for feature extraction. RT-DETR's key innovation is its hybrid architecture, combining CNNs for local feature extraction with transformers for modeling global dependencies. The training configuration includes 50 epochs, a learning rate of 5×10^{-5} , and 300 warm-up steps, with a batch size of 16 samples per device.

3.6. Evaluation Metrics

The mAP metric is widely used for evaluating object detection models as it combines both precision and localization [17]. The mAP calculation for single-class tasks begins with the Average Precision (AP), based on the IoU metric, which measures the overlap between the predicted and ground truth bounding boxes [18].

The mAP metric is derived from a precision-recall curve, where AP is the area under this curve. The mAP is the average AP across a range of IoU thresholds, reflecting the model's ability to correctly detect under different overlap criteria such as follows: (i) **mAP 50:95:**: average precision over IoU thresholds from 0.5 to 0.95, assessing the model's consistency across various levels of overlap; (ii) **mAP 50:** requires a minimum 50% overlap for a detection to be considered correct, generally resulting in higher mAP values; and (iii) **mAP 75:** requires a minimum 75% overlap, being a more stringent metric that reflects higher precision.

The latency and FPS of a model measure its response time and processing capability. Latency is calculated by measuring the time taken to process a single image, while FPS is the inverse of the average latency, indicating frames per second. These values were obtained using the best saved model, reflecting the optimized performance.

4. RESULTS

Table 1 presents the outcomes related to mAP. RT-DETR demonstrated the highest detection capability, with a mAP 50:95 of 0.75, mAP 50 of 0.88 (quite close to the best approach in this case, i.e., Roboflow), and the highest mAP 75 of 0.83. This indicates strong performance across different IoU thresholds. RoboFlow and YOLOv8 presented competitive results, with high mAP 50 values (0.89 and 0.88, respectively), although RT-DETR stands out at higher IoU thresholds. RetinaNet and DETR lag behind in terms of mAP 50:95 and mAP 75, especially DETR, which has the worst performance of all.

Model	mAP 50:95	mAP 50	mAP 75
Roboflow	0.70	0.89	-
YOLOv8	0.69	0.88	0.81
RetinaNet	0.61	0.80	0.64
DETR	0.53	0.77	0.59
RT-DETR	0.75	0.88	0.83

Table 1: mAP values.

In Table 2, DETR compensates with higher speed, achieving the highest FPS (143.47) and the lowest latency (7 ms), though it has the longest training time (4.86 hours). YOLOv8 strikes a good balance with fast training time (0.55 hours), high FPS (109.88), and low latency (9.10 ms).

Surprisingly, RT-DETR did not perform well in temporal terms as promised. Its FPS and latency are considerably worse compared to the best models in this regard (DETR and YOLOv8). Although its training time is shorter than DETR's, it is still far inferior to what is achieved with YOLOv8. Thus, these results do not support the claim of RT-DETR's authors.

Model	Temp_Train (h)	FPS	Latency (ms)
Roboflow	-	2.31	431.8
YOLOv8	0.55	109.88	9.1011
RetinaNet	3.38	17.79	56.301
DETR	4.86	143.47	7.00
RT-DETR	3.23	30	33.00

Table 2: Training time, FPS, and latency values.

As shown in Figure 3, it is possible to observe that the YOLOv8 model is able to detect vessels that are close to each other very well. In Figure 2, it is also noticeable that the model has difficulties when the color of the vessels is very similar to that of the sea. A clear example of this can be seen in the upper right corner, where the model was unable to detect a small vessel.

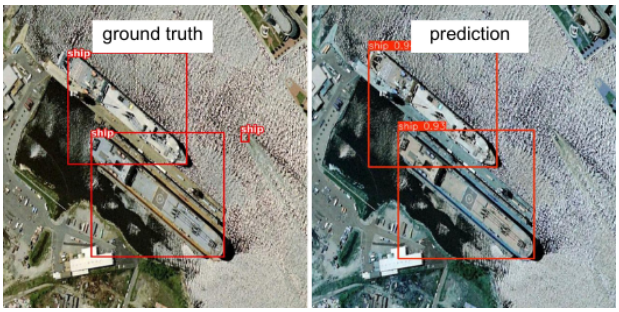

Figure 2: Comparison 1 between ground truth and object prediction via YOLOv8.

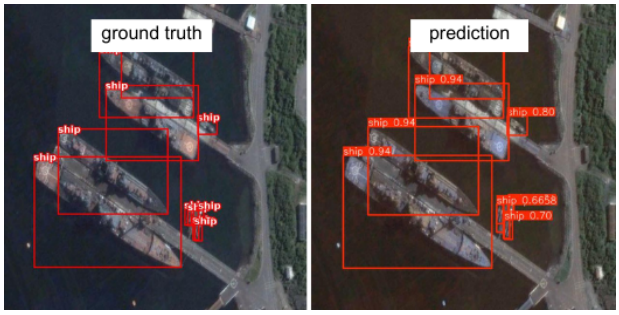

Figure 3: Comparison 2 between ground truth and object prediction via YOLOv8.

5. DISCUSSION

In the experiments, it is important to highlight that most of the models were tested with 50 training epochs to ensure a fair comparison. The only model trained with a different number of epochs was that from the Roboflow platform, which was trained with 300 epochs, as it is not possible to modify this parameter on the platform.

Considering the accuracy metric, mAP, RT-DETR was the most outstanding approach. But, for our surprise, this model presented one of the worst performance under the temporal perspective.

Taking it account all the metrics, the conclusion is that YOLOv8 was the most suitable model. Its mAP 50 is equal to RT-DETR and very close to the performance of Roboflow (the best in this case), and its mAP 75 was quite close to the best approach (RT-DETR). YOLOv8 was the fastest model to train, and its FPS and latency values were inferior to DETR but considerably better than RT-DETR. These results corroborate the popularity of the YOLO family of object detectors.

6. CONCLUSIONS

This study aimed to propose an intelligent approach for vessel detection using CNNs and ViTs. The results indicate that the YOLOv8 model presented the best overall performance (accuracy and temporal), demonstrating high assertiveness in different contexts and vessel sizes. These results have the potential to significantly improve maritime safety and environmental monitoring by enabling accurate detection in a variety of scenarios.

Future research includes the use of other deep learning models and more representative databases, considering the national territory. Additionally, the problem will be approached as a multi-class one, not only identifying the presence of a vessel but also determining its type.

ACKNOWLEDGMENTS

This research was developed within the project *Classificação de imagens e dados via redes neurais profundas para múltiplos domínios* (**IDeepS**: https://github.com/vsantjr/IDeepS) which is supported by LNCC via resources of the SDumont supercomputer. This reseach was also supported by CAPES and CNPq.

7. REFERENCES

[1] M. Silva. The ships of the future, July 2024. Accessed: [Date of Access].

[2] BBC News. The growing environmental impact of cruise ships, which will break records in 2024, July 2024. Accessed: July 10, 2024.

[3] L. A. Fernandes and M. Costa. Maritime security: Challenges and opportunities in the age of globalization. *Journal of Marine Science and Engineering*, 9(5):489, 2021.

[4] Jianwei Li, Jie Chen, Pu Cheng, Zhentao Yu, Lu Yu, and Cheng Chi. A survey on deep-learning-based real-time sar ship detection. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 16:3218–3247, 2023.

[5] Xinqiang Chen, Hao Wu, Bing Han, Wei Liu, Jakub Montewka, and Ryan Wen Liu. Orientation-aware ship detection via a rotation feature decoupling supported deep learning approach. *Engineering Applications of Artificial Intelligence*, 125:106686, 2023.

[6] Roboflow. Train a model in roboflow, 2024. Acesso em: 03 set. 2024.

[7] Joseph Redmon, Santosh Kumar Divvala, Ross B. Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. *CoRR*, abs/1506.02640, 2015.

[8] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollar. Focal loss for dense object detection. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 2999–3007. IEEE, 2017.

[9] Faheem Rustamy. Detection transformer (detr) vs yolo for object detection, 2023. Accessed: 3 set. 2024.

[10] Yian Zhao et al. Detrs beat yolos on real-time object detection. *arXiv*, 2024. Accessed: 02 Sep. 2024.

[11] Weiming Chen, Bing Han, Zheng Yang, and Xinbo Gao. Mssdet: Multi-scale ship-detection framework in optical remote-sensing images and new benchmark. *Remote Sensing*, 14(21), 2022.

[12] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. Yolov4: Optimal speed and accuracy of object detection. *CoRR*, abs/2004.10934, 2020.

[13] Kaiqi Huang Xizhou Zhang, Hanwang Zhang and Xiaogang Wang. Deformable detr: Deformable transformers for end-to-end object detection. *arXiv preprint arXiv:2005.12872*, 2020. Acesso em: 03 set. 2024.

[14] VisionWizard. Detr: An end-to-end object detection model, 2020. Accessed: 3 set. 2024.

[15] Ahmet Akdogan. Detr: End-to-end object detection with transformers and implementation in python, 2022. Accessed: 3 set. 2024.

[16] Rohan Volety. Author page on labellerr, 2024. Accessed: [Accessed: 3 set. 2024].

[17] Syed Sahil Abbas Zaidi, Mohammad Samar Ansari, Asra Aslam, Nadia Kanwal, Mamoona Asghar, and Brian Lee. A survey of modern deep learning based object detection models. *Digital Signal Processing*, 126:103514, 2022.

[18] Madhur Zanwar. Calculating map of models on edge device, 2024. Acesso em: 03 set. 2024.