Fayoum University Faculty of Engineering, 2025, Vol: 8(1), 68-78
https://fuje.journals.ekb.eg/
ISSN Online: 2537-0634
ISSN Print: 2537-0626

Airbus Ship Classification, Detection and Segmentation using Cutting-Edge Deep Learning Techniques

Fatma M. A. Mazen[1*] and Amna Mazen [2,3]

1Electronics and Communication Engineering Dept., Fayoum University, Faculty of Engineering, Fayoum, 63514, Egypt.
2Electrical Engineering Dept., University of Detroit Mercy, Detroit, MI, USA.
3Electrical Engineering Dept., Fayoum University, Faculty of Engineering, Fayoum, 63514, Egypt.
*Corresponding author: Fatma M. A. Mazen (fma04@fayoum.edu.eg).

How to cite this paper: Mazen, F.M.A. and Mazen, A. (2025). Airbus Ship Classification, Detection and Segmentation using Cutting-Edge Deep Learning Techniques. t. *Fayoum University Journal of Engineering,* Vol: 8(1), 68-78.
https://dx.doi.org/10.21608/fuje.2024.300746.1083

Abstract

With the increase in ship traffic comes an increase in the likelihood of at-sea offences. Problems such as environmentally disastrous ship accidents, piracy, illicit fishing, drug trafficking, and illegal cargo movement can all be addressed by detecting ships as rapidly as feasible in satellite images. Automatic ship detection in remote sensing images is a challenging problem due to the complexity of scene clutter and the diversity of ship scale and position. In this study, we set up a pipeline with two models: a classifier for recognizing the presence of ships in images and a mask predictor for ship location. Because most images do not contain ships, they must pass through a binary classifier that predicts ships' existence. Subsequently, images containing ships undergo processing by the mask predictor, yielding a mask specific to each image. The proposed binary classification algorithm achieved benchmark results on the Airbus ship detection dataset with 98.26% accuracy, outperforming the scores obtained using traditional methods. The Cascaded Mask R-CNN network performance outperformed the Mask R-CNN, QueryInst, and DetectoRS networks based on mean average precision.

Keywords

Airbus Ship detection; Cascaded Mask R-CNN; Mask R-CNN; QueryInst, DetectoRS; Instance Segmentation

1. Introduction

Utilizing remote sensing imagery for detecting ships is vital to maritime security. This includes the protection against unauthorized fishing activities, the monitoring of ship traffic, the surveillance of sea pollution, and the regulation of oil discharge. While ship detection can be accomplished manually through the Automated Identification System (AIS) for ships equipped with Very High Frequency (VHF) transponders, this method is prone to errors due to the potential disconnection of transponders by human operators. Therefore, there is an imperative need for an automatic technique that does not rely on human intervention, as stated in Ramani et al (2019).

The advancement of remote sensing technology has resulted in the availability of high-resolution imagery from aerial and satellite sensors. As a result, object recognition in remote-sensing images is critical for military and civilian applications. However, variations in targets, occlusion, clutter, and complicated backgrounds make identifying targets challenging. As a result, object recognition and instance segmentation are among the most challenging topics in computer vision that have received much attention in the past, as highlighted in Hafiz and Bhat (2020). One of the techniques used to collect images for object detection is Synthetic Aperture Radar (SAR), which uses radio waves to acquire images. Unlike optical images, the wavelengths used by SAR are unaffected by weather conditions or time of day, making them suitable for ship detection and segmentation applications.

Despite the efforts of numerous researchers in developing algorithms for the recognition and detection of ships in remote sensing images, the task remains challenging due to the complexity of various scene interference variables such as clouds, waves, and other issues Hu et al (2020); Li et al (2018). These interference variables, such as mists, clouds, and ocean waves, frequently result in low contrast and blurring of ship targets, making them difficult to discern from the background. Furthermore, the diversity in the size and shape of ships in remote sensing images presents additional challenges for detection models, as ships may be docked close together, making it difficult to identify each ship individually. Despite these difficulties, there remains significant potential for further research in this area, particularly in the face of the multi-scale changes and complicated environmental interference often encountered in remote sensing images of ships.

2. Related Research

Huang et al. (2021 Recent advancements in the field of object detection have led to the development of highly accurate and efficient Convolutional Neural Network (CNN) based algorithms, such as You Only Look Once (YOLO)Redmon and Farhadi (2017), Single Shot MultiBox Detector (SSD)Liu et al (2016), and Faster Region-Based Convolutional Neural Networks (R-CNN) Faster (2015). In a recent study, Patel et al (2022) developed an automatic ship detection system based on deep learning techniques. They trained three YOLO models, YOLOv3, YOLOv4, and YOLOv5, using the Kaggle Airbus ship detection dataset Kaggle (2019) and Shipsnet Rhammell (2018). The performance of the models was evaluated using accuracy as the metric. The study demonstrated that the YOLOv5 object detection model outperformed the other YOLO versions.

Sharma et al (2022) conducted a comparison of state-of-the-art object detection models for ship detection in satellite images. The models evaluated include YOLOv3, YOLOv4, RetinaNet152, EfficientDet-D2, and Faster-R-CNN, tested using the Airbus ship detection dataset. To improve the mean Average Precision (mAP), the authors integrated the YOLO v4 model with a custom selection of anchor boxes using the Kmeans++ clustering algorithm. The dataset was converted from Run Length Encoding (RLE) to YOLO format to conduct experiments on YOLO models. For the other models, Tensorflow object detection API was utilized. The dataset was converted into TensorFlow records to perform experiments on EfficientDet-D2, RetinaNet152, and Faster R-CNN. The experiment results indicated that the K-means++ clustering model outperformed all other object detection models in terms of the mAP of smallsize ships. In contrast, Faster R-CNN achieved the best results for mAP of medium and large-size ship detection. Regarding speed, YOLOv3 achieved the shortest detection time among all models. The RetinaNet152 had difficulty detecting large ships, while EfficientDet-D2 failed to detect medium and small ships.

Karki and Kulkarni (2021) utilized Keras and Fastai to train classification and segmentation models for ship detection and segmentation in satellite images. The experiments were conducted using the Kaggle competition's Airbus ship detection dataset. Due to hardware constraints, all images were resized to 224x224. To minimize false positives, the authors employed a ResNet34 pre-trained model to train a classifier to determine whether an image contained a ship. The authors also used Fastai to construct a UNet segmentation model with a Googlenet encoder and self-attention mechanism and a UNet model with an EfficientNet encoder using Keras. The results indicated that the Googlenet encoder model better segmented smaller and closer ships than the EfficientNet encoder model.

Rakhi et al (2022) presented an enhanced Mask R-CNN model for ship detection in satellite images. The proposed method can detect and segment ships at the pixel level. Their main contribution lies in adding more Convolutional layers in the segmentation head and adjusting hyper-parameters to improve overall output. They managed to reduce the number of false positives while improving overall performance and detecting small ships. They achieved an F2 score of 0.82956 on the test set.

The rest of the paper is structured as follows: Section 2 summarizes the most recent research for Airbus ship detection in satellite images, including those that employ deep learning models for ship classification and segmentation. Section 3 then briefly analyzes the Airbus ship detection dataset and the proposed classification and instance segmentation pipeline. Section 4 contains the results of the proposed binary classifier and instance segmentation model, as well as a thorough analysis and evaluation of the reported results. Finally, Section 5 concludes the study's key findings and suggests future research directions.

3. Methodology

This section is organized into three subsections. The first subsection provides a comprehensive description of the dataset. The second subsection focuses on the detailed classification phase. Lastly, the third subsection outlines the architectures utilized in the segmentation phase.

3.1 The Dataset

This paper uses the Kaggle-hosted Airbus Ship Detection dataset Kaggle (2019). This dataset pertains to a ship detection competition hosted by Airbus. This leading European aerospace company specializes in global development, manufacturing, and marketing civil and military aerospace products. The objective of the competition was to detect the presence of ships in satellite images and to place an aligned bounding box segment around them. The second version of this dataset, which was created by concatenating the training and testing images of the first version and adding a new set for testing purposes, is utilized in this study. As a result, the size of the training set in the second version is four times larger than in the first version. The dataset comprises approximately 192,000 square satellite images with a resolution of 768x768 pixels. Additionally, the dataset includes a CSV file that lists all image IDs and their corresponding pixel coordinates, representing ships' segmentation boxes. **Figure 1** illustrates some examples of images and their corresponding ship masks overlaid on them. Unfortunately, the dataset is plagued by a severe class imbalance, with only 42,556 out of 192,000 images containing ships. The absence of pixel coordinates in specific images indicates that these images do not contain ships. The ground truth segmentation masks are encoded as RLEs and have been transformed into the COCO format for ease of training.

This dataset is well-suited for testing image processing techniques, encompassing many real-world challenges. These include the diverse characteristics of ships, such as varying sizes and orientations, as well as the impact of weather conditions on image quality. Additionally, the dataset also features images with mists and clouds, as depicted in **Figure 2(a)** and **2(b)**, as well as images that depict ships close to one another, as illustrated in **Figure 2(c**.

Objects like islands and ports are attached to ships, as depicted in Fig 2d. These characteristics make it challenging to detect ships in remote sensing images, and thus, the dataset is useful for testing the effectiveness of proposed detection networks. Training a model with this dataset will make it more robust to variations in weather conditions and ship parameters.

Using this dataset, this work proposes a classification model applied to the entire dataset, followed by a segmentation model for images with ships only. In the classification phase, a binary classifier is trained to detect whether the image has a ship. However, images with ships are further processed throughout the instance segmentation process to create a mask over ships.

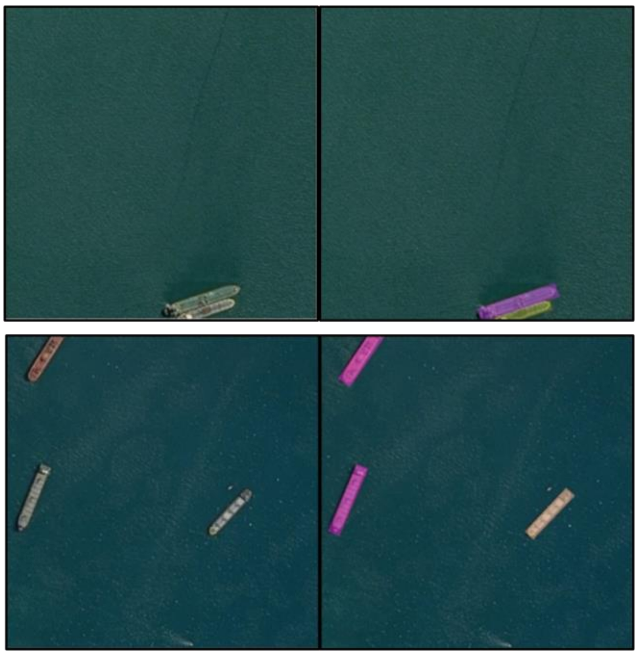


Figure 1. Sample of dataset pictures on the left and their corresponding mask overlaid on the right

3.2 Classification Phase

This paper uses the transfer learning technique to create the classifier rather than building a convolutional neural network from scratch. We used the pre-trained ResNet34 model from the Fastai library to implement the classifier. Fastai is a Pytorch library that facilitates the underlying setup for neural network training Howard and Gugger (2020). The number of training images, data augmentation techniques, and learning rate (LR) are the tuning parameters that affect the accuracy of the transfer learning-based model. Thus, different data augmentation techniques are considered to increase the number of images and make the trained model more robust for variations. Horizontal flipping, rotation, zoom, and lighting are the data augmentation techniques used in this work. In addition, progressive image resizing is used to build deep learning-based classifiers to achieve better generalization. It involves starting with small images during training and gradually switching to larger ones as the model improves. This approach speeds up training initially and enhances the final accuracy. In our work, we applied this strategy using Fastai, starting with 256x256 images and progressively increasing to 384x384 and 512x512 as training progressed.

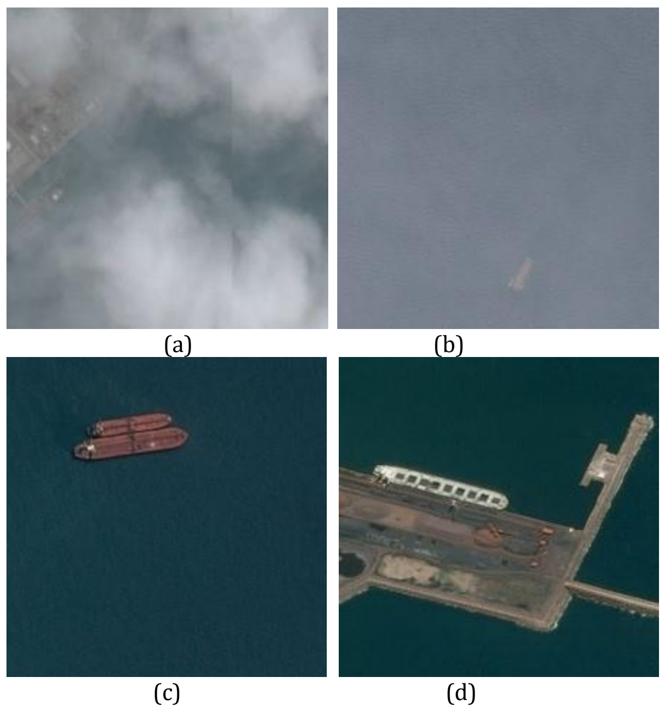


Figure 2. Challenges in Airbus ship detection dataset: (a) Mists, (b) Clouds, (c) Nearly touching ships, (d) Islands and ports attached to ships

The rate at which the model converges to a local minimum during optimization is largely determined by its LR. As a result, choosing a suitable LR value is an essential stage in the optimization procedure. The training procedure takes a lengthy time if the LR is too small, and it may skip the best answer if it is too high. In our work, we estimated the initial LR using a method known as the Cyclical Learning Rate (CLR) finder. The CLR finder records the loss at each step when training a mini-batch at a very low LR and then gradually increasing it. This procedure aids in identifying a good beginning LR, which in our instance was 3e-03, indicated in **Figure 3** by a red dot. We also used the idea of differential learning rates during training, which assigns distinct LR values to various neural network components. This method is applied to further optimize the learning process. We can group the layers in a CNN and provide each group a set of variable LR values. The first layers (green) in **Figure 4** use a tiny LR to capture fundamental properties such as edges and forms. In order to highlight the distinctive aspects of the dataset, the middle layers (yellow) have a greater LR. Lastly, the last layers (red) make use of the highest LR appropriate for their function.

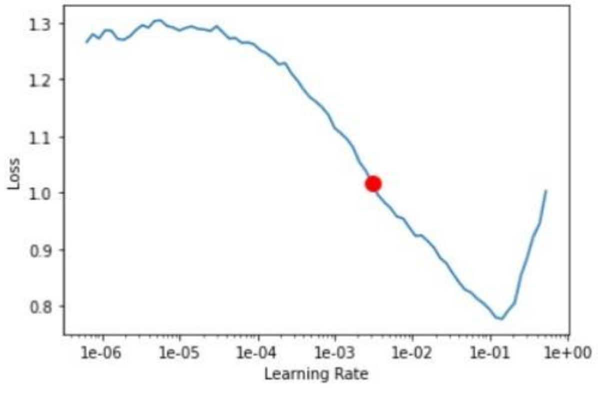


Figure 3. Cyclical learning rate

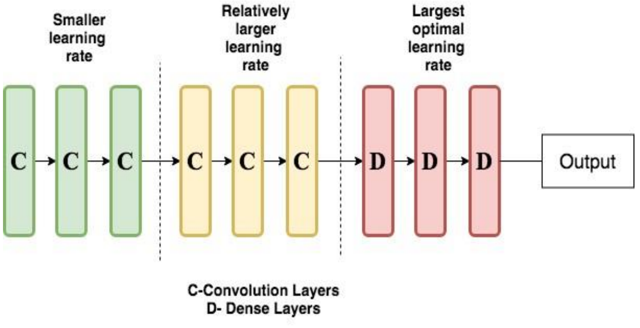


Figure 4. Sample CNN with differential LR

3.3 Instance Segmentation Phase

The goal of the instance segmentation phase is to locate and identify ships at the pixel level in images that contain ships. We use four well-known instance segmentation techniques: DetectoRS, QueryInst, Cascaded Mask R-CNN, and Mask R-CNN. To find the most effective approach, we assess compare the four approaches' respective performances.

3.3.1 Mask R-CNN

Mask R-CNN He et al (2017) is a deep learning framework used in instance segmentation tasks in computer vision. It is an extension of the Faster R-CNN object detection model, which combines object detection and instance segmentation into a single architecture. It works by breaking down the process into several steps. First, it uses a backbone model to extract the image's features. Then, it uses a region proposal network to determine where objects might be in the image. After that, it applies a pooling layer and converts all the regions to the same shape. Finally, these regions are passed through a fully connected network to predict the class label and bounding boxes and generate the segmentation mask.

3.3.2 Cascaded Mask R-CNN

Cascade Mask R-CNN is an extension of the Mask R-CNN architecture designed for more accurate object detection and instance segmentation. It builds upon the capabilities of Mask R-CNN by introducing a cascade of detectors, where each detector refines the results of the previous one. **Figure 5** depicts the network architecture of Cascaded Mask R-CNN, where 'B' represents bounding boxes used for object localization. The 'C' represents the classification component that labels the detected objects. The 'S' represents the segmentation component in the mask branch, which generates pixellevel masks for each instance. Finally, we extract the features from the input data and then apply some convolution operations using the convolutional head ’H’.

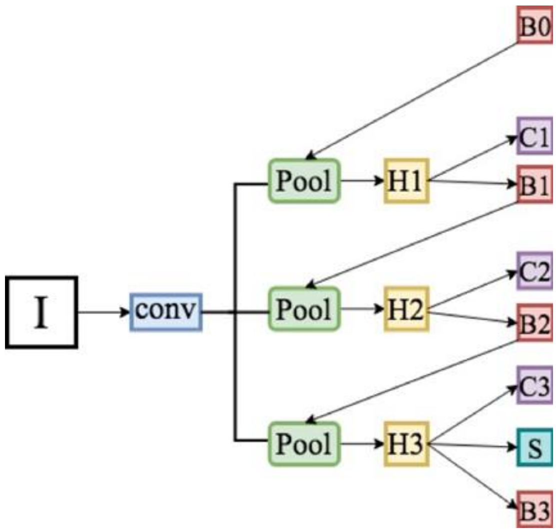


Figure 5. Cascaded Mask R-CNN

3.3.3 QueryInst

QueryInst is a simple and effective query-based instance segmentation technique. It is a multi-stage end-to-end system that considers instances of interest as learnable queries. QueryInst approach flips the traditional paradigm. Instead of starting with queries and searching for relevant instances, it starts with instances (objects in the image). It treats them as queries to retrieve information about other instances or the scene. It comprises a query-based object detector and six dynamic mask heads supervised simultaneously. **Figure 6** depicts only two of the six parallel stages of QueryInst architecture.

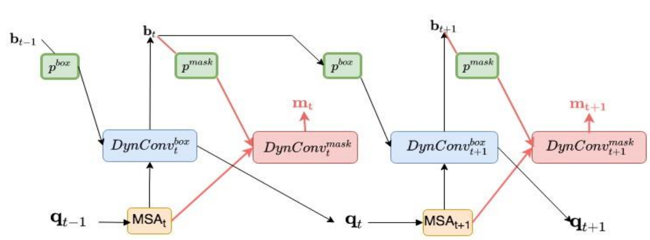


Figure 6. A quick overview of QueryInst architecture

3.3.4 DetectoRS

DetectoRS Qiao et al (2020) is a state-of-the-art computer vision model and framework for object detection in images. It is designed to identify and locate objects using bounding boxes within complex and cluttered scenes accurately. The model incorporates a recursive feature pyramid, a multi-scale representation of image features. This pyramid helps the model capture objects at various scales and is crucial for detecting objects of different sizes within the same image. Atrous convolutions, also known as dilated convolutions, are a type of convolutional operation used in deep learning for feature extraction. DetectoRS introduces ”switchable atrous convolution,” shown in Figure 7, which allows the model to adaptively choose the dilation rate (spacing between sampled pixels) during the convolutional operation.

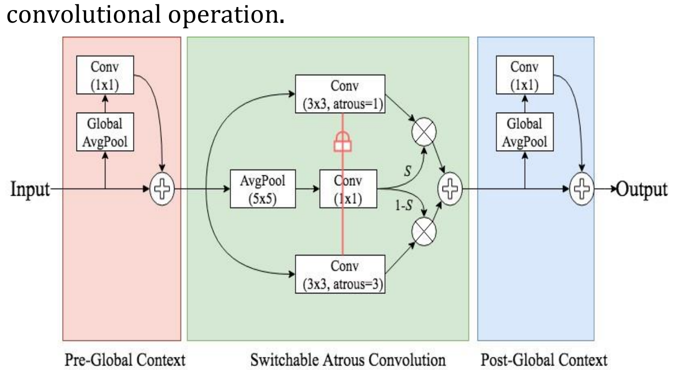


Figure 7. Switchable Atrous Convolution

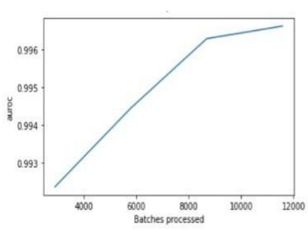
This adaptability helps the model capture fine-grained details and context in objects of interest.

4. Experimental Results and Discussion

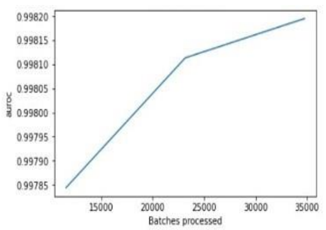
This section assesses the performance of the proposed binary classifier, ResNet34, using Area Under the Receiver Operating Characteristic (AuROC) curve and various evaluation metrics such as accuracy, precision, recall, and specificity. In addition, we analyze the detection and segmentation results for four architectures: Mask C-RNN, Cascaded C-RNN, QueryInst, and DetectoRS.

4.1 Classification phase

This work uses the AuROC curve and different evaluation metrics based on the confusion matrix to assess the performance of the proposed ResNet34 binary classifier. The AuROC calculates the discriminability degree between the predicted probabilities of positive and negative classes for evaluating the performance of binary classification models Kailkhura et al (2020). **Figure 8** shows AuROC curves for the proposed ResNet34 classifier using the initial image size of 256×256 and the final size of 512×512. The AuROC values of 99.6 and 99.8 in **Figure 8(a)** and **Figure 8(b)**, respectively, reflect a high confidence level in the predictions generated by the models. This suggests that the models exhibit a strong capability in accurately distinguishing and classifying the instances belonging to each class.



(a)



(b)

Figure 8. AuROC curves of the proposed ResNet34 classifier: (a) AuROC for 256×256 image, (b) AuROC for 512×512 image

In a binary classifier, the confusion matrix has four parts: True Positives (TP), True Negatives (TN), False Positives (FP), and False Negatives (FN). TP is when the model correctly identifies positives. TN is when it correctly identifies negatives. FP is when it mistakenly identifies positives, and FN is when it mistakenly identifies negatives. In Fig.9, we present the confusion matrix for the ResNet34 classifier using 512×512 images. This matrix compares predicted versus actual classes. Here, ”0” represents ” no ship,” and ”1” denotes ”ship”. This work considers four evaluation metrics: accuracy, precision, recall, and specificity.

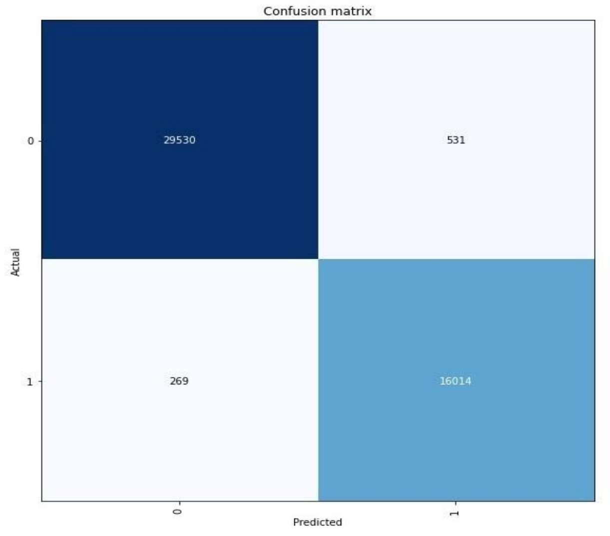


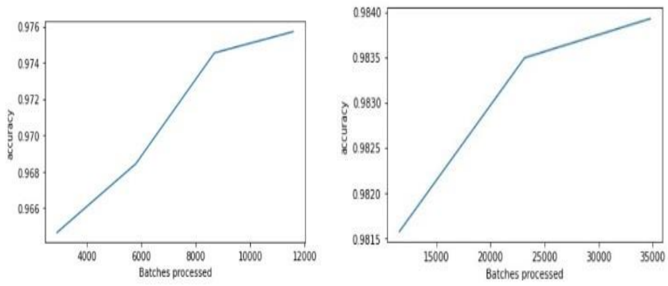
Figure 9. Confusion matrix of the proposed ResNet34 classifier

Accuracy is a common metric in evaluating classification models. It calculates the fraction of correct predictions out of all predictions made. Mathematically, accuracy is given by Equation 1. ResNet34 achieved 97.6% and 98.4% for 256x256 and 512x512 image sizes, respectively. In the presence of class imbalance within a dataset, accuracy may not accurately measure model performance. The confusion matrix enables evaluating the model's performance using metrics such as Precision, Recall, and Specificity for each class.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{1}$$

Recall (also known as sensitivity or true positive rate) is the proportion of correctly predicted positive instances out of the total actual positive instances as shown in Equation 2. It quantifies how well the classifier captures positive instances. Precision is the proportion of correctly predicted positive instances out of the total instances predicted as positive, as illustrated in Equation 3. It quantifies how well the classifier identifies positive instances. Finally, Specificity (also known as true negative rate) is the proportion of correctly predicted negative instances out of the total actual negative instances as expressed in

Equation 4. It quantifies how well the classifier identifies negative instances.



(a) (b)

Figure 10. Accuracy of the proposed ResNet34 classifier: (a) Accuracy curve for 256×256 image, (b) Accuracy curve for 512×512 image

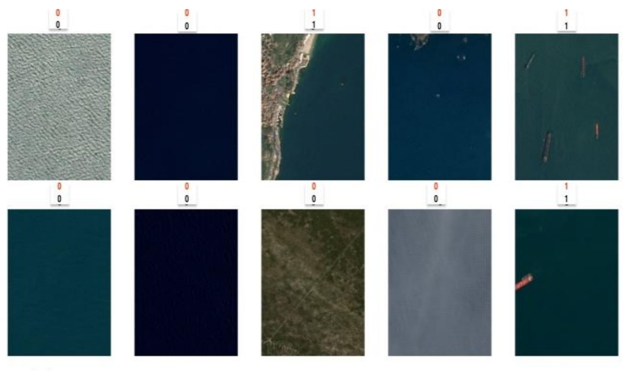


Figure 11. Sample predictions of the ResNet classifier on the validation dataset. The top label is the true label, and the bottom one is the predicted label

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \quad (2)$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \quad (3)$$

$$\text{Specificity} = \frac{\text{TN}}{\text{TN} + \text{FP}} \quad (4)$$

The confusion matrix shows our classifier's precision is 98.35%, recall is 96.79%, and specificity is 99.10%. This enhances its effectiveness in correctly identifying ”ship” images. Figure 11 shows the ResNet classifier's predictions on test images. The top red label is the true label, and the bottom is the predicted label. The classifier accurately identifies both ”ship” and ”no ship” images.

4.2 Instance Segmentation phase

In the segmentation phase of our study, we evaluated four distinct models: Cascaded C-RNN, Mask C-RNN, QueryInst, and DetectoRS. Each model was tested under uniform conditions to ensure a fair comparison. The fixed parameters are step-LR policy, linear LR warmup, a training duration of 12 epochs, and utilizing the COCO dataset for training purposes. In this work, we trained all models with LR=0.02/8 to satisfy the linear scaling rule and prevent the gradient explosion problem in training using a single GPU. The models also varied in certain parameters depending on their specific configuration. These variable parameters included the model type, the backbone architecture, and the choice of optimizer, as detailed in **Table 1.** displays the ground truth masks for a pair of images from the validation set.

(a)

(b) (c)

(d) (e)

Figure 12. Comparison of segmentation models: (a) Ground truth, (b) Cascaded R-CNN segmentation, (c) Mask R-CNN segmentation, (d) QueryInst segmentation, (e) DetectoRS segmentation.

Table 1 Type, backbone, and optimizer settings for the proposed models.

	DetectoRS	QueryInst	Mask R-CNN	Cascaded Mask R-CNN
Model type	HybridTask-Cascade	QueryInst	Mask R-CNN	Cascaded R-CNN
Backbone	DetectoRS ResNet50	ResNet50	ResNeXt101	ResNet50
Optimizer	SGD	AdamW	SGD	SGD

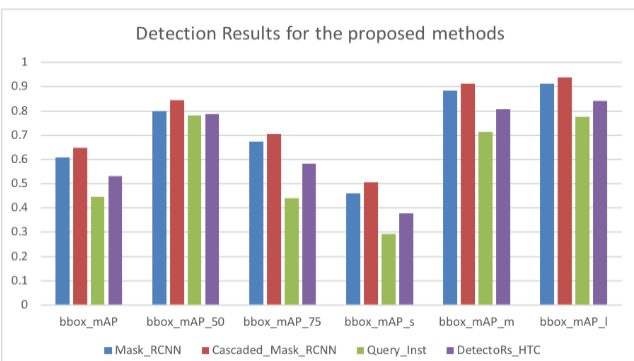


Figure 13. Detection Results for the proposed methods

To assess the accuracy of object detection, various versions of the mAP given by **Equation 5** have been utilized where Average Precision (AP) represents the area under the precision-recall curve (where *n* refers to number of classes).

$$mAP = \frac{1}{n}\sum_{j=1}^{n} AP_j \qquad (5)$$

The mAP at an Intersection over Union (IoU) threshold of 0.5 (mAP@50) has been employed, focusing on easily detectable objects. Additionally, a more comprehensive evaluation has been conducted by considering a broader range of IoU thresholds (0.5-0.95), denoted as (mAP@50-95). This approach enables the assessment of both high and low overlap between predicted and ground truth bounding boxes.

Furthermore, to gain insights into the model's performance in detecting objects of varying sizes within an image, specific metrics such as mAP_s, mAP_m, and mAP_l have been utilized to evaluate the detection of small, medium, and large objects, respectively.

All segmentation models can successfully predict all ships in easy-to-detect images**.** However, some models, generated some false positives in images. For instance, **Figure**

12(a) depicts the ground truth of an image form the dataset, where the ships are located near shorelines. Cascaded R-CNN and Mask-RCNN have successfully segment all ships correctly as shown in **Figure 12(b)** and **Figure 12(c),** respectively. However, QueryInst and DetectoRS, generated some false positives, highlighted by red circles as indicated in **Figure 12(d)** and **Figure 12(e),** respectively. **Figures 13 and 14** present a comprehensive view of these instance segmentation models' detection and segmentation performance. These results highlight the superior performance of the Cascaded Mask R-CNN model based on mAP, particularly in its ability to detect smaller ships in challenging conditions, such as cloudy environments, outperforming the other models in these aspects.

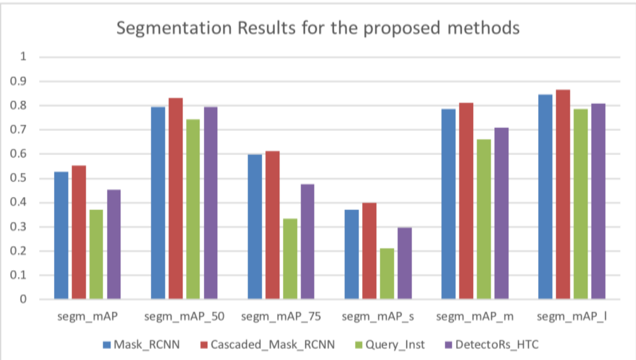


Figure 14. Segmentation Results for the proposed methods

7. Conclusion and Future Work

This paper introduces a two-stage pipeline for detecting and segmenting ships in satellite images from remote sensing, explicitly focusing on Airbus data. The first stage involves a binary classification system designed to ascertain the presence of ships. The second stage utilizes four advanced instance segmentation models to generate detailed masks of the ships. To augment the efficacy of the binary classifier, we applied a range of data augmentation techniques, including horizontal flipping, rotation, zooming out, and progressive resizing of images. Employing Transfer Learning and Progressive Resizing, the ResNet34 model demonstrated notable performance boost in binary classification tasks, achieving accuracy rate exceeding 98%. Comparative analysis of the state-of-the-art instance segmentation methods proposed in this work indicates that the Cascaded Mask R-CNN outperformed the other instance segmentation models based on the mAP value. Experimental results on the Airbus dataset demonstrate the effectiveness of the proposed approaches in detecting ships of various scales despite challenging environmental conditions.

Future work involves exploring and testing additional instance segmentation models to enhance the efficiency and accuracy of these models while reducing their computational demands.

Statements and Declarations

Funding: No funding for this study.

Conflict of Interest: Authors have not received research grants from any company. Authors have not received a speaker honorarium from any company. Authors do not own stock in any Company. Authors are not a member of any committee. Authors declare that they have no conflict of interest. Authors have no competing interests to declare that are relevant to the content of this article.

Data Availability Statement: Not applicable.

References

Faster R (2015) Towards real-time object detection with region proposal networks. Advances in neural information processing systems 9199(10.5555):2969239–2969250

Hafiz AM, Bhat GM (2020) A survey on instance segmentation: state of the art. International journal of multimedia information retrieval 9(3):171–189

He K, Gkioxari G, Doll´ar P, et al (2017) Mask r-cnn. In: Proceedings of the IEEE international conference on computer vision, pp 2961–2969

Howard J, Gugger S (2020) Deep Learning for Coders with fastai and PyTorch. O'Reilly Media

Hu J, Zhi X, Zhang W, et al (2020) Salient ship detection via background prior and foreground constraint in remote sensing images. Remote Sensing 12(20):3370

Kaggle (2019) Airbus ship detection challenge. https://www.kaggle.com/ competitions/airbus-ship-detection/data, [Online; accessed June 30, 2024]

Kailkhura V, Aravindh S, Jha SS, et al (2020) Ensemble learning-based approach for crack detection using cnn. In: 2020 4th International Conference on Trends in Electronics and Informatics (ICOEI)(48184), IEEE, pp 808–815

Karki S, Kulkarni S (2021) Ship detection and segmentation using unet. In: 2021 International Conference on Advances in Electrical, Computing, Communication and Sustainable Technologies (ICAECT), IEEE, pp 1–7

Li L, Zhou F, Zheng Y, et al (2018) Saliency detection based on foreground appearance and background-prior. Neurocomputing 301:46–61

Liu W, Anguelov D, Erhan D, et al (2016) Ssd: Single shot multibox detector. In: European conference on computer vision, Springer, pp 21–37

Patel K, Bhatt C, Mazzeo PL (2022) Deep learning-based automatic detection of ships: an experimental study using satellite images. Journal of imaging 8(7):182

Qiao S, Chen LC, Yuille A (2020) Detectors: Detecting objects with recursive feature pyramid and switchable atrous convolution. arXiv preprint arXiv:200602334

Rakhi AM, Dhorajiya AP, Saranya P (2022) Enhanced mask-rcnn for ship detection and segmentation. In: International Conference on Ubiquitous Computing and Intelligent Information Systems, Springer, pp 199–210

Ramani S, Prabakaran N, Kannadasan R, et al (2019) Real time detection and segmentation of ships in satellite images. Int J Sci Technol Res 8(12):2691–2695

Redmon J, Farhadi A (2017) Yolo9000: better, faster, stronger. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 7263–7271

Rhammell R (2018) Ships in satellite imagery. https://www.kaggle.com/rhammell/ ships-in-satellite-imagery, [Online; accessed June 30,2024]

Sharma R, Sharma H, Meena T, et al (2022) Performance evaluation of deep learning models for ship detection. In: International Conference on Computer Vision and Image Processing, Springer, pp 273–287