WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal Homepage: www.ijrpr.com ISSN 2582-7421

Deep Learning-Based Target Detection Using SAR Images

Dr. Chethan Chandra S Basavaraddi[1], Ms. Tanushree V Mugalikatti[2], Aishwarya M[3], Aliya Anjum M[4], Chandana U[5], Impana K R[6]

[1] Associate Professor, Department of Computer, Science and Engineering G M University Davangere, India Chethanchandrasb.cse@gmail.com
2Assistant professor Department of Computer Science and Engineering G M University Davangere, India tanushreevm@gmu.ac.in
[3]Department of Computer Science and Engineering G M Institute of Technology Davangere, India aishwaryam205009@gmail.com
[4]Department of Computer Science and Engineering G M Institute of Technology Davangere, India aliyaanjum1979@gmail.com
[5]Department of Computer Science and Engineering G M Institute of Technology Davangere, India chandanaujjinappa2005@gmail.com
[6]Department of Computer Science and Engineering G M Institute of Technology Davangere, India impanarevanasiddappa@gmail.com

ABSTRACT—

Synthetic Aperture Radar (SAR) imagery plays an important role in maritime surveillance and ship detection because it can capture images of the Earth's surface under different weather conditions and during both day and night. Automatic ship detection in SAR images is an important task for applications such as maritime traffic monitoring, coastal surveillance, border security, and marine safety. However, detecting ships in SAR images is challenging due to speckle noise, variations in target size, complex sea backgrounds, and differences in radar backscatter intensity. This project presents a JavaScript-based approach for automatic target detection in SAR images using the OpenSARShip dataset. The main objective is to develop a simple and accessible web-based system that processes SAR images and identifies potential ship targets using CFAR and threshold-based detection techniques. The system allows users to upload SAR images through a web interface and performs image-based detection using JavaScript. The detection process analyses pixel intensity values to distinguish potential targets from the surrounding background. The detected regions are then highlighted in the processed image to make target identification easier. The proposed system is implemented using HTML, CSS, and JavaScript, providing an interactive interface for image uploading and visualization of detection results. By combining SAR image processing with a browser-based implementation, the project demonstrates a practical approach to ship detection without requiring a complex desktop-based software environment. The developed system focuses on demonstrating the detection process and visual identification of potential ship targets in SAR imagery. Quantitative performance measures such as precision, recall, and F1-score have not yet been calculated. The project provides a foundation for further improvements, including more advanced noise reduction techniques, adaptive threshold selection, and quantitative evaluation of detection performance.

Index Terms—Synthetic Aperture Radar, OpenSARShip, Ship Detection, CFAR, Thresholding, Image Processing, Target Detection.

I. INTRODUCTION

Traditional image-processing techniques have been widely used for detecting targets in SAR images. One commonly used approach is Constant False Alarm Rate (CFAR), which identifies possible targets by comparing their intensity with the surrounding background. In this project, Ordered Statistics CFAR (OS-CFAR) is considered as the traditional detection method. It uses ordered values from reference cells to estimate the background level and calculate a suitable detection threshold. This helps identify possible target regions without requiring a deep-learning model. However, its performance can be affected by complex backgrounds, clutter, and variations in image intensity [3], [4], [6].

Deep-learning-based methods have gained attention because they can learn important features directly from training images. Different object detection models, including convolutional neural networks and YOLO-based approaches, have been studied for detecting ships in SAR imagery. These methods can identify target locations and provide bounding boxes around detected objects. Previous studies have explored deep-learning techniques for improving ship detection and recognition in different SAR conditions [16]-[20].

The OpenSARShip dataset is one of the available datasets used in SAR ship detection research. It provides SAR image samples that can support the development and evaluation of ship detection techniques [1]. Using a suitable dataset is important because the model needs labelled image examples to learn the visual patterns associated with ship targets. The quality and variety of the training data can influence how well the model performs on unseen images.

Based on these observations, this work presents a Python-based SAR ship detection system that uses OS-CFAR and YOLOv5. The system first accepts a SAR image and performs preprocessing to improve its suitability for detection. The OS-CFAR method analyses local background information and identifies candidate target regions. The YOLOv5 model processes the image and predicts possible ship locations using bounding boxes and confidence scores. The results generated by both methods can then be examined and compared.

The system uses Python for backend processing, OpenCV and NumPy for image-processing operations, and PyTorch for running the YOLOv5 model. Flask manages the backend operations and connects the processing modules with the application interface. This implementation provides a way to upload SAR images, process them, and view the corresponding detection results.

The performance of the YOLOv5 model is evaluated using commonly used object detection metrics, including precision, recall, F1-score, and mean Average Precision (mAP). These measures help assess how accurately the model identifies ship targets and how well it locates them within the images. The visual outputs of OS-CFAR and YOLOv5 are also compared to understand the differences between statistical threshold-based detection and deep-learning-based detection.

The main contribution of this work is the implementation of two different detection approaches within a single SAR ship detection system. OS-CFAR provides a traditional statistical detection method, while YOLOv5 provides a learning-based approach for identifying ship targets. Their outputs and evaluation results help demonstrate the advantages and limitations of each method when processing SAR images.

The remainder of this paper is organized as follows. Section II presents the related work and literature review. Section III explains the proposed methodology and workflow. Section IV describes the system implementation, including preprocessing, OS-CFAR, and YOLOv5. Section V presents the experimental results and performance evaluation. Finally, Section VI concludes the work.

II. RELATED WORK

Ship detection in Synthetic Aperture Radar (SAR) images has received considerable attention because it supports maritime surveillance, coastal monitoring and vessel identification. Researchers have explored different techniques, including traditional image processing, statistical detection methods and deep-learning-based approaches. These methods aim to identify ship targets accurately, even when the images contain noise and complex backgrounds.

A. Traditional CFAR-Based Ship Detection

Traditional SAR ship detection methods commonly use statistical techniques to distinguish ship targets from the surrounding sea background. One widely used approach is the Constant False Alarm Rate (CFAR) algorithm. Rohling [3] discussed CFAR thresholding in situations involving clutter and multiple targets. This method estimates the background level using surrounding reference cells and determines a threshold for identifying possible targets.

Gao et al. [4] proposed an adaptive and fast CFAR algorithm for target detection in high-resolution SAR images. Their approach focuses on improving detection by adapting to background conditions. Similarly, Qin et al. [6] presented a CFAR detection algorithm for SAR images with generalized gamma-distributed backgrounds. These studies show the importance of adaptive threshold selection for detecting targets in different SAR environments. However, CFAR-based methods can still face difficulties when the background contains complex clutter or when target intensity is similar to the surrounding sea.

B. Speckle Noise Reduction in SAR Images

Speckle noise is one of the major challenges in SAR image processing because it can affect image quality and make target identification difficult. Huang et al. [11] introduced a method that combines speckle noise reduction with ship target detection in SAR images. Their work highlights the importance of reducing noise before performing target detection.

Chen et al. [12] also investigated speckle-free SAR image ship detection, focusing on improving the detection process by addressing the effects of speckle noise. These studies indicate that suitable preprocessing techniques can help improve the visibility of targets and support more reliable detection.

C. Deep-Learning-Based SAR Ship Detection

With the development of deep learning, researchers have started using neural-network-based approaches for automatic ship detection in SAR images. Unlike traditional methods that mainly depend on manually selected thresholds, deep-learning models learn useful features directly from training images.

Li et al. [16] investigated ship detection in SAR images using an improved Faster R-CNN approach. Kang et al. [17] proposed a contextual region-based convolutional neural network with multilayer fusion for SAR ship detection. These approaches demonstrate how deep-learning architectures can use image features and contextual information to identify ship targets.

Wang et al. [18] presented an automatic ship detection method based on RetinaNet using multi-resolution Gaofen-3 imagery. In addition, Li et al. [19] reviewed the development of deep-learning techniques for SAR ship detection, discussing earlier methods and recent developments. Zhang et al. [20] also reviewed recent advances in ship detection using single-channel SAR images. These studies highlight the growing role of deep learning in improving automatic target detection and localization.

D. Research Gap and Proposed Approach

The existing studies show that traditional CFAR methods are useful for identifying potential targets based on local background intensity, while deep-learning-based methods can learn image features and locate objects using bounding boxes. However, traditional detection methods may produce unwanted detections in complex backgrounds, whereas deep-learning approaches depend on suitable labelled training data and computational resources.

Based on these observations, the proposed work focuses on developing a Python-based SAR ship detection system that combines OS-CFAR and YOLOv5. OS-CFAR is used to identify potential target regions by estimating the local background, while YOLOv5 is used to detect ship targets and generate bounding boxes with confidence scores. The outputs of both approaches are considered to understand their detection behaviour on SAR images.

D. Statistical and Signal-Based Ship Detection Methods

Apart from CFAR, researchers have explored other statistical and signal-processing techniques for identifying ships in SAR images. Wang et al. [5] studied ship detection using the alpha-stable distribution, which provides a statistical approach for analysing the characteristics of SAR image backgrounds. Such methods are useful because SAR images contain variations in intensity that make it difficult to separate actual targets from surrounding sea regions.

Eldhuset [7] developed an automatic ship and ship-wake detection system for spaceborne SAR images in coastal regions. This work demonstrates the importance of automatic detection techniques for analysing large coastal areas. Marino et al. [9] presented a comparison of different ship detection algorithms using spectral analysis of SAR images. Their study highlights the need to examine different detection approaches to understand their performance under varying image conditions.

Stasolla et al. [10] conducted a comparative study of operational vessel detection methods for maritime surveillance using satellite-based SAR images. Such comparative research is important because different detection algorithms may behave differently depending on the image background, target characteristics and processing conditions.

E. SAR Datasets for Automatic Ship Detection

The availability of labelled SAR datasets has supported the development of automatic ship detection systems. Huang et al. [1] introduced the OpenSARShip dataset, which is dedicated to interpreting ships in Sentinel-1 SAR imagery. Publicly available datasets provide researchers with image samples that can be used for developing and evaluating detection approaches.

Wang et al. [13] introduced a SAR ship detection dataset designed for deep learning under complex backgrounds. Their work supports the development of models that learn ship-related features from labelled SAR images. Zhang et al. [14] presented the SAR Ship Detection Dataset (SSDD), including its official release and a detailed analysis of the dataset.

Wei et al. [15] introduced the High-Resolution SAR Images Dataset (HRSID) for ship detection and instance segmentation. These datasets provide useful resources for training and testing deep-learning-based detection models. In the present work, the OpenSARShip dataset is considered for studying ship detection using OS-CFAR and YOLOv5.

F. Comparison of Traditional and Deep-Learning Approaches

Traditional statistical methods and deep-learning approaches follow different strategies for detecting ships in SAR images. Traditional methods such as CFAR mainly analyse local image intensity and background characteristics to identify possible targets [3], [4]. These methods provide a statistical approach to detection but can be affected by complex backgrounds and unwanted image responses.

Deep-learning-based approaches use labelled training images to learn useful features for identifying and locating objects. Studies involving improved Faster R-CNN [16], contextual region-based CNN [17] and RetinaNet [18] demonstrate the application of different neural-network architectures to SAR ship detection.

Li et al. [19] discussed the development of deep-learning-based SAR ship detection methods and highlighted the progress made in this research area. Zhang et al. [20] also reviewed recent developments in single-channel SAR ship detection. These studies show that selecting an appropriate detection method is important for improving the identification and localization of ship targets.

G. Research Gap and Motivation for the Proposed Work

The reviewed studies demonstrate that several approaches have been developed for ship detection in SAR images. Statistical methods, including CFAR, estimate the local background to identify potential targets, while deep-learning approaches use learned image features to detect and locate objects.

However, each approach has its own requirements and limitations. Statistical methods depend on background estimation and threshold selection, while deep-learning models require labelled datasets and suitable computational resources. Therefore, studying both approaches within a common detection system can help understand their differences.

Motivated by these observations, this work presents a Python-based SAR ship detection system that combines OS-CFAR and YOLOv5. The OS-CFAR method is used for statistical target identification, whereas YOLOv5 performs deep-learning-based ship detection. The system allows the detection outputs of both methods to be examined and compared, providing a practical way to study traditional and deep-learning-based approaches for SAR ship detection.

III. METHODOLOGY

The proposed system is developed using Python by integrating two different techniques for ship detection in Synthetic Aperture Radar (SAR) images: Ordered Statistics Constant False Alarm Rate (OS-CFAR) and YOLOv5. The OS-CFAR algorithm identifies potential ship regions by analysing

pixel intensity and estimating the surrounding background level. YOLOv5 uses deep-learning techniques to recognize ship targets and predict their locations. Both approaches are implemented within the same system to examine their detection outputs and compare their performance.

A. Python-Based System Development

Python is selected as the primary programming language because it supports image processing, numerical operations, and deep-learning model development. The system uses different Python libraries to perform individual processing tasks.

- OpenCV: Used for reading, processing, and handling SAR images.

- NumPy: Supports numerical calculations and pixel-intensity operations.

- PyTorch: Provides the framework for loading and running the YOLOv5 model.

- Flask: Manages backend processing and communication between the detection modules and the web interface.

The system is organized into separate processing modules. This makes it possible to execute OS-CFAR and YOLOv5 independently and collect their respective outputs for comparison.

B. OS-CFAR-Based Ship Detection

Ordered Statistics Constant False Alarm Rate (OS-CFAR) is implemented as a traditional statistical approach for identifying potential ship targets in SAR images. The method estimates the local background intensity and calculates a detection threshold according to the surrounding image conditions.

The detection process begins by selecting a test cell, which represents the pixel or region currently being examined. A sliding window moves across the image to analyse the surrounding area. The window contains the following components:

1. Test Cell (CUT): The pixel or region whose intensity is checked to determine whether it represents a possible target.

2. Guard Cells: Cells surrounding the test cell that are excluded from background estimation to reduce the influence of the target response.

3. Reference Cells: Cells used to estimate the local background or clutter level.

The intensity values from the reference cells are collected and arranged in ascending order. Instead of using all reference values directly, OS-CFAR selects a particular ordered value, known as the order statistic, to estimate the background level.

Using this estimated background level, the algorithm calculates a detection threshold. The intensity of the test cell is then compared with this threshold.

- If the test-cell intensity exceeds the threshold, it is identified as a potential target

- If the test-cell intensity does not exceed the threshold, it is classified as background.

This process is repeated across different image locations to generate candidate target regions. The resulting detections are highlighted in the output image.

OS-CFAR provides an adaptive approach because the threshold depends on the estimated local background. However, strong clutter and intensity variations can still result in false detections. CFAR thresholding and adaptive detection methods have been investigated in earlier radar detection studies [3], [4], [6].

C. YOLOv5-Based Deep Learning Detection

To identify ship targets more selectively, the proposed system also integrates YOLOv5, a deep-learning-based object detection model. Unlike OS-CFAR, which mainly depends on intensity variations, YOLOv5 learns patterns and visual features from annotated training images.

The model is implemented using PyTorch. During training, SAR images and their corresponding annotations are provided to the model. The annotations contain the locations of ship targets, represented using bounding boxes. The model learns to identify relevant target features and predict their positions in the images.

The YOLOv5 training process includes the following stages:

1. Input Image Processing

The SAR images are provided to the model in the required input format. The images and their corresponding annotations are prepared for model training.

2. Feature Learning

The model learns important visual patterns from the training images. These learned features help distinguish ship targets from surrounding background regions.

3. Bounding Box Prediction

The model predicts bounding boxes to indicate the positions and dimensions of possible ship targets in the image.

4. Loss Calculation

During training, the model uses three main loss components:

- Bounding Box Loss: Measures how closely the predicted bounding boxes match the actual target locations

- Objectness Loss: Helps the model determine whether a predicted region contains an object.

- Classification Loss: Measures whether the predicted object category is correct.

5. Model Optimization

Stochastic Gradient Descent (SGD) optimization is used to update model parameters during training. The implementation also supports pretrained weights and data augmentation.

6. Target Detection

After training, the model is used to process new SAR images. It predicts the detected ship locations and produces bounding boxes, object labels, and confidence scores.

The confidence score indicates how confident the model is about a particular prediction. The detection results can therefore be used to examine the identified ship regions. Previous research has explored CNN-based and other deep-learning approaches for SAR target detection [16]–[20].

D. Integration of OS-CFAR and YOLOv5

Both detection methods are integrated into the Python-based system to provide two different ways of analysing SAR images.

The OS-CFAR module processes the image using local background estimation and statistical thresholding. It identifies regions that have intensity values above the calculated detection threshold. The YOLOv5 module processes the image using features learned during model training and identifies possible ship targets through bounding boxes and confidence scores.

The integration allows the system to process the same SAR image using both approaches. Their outputs are collected separately and presented for visual comparison. This helps demonstrate the difference between traditional statistical detection and deep-learning-based object recognition.

E. Backend Processing Using Flask

Flask is used to manage the backend operations of the proposed application. When a user uploads a SAR image through the web interface, the image is transferred to the Flask backend for processing.

The backend coordinates the execution of the Python-based image-processing and detection modules. It invokes the preprocessing, OS-CFAR, and YOLOv5 scripts, collects their generated outputs, and returns the results to the frontend.

This arrangement separates the user interface from the detection operations and provides a structured way to manage image processing and model execution.

F. Detection Output and Visualization

After the processing is completed, the detection results are displayed through the web interface. The system presents the original SAR image and the corresponding outputs generated by OS-CFAR and YOLOv5.

The OS-CFAR output displays candidate regions identified through threshold-based detection. The YOLOv5 output displays detected ship targets using bounding boxes and confidence scores.

Displaying both outputs makes it easier to observe how the two methods identify possible ship regions. It also helps users understand the effect of background clutter on statistical detection and the role of learned image features in deep-learning-based detection.

G. Performance Evaluation

The YOLOv5 model is evaluated using a test dataset containing 232 SAR images and 546 ship instances. The evaluation uses standard object detection metrics to measure the model's performance.

The following metrics are considered:

- Precision: Measures the proportion of predicted ship detections that are correct.

- Recall: Measures the proportion of actual ship targets that are successfully detected.

- F1-score: Represents the balance between precision and recall.

- mAP@0.5: Measures mean Average Precision at an Intersection over Union (IoU) threshold of 0.5.

- mAP@0.5:0.95: Measures mean Average Precision across multiple IoU thresholds, providing a stricter evaluation of detection and localization.

The evaluation results help assess the model's ability to identify and localize ship targets in the selected test images. The OS-CFAR and YOLOv5 outputs are also compared visually to understand the differences between their detection approaches.

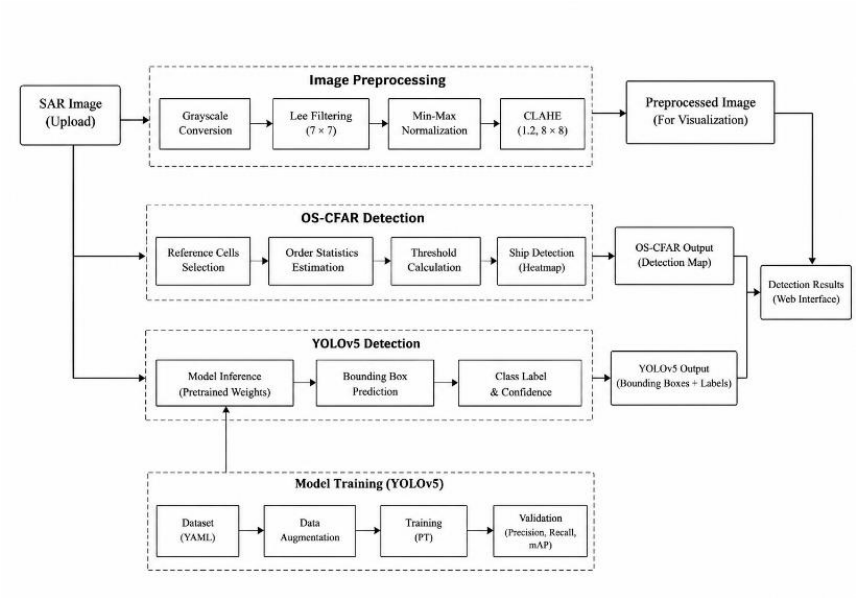


Fig. 1 Overall workflow of the proposed SAR ship detection system.

IV. SYSTEM IMPLEMENTATION

The proposed SAR ship detection system is implemented using Python and a combination of image-processing and deep-learning technologies. The main objective is to identify possible ship targets in SAR images using two approaches: OS-CFAR and YOLOv5. The system also includes a web interface that allows users to upload images and view the detection results. Python and Flask are used for backend processing, while OpenCV, NumPy, and PyTorch support image processing and deep-learning operations.

A. Software Environment

Python is used as the main programming language because it supports both image processing and deep-learning applications. OpenCV and NumPy are used to handle image operations and numerical calculations. PyTorch supports the implementation and execution of the YOLOv5 model. Flask manages backend operations and connects the processing modules with the web interface.

The frontend is developed using HTML, CSS, and JavaScript. HTML provides the structure of the application, CSS is used to design the interface, and JavaScript handles user interactions. These technologies work together to provide a simple environment for uploading SAR images and viewing the detection outputs.

B. Image Upload

The system allows users to upload SAR images in JPG, JPEG, PNG, TIF, and TIFF formats through the web interface. Once an image is selected, it is transferred to the Flask backend and temporarily stored for further processing.

The uploaded image is then provided as input to the image-processing and detection modules. This allows the same image to be examined using both OS-CFAR and YOLOv5, making their outputs easier to compare.

C. Image Preprocessing

Before applying the detection algorithms, the input SAR image passes through a preprocessing stage. This stage improves the image quality and prepares it for further analysis.

The following operations are performed:

1. Grayscale Conversion: The input image is converted into grayscale so that the processing can focus on intensity information.

2. Lee Filtering: A 7 × 7 Lee filter is applied to reduce speckle noise present in SAR images. Speckle noise can affect the appearance of targets and make detection more difficult.

3. Normalization: The image intensity values are scaled to the range of 0–255 to maintain a consistent intensity range.

4. Contrast Enhancement: Contrast Limited Adaptive Histogram Equalization (CLAHE) is applied with a clip limit of 1.2 and an 8 × 8 tile grid. This improves local contrast and makes image details easier to observe.

The processed images are saved and made available for visualization and subsequent detection.

D. OS-CFAR Detection Implementation

The OS-CFAR algorithm is implemented in Python to identify possible ship regions by analysing the local intensity information in the SAR image.

The algorithm examines the image using a sliding-window approach. It collects intensity values from the reference cells surrounding the test cell and arranges them in ascending order. A selected ordered value is used to estimate the local background level.

Based on this estimated background, the algorithm calculates a detection threshold. The intensity of the test cell is then compared with the threshold. If the intensity exceeds the threshold, the pixel is marked as a potential target; otherwise, it is treated as background.

This process is repeated across the image to identify candidate target regions. The detected regions are highlighted in the output image using bounding boxes. OS-CFAR follows the statistical detection approach discussed in earlier CFAR research, where adaptive thresholding is used to identify targets under different background and clutter conditions [3], [4], [6].

E. YOLOv5 Detection Implementation

YOLOv5 is implemented using PyTorch to identify ship targets through deep-learning-based object detection. The model is trained using annotated SAR images so that it can learn the important features and patterns associated with ship targets.

The training implementation supports pretrained weights and data augmentation to assist the learning process. Stochastic Gradient Descent (SGD) is used for model optimization. During training, the model uses three loss components:

- Bounding-box loss: Helps predict the correct position and size of ship targets.

- Objectness loss: Helps determine whether an object is present in a predicted region.

- Classification loss: Helps identify the correct object category.

After training, the model is used to detect ships in input SAR images. During inference, it generates bounding boxes, object labels, and confidence scores for the detected targets. These outputs are used to highlight the identified ship regions in the application.

Deep-learning approaches, including CNN-based detection and other object detection architectures, have been explored in previous SAR target detection studies [16]–[20]. The YOLOv5 model in this work is used as the selected detection model.

F. Backend Processing

Flask is used to manage communication between the frontend and the Python-based processing modules. When a user uploads a SAR image, the Flask backend receives the image and coordinates the required processing operations.

The backend executes the preprocessing, OS-CFAR, and YOLOv5 scripts using subprocess calls. After execution, it collects the generated image outputs and detection information. These results are then returned to the frontend for display.

This approach keeps the image-processing operations separate from the user interface and makes it easier to manage the different detection modules.

G. Web Interface and Result Visualization

The web interface is developed using HTML, CSS, and JavaScript. It provides an option to upload SAR images and view the results generated by the detection algorithms.

The interface displays the original input image, preprocessed image outputs, OS-CFAR detection results, and YOLOv5 detection results. For YOLOv5, the detected ship regions are displayed with bounding boxes and confidence scores. The OS-CFAR output highlights candidate regions identified using the calculated detection threshold.

The system also provides comparative visualizations so that users can observe the differences between the two detection approaches. OS-CFAR identifies candidate regions based on local intensity variations, whereas YOLOv5 uses learned image features to detect and localize ship targets. Overall, the implementation brings image processing, statistical target detection, deep-learning-based detection, and web-based visualization together in one application.

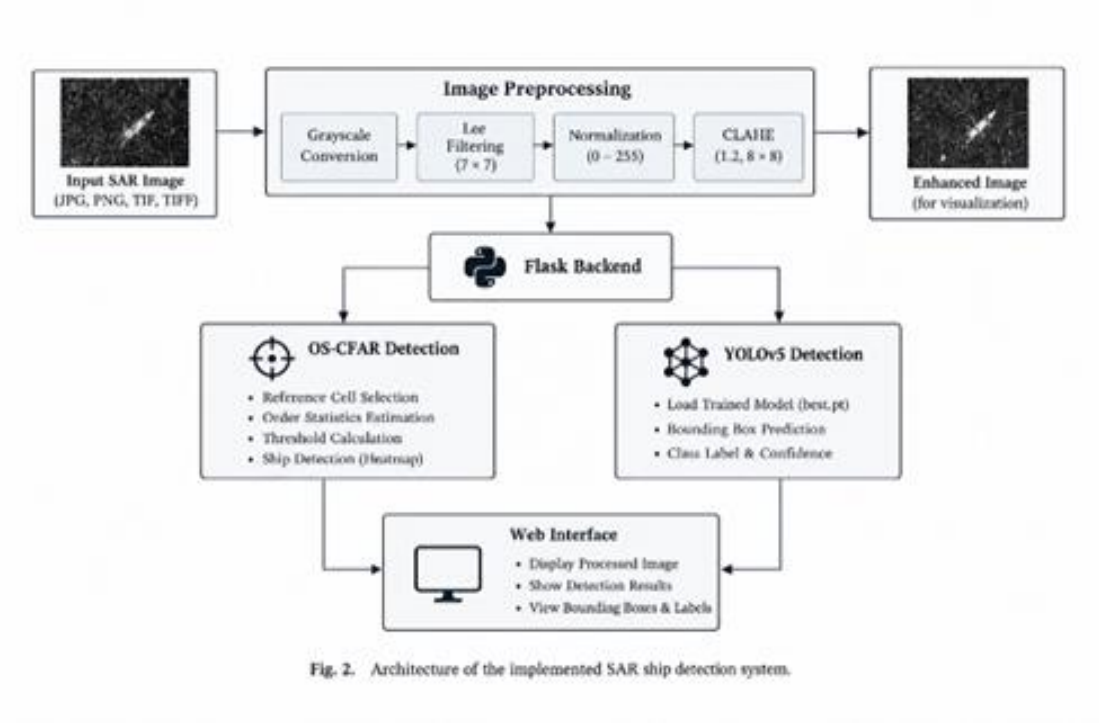


Fig. 2. Architecture of the implemented SAR ship detection system

V. RESULTS AND DISCUSSION

A. SAR Image Input

The first step of the proposed system is to upload a SAR image for ship detection. The uploaded image is displayed in the web application and then used as input for the detection process. In this project, two approaches, CFAR and YOLOv5, are used to identify possible ship regions in SAR images. The original input image is shown in Fig. 3.

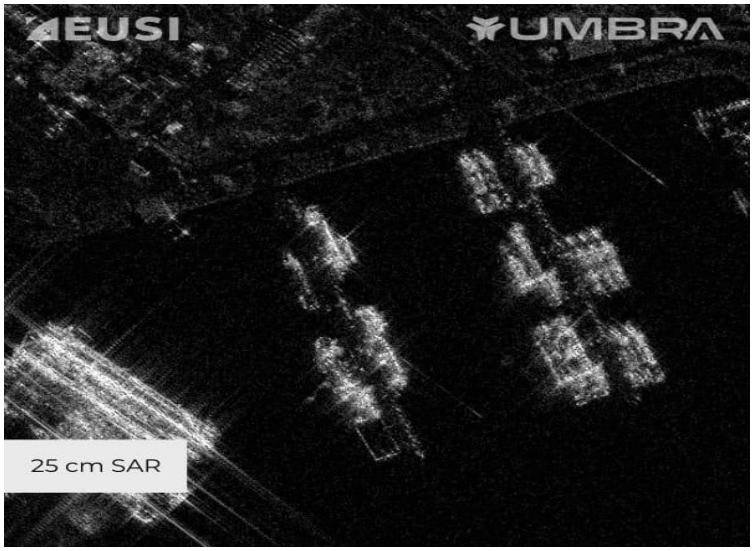


Fig. 3. Original SAR image used for ship detection.

B. CFAR-Based Ship Detection

The CFAR method is used as a baseline for detecting potential ship regions in SAR images. It works by analysing the intensity values of the image and identifying regions that differ from the surrounding background. The detected candidate regions are highlighted using green bounding boxes, as shown in Fig.4.

In the displayed example, the CFAR method identifies 61 candidate regions. However, several regions may correspond to background clutter rather than actual ships. This shows that intensity-based detection can produce unwanted detections, especially when the SAR image contains noise and complex background patterns.

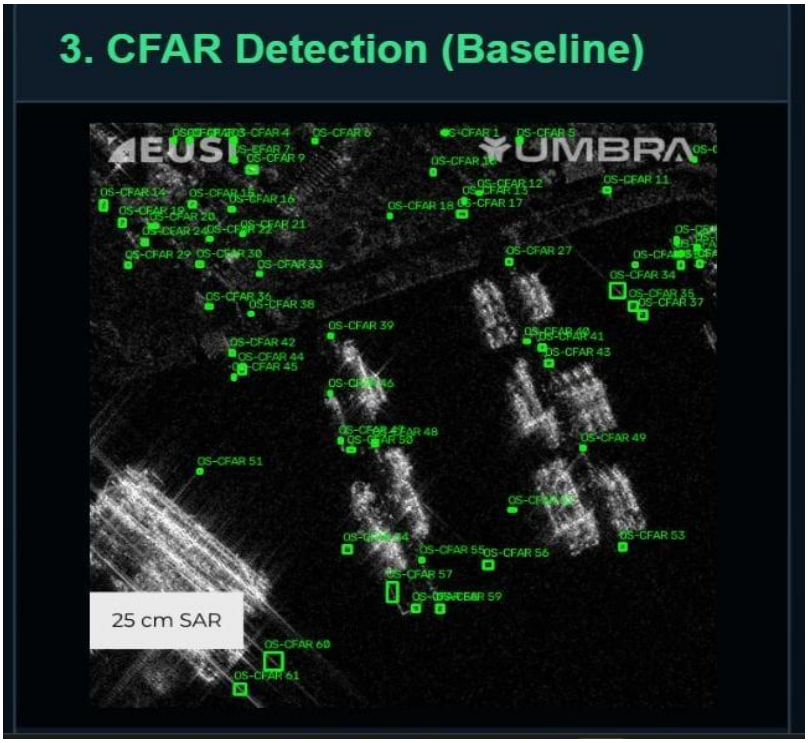


Fig. 4. Ship detection results using the CFAR baseline method.

C. YOLOv5-Based Ship Detection

To improve ship identification, the project also uses the YOLOv5 deep-learning model. The model analyses the SAR image and predicts the possible ship locations with bounding boxes and confidence scores.

As shown in Fig. 5, YOLOv5 identifies two ship targets in the displayed image. The detected ships have confidence scores of 0.91 and 0.82. The application dashboard reports an average confidence value of 0.861 for this image.

Compared with the CFAR output, YOLOv5 provides more selective detections in this example by identifying specific ship regions instead of marking a large number of candidate areas.

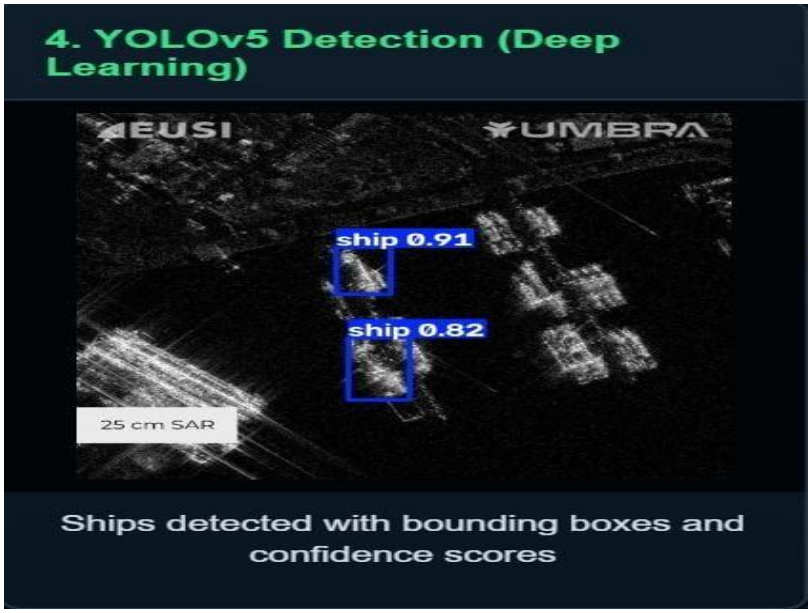


Fig. 5. Ship detection results using the YOLOv5 deep-learning model.

D. Visual Comparison of CFAR and YOLOv5

Fig. 6 presents the detection results of CFAR and YOLOv5 on the same SAR image. The CFAR method marks several candidate regions, whereas YOLOv5 identifies two ship targets using bounding boxes and confidence scores.

This comparison helps illustrate the difference between the two approaches. CFAR mainly depends on image intensity variations, while YOLOv5 uses features learned during model training to recognise ship objects. In the displayed example, YOLOv5 produces fewer and more focused detections.

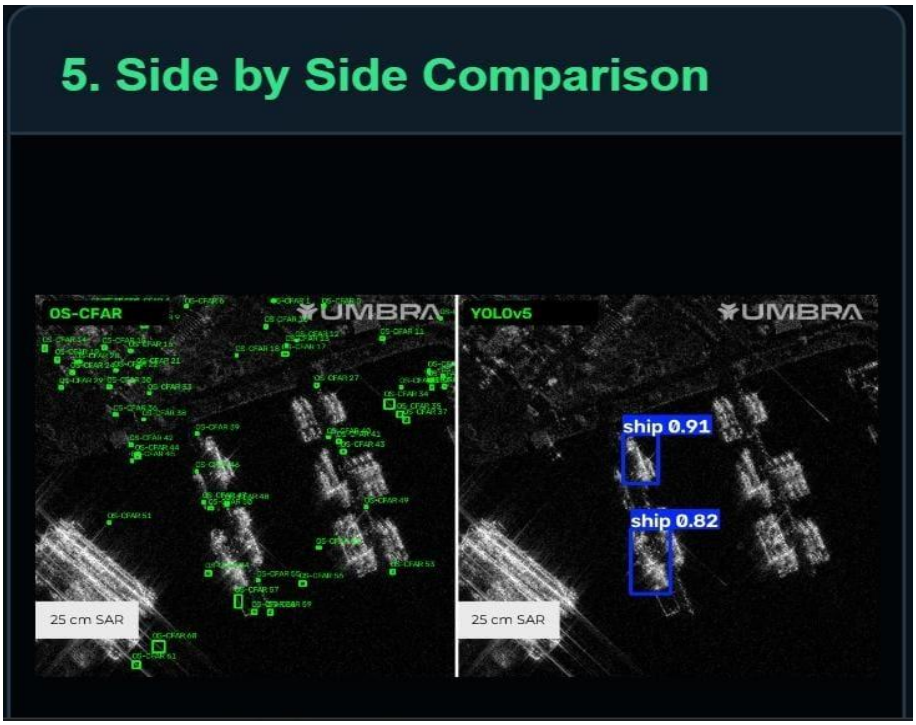


Fig. 6. Visual comparison between CFAR and YOLOv5 detection results.

E. Quantitative Performance Evaluation

The YOLOv5 model was evaluated using a test set containing 232 images and 546 ship instances. Its performance was measured using precision, recall, F1-score, and mean Average Precision (mAP).

The evaluation results are presented below.

TABLE I. YOLOv5 TEST-SET PERFORMANCE

Evaluation Metric	**YOLOv5**
Precision	0.957
Recall	0.937
F1-score	0.947
mAP@0.5	0.979
mAP@0.5:0.95	0.648

The model achieved a precision of 95.7% and a recall of 93.7%, with an F1-score of 94.7%. It also achieved an mAP@0.5 of 97.9%. These results show that the model was able to identify ship targets effectively on the evaluated test set.

The mAP@0.5:0.95 value was 64.8%, showing that achieving accurate bounding-box locations becomes more challenging when stricter IoU thresholds are considered.

F. Graphical Performance Comparison

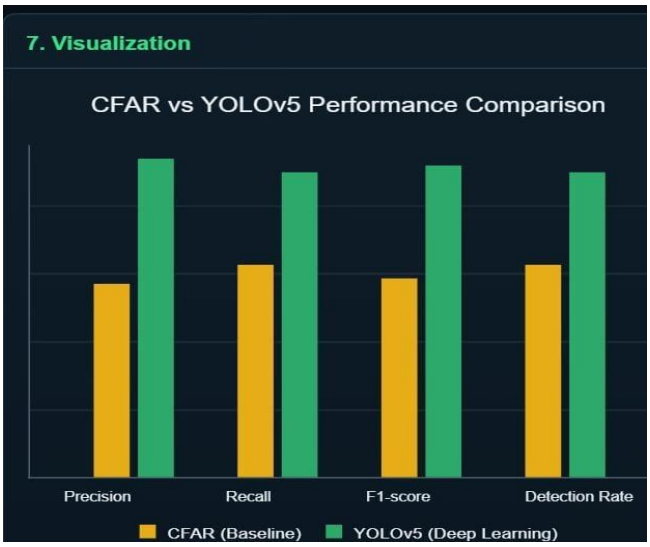


Fig. 8. Graphical comparison of CFAR and YOLOv5 performance metrics.

Fig. 8 presents a graphical comparison of the CFAR and YOLOv5 detection approaches. The graph provides a visual representation of the performance differences between the two methods.

The displayed results show higher values for YOLOv5 than for the CFAR baseline. This suggests that the deep-learning approach provides better ship identification in the evaluated experiment. However, a reliable numerical comparison requires both methods to be tested using the same images, ground-truth annotations, and evaluation metrics.

G. Overall Discussion

The results show how two different approaches can be used for ship detection in SAR images. The CFAR method identifies possible targets by analysing image intensity, but it may also detect unwanted regions because of noise and background clutter. YOLOv5, on the other hand, uses learned image features to identify ship targets and provides bounding boxes along with confidence scores.

Based on the test-set evaluation, YOLOv5 achieved a precision of 95.7%, recall of 93.7%, and mAP@0.5 of 97.9%. These results indicate that YOLOv5 performed well on the evaluated test set. The comparison also highlights the limitations of relying only on intensity-based detection and the potential benefits of using deep-learning techniques for SAR ship detection.

VI. CONCLUSION AND FUTURE WORK

This project presents a web-based approach for ship detection in Synthetic Aperture Radar (SAR) images using the OpenSARShip dataset. The proposed system implements JavaScript-based CFAR/threshold processing to analyze image intensity variations and identify potential ship regions. The detected candidate regions are highlighted using bounding boxes, allowing users to visualize the possible target locations through the web application.

The system provides a simple and accessible platform for uploading SAR images, processing them, and displaying the detection output. The implementation demonstrates how intensity-based thresholding techniques can be applied to SAR imagery without requiring a separately trained deep-learning model. However, the detection output may be affected by speckle noise, background clutter, and variations in target intensity. Further quantitative evaluation is required to determine the detection performance using ground-truth annotations.

The proposed system can be improved in several ways to enhance ship detection performance and reliability. Future work will focus on the following areas:

1. Improved Detection Accuracy: Optimize threshold parameters and detection procedures to improve the identification of ship regions in different SAR images.
2. False Detection Reduction: Introduce suitable filtering and noise-reduction techniques to minimize false detections caused by background clutter and speckle noise.
3. Quantitative Performance Evaluation: Use OpenSARShip ground-truth annotations to calculate precision, recall, F1-score, and other relevant evaluation metrics.
4. Deep Learning Integration: Explore deep-learning-based approaches such as YOLO for ship detection and compare their performance with the implemented threshold-based method.

5. Processing Optimization: Improve the processing speed and efficiency of the application to support the analysis of larger SAR images.
6. Enhanced Web Application: Improve the user interface by adding features such as detection-result downloading, image history, and detailed detection information.

These improvements can help develop a more reliable and efficient SAR-based ship detection system for potential applications in maritime surveillance and monitoring.

ACKNOWLEDGMENT

The authors would like to express their sincere gratitude to **GM Institute of Technology, Davangere**, for providing the necessary facilities and academic support to carry out this project.

The authors are thankful to the **Department of Computer Science and Engineering** for its encouragement, guidance, and support throughout the project work. We also extend our heartfelt thanks to our project guide, Head of the Department, and all the faculty members for their valuable suggestions, cooperation, and continuous support.

Finally, we would like to thank everyone who directly or indirectly contributed to the successful completion of this project.

REFERENCES

1. L. Huang, B. Liu, B. Li, W. Guo, W. Yu, Z. Zhang, and W. Yu, "OpenSARShip: A dataset dedicated to Sentinel-1 ship interpretation," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 11, no. 1, pp. 195–208, Jan. 2018, doi: 10.1109/JSTARS.2017.2755672.
2. R. Torres et al., "GMES Sentinel-1 mission," *Remote Sensing of Environment*, vol. 120, pp. 9–24, May 2012, doi: 10.1016/j.rse.2011.05.028.
3. H. Rohling, "Radar CFAR thresholding in clutter and multiple target situations," *IEEE Transactions on Aerospace and Electronic Systems*, vol. AES-19, no. 4, pp. 608–621, Jul. 1983, doi: 10.1109/TAES.1983.309350.
4. G. Gao, L. Liu, L. Zhao, G. Shi, and G. Kuang, "An adaptive and fast CFAR algorithm based on automatic censoring for target detection in high-resolution SAR images," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 47, no. 6, pp. 1685–1697, Jun. 2009, doi: 10.1109/TGRS.2008.2006504.
5. C. Wang, M. Liao, and X. Li, "Ship detection in SAR image based on the alpha-stable distribution," *Sensors*, vol. 8, no. 8, pp. 4948–4960, Aug. 2008, doi: 10.3390/s8084948.
6. X. Qin, S. Zhou, H. Zou, and G. Gao, "A CFAR detection algorithm for generalized gamma distributed background in high-resolution SAR images," *IEEE Geoscience and Remote Sensing Letters*, vol. 10, no. 4, pp. 806–810, Jul. 2013, doi: 10.1109/LGRS.2012.2224317.
7. K. Eldhuset, "An automatic ship and ship wake detection system for spaceborne SAR images in coastal regions," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 34, no. 4, pp. 1010–1019, Jul. 1996, doi: 10.1109/36.508418.
8. Z. Zhao, K. Ji, X. Xing, H. Zou, and S. Zhou, "Ship surveillance by integration of space-borne SAR and AIS—Review of current research," *The Journal of Navigation*, vol. 67, no. 1, pp. 177–189, Jan. 2014, doi: 10.1017/S0373463313000659.

A. Marino, M. J. Sanjuan-Ferrer, I. Hajnsek, and K. Ouchi, "Ship detection with spectral analysis of synthetic aperture radar: A comparison of new and well-known algorithms," *Remote Sensing*, vol. 7, no. 5, pp. 5416–5439, 2015, doi: 10.3390/rs70505416.

9. M. Stasolla, J. J. Mallorqui, G. Margarit, C. Santamaria, and N. Walker, "A comparative study of operational vessel detectors for maritime surveillance using satellite-borne synthetic aperture radar," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 9, no. 6, pp. 2687–2701, Jun. 2016, doi: 10.1109/JSTARS.2016.2551730.
10. S. Huang, D. Liu, G. Gao, and X. Guo, "A novel method for speckle noise reduction and ship target detection in SAR images," *Pattern Recognition*, vol. 42, no. 7, pp. 1533–1542, Jul. 2009, doi: 10.1016/j.patcog.2009.01.013.
11. S. W. Chen, X. C. Cui, X. S. Wang, and S. P. Xiao, "Speckle-free SAR image ship detection," *IEEE Transactions on Image Processing*, vol. 30, pp. 5969–5983, 2021, doi: 10.1109/TIP.2021.3089936.
12. Y. Wang, C. Wang, H. Zhang, Y. Dong, and S. Wei, "A SAR dataset of ship detection for deep learning under complex backgrounds," *Remote Sensing*, vol. 11, no. 7, Art. no. 765, 2019, doi: 10.3390/rs11070765.
13. T. Zhang et al., "SAR ship detection dataset (SSDD): Official release and comprehensive data analysis," *Remote Sensing*, vol. 13, no. 18, Art. no. 3690, 2021, doi: 10.3390/rs13183690.
14. S. Wei, X. Zeng, Q. Qu, M. Wang, H. Su, and J. Shi, "HRSID: A high-resolution SAR images dataset for ship detection and instance segmentation," *IEEE Access*, vol. 8, pp. 120234–120254, 2020, doi: 10.1109/ACCESS.2020.3005861.

15. J. Li, C. Qu, and J. Shao, "Ship detection in SAR images based on an improved Faster R-CNN," in *Proc. 2017 SAR in Big Data Era: Models, Methods and Applications (BIGSARDATA)*, Beijing, China, Nov. 2017, pp. 1–6, doi: 10.1109/BIGSARDATA.2017.8124934.

16. M. Kang, K. Ji, X. Leng, and Z. Lin, "Contextual region-based convolutional neural network with multilayer fusion for SAR ship detection," *Remote Sensing*, vol. 9, no. 8, Art. no. 860, 2017, doi: 10.3390/rs9080860.

17. Y. Wang, C. Wang, H. Zhang, Y. Dong, and S. Wei, "Automatic ship detection based on RetinaNet using multi-resolution Gaofen-3 imagery," *Remote Sensing*, vol. 11, no. 5, Art. no. 531, 2019, doi: 10.3390/rs11050531.

18. J. Li, C. Xu, H. Su, L. Gao, and T. Wang, "Deep learning for SAR ship detection: Past, present and future," *Remote Sensing*, vol. 14, no. 11, Art. no. 2712, 2022, doi: 10.3390/rs14112712.

19. C. Zhang, P. Liu, H. Wang, and Y. Jin, "A review of recent advance of ship detection in single-channel SAR images," *Waves in Random and Complex Media*, vol. 33, no. 5–6, pp. 1442–1473, 2023, doi: 10.1080/17455030.2022.2078016.