Satellite Ship Detection Using YOLOv8

Developed by: Adithi Jahnavi
Year: 2026

1. Abstract

This project presents an end-to-end deep learning system for detecting ships in satellite imagery using the YOLOv8 object detection model. The system was trained and evaluated in Google Colab using annotated satellite datasets. The best-performing model weights were exported and integrated into a Streamlit-based web application. The app was containerized using Docker and deployed on Hugging Face Spaces for real-time inference.

Additionally, a custom frontend website was built using React and TypeScript and deployed on Vercel to showcase the project and embed the live demo. The project demonstrates the complete pipeline from model training to full-stack deployment.

2. Problem Statement

Maritime monitoring plays a crucial role in port surveillance, illegal fishing detection, maritime security, and vessel traffic management. Manual inspection of satellite images is time-consuming and inefficient.

The goal of this project is to design and deploy an automated deep learning system capable of accurately detecting ships in satellite imagery and making the system publicly accessible via a web interface.

3. Model Development (Google Colab)

The YOLOv8 model was trained in Google Colab to utilize GPU acceleration. The dataset was structured in YOLO format with separate training and validation splits.

Pretrained YOLOv8 weights were fine-tuned using custom training parameters such as image size and number of epochs. Model performance was evaluated using precision, recall, and mean Average Precision (mAP). The best-performing weights (`best.pt`) were exported for deployment.

4. GitHub Repository Setup

A GitHub repository was created to maintain version control and organize the project files. The repository contains the Streamlit inference application, trained model weights, Docker configuration, dependency file, and project documentation. Using GitHub allowed systematic tracking of changes during debugging and simplified integration with Hugging Face and Vercel deployments.

5. Deployment on Hugging Face (Docker + Streamlit)

A Docker-based Hugging Face Space was configured to host the inference application. The Streamlit app loads the trained YOLOv8 model and processes uploaded satellite images in real time.

The Dockerfile was configured to handle dependency installation, Python compatibility, and system library requirements. After debugging CORS and file upload issues, the application was successfully deployed and made publicly accessible.

6. Frontend Development (React + TypeScript)

A custom frontend website was built using React, TypeScript, and Vite. The interface includes:

- Animated background
- Structured project sections
- Embedded live demo via Hugging Face iframe

Custom styling and layout enhancements were applied to present the project professionally.

7. Deployment on Vercel

The frontend repository was deployed using Vercel with automatic builds from GitHub. The final website hosts the project overview, technical documentation, and live demo.

8. System Architecture

User → React Frontend (Vercel) → Embedded Hugging Face App → Streamlit Backend → YOLOv8 Model → Prediction Output

This architecture separates model training, backend inference, and frontend presentation for modularity and scalability.

9. Challenges Faced

Key challenges included:

- Hugging Face 403 upload errors
- Docker configuration debugging

- Streamlit file path handling
- CORS restrictions

Each issue was resolved through systematic debugging and environment configuration adjustments.

10. Future Improvements

- Multi-class vessel detection
- Real-time satellite feed integration
- Model optimization for faster inference
- API-based backend architecture
- Analytics dashboard for prediction statistics

11. Conclusion

This project demonstrates the complete lifecycle of a detection model — from dataset preparation and model training to backend deployment, frontend development, and public hosting. It highlights practical deep learning implementation skills along with full-stack deployment experience.