Safety Equipment Detection

Using YOLOv8

GenIgnite Hackathon 2025
Team Name: TechTiaras

Team Members: SnehaRaj, SnehaKumari,TrijyaChoubey & Saumya Maurya

Event: GenIgnite Hackathon 2025

Date: 18 October 2025

The Challenge We Solved

The Problem

Detectingsafety-related objects such as fire extinguishers, alarms, and first-aid kits within **space-station environments** is challenging when performed manually. Varying lighting, object orientation, and occlusions inside spacecraft make consistent detection difficult.

Our Solution

An automatedAI-based system built using **YOLOv8**, trained on the **Falcon synthetic space-station dataset**, to accurately identify safety equipment under multiple conditions. The trained model is deployed through a **Streamlit web app**, offering a simple and interactive interface for real-time detection and visualization..

Implementation Roadmap

Ourprojectfollowedastructuredfive-stepdevelopmentprocesstotransformraw data into production-ready detection capabilities.

→ Dataset Preparation

Collected and annotated safety equipment images, then split into training (70%), validation (15%), and test sets (15%) to ensure robust model generalisation.

→ Environment Setup

Configured Anaconda environment with Python 3.10 and installed all required dependencies including YOLOv8, OpenCV, and Streamlit for development.

→ Model Training

Selected YOLOv8 for its excellent speed-accuracy trade-off and trained it on our annotated safety equipment dataset across multiple epochs.

→ Deployment

Integrated the trained model into a Streamlit application for real-time detection, providing an intuitive interface for end-users.

Model Performance Metrics

90.8%
Precision

Accuracy of positive predictions4 when model detects equipment, it is correct 90.8% of the time

65.8%
Recall

Sensitivity for capturing true positives4identifies 65.8% of all safety equipment present in images

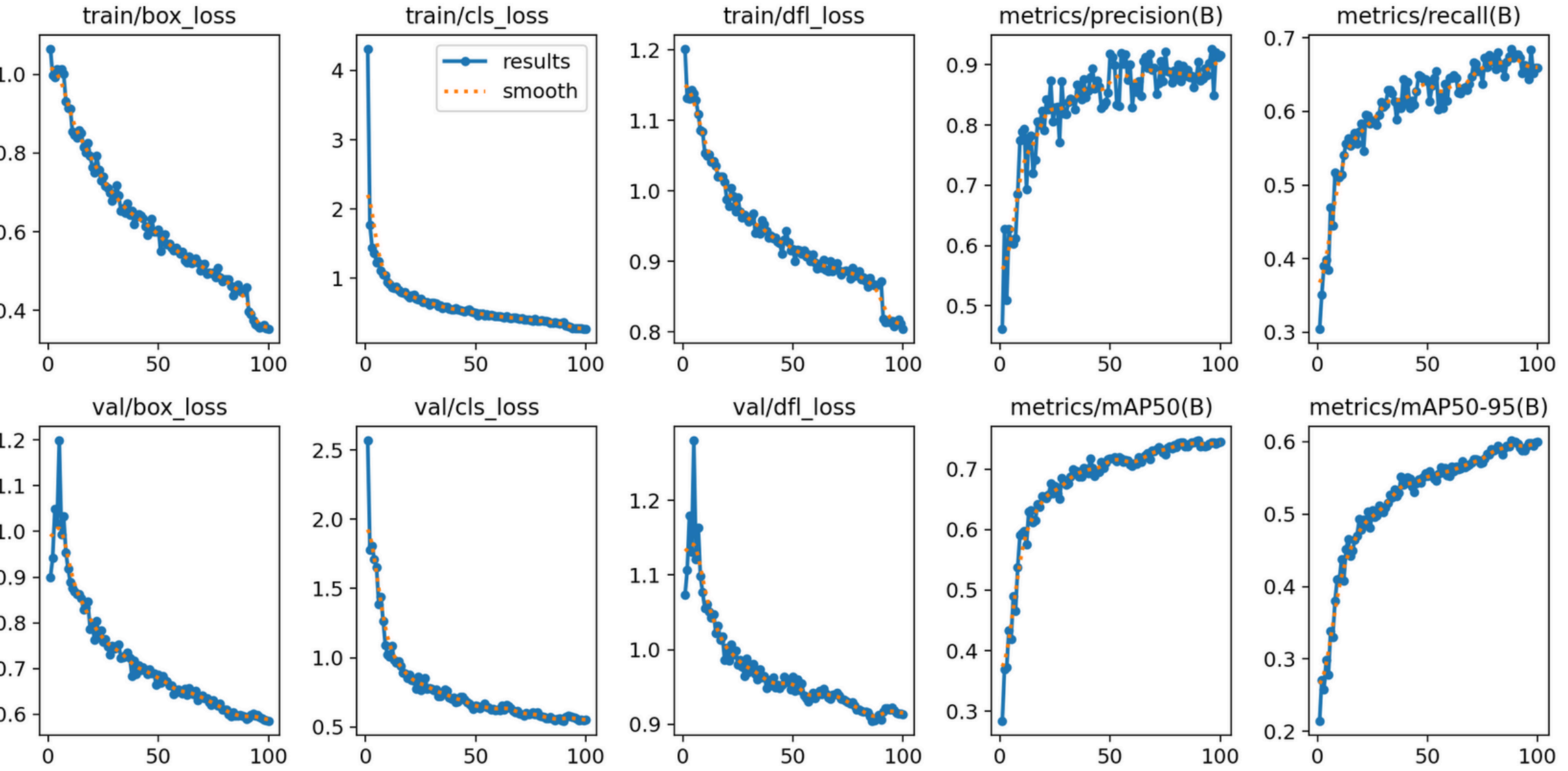
0.745
mAP@0.5

Mean average precision at 50% intersection-over-union threshold provides reliable detection benchmark

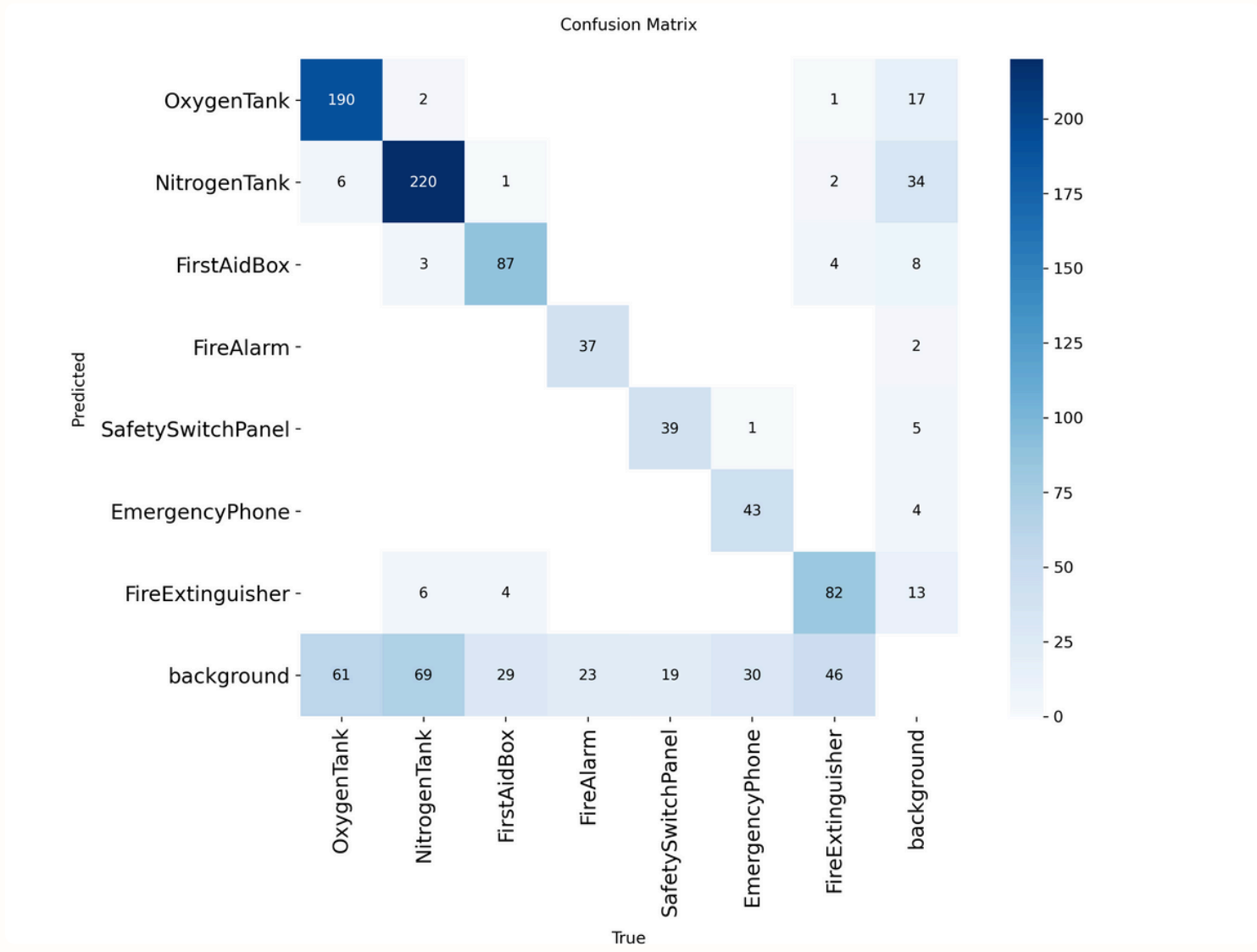
0.602
mAP@0.5-0.95

Rigorous metric across multiple IoU thresholds ensures detection quality across varying scenarios

Detection Results in Action



Challenges Overcome



We encountered several technical obstacles during development. Here's how we solved them to achieve our final results.

GPU Constraints

Limited GPU availability forced us to optimise for CPU training. We modified the training configuration to run on CPU by adjusting device parameters in train.py, then increased epoch counts to compensate for slower iteration cycles and achieve comparable accuracy.

Dataset Imbalance

Initial dataset contained uneven distributions across safety equipment classes. We implemented data cleaning and balancing techniques, including augmentation and sampling strategies, to ensure the model learned equitably across all object types.

Model Accuracy

Early training iterations showed suboptimal metrics. By systematically increasing training epochs, fine-tuning hyperparameters, and improving data quality, we progressively improved precision and recall to production-ready levels.

Technology Stack

OursolutionleveragesmodernPythonlibrariesandmachine learning frameworks optimised for computer vision tasks.

Core Technologies

- Python 3.10 4 primary programming language
- YOLOv8 (Ultralytics) 4 real-time object detection
- OpenCV 4 image processing and computer vision
- NumPy 4 numerical computing
- Pillow 4 image manipulation
- Streamlit 4 web application interface
- Anaconda 4 environment management

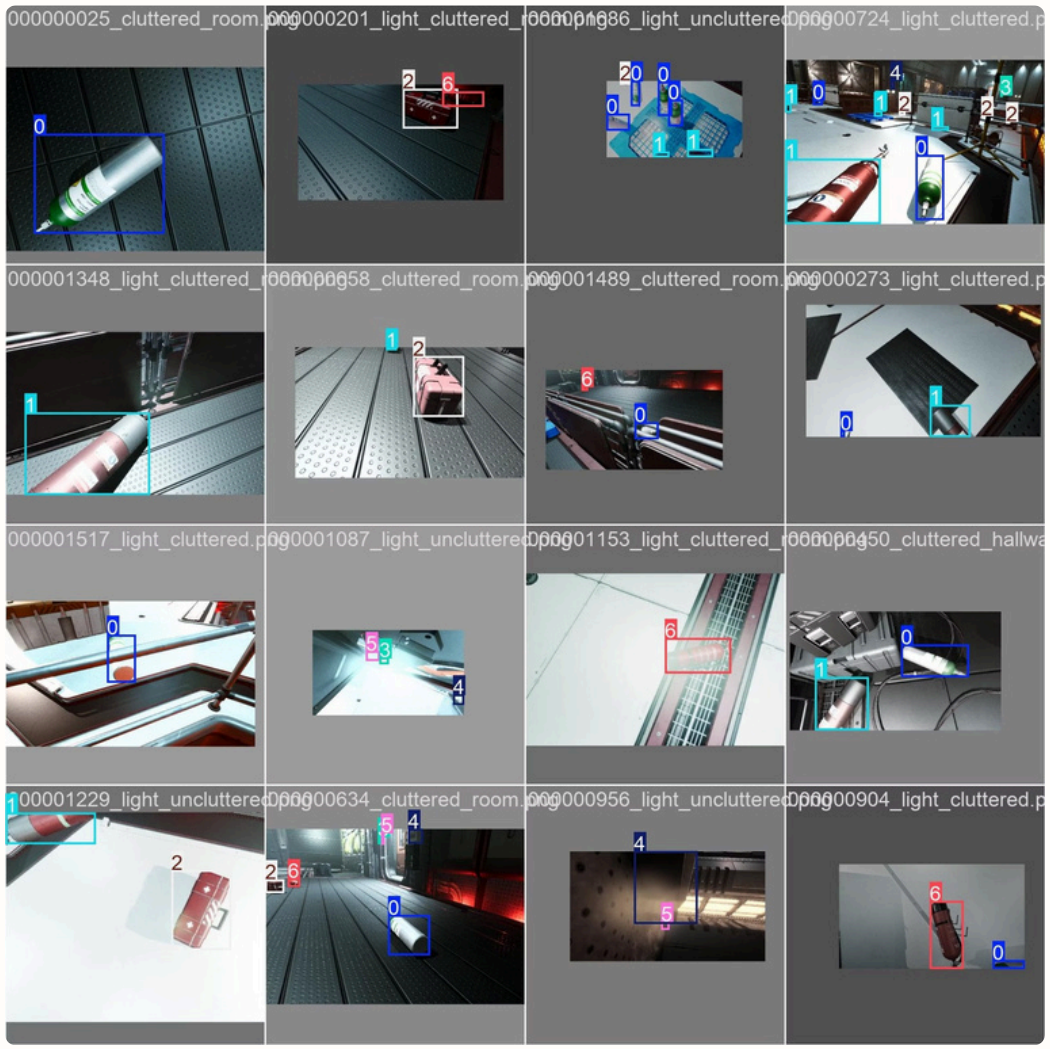
Key Commands

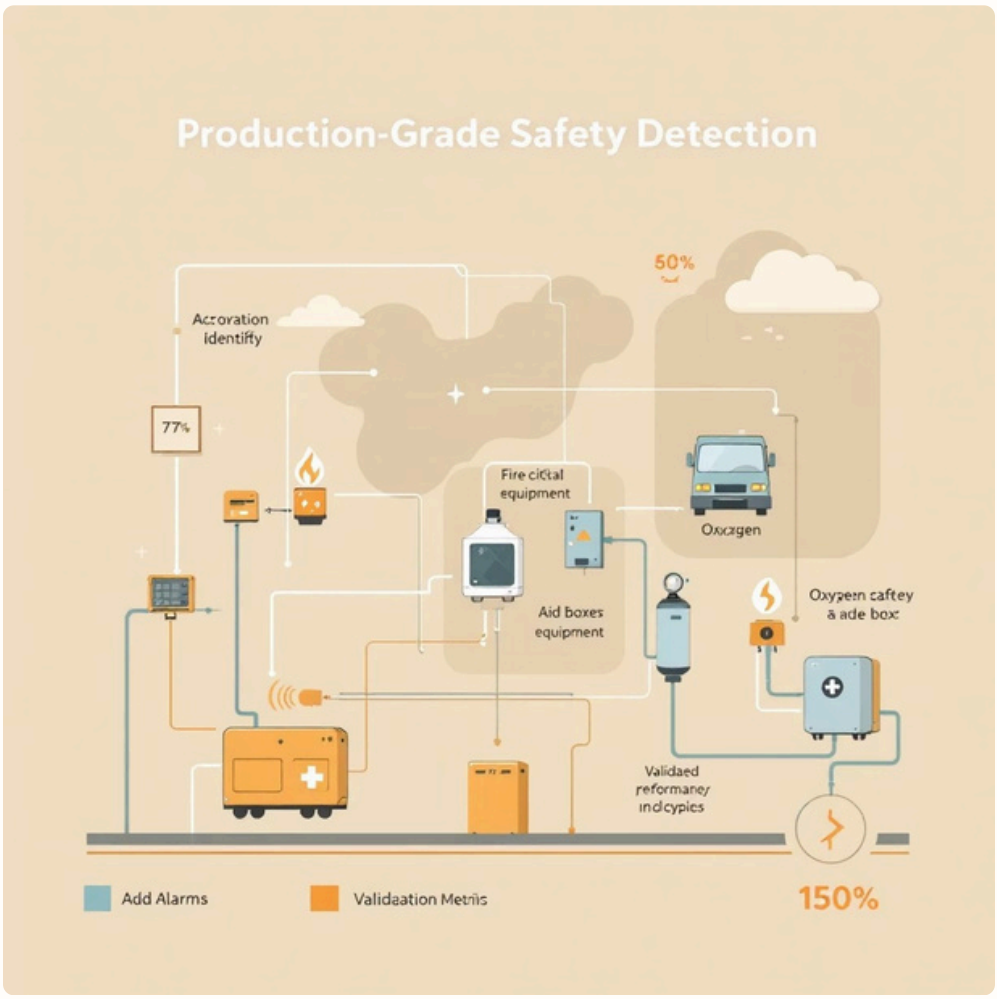
```
conda create -n SafetyDetect python=3.10
conda activate SafetyDetect

pip install ultralytics opencv-python
pip install pillow numpy streamlit
yolo train model=yolov8s.pt \
 data=safety_params.yaml epochs=100 \
 imgsz=640

yolo val model=runs/detect/train/\
 weights/best.pt data=safety_params.yaml
```

Results Achieved





Through systematic development and iterative refinement, we achieved a production-grade safety detection system with validated performance metrics.

Seven Classes Detected

Successfully trained model recognises fire alarms, oxygen tanks, first aid boxes, and four additional critical safety equipment types with high confidence.

mAP@0.5: 0.697

Achieved validation mean average precision of 0.697 at 50% IoU, demonstrating reliable and accurate detection across our safety equipment dataset.

Lightweight Deployment

Streamlit application provides intuitive real-time detection interface that runs efficiently on standard hardware, enabling accessible deployment across facilities.

Future Enhancements

Ourcurrentsystemestablishesasolidfoundation.Herearethestrategic improvements planned to expand capabilities and impact.

GPU Acceleration

Transition to GPU training to dramatically reduce iteration cycles, enable larger batch sizes, and train more sophisticated model variants for improved accuracy.

Live CCTV Integration

Connect the system to facility security cameras for continuous real-time monitoring, enabling automatic compliance verification across multiple locations simultaneously.

Edge Deployment

Deploy models to edge devices using frameworks like Falcon, enabling local processing with automatic updates and reduced dependency on centralised infrastructure.

Alert Dashboard

Develop intelligent alerting system that notifies security personnel of missing or improperly placed safety equipment, with historical compliance tracking.

Model Performance Metrics

OurYOLOv8modeldemonstratesstrongdetectioncapabilitiesacrosssafetyequipment classes with balanced precision and recall metrics.

89.1%
Precision
Accuracy of positive predictions4 when model detects equipment, it is correct 88.2% of the time

83.4%
Recall
Sensitivity for capturing true positives4identifies 60.1% of all safety equipment present in images

0.697
mAP@0.5
Mean average precision at 50% intersection-over-union threshold provides reliable detection benchmark

0.575
mAP@0.5:0.95
Rigorous metric across multiple IoU thresholds ensures detection quality across varying scenarios