

Duality AI - Space Station Hackathon Report



Team Name : Graviton

Project Name: Develop an AI-powered object detection system for a space station environment using synthetic data generated from Falcon, Duality AI's digital twin platform.

Brief Tagline: Leveraging synthetic data and AI-driven object detection to ensure the safety and operational efficiency of space station environments.

Methodology

Our methodology focused on training a YOLOv8 object detection model using Falcon synthetic data to identify critical space station objects (Toolbox, Oxygen Tank, Fire Extinguisher).

1. Data Preparation:

- Collected synthetic space station dataset from Falcon with YOLO-compatible labels.
- Dataset included Train, Validation, and Test splits with varied lighting, angles, and occlusions.
- Preprocessed images and verified class balance for accurate detection.

2. Model Training:


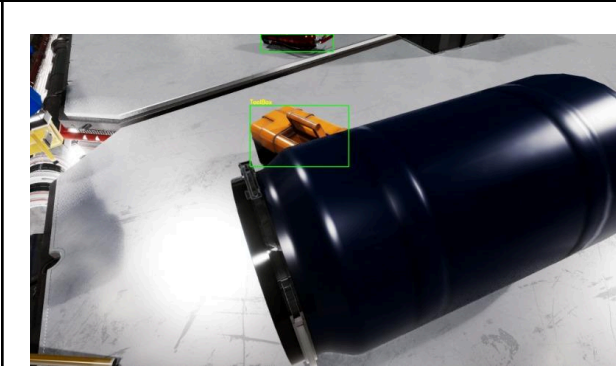
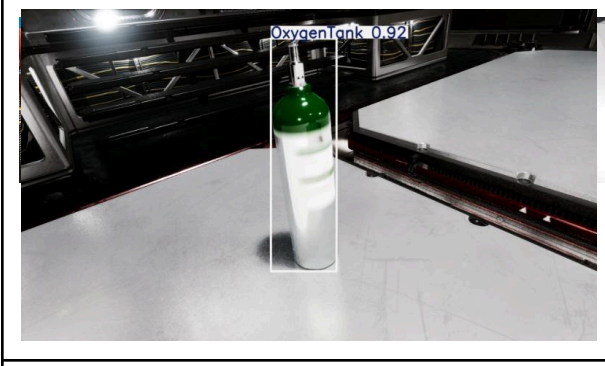
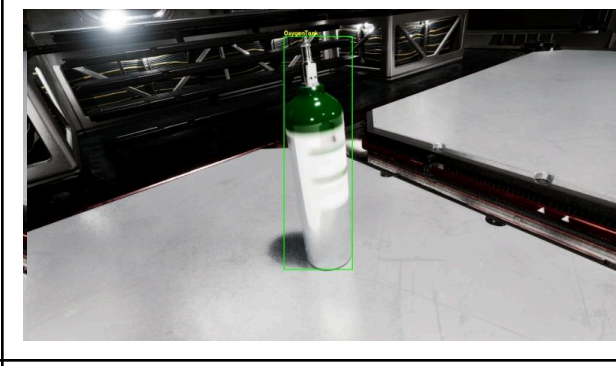
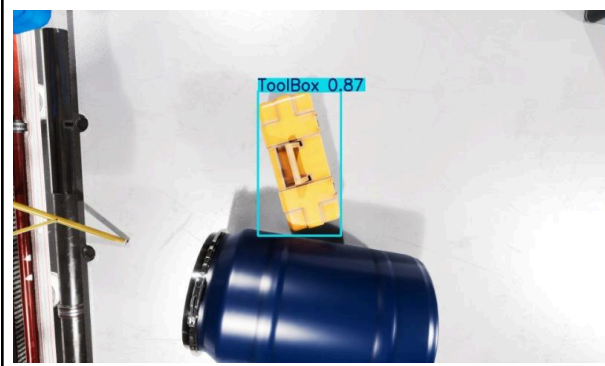
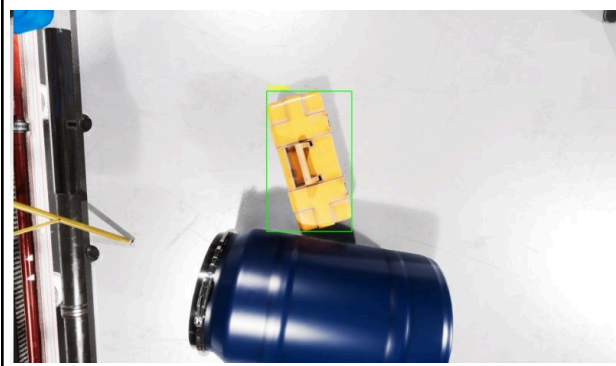
- Implemented YOLOv8 for multi-class object detection.
- Initial training with default hyperparameters to establish baseline performance.
- Applied data augmentation (rotation, brightness, and occlusion simulation) to improve generalization.

3. Model Fine-Tuning:

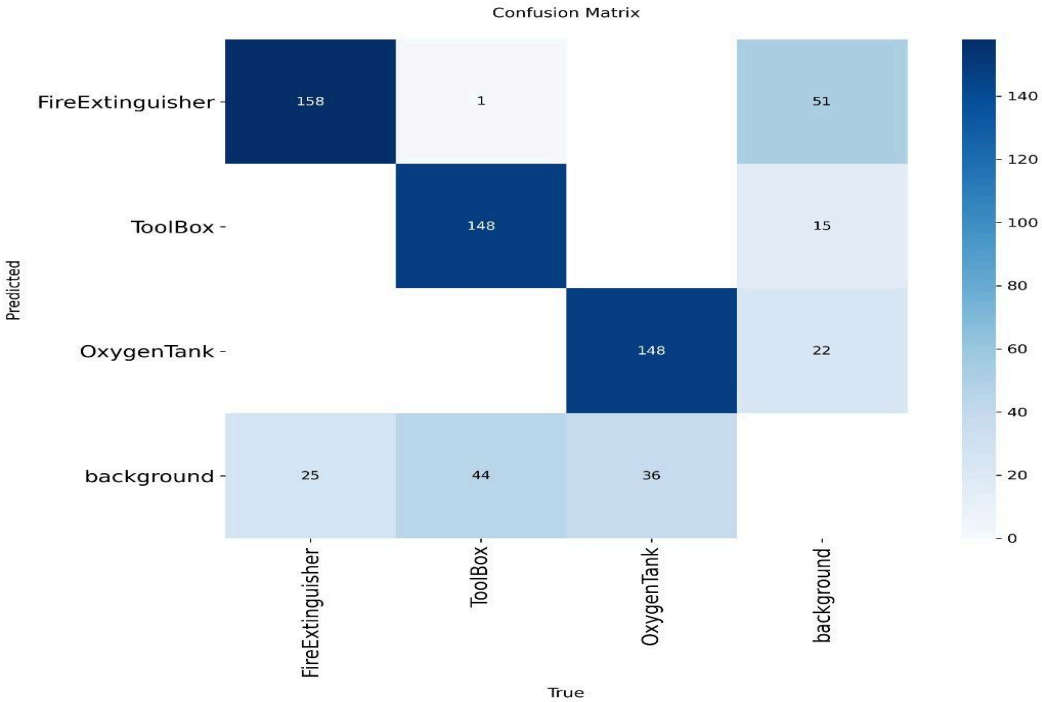
- Optimized epochs, learning rate, and batch size to enhance mAP scores and reduce inference time.
 - Introduced early stopping and validation checks to prevent overfitting.
 - Conducted failure case analysis and retrained with augmented samples for classes with lower recall (mainly Oxygen Tank).
-

Results

After training and fine-tuning our YOLOv8 object detection model on the Falcon synthetic space station dataset, we evaluated its performance using mAP scores, precision, recall, and confusion matrix analysis.

Model Detection Results	Original Results
	
	
	

Performance Metrics



METRICS

Model	YOLOv8
mAP	0.94359
Recall Percent	98
Precision Percent	91

Challenges

→ **Challenge 1**: Misclassification of Oxygen Tanks

Issue: Frequent occlusion of Oxygen Tanks led to reduced recall and false positives.

→ **Challenge 2**: Overfitting on Training Data

Issue: Model performed well on training but poorly on validation initially.

→ **Challenge 3**: Similar-looking Objects Under Poor Lighting

Issue: Fire Extinguishers and Oxygen Tanks looked similar under dim lighting, confusing the model.

→ **Challenge 4**: High Inference Latency on CPU

Issue: On low-resource devices, the model inference time exceeded 100ms.

→ **Challenge 5**: Training YOLOv8 requires GPU acceleration for optimal speed.

Issue: Some team members lacked access to high-end GPUs or only had limited GPU memory (e.g., <4GB VRAM), which caused slow training or memory errors.

→ **Challenge 6**: The synthetic dataset, training logs, and multiple checkpoints occupied a lot of disk space (often >3GB).

Issue: Limited SSD space on local machines led to I/O errors or storage warnings.

Solutions

- **For frequent occlusion of Oxygen Tanks** led to reduced recall and false positives, we augmented the dataset by artificially **adding more occluded versions of Oxygen Tanks** using image processing techniques.
 - For overfitting on training data we used **early stopping, dropout layers**, and **data augmentation** to improve generalization.
 - Applied **brightness and contrast augmentation** during training to simulate poor lighting.
 - High inference latency on CPU is handled by **model pruning and quantization** to reduce computational load.
 - The synthetic dataset, training logs, and checkpoints occupying a lot of disk space is checked by regularly **deleting unused checkpoints, intermediate outputs, and logs** from the **runs/** directory.
-

Conclusion

Our project successfully demonstrated how synthetic data and YOLOv8 can be combined to build a reliable object detection system for inaccessible environments like space stations.

- We trained and deployed a model capable of identifying Toolbox, Oxygen Tank, and Fire Extinguisher with high accuracy using entirely synthetic inputs.
- The system is integrated with a **Streamlit web app** that allows real-time webcam or image input for live detection.

Future Work

- Continuously update the model using new synthetic data generated in Falcon as scenarios evolve (e.g., new lighting or objects).
 - Extend the model to detect more object categories such as wires, helmets, and control panels.
 - Deploy the model as a microservice or integrate it into a larger robotic automation pipeline.
 - Implement cloud-based retraining pipelines for adaptive learning in space exploration scenarios.
-