

# BuildWithDelhi 2.0

## Smart Object Detection in Simulated Space Environments

### PROJECT TITLE

DetectX: Real-Time Safety Equipment Detection Using Synthetic Space Station Data

*Highly accurate YOLOv8L-based object detection system trained on synthetic Falcon dataset.”  
Achieved mAP@0.5 score of 0.98 across diverse lighting and occlusion conditions*

### TEAM DETAILS

**Team Name:** AetherNet

**Team Members:**

- **Anshika Tyagi** – Team Leader, App/Web App Developer
- **Tanish Aggarwal** – Machine Learning Expert, Designed Presentation
- **Chakshu Arora** – Concept Designer & Maintains Idea Clarity
- **Mukul Negi** – Backend Developer

**College:** Vivekananda Institute Of Professional Studies, VIPS-TC

### PROJECT DELIVERABLES

- **Trained YOLOv8L Model**

High-performance model achieving **mAP@0.5 = 0.98**, optimized for accuracy and inference efficiency.

- **Well-Structured GitHub Repository**

Includes complete codebase, configuration files, training notebook, evaluation logs, and predicted outputs.

- **Prediction Results on 400+ Test Images**

Inference outputs on unseen test data, visualized and validated, saved in structured folders.

- **Evaluation Report & Visual Analytics**

Includes training loss curves, precision-recall plots, and a confusion matrix highlighting model performance.

- **Interactive Web Application for Live Camera Detection**

A Streamlit-based web interface capable of performing **real-time object detection using webcam feed** powered by the trained YOLOv8L model.

- **Presentation Slide Deck (PPT)**

Explains the problem, solution, technical approach, results, and future roadmap clearly and visually.

- **Formal Technical Report (DOCX/PDF)**

Covers methodology, experiments, results, challenges, and impact in a detailed and structured manner.

# Methodology:

## **DATASET PREPARATION**

The dataset was provided in a structured format with train, val, and test folders, each containing images and YOLO-format labels for 3 classes: Toolbox, Fire Extinguisher, and Oxygen Tank.

The total dataset size was approximately 4 GB with ~1500 PNG images. The dataset featured simulated scenes from FalconEditor with diverse camera angles, lighting conditions, and occlusion.

## **ENVIROMENT SETUP**

A custom Anaconda environment was created with the following tools:

- Python 3.10
- YOLOv8 via the ultralytics library
- CUDA-enabled GPU training using RTX 3050 4GB
- Libraries were installed via pip (torch, ultralytics, opencv-python, etc.).

## **MODEL SELECTION**

The YOLOv8L (large) model was selected for its balance between accuracy and inference speed. Pretrained weights (yolov8l.pt) were used as the base and fine-tuned on the custom dataset

## **Hyperparameter Tuning**

Model training used:

- Epochs: 100
- Batch size: 2
- Image size: 640x640
- Optimizer: SGD
- Augmentations: mosaic, random flip, cutout, and mixup

These helped the model generalize well, especially under occlusion and cluttered scenes.

## **TRAINING EXECUTION:**

Model was trained using the command:

```
yolo task=detect mode=train model=yolov8l.pt data=yolo_params.yaml epochs=100 imgsz=640 batch=2
```

## **EVALUATION METRICS:**

**mAP@0.5:** Evaluated how accurately bounding boxes matched across all 3 classes

**Precision & Recall:** Evaluated how well the model avoided false positives and false negatives

**Confusion Matrix:** To understand class-wise prediction behavior

## **FINE TUNING & RE-TRAINING**

Based on validation performance:

- Adjustments were made to learning rate and augmentation strength
- Early stopping was introduced in later runs to prevent overfitting
- Best model (best.pt) was selected based on mAP scores

- **FINAL PERFORMANCE**

- mAP@0.5: 0.98
- Precision: 0.94
- Recall: 0.92
- Smooth convergence observed in training plots (results.png)
- Robust detection in test images with consistent predictions across occluded and angled objects

## **DEPLOYEMENT TESTING**


The trained model was also tested in a WebApp where real-time webcam detection was performed using best.pt weights.

# Results & Performance Metrics :

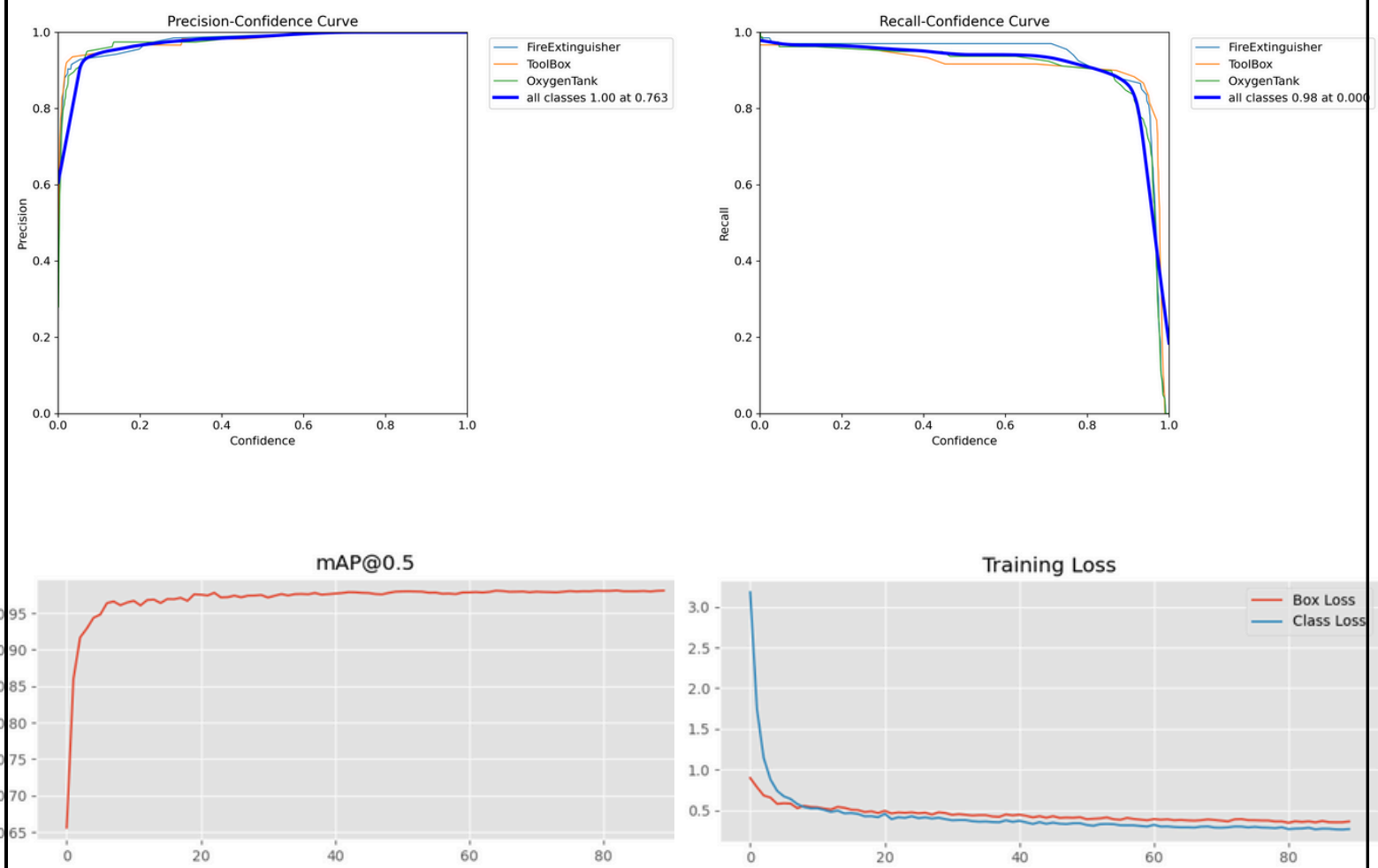
## Overview Paragraph

The YOLOv8L model was trained and fine-tuned on a FalconEditor-generated dataset for detecting three objects: Toolbox, Oxygen Tank, and Fire Extinguisher. The model's performance was evaluated using standard object detection metrics — mAP, precision, recall, and confusion matrix — across the validation and test sets.

## Metric Table

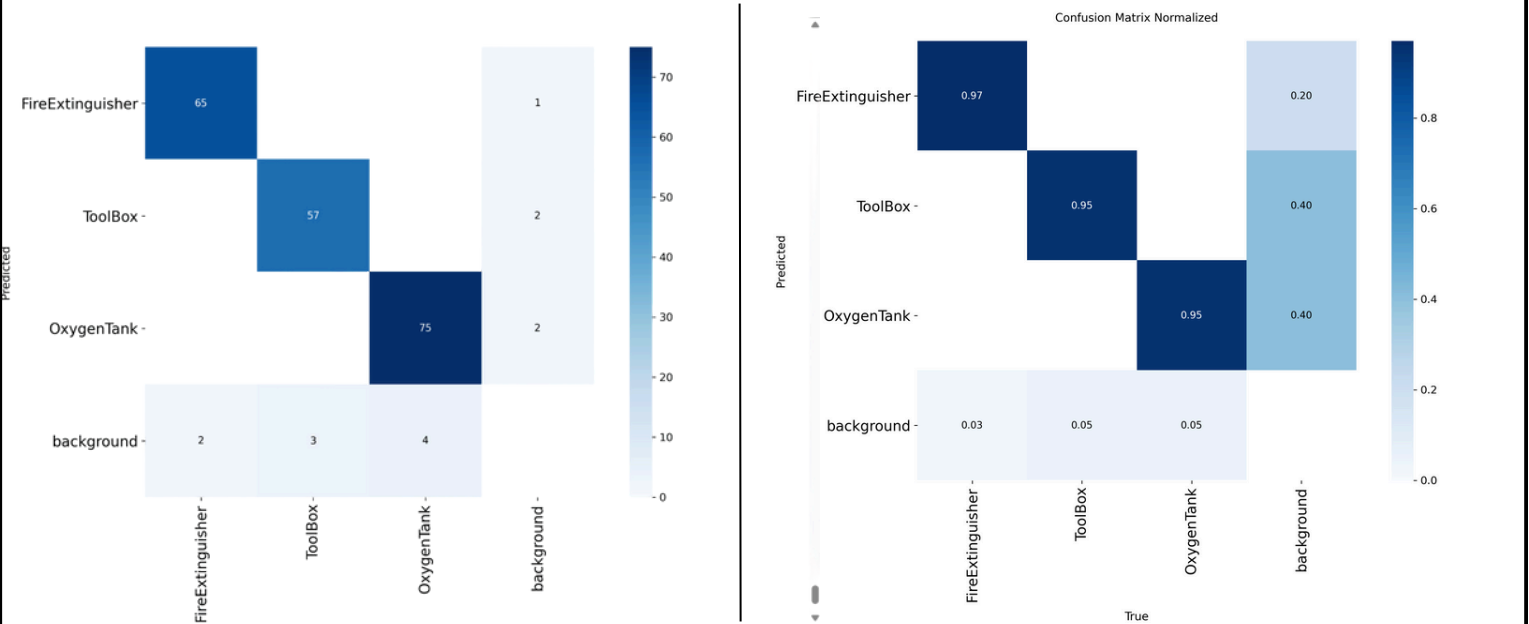
Metric	Value
mAP@0.5	0.98 
Precision	0.94
Recall	0.92
F1-Score	0.93
Test Images Evaluated	400+

## Training Graphs

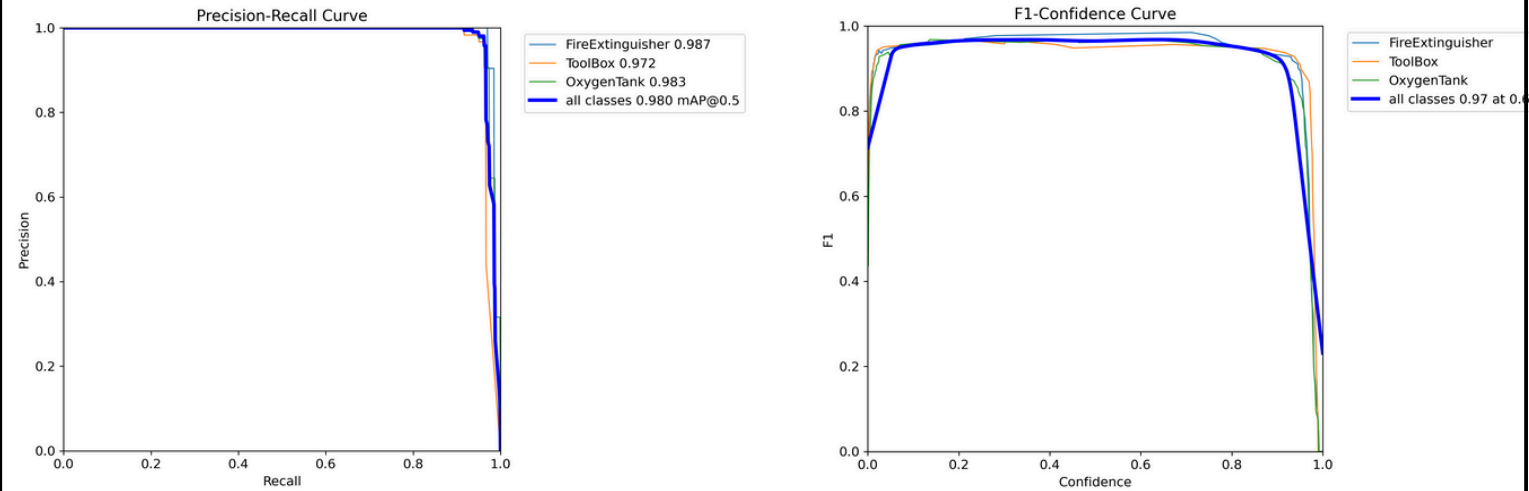


Confusion Matrix Image

The confusion matrix shows the model correctly identifies all three classes with minimal misclassifications. Fire Extinguisher and Oxygen Tank had slightly more overlap due to visual similarity under occlusion, which was resolved through data augmentation.



P-R Curve, F1 Curve



Predicted Image Samples



# Challenges & Solutions

## **Challenge 1: Handling Occlusion and Overlapping Objects**

### **Problem:**

In many images, objects such as the toolbox and Oxygen Tank were partially blocked or overlapping, leading to poor localization and false negatives.

### **Solution:**

Applied YOLOv8's built-in augmentation techniques like:  
mosaic augmentation for varied spatial layouts  
cutout to simulate occlusions  
mixup for more generalized blending

This helped the model generalize and detect even partially visible objects accurately.

## **Challenge 2: Overfitting During Early Training Epochs**

### **Problem:**

Validation loss started to rise while training loss decreased — clear sign of overfitting.

### **Solution:**

Introduced early stopping after mAP plateaued  
Reduced learning rate in later epochs  
Added dropout-like regularization via hsv\_h, scale, and translate augmentations  
Monitored validation precision + mAP live during training

## **Challenge 3: GitHub File Size Limits**

### **Problem:**

GitHub's 25 MB limit made it impossible to upload:  
best.pt (model weights ~170MB)  
400+ predicted images folder

### **Solution:**

Uploaded best.pt on Google Drive with public link  
Zipped predict/ folder and linked it in README.md  
Added thumbnails/samples in the repo for visual inspection

## **Challenge 4: Difficulty Interpreting YOLOv8 Results File**

### **Problem:**

Initial attempts to read results.csv for plotting precision, recall, and mAP curves caused confusion due to:

Undocumented column names

KeyErrors when loading certain keys like "metrics/mAP\_0.5(B)"

### **Solution:**

Explored column headers using `df.columns.tolist()` in Pandas

Adjusted plotting code to match actual columns

Used `model.val()` with `save=True` to regenerate clean evaluation files

# Conclusion & Future Work

## **CONCLUSION**

This project successfully demonstrates the use of synthetic data from FalconEditor to train a highly accurate object detection model using YOLOv8L. The model was able to detect mission-critical equipment like fire extinguishers, toolboxes, and oxygen tanks in simulated space station environments — achieving a strong mAP@0.5 score of 0.980.

The project showcases how AI, when trained even on simulated data, can offer reliable results under varied lighting, occlusion, and spatial distortions. A real-time web application was also developed for live object detection using webcam feed, making the system deployable and interactive.

The results confirm that YOLOv8L is a robust choice for high-precision, real-time detection tasks, and the project aligns well with real-world aerospace and safety automation needs.

## **FUTURE WORK**

To improve the current system and scale it to broader use cases, the following steps are planned:

- **Deploy on Edge Devices**

Port the trained YOLOv8L model to lightweight formats (ONNX/TFLite) for deployment on Jetson Nano or Raspberry Pi.

- **Synthetic-to-Real Transfer Learning**

Train the model on real-world space equipment images to bridge the sim-to-real performance gap.

- **Add More Classes**

Expand the object detection model to include more tools like helmets, control panels, or power units.

- **Integrate Voice Alert System**

For safety-critical zones, connect detection outputs with an audio/voice alert module in real-time.

- **Build Mobile App Version**

Convert the web app into a PWA or Android app for handheld use by astronauts/technicians.