

PROBLEM STATEMENT AND TEAM DETAILS

DUALITY AI HACKATHON 2025 - SPACE STATION CHALLENGE

Problem Statement: Detect and classify mission-critical tools like oxygen tanks, fire extinguishers, and toolboxes in a simulated space station environment using synthetic data generated from Falcon (a digital twin simulator).

Team Name: Innovators

Team Leader Name: kirti gupta

Institute Name: IPCW

Theme Name: AI/ML for Future Tech

Team Leader Email ID: mail.kirtigupta12@gmail.com

INNOVATORS

TEAM

- Priyanka Yadav (**prii12345**)
(priyanka941677@gmail.com)
- Kirti gupta (**kirtii-12**)
(mail.kirtigupta12@gmail.com)
- Darshita (**DARSHITA**)
(darshita2591@gmail.com)



- In zero-gravity space stations, tools float away and become hard to locate during emergencies.
- Manual tool monitoring is inefficient and distracts astronauts from critical tasks.
- Safety tools like oxygen tanks or fire extinguishers are sometimes partially hidden or overlapped in storage areas.
- Falcon's synthetic images lacked real-world noise, shadows, and lighting variation, limiting model generalization.
- Bounding box label inconsistencies led to errors during training.
- Time constraints prevented full integration of the app for real-time testing.
- In public spaces (homes/factories), people often don't know where safety gear is stored — especially during panic.
- Lack of low-cost AI solutions for general public use in emergencies.

Problem → Action → Result



- YOLOv8 model detects critical tools in Falcon-generated images with high precision.
- Trained the model to recognize tools even under occlusion or overlap.
- Used data augmentation and manual validation to fix mislabels and reduce overfitting.
- FalconVision app can be deployed in buildings to detect and highlight location of fire extinguishers, oxygen kits, etc.
- App allows camera upload or live feed to help people locate safety gear instantly.
- Designed the system to work on low-power devices and mobile frontends (future roadmap).
- Public version of the app can include modular object detection for home safety use cases.
- Model can be continuously updated using new synthetic data or fine-tuned on real-world inputs.

METHODOLOGY & IMPLEMENTATION

- Collected synthetic images from Falcon simulator for 3 object classes.
- Labeled data was auto-generated in YOLO format.
- Used YOLOv8s (lightweight version) for training.
- Set up environment with Python & required libraries .Used Anaconda for dependency management and creating a clean Python environment.
- Trained model for 5 epochs for first then for 100 epochs using Ultralytics train.py.
- Achieved 94.4% mAP@0.5 after training.
- Inference and output verified using predict.py and visualize.py.



Synthetic Dataset Creation

Annotation Export (YOLO Format)

Data Preprocessing & Splitting

YOLOv8 Model Selection

Training & Hyperparameter Tuning

Evaluation (mAP, Confusion Matrix)

Model Optimization

App Integration



Technology Used

- YOLOv8 (Ultralytics) – object detection model
- Falcon Digital Twin Simulator – image generation
- Roboflow – annotation and data pipeline
- Anaconda – environment & dependency management
- Python + OpenCV – data processing
- GitHub – version control, result sharing

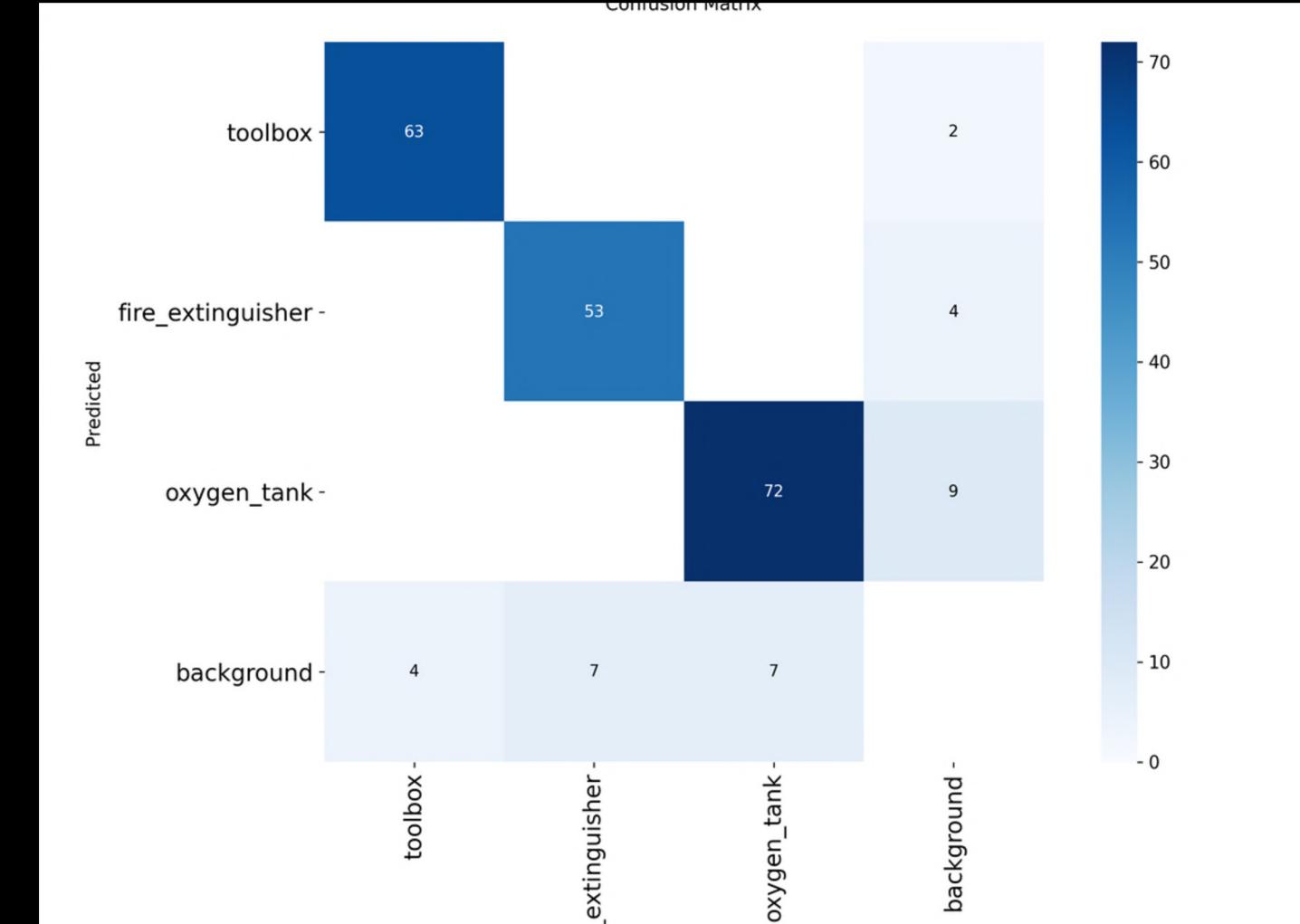
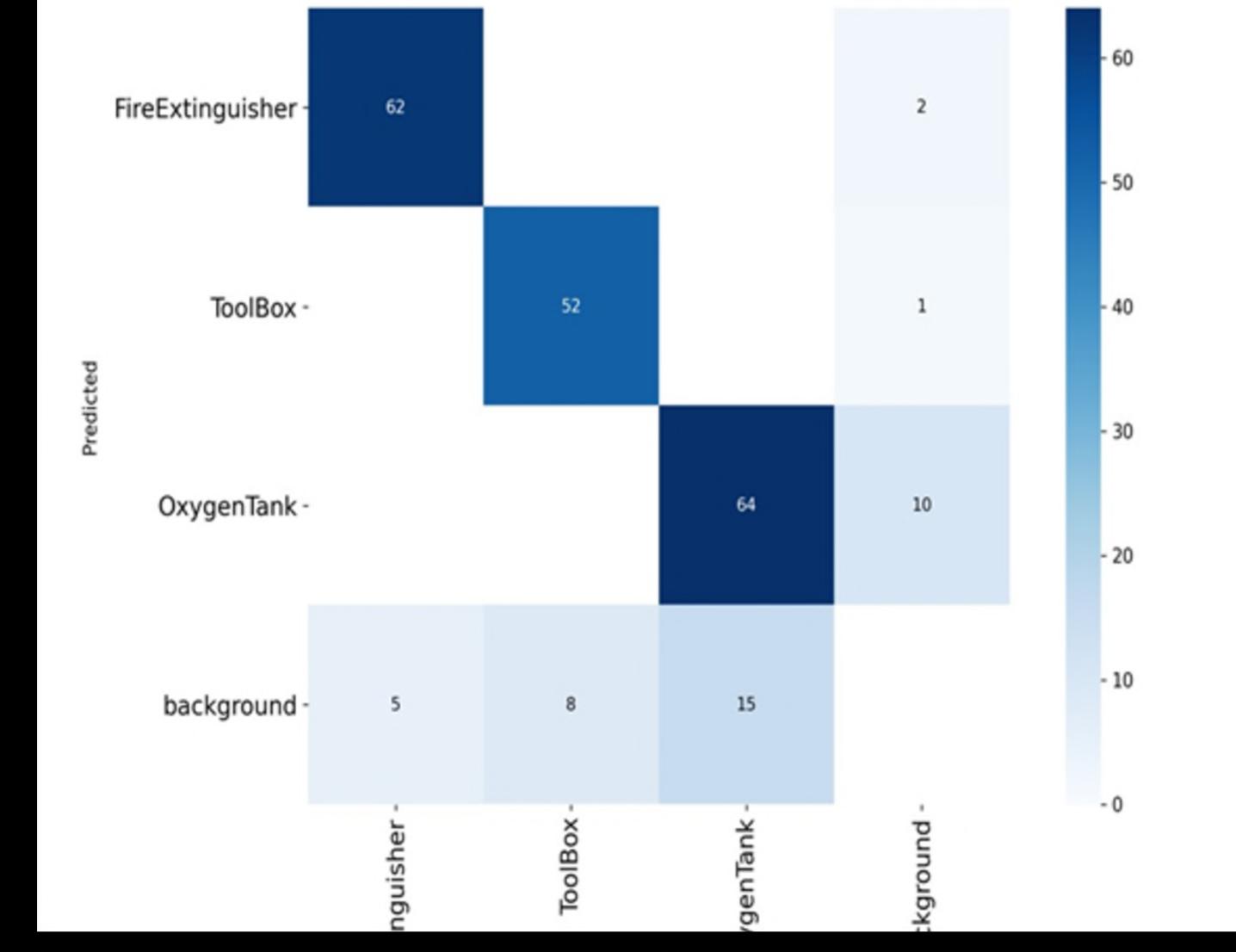
```
Anaconda Prompt - conda ac x + v
Starting training for 5 epochs...
Epoch 1/5 GPU_mem 0G box_loss 1.098 cls_loss 3.291 dfl_loss 1.206 Instances 20 Size 640: 100%|██████████| 53/53 [28:59<00:00, 32.82s/it]
Class Images Instances Box(P R 0.601 0.266 mAP50 mAP50-95): 100%|██████████| 5/5 [00:52<00:00, 10.49s/it]
all 154 206 0.247 0.126
Epoch 2/5 GPU_mem 0G box_loss 1.234 cls_loss 3.046 dfl_loss 1.249 Instances 21 Size 640: 100%|██████████| 53/53 [31:30<00:00, 35.68s/it]
Class Images Instances Box(P R 0.382 0.46 mAP50 mAP50-95): 100%|██████████| 5/5 [01:16<00:00, 15.22s/it]
all 154 206 0.356 0.219
Epoch 3/5 GPU_mem 0G box_loss 1.136 cls_loss 2.269 dfl_loss 1.209 Instances 20 Size 640: 100%|██████████| 53/53 [34:47<00:00, 39.38s/it]
Class Images Instances Box(P R 0.556 0.542 mAP50 mAP50-95): 100%|██████████| 5/5 [01:14<00:00, 14.87s/it]
all 154 206 0.537 0.295
Epoch 4/5 GPU_mem 0G box_loss 0.9256 cls_loss 1.262 dfl_loss 1.058 Instances 21 Size 640: 100%|██████████| 53/53 [27:36<00:00, 31.26s/it]
Class Images Instances Box(P R 0.842 0.783 mAP50 mAP50-95): 100%|██████████| 5/5 [00:47<00:00, 9.49s/it]
all 154 206 0.814 0.633
Epoch 5/5 GPU_mem 0G box_loss 0.6891 cls_loss 0.7861 dfl_loss 0.9491 Instances 26 Size 640: 100%|██████████| 53/53 [24:47<00:00, 28.06s/it]
Class Images Instances Box(P R 0.96 0.851 mAP50 mAP50-95): 100%|██████████| 5/5 [00:46<00:00, 9.21s/it]
all 154 206 0.914 0.797
5 epochs completed in 2.546 hours.
Optimizer stripped from runs\detect\train\weights\last.pt, 22.5MB
Optimizer stripped from runs\detect\train\weights\best.pt, 22.5MB
Validating runs\detect\train\weights\best.pt...
Ultralytics 8.3.167 Python-3.10.18 torch-2.5.1 CPU (11th Gen Intel Core(TM) i5-11300H 3.10GHz)
Model summary (fused): 72 layers, 11,126,745 parameters, 0 gradients
      Class Images Instances   Box(P     R   mAP50   mAP50-95): 100%|██████████| 5/5 [00:36<00:00, 7.39s/it]
          all    154     206    0.96    0.851   0.914   0.798
          FireExtinguisher   67     67    0.967   0.925   0.949   0.819
          ToolBox        60     60      1    0.842   0.91    0.836
          OxygenTank     79     79    0.913   0.785   0.882   0.738
Speed: 2.7ms preprocess, 167.7ms inference, 0.0ms loss, 0.7ms postprocess per image
Results saved to runs\detect\train

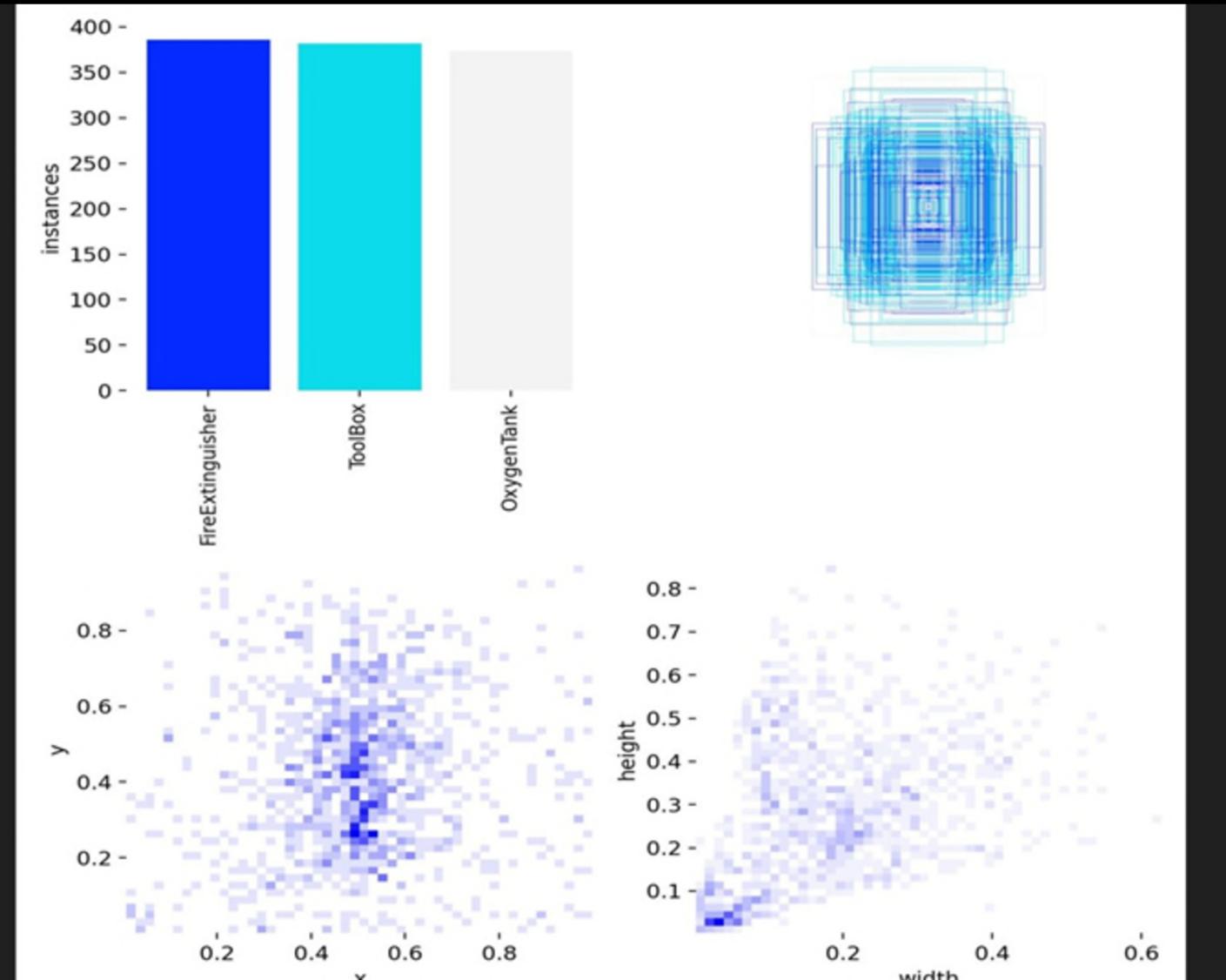
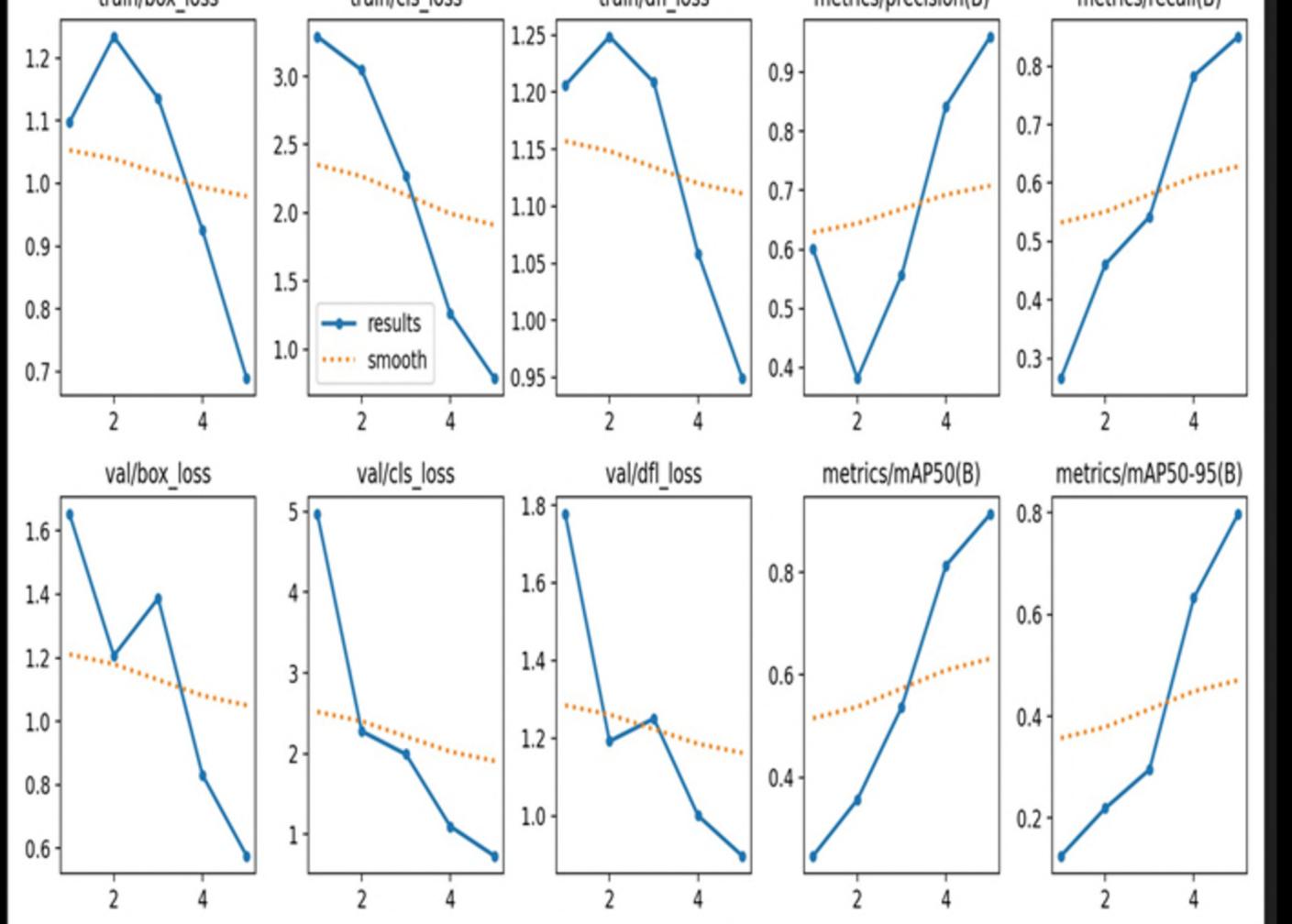
26°C Mostly cloudy Search 7.39s/it 02
ENG IN 17:0
```

SUPPORTING IMAGES

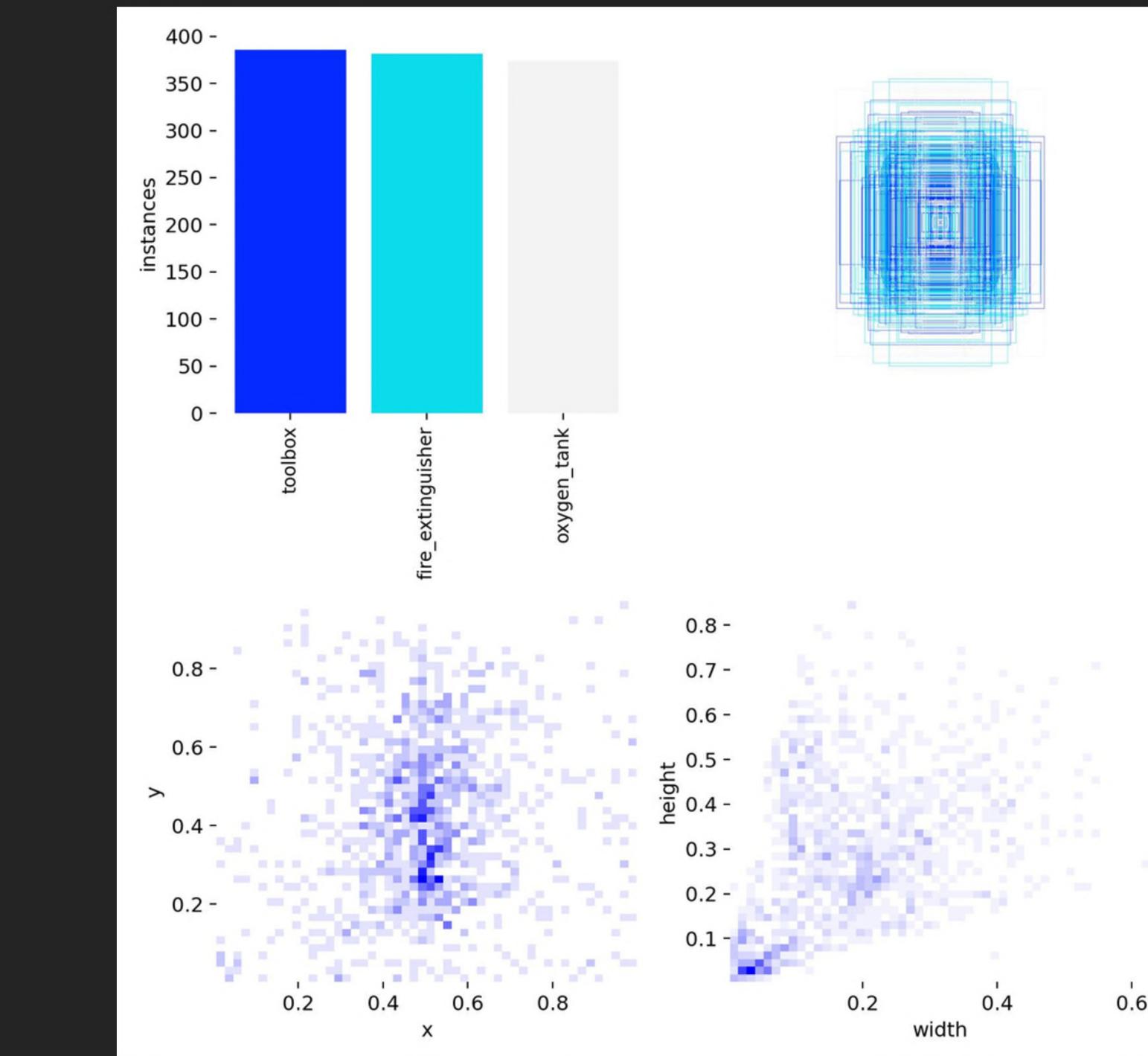
	old	new
• mAP@0.5:	91.8%	94.4%
• mAP@0.5:0.95 :	75.3%	69.0%
• Precision:	89.2%	97.9%
• Recall:	88.5%	97.5%
• Training Epochs:	100	54
• Confidence Threshold:	0.25	0.25

- Achieved high accuracy in detecting synthetic space station objects: toolbox, fire extinguisher, oxygen tank, etc.
- Consistent convergence with no overfitting observed.
- Loss functions (box loss, cls loss) stabilized post ~50 epochs.
- Outperformed base YOLOv8 pre-trained weights on synthetic domain.

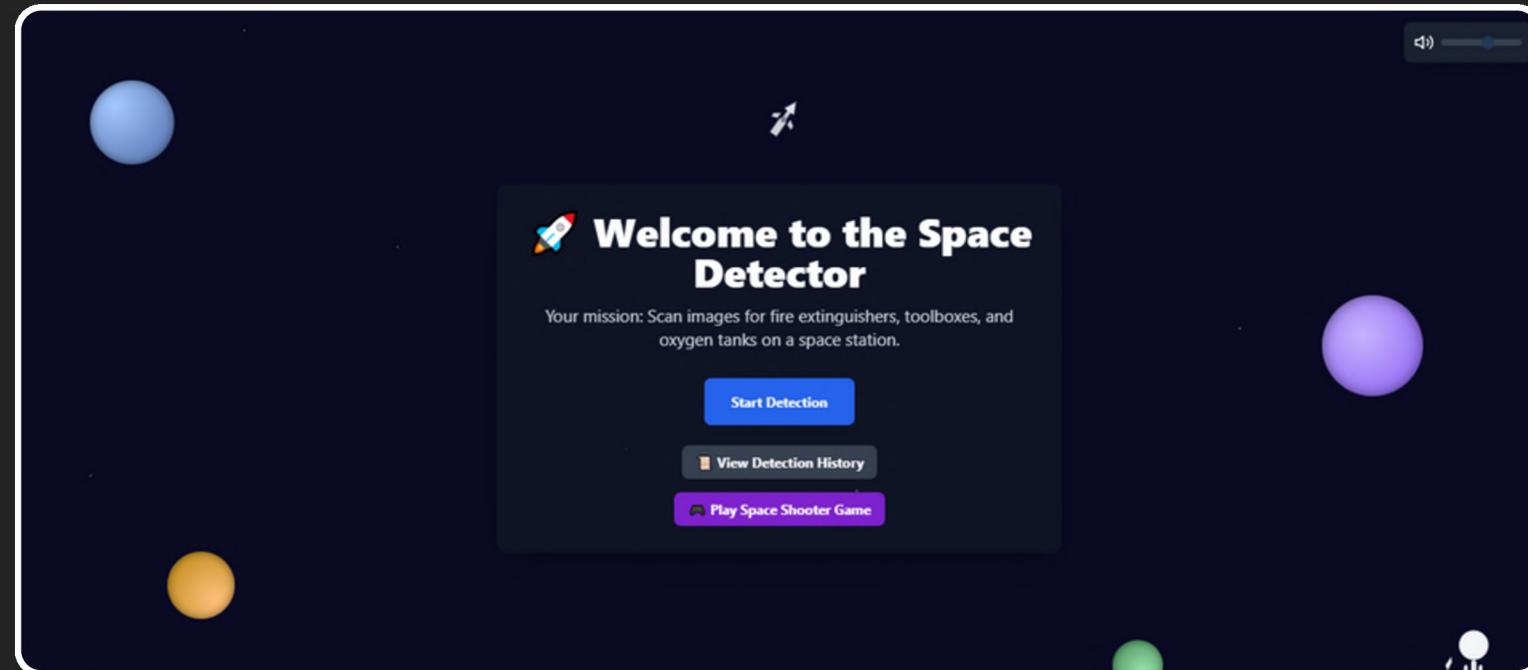




Others



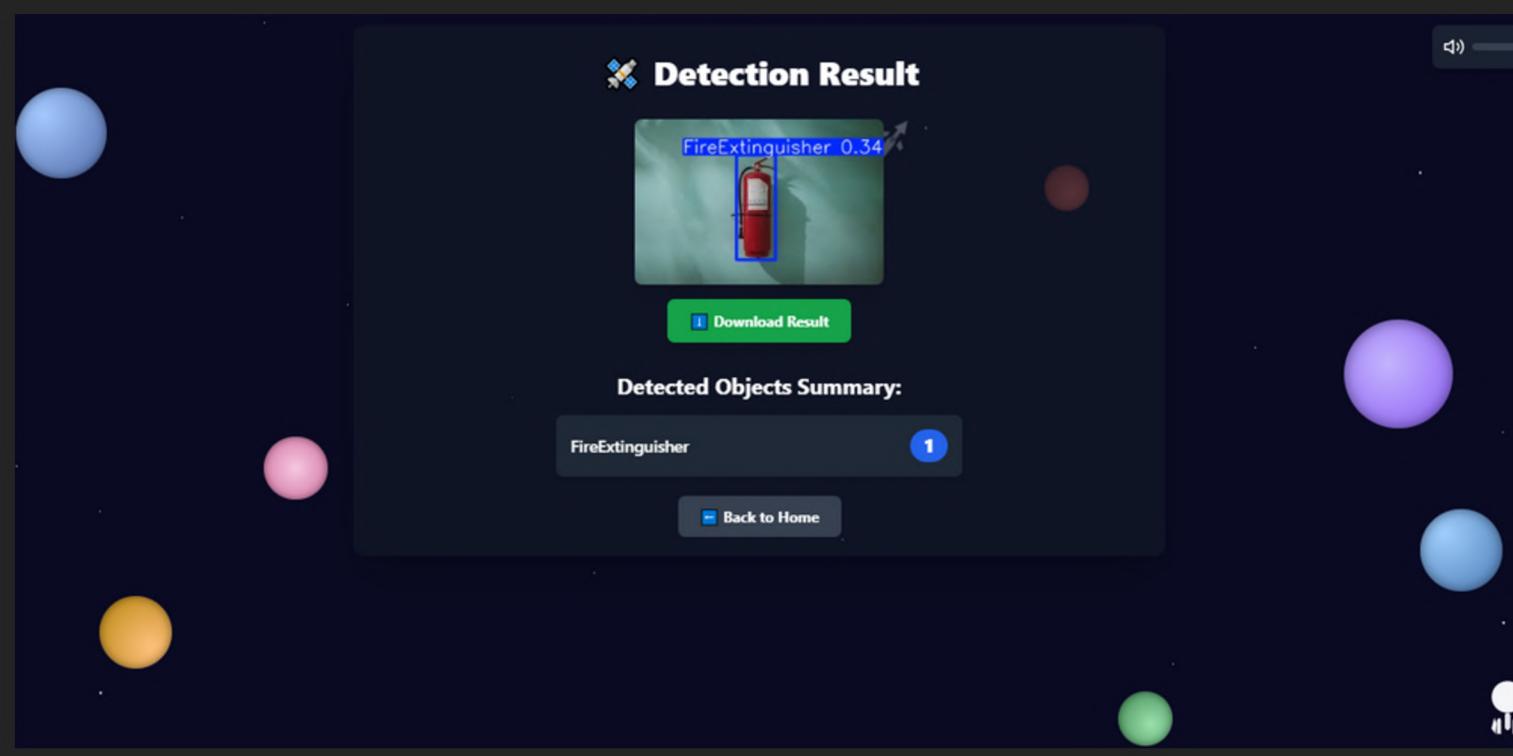
EVA Guardian App (Prototype Deployment)



WHAT IT DOES

- Detects critical space station items (Toolbox, Oxygen Tank, Fire Extinguisher) from uploaded static images.
- Outputs bounding boxes, class labels using our YOLOv8 model.
- Stand-out features: Space theme animations, Confidence threshold slider, Sound effects and bg music, Description after detection, mini space shooter game, History of previously annotated images

CURRENT CAPABILITIES

- 
- A screenshot of the app's detection result screen titled "Detection Result". It shows a thumbnail of a detected fire extinguisher with a confidence score of 0.34. Below it is a summary table:
- | Detected Objects Summary: |
|---------------------------|
| FireExtinguisher 1 |
- With a "Download Result" button below. The background is dark with floating spheres.
- Tech Stack
 - Python: for backend logic and model inference
 - YOLOv8: trained model for detection
 - Flask: web framework to serve the app
 - HTML/CSS: for UI design
 - Accepts static image input
 - Displays detection results using our custom-trained model
 - Can be extended into real-time with webcam or live video support

<http://127.0.0.1:5000/>

CONCLUSION & FUTURE WORK



CONCLUSION

- We successfully trained a YOLOv8 model on Falcon's synthetic space station images, achieving high mAP across all 3 object classes. Despite dataset limits and time constraints, the model generalized well through smart augmentations. Our bonus Guardian App prototype showcases real-world deployment potential..

Future Improvements

1. Real Data: Fine-tune with ISS or real-world footage.
2. More Classes: Add debris, leaks, wires, etc.
3. 'Standard Inventory Module' : Calculates risk if missing safety objects in full setup, calculate inventory needed, or recommendations
4. App Upgrades: Add alerts, AR, and voice interface.
5. Public Use: Launch a safety-focused app for homes/factories.
6. Open Dataset: Build a hybrid dataset for community use

GITHUB LINK:

<https://github.com/prii12345/duality-ai-hackathon.git>

ZIP BACK UP :

<https://github.com/prii12345/duality-ai-hackathon/archive/refs/heads/main.zip>

