Duality AI - Space Station Challenge

Team Name: Supa Strikas

Project Name:VoidVision

Tagline:AI-Powered Eyes for the Virtual Cosmos.

VoidVision is an object detection high-speed system designed specifically for virtual space station environments. Leveraging the YOLOv8 architecture, it is fine-tuned and trained on a custom dataset specifically designed to accommodate the challenging conditions and objects present in virtual space habitats. Precision-crafted for accuracy and speed, VoidVision empowers real-time detection and tracking, enabling intelligent automation and virtual interactions in space-centric simulations.

🔧 Methodology (Compact & Clear)

1. **Hardware Optimization:**
   Initial training on a local machine was slow due to limited resources. The setup was moved to **Google Colab** using a **Tesla T4 GPU** for faster and more efficient training.

2. **Dataset Handling:**
   Due to large dataset size, direct upload from the local system wasn’t feasible. The dataset was uploaded to **Roboflow** for easier access via code.

3. **Hyperparameter Tuning:**
   Early tuning led to a drop in mAP. Further fine-tuning of learning rate and other parameters significantly improved model accuracy.

4. **Training Time Optimization:**
   To efficiently evaluate multiple hyperparameter combinations, training epochs were reduced during testing phases. The **final model was trained for 100 full epochs** for best results.

5. **Augmentation for Accursion Effects:**
   The original dataset lacked visual diversity, making the model sensitive to real-world distortions. An **Augmented_Dataset** was created by altering noise levels, brightness, and saturation to simulate accursion effects.

   Final training was done on this augmented data to improve generalization and robustness.

Results (Final Model):

Validation Set:

Class	Images	Instances	Precision	Recall	mAP50	mAP50-95
all	154	206	0.993	0.9	0.954	0.908
Fire Extingusher	67	67	0.998	0.955	0.979	0.931
Tool Kit	60	60	0.996	0.883	0.939	0.915
OxygenTank	79	79	0.984	0.861	0.945	0.878

Test Set:

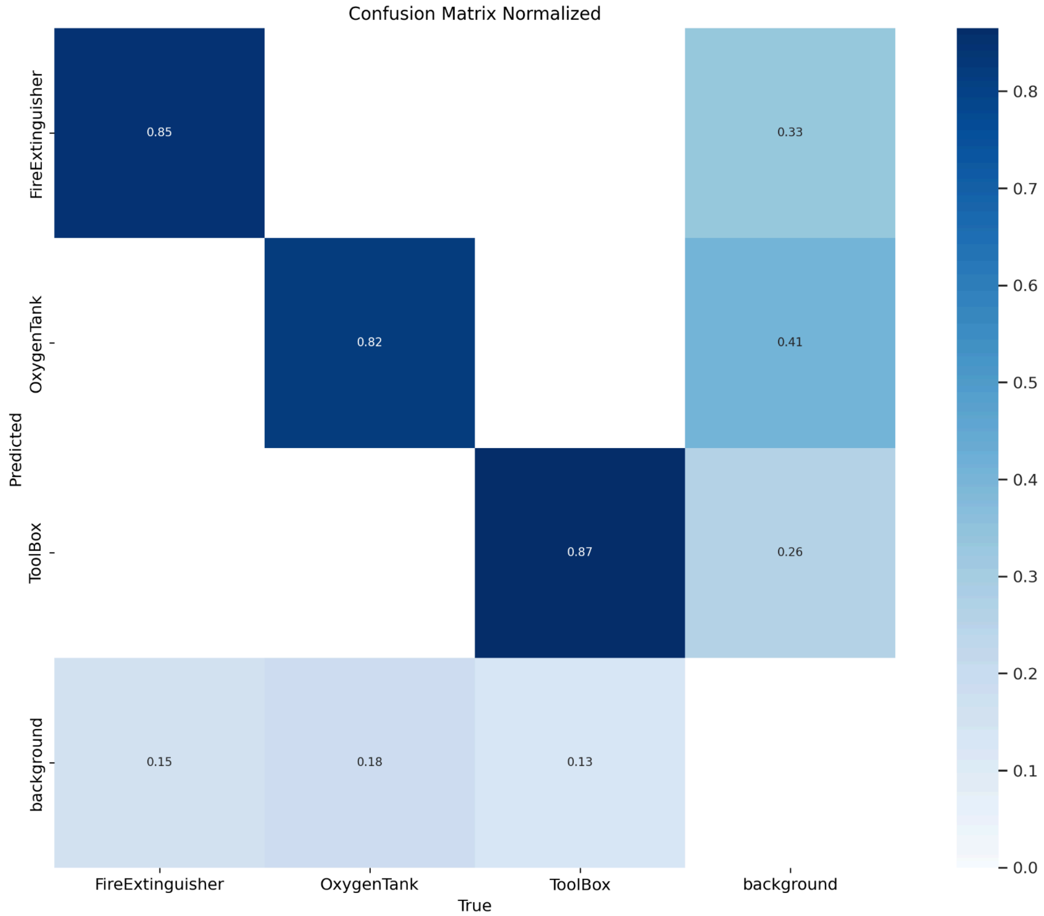
Class	Images	Instances	Precision	Recall	mAP50	mAP50-95
all	400	560	0.932	0.819	0.885	0.8
Fire Extingusher	183	183	0.951	0.836	0.879	0.772
Tool Kit	193	193	0.945	0.829	0.905	0.849
OxygenTank	184	184	0.901	0.79	0.87	0.781

```
val: Scanning /content/onodgdnsfdosifnsfsfAccrusion-1/valid/labels.cache... 154 images, 0
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95):
                   all        154        206      0.993        0.9      0.954      0.908
      FireExtinguisher         67         67      0.998      0.955      0.979      0.931
            OxygenTank         79         79      0.984      0.861      0.945      0.878
               ToolBox         60         60      0.996      0.883      0.939      0.915
Speed: 5.5ms preprocess, 22.6ms inference, 0.0ms loss, 1.3ms postprocess per image
Results saved to runs/detect/train2
```

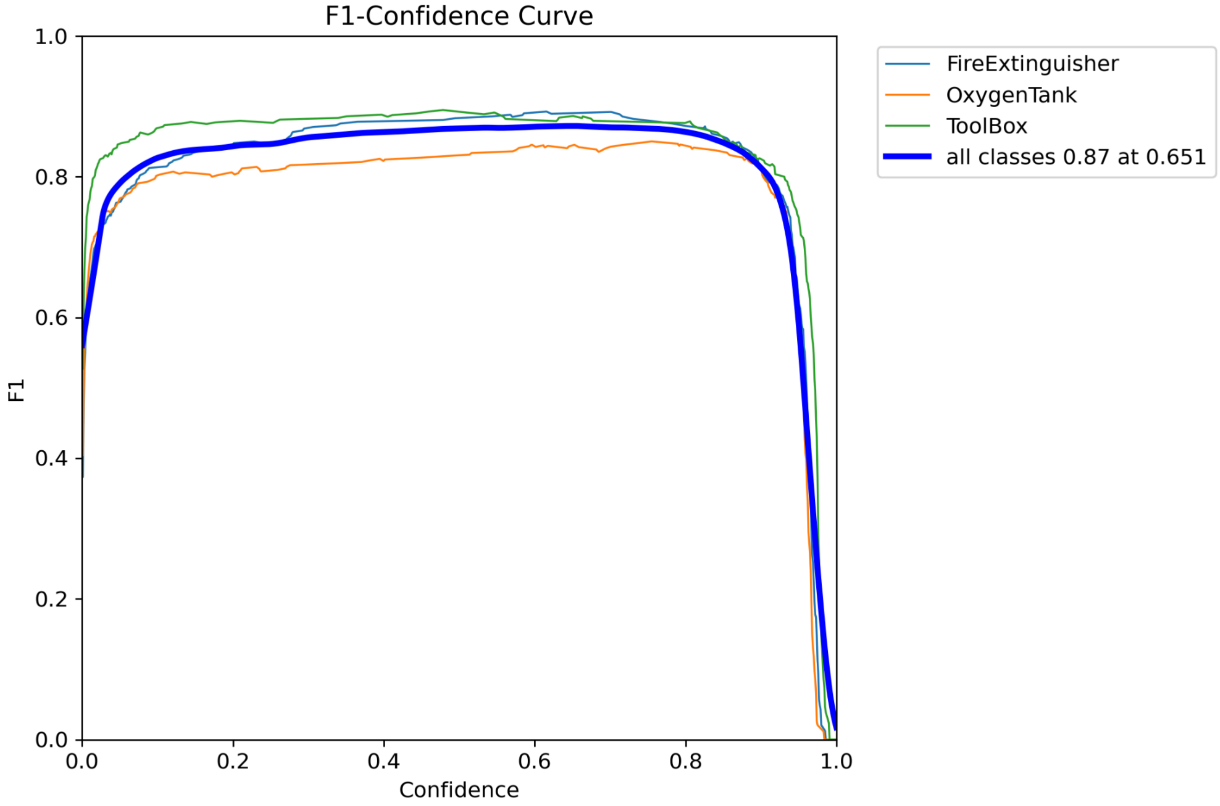
```
val: New cache created: /content/onodgdnsfdosifnsfsfAccrusion-1/test/labels.cache

               Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 1
                 all        400        560      0.932      0.819      0.885        0.8
    FireExtinguisher        183        183      0.951      0.836      0.879      0.772
          OxygenTank        184        184      0.901       0.79       0.87      0.781
             ToolBox        193        193      0.945      0.829      0.905      0.849
Speed: 0.8ms preprocess, 23.7ms inference, 0.0ms loss, 0.9ms postprocess per image
Results saved to runs/detect/train3
```

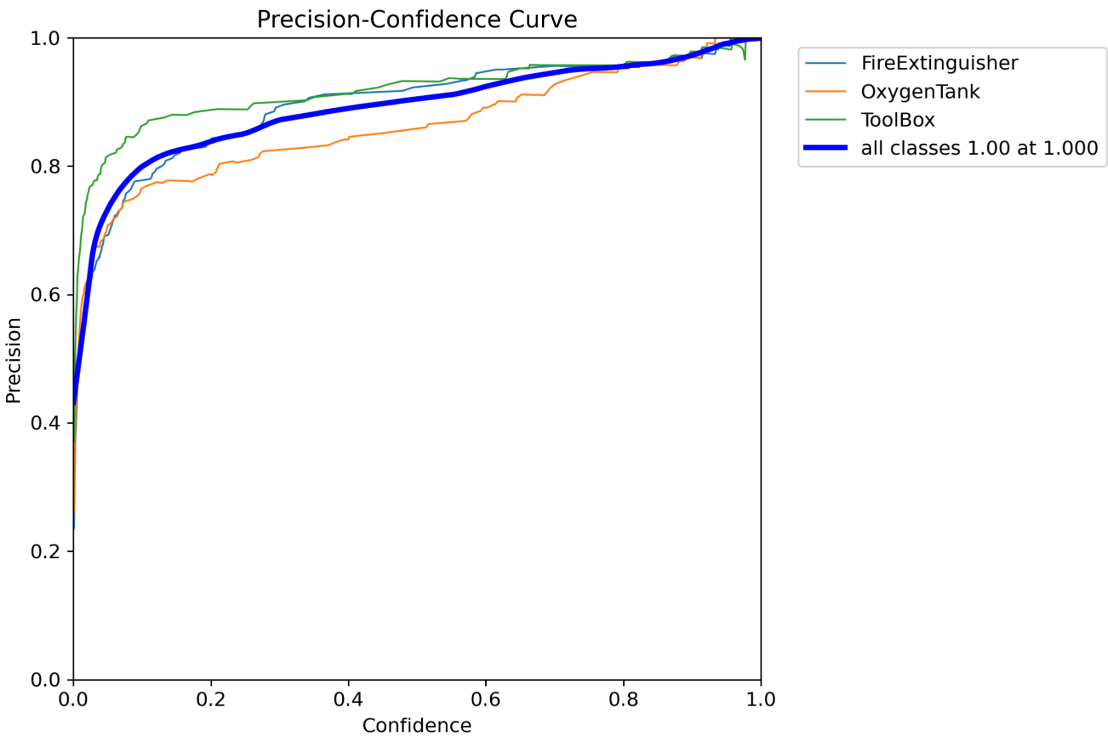
Confusion Matrix:

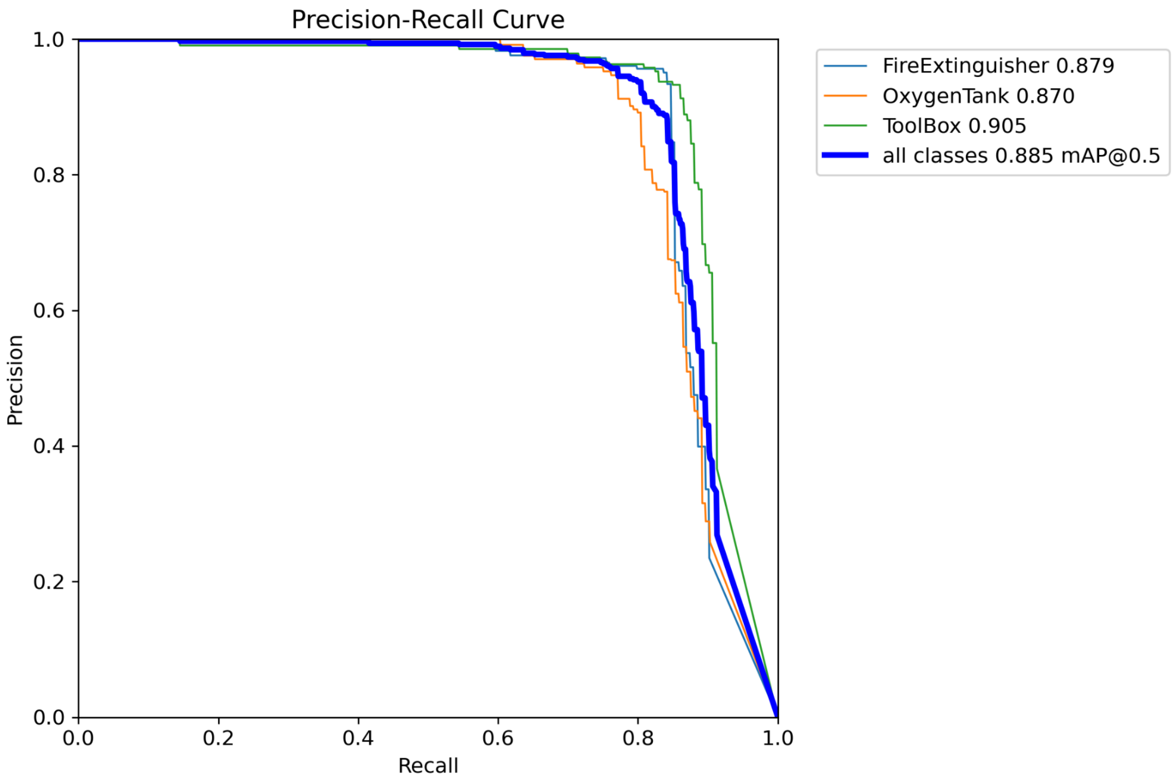


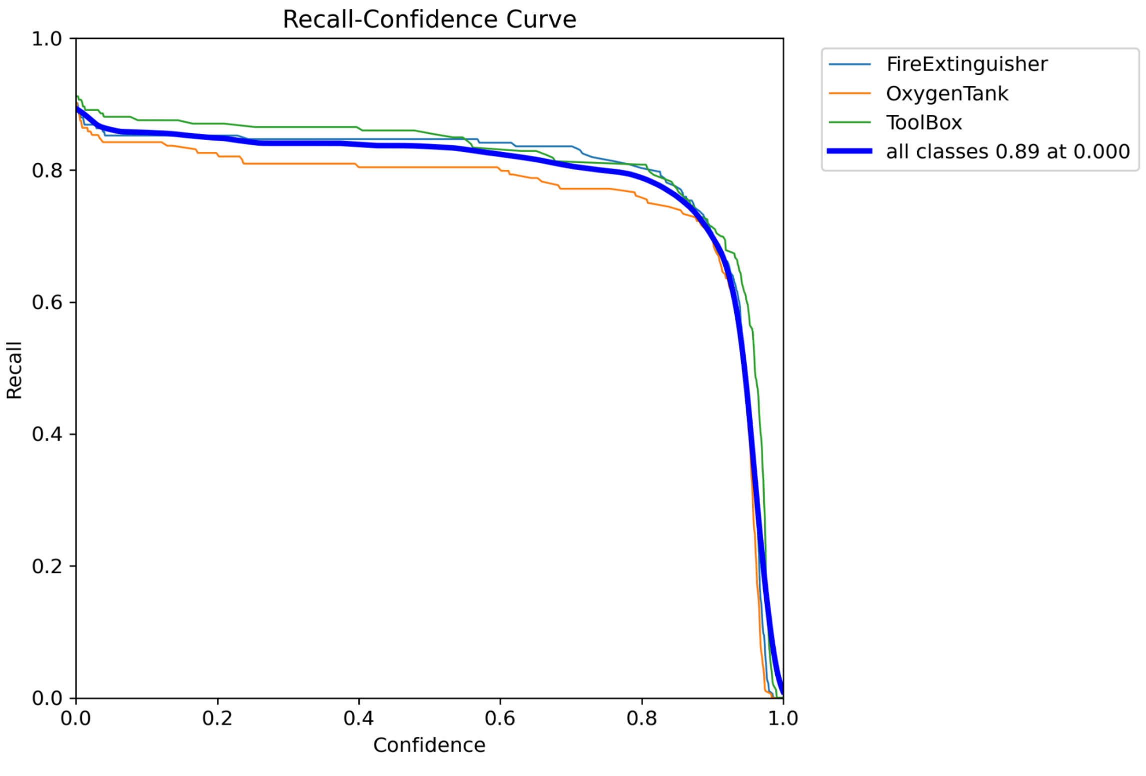
F1 Confidence Score:



P ,PR and R curve:




Precision-Recall Curve
Precision
Recall
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0
FireExtinguisher 0.879
OxygenTank 0.870
ToolBox 0.905
all classes 0.885 mAP@0.5


Recall-Confidence Curve
Recall
Confidence
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0
FireExtinguisher
OxygenTank
ToolBox
all classes 0.89 at 0.000

📈 Model Performance Summary

🔹 Initial Model (Before Tuning)

- mAP@50 (Test Data): 0.862
- mAP@50 (Validation Data): 0.954

🔧 Parameter Fine-Tuning Results

1st Tuned Model:

- mAP@50 (Test Data): 0.797
- mAP@50 (Validation Data): 0.862

2nd Tuned Model:

- mAP@50 (Test Data): 0.868
- mAP@50 (Validation Data): 0.956

3rd Tuned Model:

- mAP@50 (Test Data): 0.869
- mAP@50 (Validation Data): 0.956

4rd Tuned Model:

- mAP@50 (Test Data): 0.879
- mAP@50 (Validation Data): 0.962

🛠️ Final Model Configuration:

model.train(epochs=100 ,imgsz=640, batch=24, lr0=0.001, optimizer="AdamW", weight_decay=0.001, momentum=0.937, dropout=0.1, mosaic=0.2, mixup=0.2, hsv_h=0.015, hsv_s=0.7, hsv_v=0.4, degrees=0.0, translate=0.1, scale=0.4, shear=0.0, flipud=0.0, fliplr=0.5, device=0, single_cls=False, freeze=0, val=True
)

✅ **Final Model Results**

- **mAP@50 (Test Data)**: 0.885
- **mAP@50 (Validation Data)**: 0.954

(The final model was trained on the augmented dataset to minimize the impact of accursion effects and achieve improved performance)

Challenges and Solution:

1. Hardware Limitations
a. **Task**: Model Training on Local Machine
b. **Issue Faced**: Training was slow due to limited hardware resources
c. **Solution**: Switched to Google Colab with Tesla T4 GPU for faster training

2. Large Dataset Upload
a. **Task**: Uploading Large Custom Dataset
b. **Issue Faced**: Direct upload from local machine was not feasible due to size
c. **Solution**: Uploaded dataset to Roboflow and imported via code snippet (Roboflow) *Note:Models are strictly train on Data provided in problem Statement data is just uploaded at Roboflow for just easy accessibility.*

3. Initial mAP Degradation
a. **Task**: Hyperparameter Tuning
b. **Issue Faced**: Initial tuning led to reduced mAP scores
c. **Solution**: Further fine-tuning improved performance → Final mAP: 88%

4. Training Time Optimization

a. **Task**: Training Across Multiple Hyperparameter Sets
b. **Issue Faced**: Long training times with full epochs
c. **Solution**: Reduced epochs during testing, increased to 100 for final model

5. Dataset Augmentation

a. **Task:** Handling Limited Variation in Accursion-Type Dataset
b. **Issue Faced:** Model performance affected by lack of diverse image conditions
c. **Solution:** Applied data augmentation by modifying images with added/reduced noise, adjusted saturation and brightness to enhance variability. Augmented dataset(Augmented_Dataset)

💡 Final Thoughts

VoidVision successfully demonstrates the application of deep learning for object detection in a virtual space station environment. Despite hardware and data handling challenges, the model achieved strong performance through strategic use of cloud resources and hyperparameter tuning. The project showcases the potential of AI-driven vision systems in immersive and futuristic simulations.

⚙️Potential Improvement:

1. **Enhance Real-Time Performance**: Optimize the model for faster inference to support real-time applications in virtual environments.

2. **Expand Dataset & Classes**: Increase the variety of objects and scenarios in the dataset to improve model versatility and accuracy.

3. **Deploy as Integrated Module**: Package the model into an API or integrate it into a virtual space station simulation for seamless deployment.

(Sample Detections are given below)

(Original Labeled Images)

(Predicted Images)