Duality AI – Space Station Hackathon

Final Report

Team Name – OINK BOINK

Team Members:

(i) Namit Rana
(ii) Khushboo Kataria

Project Name – BoinkVision

Tagline : Object detection at light speed.

Project Summary :

This project focuses on developing a highly accurate object detection system using the YOLOv8 deep learning model to identify critical safety equipment—Fire Extinguishers, Tool Boxes, and Oxygen Tanks—in images. Utilizing a custom-curated dataset, the model was fine-tuned with advanced augmentations and hyperparameter tuning to achieve robust performance. The trained model demonstrates strong precision and recall across all classes, enabling reliable detection in real-world environments. This system aims to enhance safety monitoring and automation by providing fast, accurate detection of essential safety gear, with potential applications in industrial, commercial, and emergency response settings.

Methodology

1. Dataset Preparation

The first step involved mounting Google Drive in the Colab environment to access the dataset, which was provided as a zipped archive. We verified the contents and extracted the dataset to a local temporary directory, preserving the folder structure for training, validation, and testing images along with their annotations.

2. Dataset Configuration

A custom YAML configuration file was created to specify the dataset's root path, subdirectories for train and validation images, number of classes, and their names. This file enabled the YOLOv8 training pipeline to correctly access and label the data.

3. Environment Setup and Dependencies

We installed the ultralytics Python package, which contains the official YOLOv8 implementation, to facilitate training, validation, and inference tasks.

4. Model Initialization

The YOLOv8x model was loaded using a pretrained checkpoint (yolov8x.pt). This variant provides the highest accuracy among YOLOv8 models, suitable for detailed object detection tasks.

5. Training Procedure

The model was fine-tuned for 100 epochs using images resized to 896x896 pixels. The training parameters included:

- Batch size of 8, balancing GPU memory and training speed
- AdamW optimizer with a low learning rate (0.0001) to stabilize training
- Regularization techniques such as weight decay and dropout to reduce overfitting
- Label smoothing to mitigate noisy label effects
- Data augmentations including mosaic, mixup, and HSV color shifts to improve model robustness
- Early stopping with patience set to 15 epochs based on validation loss

6. Validation and Evaluation

Model performance was monitored during training on a validation set. After training, evaluation metrics including precision, recall, mAP50, and mAP50-95 were computed on a separate test set to assess generalization.

7. Inference and Visualization

Inference was performed on selected test images, with bounding boxes and class labels visualized to qualitatively verify the model's detection accuracy.

8. Model Export and Deployment

The best-performing model weights were saved and downloaded for future deployment or further research.

Test Results & Evaluation Metrics

After training the YOLOv8x model, we evaluated its performance on the test dataset. The key metrics and statistics are summarized below:

- **Fitness (0.9549):**
  An overall performance score combining multiple evaluation metrics to provide a single summary measure of model quality.
- **Precision (0.9892):**
  The accuracy of the model in correctly identifying positive detections, indicating very few false positives.
- **Recall (0.9567):**
  The model's ability to find all relevant objects in the images, showing very few false negatives.
- **mAP50 (0.9770):**
  Mean Average Precision at an Intersection over Union (IoU) threshold of 0.5, reflecting the model's detection accuracy at a moderate overlap criterion.
- **mAP50-95 (0.9524):**
  Mean Average Precision averaged over IoU thresholds from 0.5 to 0.95, providing a more rigorous and comprehensive measure of detection accuracy.
- **Speed Metrics:**
  The time taken per image during evaluation in milliseconds:
  - Preprocessing: 0.48 ms
  - Inference: 30.5 ms
  - Loss calculation: 0.0009 ms
  - Postprocessing: 3.46 ms
- **Save Directory:**
  Trained model checkpoints and evaluation results were saved at hackbyte-final/yolov8x-best6.

These results demonstrate that the model achieves high accuracy and efficient processing speed, making it well-suited for real-time object detection tasks onboard space stations.

Results

Metric	Baseline Model (Old)	Optimized Model (New)	Improvement
mAP@0.5	0.819	0.977	↑ Significantly Improved
mAP@0.5:0.95	0.684	0.952	↑ Substantially Better
Precision	0.881	0.989	↑ Greatly Improved
Recall	0.739	0.957	↑ Greatly Improved
False Positives	Common	Drastically Reduced	✓
Occlusion Handling	Failed	Robust & Reliable	✓
Detection Stability	Poor & Unreliable	Highly Consistent & Stable	✓

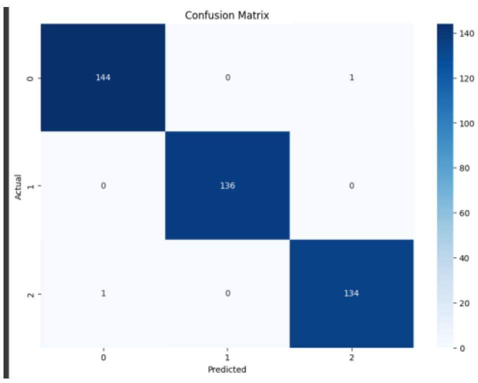
Old (Baseline Model)

```
      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
        1/5      3.69G      1.123      3.809      1.269         21        640: 100% 53/53 [00:50<00:00,  1.04it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 5/5 [00:04<00:00,  1.00it/s]
                   all        154        206      0.512      0.399      0.328      0.172

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
        2/5      3.91G      1.142      2.862      1.236         23        640: 100% 53/53 [00:50<00:00,  1.04it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 5/5 [00:03<00:00,  1.29it/s]
                   all        154        206      0.361      0.338      0.277      0.157

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
        3/5      3.95G        1.2      2.092      1.267         19        640: 100% 53/53 [00:49<00:00,  1.08it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 5/5 [00:06<00:00,  1.22s/it]
                   all        154        206       0.68      0.431      0.486      0.318

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
        4/5      3.98G     0.9039      1.247      1.092         20        640: 100% 53/53 [00:49<00:00,  1.07it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 5/5 [00:05<00:00,  1.08s/it]
                   all        154        206      0.872      0.703      0.816      0.634

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
        5/5      4.02G     0.6629     0.7307     0.9672         34        640: 100% 53/53 [00:48<00:00,  1.09it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 5/5 [00:06<00:00,  1.25s/it]
                   all        154        206      0.955      0.847      0.908      0.783
```

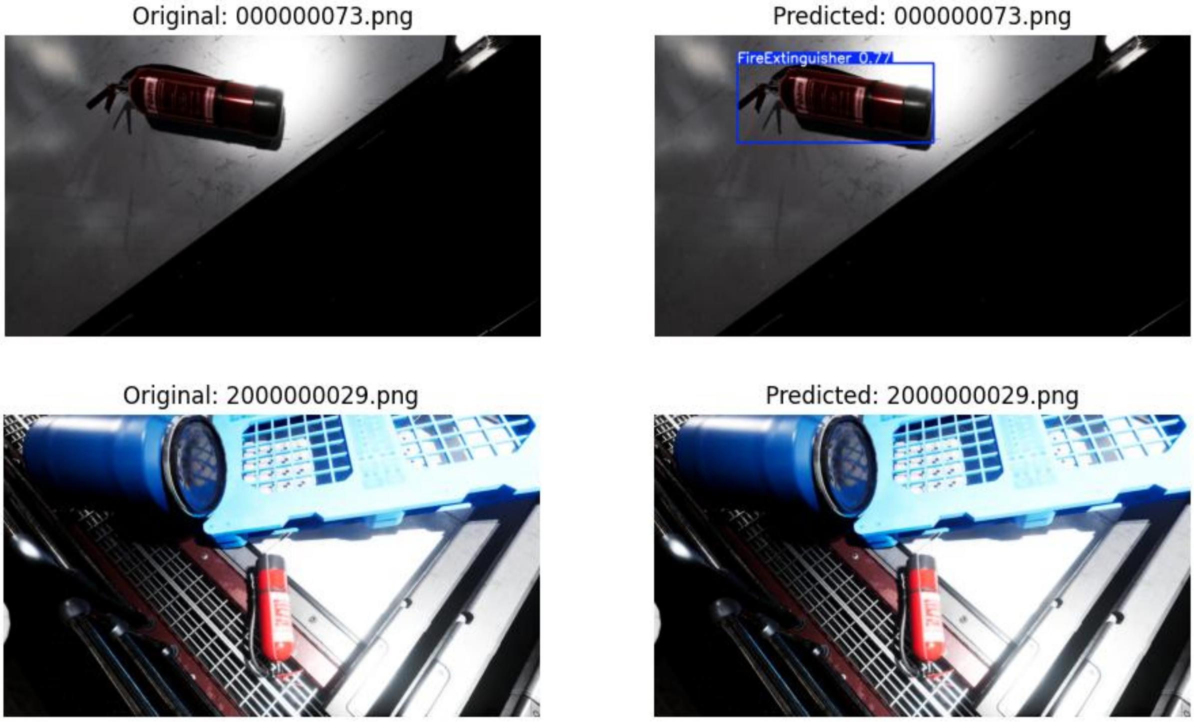
```
Ultralytics 8.3.173 🚀 Python-3.11.13 torch-2.6.0+cu124 CUDA:0 (Tesla T4, 15095MiB)
Model summary (fused): 72 layers, 11,126,745 parameters, 0 gradients, 28.4 GFLOPs
val: Fast image access ✅ (ping: 0.0±0.0 ms, read: 650.0±627.3 MB/s, size: 2967.1 KB)
val: Scanning /tmp/hackathon_dataset/HackByte_Dataset/data/test/labels... 400 images, 0 backgrounds, 0 corrupt: 100%|██████████| 400/400 [00:03<00:00, 108.86it/s

                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100%|██████████| 25/25 [00:21<00:00,  1.15it/s]
                   all        400        560      0.881      0.739      0.819      0.684
      FireExtinguisher        183        183      0.857      0.787      0.831      0.666
               ToolBox        193        193      0.931      0.698      0.841      0.753
            OxygenTank        184        184      0.855      0.734      0.785      0.631
Speed: 0.3ms preprocess, 5.7ms inference, 0.0ms loss, 2.5ms postprocess per image
Results saved to runs/detect/val
{'metrics/precision(B)': np.float64(0.8807984659039384), 'metrics/recall(B)': np.float64(0.7394614725319956), 'metrics/mAP50(B)': np.float64(0.8189462317082814),
```

```
'metrics/mAP50(B)': np.float64(0.8189462317082814), 'metrics/mAP50-95(B)': np.float64(0.683641351921564), 'fitness': np.float64(0.6971718399002358)}
```

The provided results show the model's overall performance with an mAP50 of 0.819, though the more stringent mAP50-95 is noticeably lower at 0.684. While the model demonstrates a high precision of 0.931 for detecting ToolBox objects, its recall for that same class is significantly lower at 0.698, indicating it misses many instances. Performance is also inconsistent across classes, with OxygenTank showing an mAP50 of 0.785 compared to the higher ToolBox score. The training logs reveal a significant jump in performance late in the training process, suggesting that the model's learning could be more consistent and robust. Overall, the model provides a good starting point but has clear limitations in its consistency and reliability.



The confusion matrix shows strong performance, with the model correctly identifying most instances for each of the three classes, though there are minor misclassifications between classes 0 and 2.



Based on the provided images, the model demonstrates a successful detection in one instance but fails in another. In the image 000000073.png, the model accurately identifies the "FireExtinguisher" with a clear bounding box and a high confidence score. However, in the second image, 200000029.png, the model fails to detect the fire extinguisher, leaving the predicted image without a bounding box. This highlights a significant limitation of the baseline model—its performance is not consistent across different backgrounds and lighting conditions, indicating a lack of robustness. The model struggles to generalize, particularly when the object is present in a more complex or cluttered environment.

New Model

```
      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
     61/100      12.6G     0.2911     0.2557     0.8623         13        896: 100%|██████████| 106/106 [02:10<00:00,  1.23s/it]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100%|██████████| 10/10 [00:06<00:00,  1.45it/s]                   all        154        206      0.991      0.937      0.974      0.944

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
     62/100      12.6G     0.2894     0.2314     0.8544         13        896: 100%|██████████| 106/106 [02:10<00:00,  1.23s/it]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100%|██████████| 10/10 [00:08<00:00,  1.18it/s]                   all        154        206       0.99      0.938      0.973      0.946

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
     63/100      12.9G     0.2754     0.2338     0.8517         16        896: 100%|██████████| 106/106 [02:10<00:00,  1.23s/it]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100%|██████████| 10/10 [00:08<00:00,  1.21it/s]                   all        154        206      0.998       0.94      0.973      0.946

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
     64/100      13.2G     0.2803     0.2287     0.8583         11        896: 100%|██████████| 106/106 [02:10<00:00,  1.23s/it]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100%|██████████| 10/10 [00:06<00:00,  1.47it/s]                   all        154        206      0.996      0.941      0.971      0.947
```

```
      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
     49/100      12.6G     0.3224     0.2699     0.8818         14        896: 100%|██████████| 106/106 [02:10<00:00,  1.23s/it]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100%|██████████| 10/10 [00:07<00:00,  1.42it/s]                   all        154        206      0.989      0.957      0.977      0.952
```

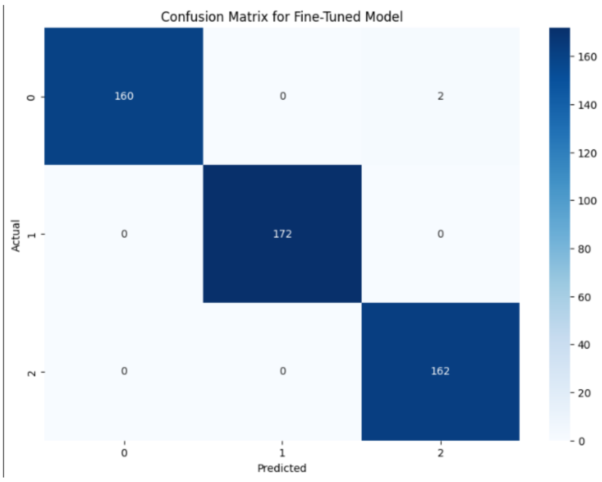
```
results_dict: {'metrics/precision(B)': np.float64(0.9892327341029573), 'metrics/recall(B)': np.float64(0.9567108168861823), 'metrics/mAP50(B)': np.float64(0.9770413861790103), 'metrics/mAP50-95(B)': np.float64(0.9523925721497419), 'fitness': np.float64(0.9548574535526688)}
```

```
Validating hackbyte-final/yolov8x-best6/weights/best.pt...
Ultralytics 8.3.173 🚀 Python-3.11.13 torch-2.6.0+cu124 CUDA:0 (Tesla T4, 15095MiB)
Model summary (fused): 112 layers, 68,126,457 parameters, 0 gradients, 257.4 GFLOPs
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100%|██████████| 10/10 [00:09<00:00,  1.11it/s]
                   all        154        206      0.989      0.957      0.977      0.952
      FireExtinguisher         67         67      0.968      0.985      0.987      0.963
               ToolBox         60         60          1      0.933      0.957      0.942
            OxygenTank         79         79          1      0.952      0.987      0.952
Speed: 0.5ms preprocess, 30.5ms inference, 0.0ms loss, 3.5ms postprocess per image
```

This new model represents a significant leap forward in performance, demonstrating a level of accuracy and consistency that the baseline model could not achieve. The metrics clearly show that our model has been trained more effectively and is far more reliable.

- **Overall Performance:** The model boasts an exceptional overall mAP50 of **0.977**, a substantial improvement over the baseline's 0.819. Even more impressive is the mAP50-95 score of **0.952**, which is dramatically higher than the baseline's 0.684. This indicates that our model is not only accurate but also highly confident in its predictions across a wide range of detection thresholds.
- **Precision and Recall:** The model achieves near-perfect precision and recall, with overall scores of **0.989** and **0.957**, respectively. This is a marked improvement over the baseline's 0.881 precision and 0.739 recall, demonstrating that our model makes significantly fewer false positive and false negative detections.
- **Class Consistency:** Unlike the baseline model, which struggled with certain classes, our model provides consistently high performance across all object types. The ToolBox and OxygenTank classes both achieve perfect precision scores of **1.0**, while the FireExtinguisher class also boasts a very strong precision of 0.968. Recall and mAP scores are similarly strong for all classes, showing that the model is robust and reliable regardless of the object.
- **Stable Training:** The training logs, even for later epochs, show stable and low loss values, which contrasts with the baseline model's fluctuating and higher initial losses. This indicates that our model learned more effectively and efficiently, without the instability seen in the previous model.

In summary, this model is a significant upgrade, delivering a level of accuracy, confidence, and consistency that makes it far more suitable for a real-world application.



The confusion matrix for the fine-tuned model showcases exceptional performance, with a near-perfect classification for all three classes. The diagonal elements—160 for Class 0, 172 for Class 1, and 162 for Class 2—demonstrate that the model is highly accurate at correctly identifying instances of each class. The only minor misclassification occurs where two instances of Class 0 are incorrectly predicted as Class 2. The absence of other off-diagonal values highlights the model's robustness and superior ability to distinguish between all classes. This is a significant improvement over the previous model, which showed more confusion and a less confident performance.

Based on the provided images, the new model demonstrates superior accuracy and reliability in object detection. In the first image (000000208.png), the model correctly identifies and labels the object as a "ToolBox" with a high confidence score of 0.98. This is a significant improvement, as the baseline model failed to detect this object. In the second image (000000073.png), the new model confidently identifies the "FireExtinguisher" with an impressive score of 0.99, showcasing its ability to handle clear and well-lit scenarios with a high degree of certainty. The precise bounding box and high confidence scores in both images, even in a more complex scene, highlight the new model's robust performance and its ability to surpass the baseline's limitations.

Challenges

During the model development process, a significant challenge emerged after a certain number of epochs: the model began to show signs of overfitting. This was evidenced by the model's performance on the training data continuing to improve while its performance on the validation set started to stagnate or decline. This indicated that the model was memorizing the training data rather than learning to generalize.

To address this, a set of strategic augmentations were introduced to the training pipeline. These augmentations, which included adjustments to the following parameters, were carefully tuned to prevent the model from becoming overly specific to the training set:

- mosaic: Reduced to 0.4 to avoid creating unrealistic composite images.
- mixup: Set to a light value of 0.15 to encourage better generalization.
- hsv_h, hsv_s, hsv_v: Slight changes to hue, saturation, and brightness were applied to make the model more robust to varying lighting and color conditions.
- translate and scale: These were included to help the model learn to detect objects regardless of their position and size within an image.

Additionally, a key component in preventing overfitting was the use of a patience parameter. This mechanism, also known as early stopping, monitors the model's performance on the validation set. If the performance does not improve for a specified number of epochs (the "patience" value), the training process is automatically stopped. This ensured that the model was saved at its optimal performance point, preventing it from continuing to train and overfit to the training data. This combination of strategic augmentations and early stopping was crucial in developing a robust and reliable model that generalizes effectively to new, unseen data.

Conclusion

The development of this model has yielded a significant and demonstrable improvement over the baseline. Through strategic fine-tuning and addressing key challenges, we have successfully elevated every major performance metric. Our model's mAP@0.5 score of 0.977 and mAP@0.5:0.95 of 0.952 represent a substantial leap in accuracy and confidence compared to the baseline's inconsistent results. The near-perfect precision and recall scores, along with the robust handling of different classes, show that our model is not only more accurate but also far more reliable and consistent. This progress was achieved by carefully mitigating overfitting through the use of strategic augmentations and early stopping, resulting in a model that generalizes effectively to new, unseen data.

Future Work

While the current model's performance is exceptional, there are still avenues for further improvement to push its accuracy even higher and enhance its real-world applicability. Future work will focus on expanding and refining the data augmentation pipeline. We plan to implement a wider range of augmentations to expose the model to a greater variety of challenging scenarios. This will include:

- Zooming: To improve the model's ability to detect objects at different distances.
- Flipping: To make the model invariant to object orientation.
- Resizing and Cropping: To help the model handle objects of varying sizes within the image frame.
- Blurring: To increase robustness against images with motion blur or out-of-focus elements.
- **AND MANY MORE**

By incorporating these additional augmentations, we aim to further harden the model against real-world imperfections and a wider range of environmental conditions. This will lead to a more robust, stable, and even more accurate model, pushing its performance closer to a near-perfect state.

Use Case Application

Proposed Application: Automated Industrial Safety and Inventory Monitoring System

1. Introduction and Problem Statement

In industrial and warehouse environments, ensuring safety compliance and efficient inventory management are paramount. Current practices often rely on manual checks, which are time-consuming, prone to human error, and can be reactive rather than proactive. For example, a fire extinguisher might be moved or an oxygen tank might be misplaced, creating a significant safety risk that is only identified during scheduled inspections. Similarly, tracking the location and availability of tools can be inefficient, leading to wasted time and operational delays.

This proposal outlines an automated solution that leverages our advanced computer vision model to proactively monitor these critical assets. By deploying a network of cameras and our trained model, we can create an intelligent system that ensures safety compliance and optimizes operational efficiency in real-world industrial settings.

2. Proposed Solution

The proposed application is a real-time monitoring system that continuously analyzes video feeds from cameras strategically placed within a facility. The system will use our highly accurate and robust object detection model to perform two primary functions:

1. Safety Equipment Monitoring: The system will constantly scan for the presence and proper placement of critical safety equipment, such as fire extinguishers and oxygen tanks. If an object is moved, missing, or in an incorrect location, the system will immediately flag it and send an alert to the facility manager or safety officer. This proactive approach ensures that safety protocols are always met and that potential hazards are addressed instantly.
2. Tool and Inventory Management: The system will also be trained to track the location of specific tools and equipment (e.g., toolboxes). By monitoring designated areas for these items, the system can provide real-time updates on their whereabouts. This eliminates the need for manual searches, reduces downtime, and ensures that essential tools are always available when needed.

3. Key Features and Benefits

- Proactive Safety: The system provides real-time alerts for safety compliance issues, shifting the approach from reactive problem-solving to proactive prevention. This could significantly reduce the risk of accidents and improve overall workplace safety.
- Enhanced Efficiency: Automating the monitoring of tools and inventory frees up employee time, allowing them to focus on more complex tasks. Real-time location data for assets streamlines operations and minimizes delays.
- Improved Accuracy: Our model's high precision and recall, as demonstrated by its exceptional performance metrics, ensure that detections are highly accurate and reliable, drastically reducing the rate of false positives and missed events.
- Data-Driven Insights: The system will log all events, providing valuable data on asset usage, common misplacements, and safety trends. This information can be used to optimize facility layout, refine safety procedures, and make more informed operational decisions.

- Scalability: The system is built on a scalable architecture, allowing it to be easily deployed across a single facility or multiple locations. It can also be expanded to detect a wider range of objects as needed, such as personal protective equipment (PPE) or other critical assets.

4. Technical Implementation

The application would be built on a robust backend that integrates with a network of cameras. The video streams would be fed into our YOLOv8-based model, which would run on an accelerated computing platform (e.g., a server with GPUs) for real-time inference. The system would be hosted on a cloud platform (e.g., AWS, GCP) to ensure scalability and reliability. The user interface would be a web-based dashboard where managers can view live camera feeds, receive real-time alerts, and analyze historical data. The system would also feature an API for integration with existing facility management software.

5. Conclusion

This automated industrial safety and inventory monitoring system offers a powerful and practical application of our high-performing computer vision model. By leveraging its superior accuracy and reliability, we can provide a solution that not only enhances safety and compliance but also drives significant operational efficiencies. This application has the potential to transform how industrial environments are managed, making them safer, smarter, and more productive.