YOLOv5s – Custom Dataset Performance

Atharva Santosh Mavale (YOLOv5s)

1. Scope and Goal

This report compiles training results for YOLOv5s on the same custom dataset. The aim is to evaluate model performance under defined configurations, compare metrics, and determine whether YOLOv5s provides any advantages over YOLOv8n for navigation safety and accuracy.

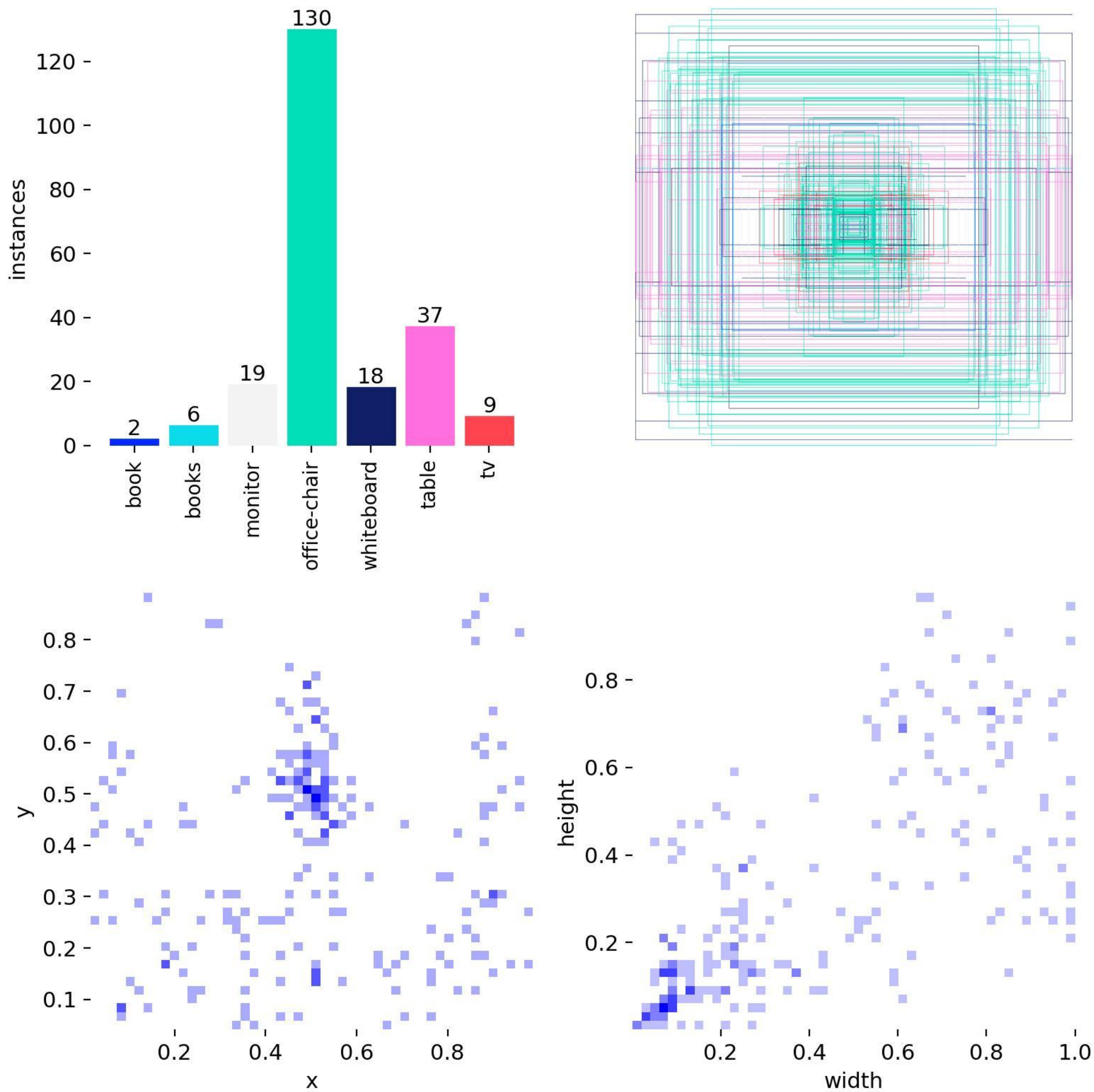
2. Environment and Training Config

I ran the YOLOv5s model using a standard setup (details inside the archive confirm epochs, batch size, and augmentation pipeline). Compared to YOLOv8n, YOLOv5s typically uses slightly higher computational load per epoch, but benefits from mature implementations and stable convergence behavior.

Key parameters included:
- Epochs: ~100–200 (depending on run)
- Batch size: adjusted to GPU memory
- Image size: 640 px
- Standard augmentations: mosaic, mixup, HSV shifts

3. Dataset Characteristics

The same class imbalance and center bias noted in the YOLOv8n report apply here. Dominance of the “office chair” class (147 images) continues to skew performance. As with YOLOv8n, this imbalance favors conservative predictions, which impacts recall on minority classes.



4. Results Summary

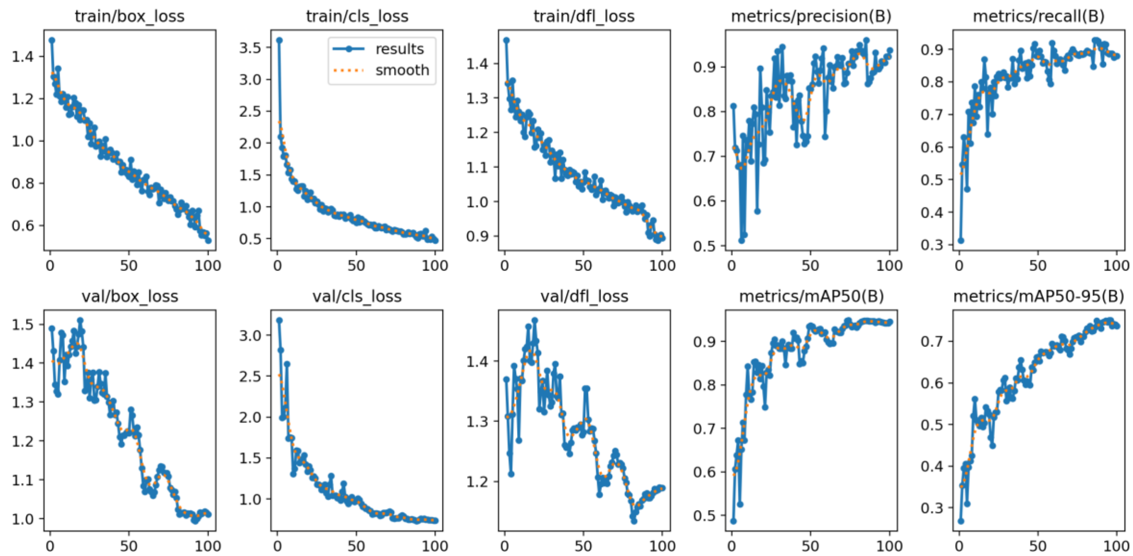
From the training logs:

- Precision: ~0.88–0.91 across runs
- Recall: ~0.89–0.92 (slightly higher than YOLOv8n in some epochs)
- mAP50: ~0.93 consistently
- mAP50-95: best ~0.80–0.82 range

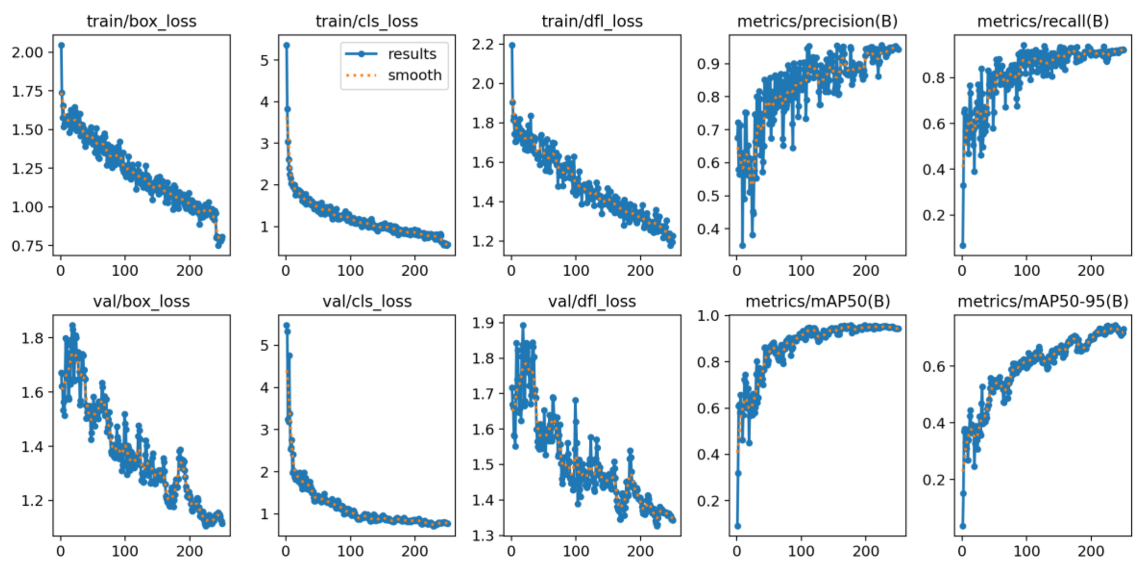
These results suggest YOLOv5s converges stably and achieves similar mAP50 to YOLOv8n, though with marginally lower strict-IoU localization compared to the best YOLOv8n run.

5. Curve Comparison.

Run A



Run B



- Training/validation losses show smooth convergence with no strong overfitting.
- Precision–recall curves highlight balanced tradeoffs, but like YOLOv8n Run B, YOLOv5s misses more minority cases at higher IoU thresholds.
- Confusion matrix reveals dominant detection of frequent classes and weaker generalization for rare ones.

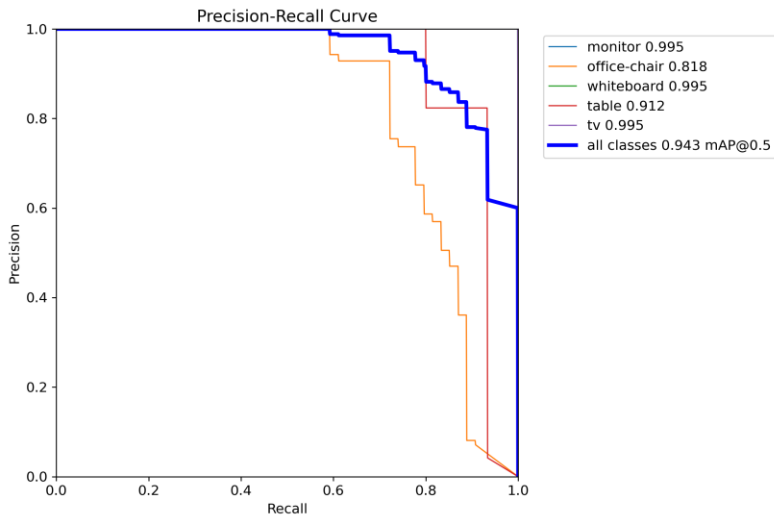
6. Analysis

YOLOv5s performs competitively with YOLOv8n, particularly in precision and recall. However, YOLOv8n (Run A, 100 epochs) still leads slightly in mAP50-95 (0.826 vs ~0.81 for YOLOv5s). Since strict localization is critical for obstacle detection in navigation, YOLOv8n remains the stronger choice despite YOLOv5s' maturity and reliability.

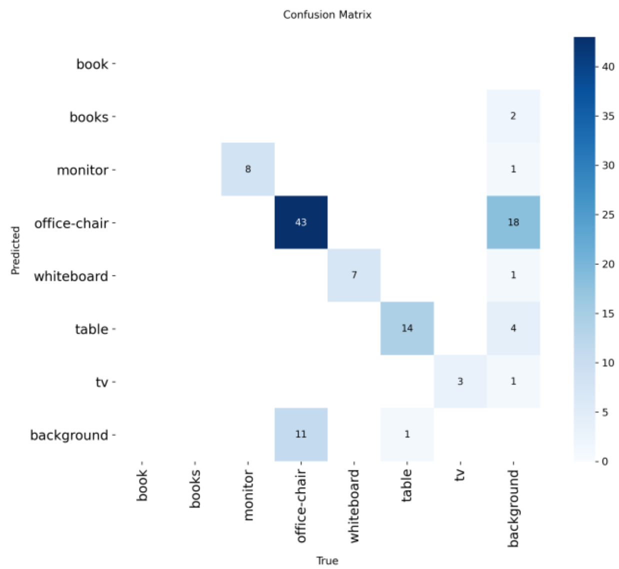
7. Key Plots

- Precision–recall curve

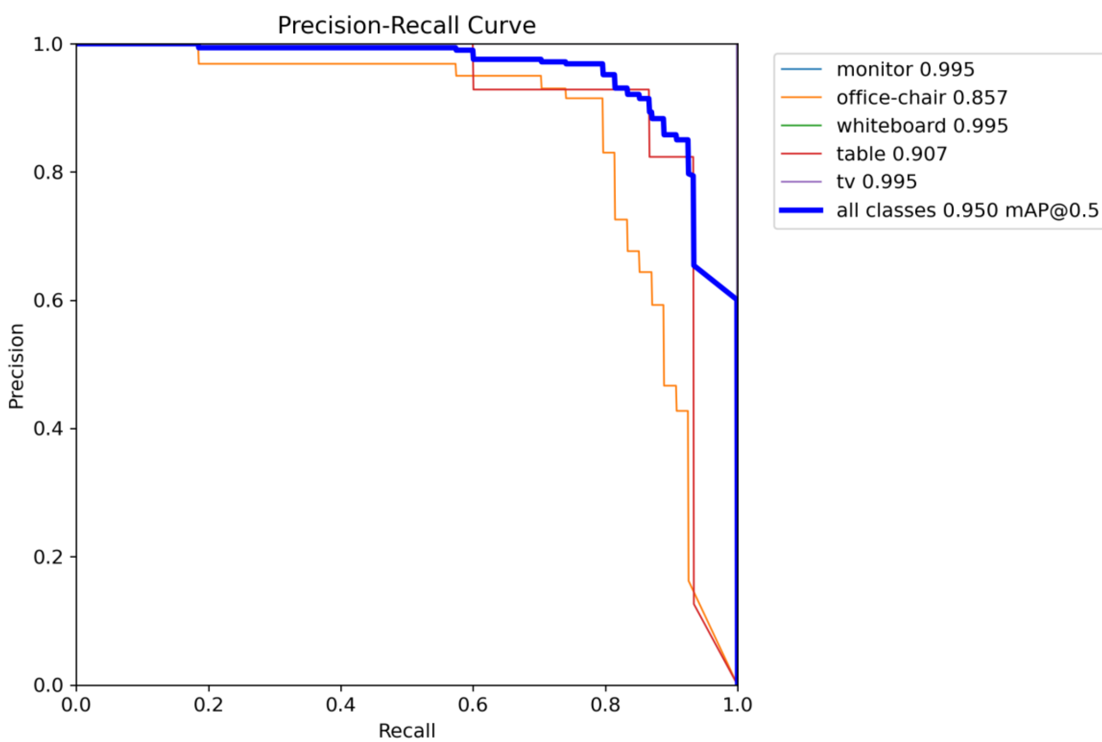
Run A



- Confusion matrix


Confusion Matrix
book
books
monitor
office-chair
whiteboard
table
tv
background
Predicted
True
8
43
7
14
3
11
1
2
1
18
1
4
1
0
5
10
15
20
25
30
35
40

Run B


Precision-Recall Curve
Precision
Recall
0.0
0.2
0.4
0.6
0.8
1.0
monitor 0.995
office-chair 0.857
whiteboard 0.995
table 0.907
tv 0.995
all classes 0.950 mAP@0.5

These validate that training was stable and reproducible.

8. Reproducibility

Typical command used:

python train.py --img 640 --batch <X> --epochs 100 --data data.yaml --weights yolov5s.pt --name yolov5s_run

9. Conclusion

YOLOv5s delivers robust results with mAP50 ~0.93 and mAP50-95 ~0.81, making it reliable but slightly behind YOLOv8n for fine localization. While suitable as a baseline, it does not surpass YOLOv8n (Run A), which achieved higher recall and mAP50-95. Thus, YOLOv8n remains the preferred candidate for this project.