Comparative Analysis between a Transformer and CNN based Object Detection Model

Pavan Ganesh Pabbineedi

Models Chosen – DeformableDETR and YOLOv8 (Why These Models?)

DETR (**DE**tection **TR**ansformer): It uses a **transformer-based architecture** to provide end-to-end object detection, treating the task as a direct set prediction problem

However, standard DETR has two significant limitations:
1. Slow convergence: **requiring 500 epochs** of training to reach competitive performance
2. Limited feature spatial resolution: **affecting** its ability to **detect small objects**

Deformable DETR: This particularly suitable for applications requiring **accurate detection of objects at various scales**, especially when small object detection is important, while still maintaining reasonable training times.

1. 10× Faster Training: Deformable DETR achieves better results in just 50 epochs (vs. 500), cutting training time from 2000+ to 325 GPU hours while improving detection quality.
2. Superior Accuracy: Two-stage Deformable DETR with ResNet50 reaches 46.9% mAP on COCO, outperforming both original DETR and Faster R-CNN (both at 42.0%).
3. Balanced Speed-Accuracy: Though slightly slower at inference (14.5 vs 27.0 FPS), Deformable DETR delivers significantly better detection performance, especially for small objects.

YOLOv8: It combines **speed and accuracy** for real-time object detection. Its **user-friendly design** and **versatility** make it easy to deploy across diverse tasks, while its **state-of-the-art performance** ensures reliable results even in complex scenarios. **Widely adopted in industries**.

Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., & Zagoruyko, S. (2020). End-to-End Object Detection with Transformers. In European Conference on Computer Vision (ECCV).
Zhu, X., Su, W., Lu, L., Li, B., Wang, X., & Dai, J. (2021). Deformable DETR: Deformable Transformers for End-to-End Object Detection. In International Conference on Learning Representations (ICLR).

Architecture

Deformable DETR Architecture:

Deformable DETR uses a transformer-based approach with these key components:

Backbone: Uses ResNet-50 as the primary feature extractor to extract multi-scale features

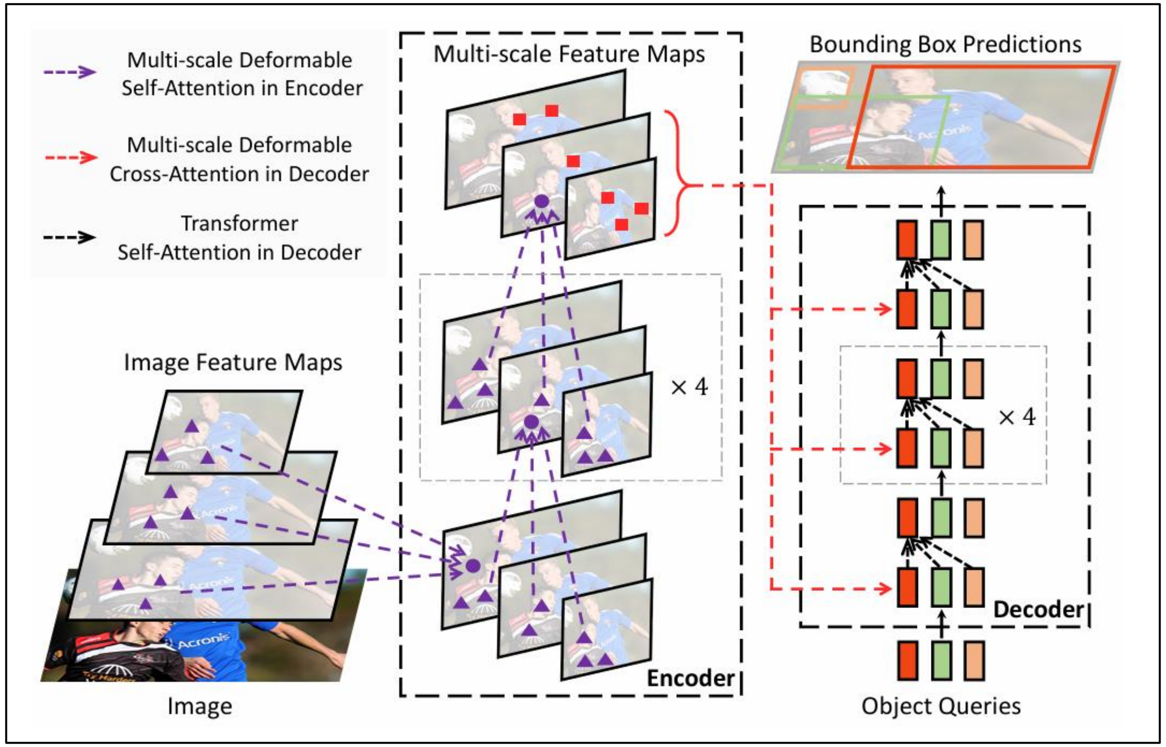
Deformable Attention Module: The core innovation that only attends to a small set of key sampling points around a reference point, reducing computational complexity from quadratic to linear

Encoder-Decoder Transformer:

- Encoder processes image features using deformable attention
- Decoder takes object queries and refines them using encoded features

Prediction Heads: Linear layer for class prediction and MLP for bounding box coordinates

Deformable DETR eliminates the need for hand-designed components like non-maximum suppression (NMS) and anchor generation, treating object detection as a direct set prediction problem.



Zhu, X., Su, W., Lu, L., Li, B., Wang, X., & Dai, J. (2021). Deformable DETR: Deformable Transformers for End-to-End Object Detection. In International Conference on Learning Representations (ICLR).

Continue...

YOLOv8 Architecture

YOLOv8 maintains a CNN-based approach with significant enhancements:

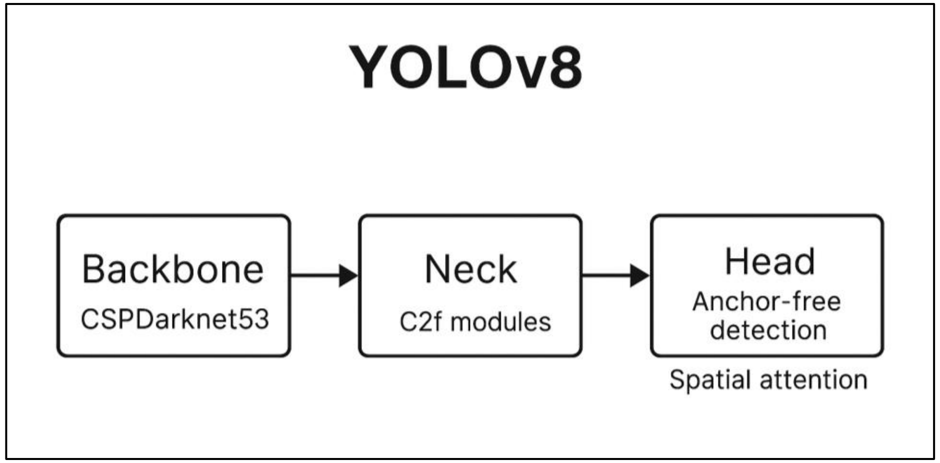
Backbone: Custom CSPDarknet53 with cross-stage partial connections to improve information flow between layers

Neck: Uses C2f modules instead of traditional Feature Pyramid Network (FPN), combining high-level semantic features with low-level spatial information

Head:

- Employs an anchor-free approach for bounding box prediction
- Multiple detection modules predict bounding boxes, objectness scores, and class probabilities
- Decoupled head separates classification and localization tasks

YOLOv8 incorporates spatial attention mechanisms and bottlenecks to reduce computational complexity while maintaining accuracy.



Feature	Deformable DETR	YOLOv8x
Base architecture	Transformer-based	CNN-based
Key innovation	Deformable attention mechanism	Anchor-free detection
Preprocessing	Complex with specific normalization	Streamlined
Post-processing	No NMS required	Includes NMS
Training epochs	10x— fewer than original DETR	Not specified
Trained Dataset	COCO 2017	COCO dataset

Jocher, G., Chaurasia, A. and Qiu, J. (2023) YOLO by Ultralytics

Evaluation Methodology: Ensuring Fair Comparison Between YOLOv8 and Deformable DETR

Fixed Parameter:

- **Consistent Test Dataset:** Both models evaluated on identical test images to ensure fair comparison (11 images)
- **Standardized Hardware:** Testing conducted on NVIDIA GeForce RTX 4060 GPU for consistent performance benchmarking
- **Uniform Confidence Threshold:** Applied same detection confidence threshold (0.6) across both models
- **Controlled Image Size:** Standardized input resolution (640px) for both models

Metrics Calculation Explanation:

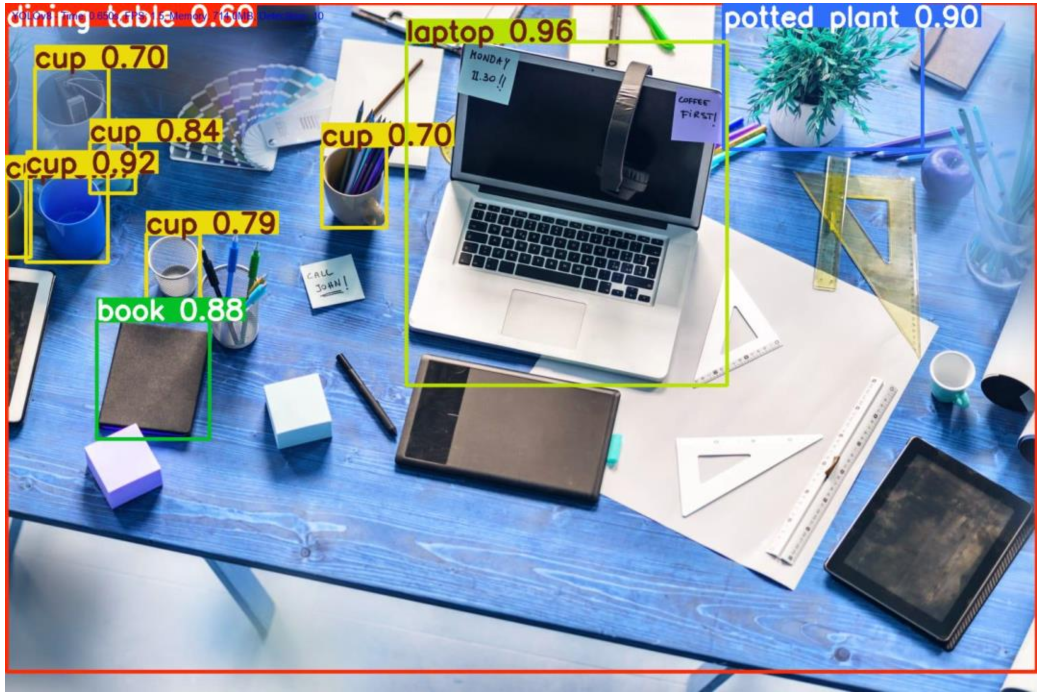
Metric	Calculation	Description
Inference Time (s)	End time - Start time	Measures the total time taken for model inference
FPS	1 / Inference Time	Frames per second - higher is better for real-time applications
Detections	len(results['scores'])	Number of objects detected above confidence threshold
Memory (MB)	torch.cuda.max_memory_allocated() / (1024 * 1024)	Peak GPU memory usage during inference
Avg Confidence	mean(results['scores'])	Average confidence score of all detections

Results

Test Image 1:

DeformableDETR

YOLOv8



Results

Test Image 3:

DeformableDETR

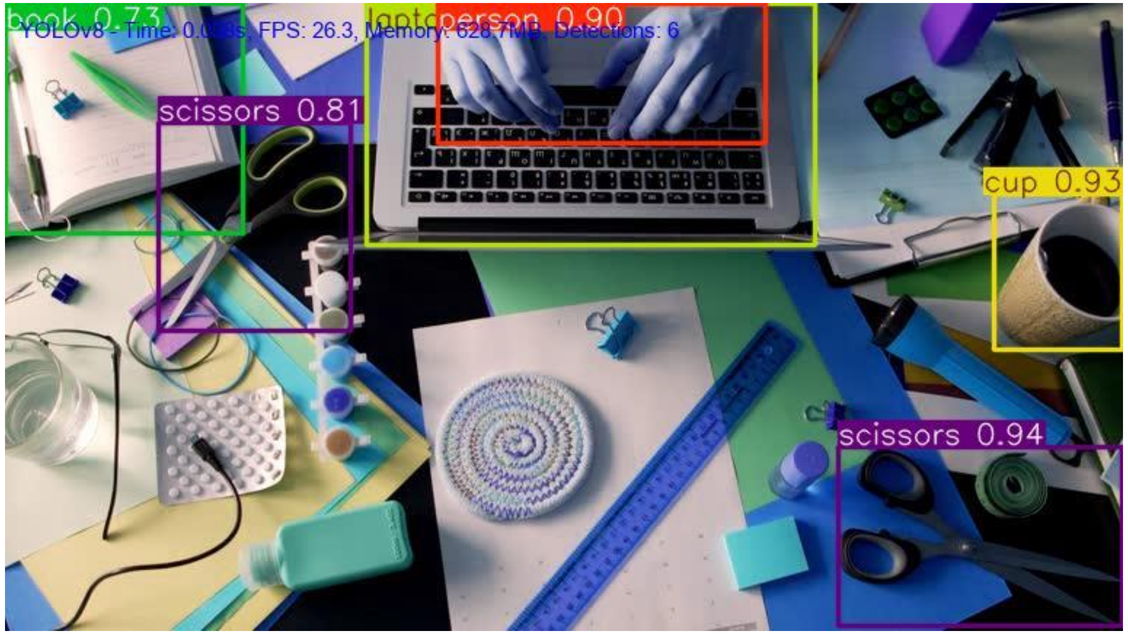
YOLOv8

Results

Test Image 5:

DeformableDETR

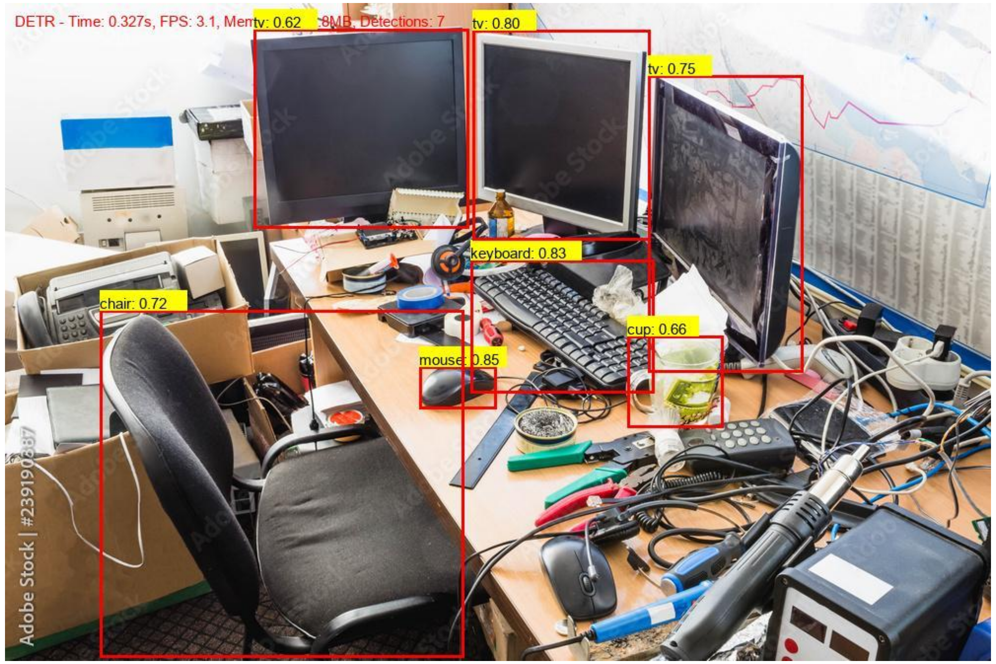
YOLOv8



Results

Test Image 6:

DeformableDETR



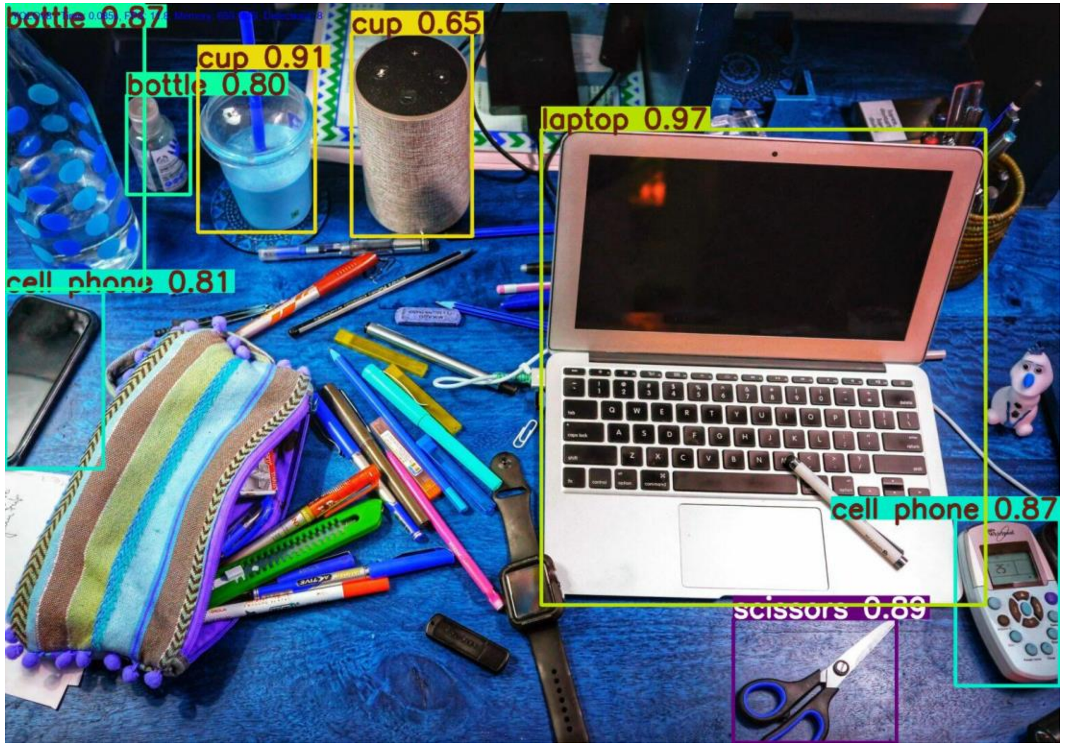
YOLOv8

Results

Test Image 6:

DeformableDETR

YOLOv8

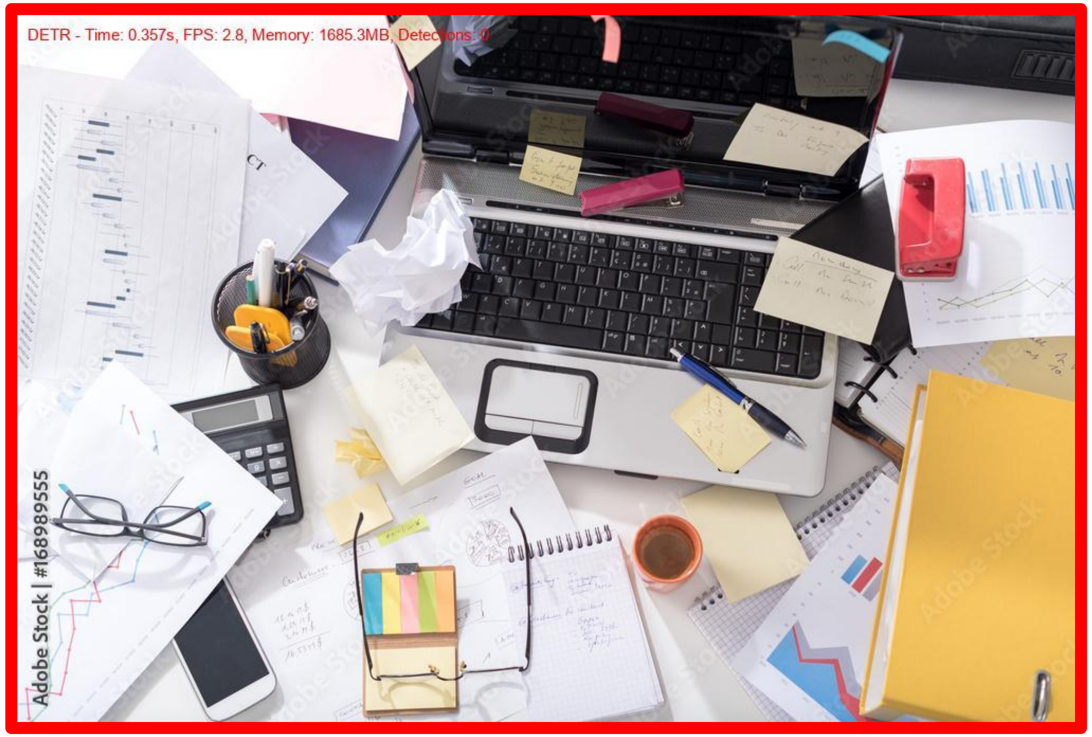


Results

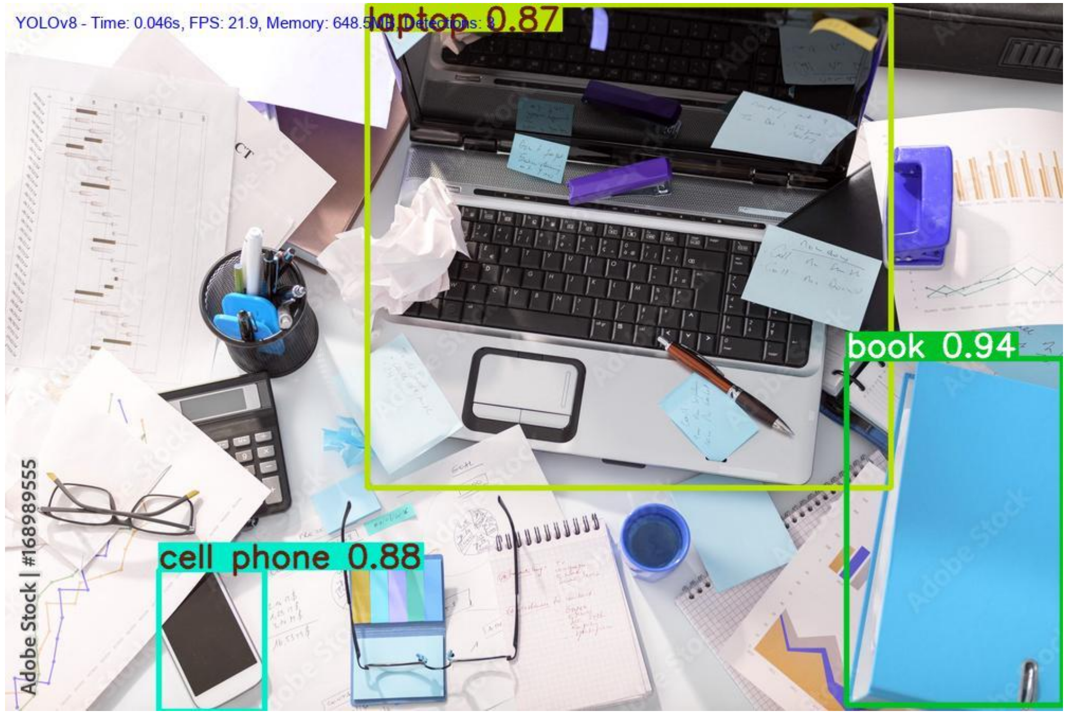
Failed Cases

Test Image 2:

DeformableDETR



YOLOv8



Outline color red – No Detections or False Detection

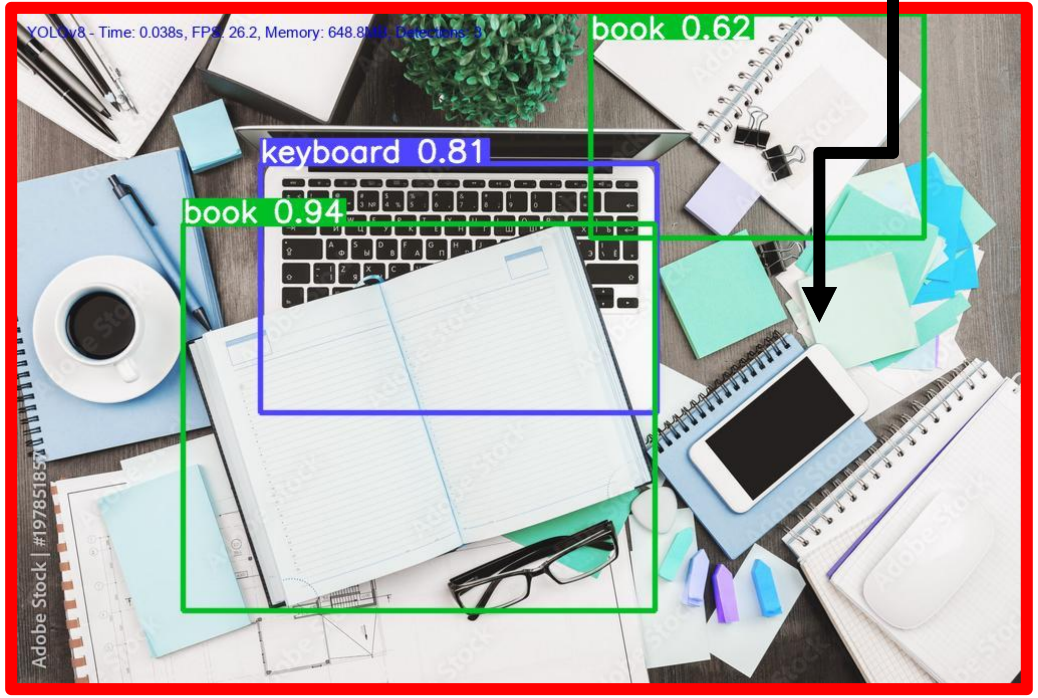
Results

Failed Cases:

YOLOv8

Test Image 3: False detection (book)

Test Image 4: No Detection (Cell Phone)



Outline color red – No Detections or False Detection

Evaluation Metric

Metrics Comparison Table:

Image	Model	Inference Time (s)	FPS	Detections	Memory (MB)	Avg Confidence
1.jpg	DETR	1.5372	0.65	6	1398.94	0.6724
1.jpg	YOLOv8	0.6498	1.54	10	713.99	0.8157
10.jpeg	DETR	0.2903	3.45	1	1684.07	0.6857
10.jpeg	YOLOv8	0.0424	23.56	5	648.77	0.9022
11.jpeg	DETR	0.2942	3.4	2	1744.32	0.7808
11.jpeg	YOLOv8	0.0601	16.64	5	628.68	0.8445
2.jpeg	DETR	0.3574	2.8	0	1685.31	N/A
2.jpeg	YOLOv8	0.0456	21.95	3	648.52	0.8953
3.jpeg	DETR	0.317	3.15	3	1597.12	0.7779
3.jpeg	YOLOv8	0.0487	20.55	5	601.87	0.8382
4.jpeg	DETR	0.3383	2.96	0	1681.6	N/A
4.jpeg	YOLOv8	0.0381	26.22	3	648.75	0.7911
5.jpg	DETR	0.354	2.82	2	1744.37	0.6902
5.jpg	YOLOv8	0.0381	26.28	6	628.68	0.8590
6.jpeg	DETR	0.3272	3.06	7	1684.79	0.7480
6.jpeg	YOLOv8	0.0679	14.73	9	648.12	0.8849
7.jpg	DETR	0.3727	2.68	4	1683.27	0.7590
7.jpg	YOLOv8	0.0682	14.67	6	648.18	0.8614
8.jpeg	DETR	0.3815	2.62	1	1683.03	0.7154
8.jpeg	YOLOv8	0.0684	14.63	3	648.77	0.8661
9.jpg	DETR	0.3688	2.71	3	1629.71	0.7202
9.jpg	YOLOv8	0.0846	11.82	8	659.33	0.8465

Summary Comparison Table:

Metric	DETR	YOLOv8
Average Inference Time (s)	0.449	0.1102
Average FPS	2.75	17.51
Average Detections	2.64	5.73
Average Memory (MB)	1656.05	647.61
Average Confidence	0.7277	0.855

Speed & Efficiency

- YOLOv8 processes images 4× faster (0.11s vs 0.45s)
- YOLOv8 achieves 17.51 FPS vs DETR's 2.75 FPS
- YOLOv8 uses 60% less memory (647MB vs 1656MB)

Detection Quality

- YOLOv8 detects 2.2× more objects per image (5.73 vs 2.64)
- YOLOv8 maintains higher confidence scores (0.855 vs 0.728)
- YOLOv8 shows more consistent performance across diverse images

Metric based on the Research Paper and Articles:

Metric	Deformable DETR	YOLOv8x
mAP (IoU=0.50:0.95)	46.3%	53.9%
mAP (IoU=0.50)	65.9%	Not specified
mAP for small objects	29.8%	Not specified
mAP for medium objects	49.3%	Not specified
mAP for large objects	60.7%	Not specified
Parameters	Not specified	68.2M
FLOPs	Not specified	257.8B
Research paper inference speed	15.7 FPS (A10e GPU)	283.3 FPS (3.53ms on A100)
CPU inference time	Not specified	479.1ms
Model source	"SenseTime/deformable-detr"	"yolov8x.pt"

Why These Differences Exist

1. **Architectural Differences**: YOLOv8's CNN-based architecture is optimized for speed, while DETR's transformer-based approach prioritizes different aspects of detection.
2. **Preprocessing Complexity**: As seen in the DETR preprocessing details output, Deformable DETR uses a more complex preprocessing pipeline that resizes images to maintain aspect ratio with shortest edge at 800 pixels and longest edge at 1333 pixels. This higher resolution processing contributes to its slower speed but might help with certain types of detections.
3. **Attention Mechanisms**: Deformable DETR uses deformable attention which, while innovative for handling objects at different scales, requires more computational resources than YOLOv8's approach.
4. **Model Optimization**: YOLOv8 has been heavily optimized for real-time inference on standard hardware, while Deformable DETR is more research-oriented.

Zhu, X., Su, W., Lu, L., Li, B., Wang, X., & Dai, J. (2021). Deformable DETR: Deformable Transformers for End-to-End Object Detection. In International Conference on Learning Representations (ICLR).

Observations:

- My observations align with published benchmarks that confirm YOLOv8x's higher mAP score (53.9% compared to Deformable DETR's 46.3%) on the COCO dataset, though I noted lower absolute performance figures in my testing due to hardware differences.
- Interestingly, I observed that despite Deformable DETR's theoretical advantage in small object detection, YOLOv8 consistently identified more small objects across my test images. This suggests that YOLOv8's overall superior accuracy effectively overcomes any specialized advantages of DETR's transformer architecture.
- Based on my comprehensive evaluation, I found YOLOv8 to be the clear winner for applications requiring real-time performance or deployment on devices with limited computational resources.

Thank you