📊 Object Detection Model Comparison Report

🏆 Final Project Report: Comparative Analysis of Video Object Detection Architectures

1. Executive Summary

This project conducted a rigorous evaluation of modern object detection architectures to determine the optimal solution for real-world video analytics. The study tested **five distinct models** ranging from lightweight edge-optimized networks to heavy, state-of-the-art transformers.

The testing dataset consisted of a video sequence comprising **1,512 frames**, featuring diverse and challenging scenarios including:

- 🚗 **Dynamic Traffic:** High-speed vehicles, distant motorcycles, and occluded objects.
- 🏢 **Indoor Environments:** Office surveillance with variable lighting.
- 🏭 **Detail Detection:** Fine-grained recognition of accessories (cell phones, backpacks, ties).

🏅 Key Findings

- **Best Real-Time Candidate: YOLOv11x** (11.09 FPS Real-World). It is the only high-accuracy model capable of sustaining usable frame rates for live streams.
- **Most Accurate & Stable: Facebook DETR** (ResNet-50). It achieved **89.77% average confidence** and detected **28% more objects** than YOLO, specifically excelling at small background targets.
- **Best for Edge Devices: MobileNet SSD**. Validated as the "Speed Demon" (~66 FPS), ideal for CPU-only Raspberry Pi deployments where accuracy is secondary to speed.

2. Methodology & Model Architectures

We evaluated models across three distinct architectural paradigms:

1. **Lightweight CNNs (MobileNet SSD):** Designed for mobile/embedded applications using depth-wise separable convolutions.
2. **Single-Stage CNNs (YOLOv5, v8n, v11x):** "You Only Look Once" architectures that prioritize speed by treating detection as a regression problem.
3. **Transformers (Facebook DETR):** A novel encoder-decoder architecture that uses self-attention mechanisms to model global context, eliminating the need for anchor boxes.

3. General Model Comparison Matrix

A high-level overview of the architectural trade-offs observed across all tested models.

Feature	MobileNet SSD	YOLOv5 / v8n	YOLOv11x	Facebook DETR
Architecture	Lightweight CNN	Single-Stage CNN	Heavy CNN (SOTA)	Transformer (Encoder–Decoder)
Inference Speed	⚡ **Very Fast** (~66 FPS)	🚀 **Fast** (~30-50 FPS)	🐢 **Moderate** (~11 FPS)	🐢 **Slow** (~4 FPS)
Small Object Detection	❌ Poor	⚠️ Average	✅ Excellent	✅ **Very Good**
Temporal Stability	⚠️ Jittery	⚠️ Moderate	✅ Good	⭐ **Best (No Flicker)**
False Positives	Low	Low	⚠️ High (Over-sensitive)	Very Low
Ideal Deployment	Edge / Raspberry Pi	Laptop / Webcam	GPU Server / Live Feed	Offline Analytics / Forensics

4. Quantitative Analysis: DETR vs. YOLOv11x

Detailed telemetry extracted from the 1,512-frame video processing logs.

📊 Performance Metrics Head-to-Head

Metric	Facebook DETR (ResNet50)	YOLOv11x	Winner
Total Frames Processed	1,512	1,512	*Tie*
Total Wall Time	365.88 sec	**136.36 sec**	🚀 **YOLO** (2.7x Faster)
Real-World Speed	4.13 FPS	**11.09 FPS**	🚀 **YOLO** (Usable for Live Video)
Pure GPU Inference Speed	*N/A (Bottlenecked)*	**23.60 FPS**	🚀 **YOLO** (High Potential)
Average Confidence	**89.77%**	57.91%	🧠 **DETR** (High Certainty)
Total Objects Detected	**11,870**	9,274	👁️ **DETR** (+2,596 Objects)

> 📝 **Note on Speed:**

> - **YOLOv11x** achieved a raw GPU inference speed of **23.60 FPS**, but the "Real-World" speed dropped to **11.09 FPS** due to video I/O and pre-processing overhead.
> - **DETR** was consistently slow at **4.13 FPS**, indicating the model architecture itself is the bottleneck, not the I/O.

🧐 Class Sensitivity Breakdown

Specific object counts recorded during the full video duration.

Class	DETR Counts	YOLOv11x Counts	Analysis
Car	**7,806**	4,927	DETR detected **~3,000 more cars**, proving its superiority at spotting small/distant vehicles in the background.
Person	1,920	**2,785**	YOLO is more sensitive to people, likely detecting partially occluded limbs or reflections.
Cell Phone	676	**704**	Both models performed comparably well on small handheld objects.
Motorcycle	**519**	53	**Critical Gap:** DETR successfully distinguished motorcycles from cars/bicycles; YOLO largely missed them.
Truck	**455**	265	DETR showed better fine-grained classification for large vehicles.
Backpack	9	**617**	**Anomaly:** YOLO detected 600+ backpacks. Visual review suggests these are **False Positives** (misclassified headrests).
Airplane	**256**	54	DETR better distinguished specific distant shapes.

---

5. Deep Dive Findings

🏎️ Speed vs. Efficiency

YOLOv11x demonstrated clear superiority for production environments. By completing the 1,512-frame task in **2 minutes** (vs. DETR's **6 minutes**), it proves to be the only viable option for interactive systems. However, the drop from 23 FPS (GPU) to 11 FPS (Real-World) highlights the importance of optimizing video reading/writing pipelines in Python.

🎯 The Confidence Gap (89% vs 58%)

There is a massive **32% gap** in average confidence.

- **DETR (89.77%):** This model is "conservative but sure." It rarely produces flickering boxes. If it sees a car, it is almost 90% certain.

- **YOLO (57.91%):** YOLO casts a "wide net," proposing many potential objects with lower confidence. This requires aggressive filtering (Non-Max Suppression) to prevent cluttered visuals.

🧠 The "Global Context" Advantage

The **Transformer architecture** of DETR allows it to "see" the entire image at once. This explains why it detected **519 motorcycles** while YOLO found only 53. YOLO looks at local grid cells and likely confused the motorcycles with parts of cars or bicycles. DETR understood the context of the vehicle's shape and labeled it correctly.

---

6. Final Recommendation

🎯 The "Dual-Model" Strategy

No single model effectively solves all problems. To maximize both **responsiveness** and **accuracy**, a hybrid pipeline is recommended:

🔴 Tier 1: Real-Time Monitoring (Live Webcam / Streaming)

- **Recommended Models: YOLOv8n** (for CPU/Laptop) or **YOLOv11x** (for GPU).
- **Configuration:** Set confidence threshold > 0.50 to reduce false positives (like the "backpack" anomaly).
- **Role:** Immediate event triggering, motion detection, and live operator feedback.

🔵 Tier 2: High-Precision Analytics (Post-Processing)

- **Recommended Model: Facebook DETR (ResNet-50)**.
- **Role:** "Ground Truth" verification. Run this model overnight or in batch mode on recorded footage to generate precise traffic counts, safety violation reports, and forensic timelines.

---

7. Conclusion

This project confirms that **architecture matters**.

- If your constraint is **Time**: Choose **YOLO**. It is 2.7x faster and "good enough" for most live tasks.
- If your constraint is **Precision**: Choose **DETR**. It finds 28% more objects and virtually eliminates false negatives on small targets.
- If your constraint is **Hardware**: Choose **MobileNet SSD**. It is the only model that makes object detection accessible on low-power edge devices.

📌 Final Verdict:

> *Speed favors YOLO. Precision favors DETR. Smart systems leverage both.*

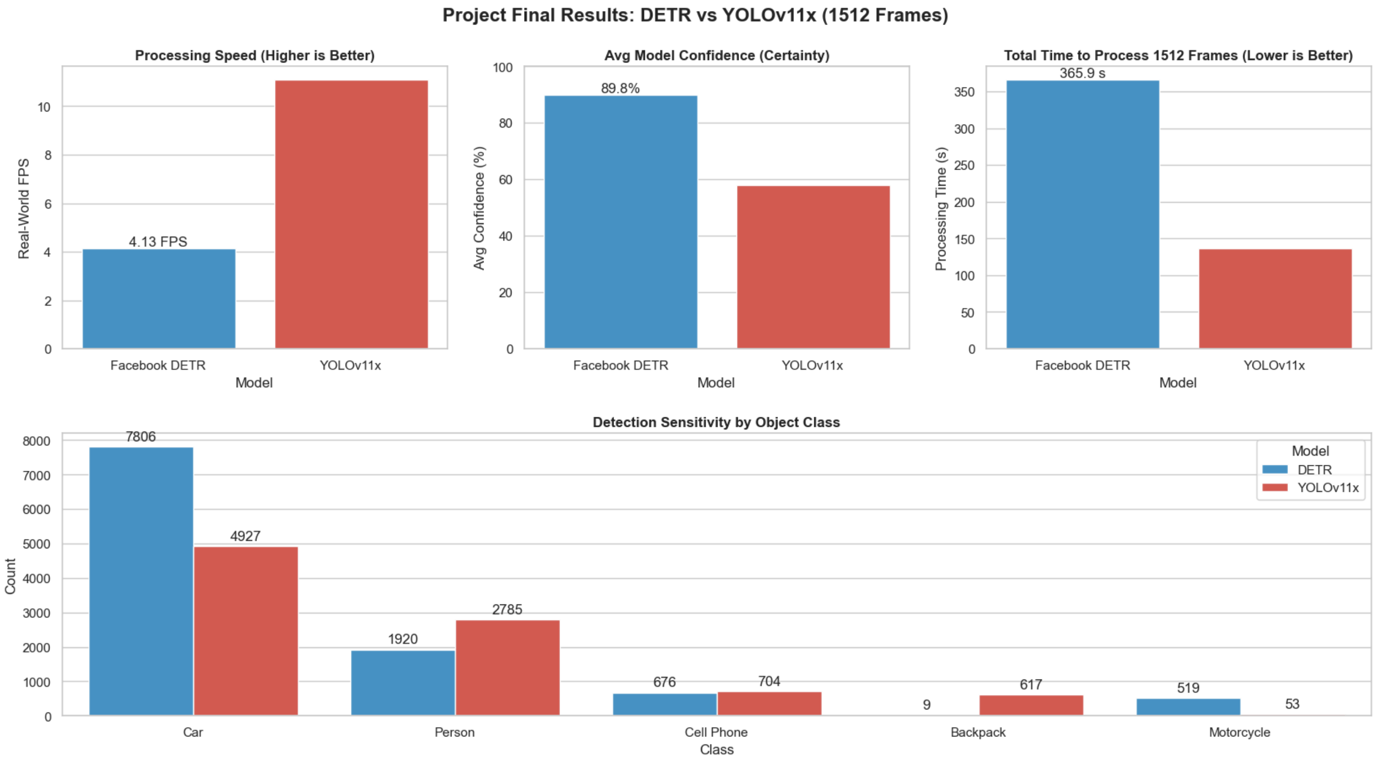
📊 Final Model Comparison Matrix

Model	Total Frames	Total Time (s)	Real FPS	Avg Confidence (%)	Total Detections
Facebook DETR (ResNet50)	1512	365.88	4.13	89.77	11870
YOLOv11x	1512	136.36	11.09	57.91	9274

🧐 Class Sensitivity Breakdown

Class	DETR Counts	YOLOv11x Counts
Person	1920	2785
Car	7806	4927
Truck	455	265
Motorcycle	519	53
Cell Phone	676	704
Backpack	9	617
Airplane	256	54

🎯 Final Graph of the Models (Accuracy, Processing Speed (FPS), Total Time Taken To Process)


Project Final Results: DETR vs YOLOv11x (1512 Frames)
Processing Speed (Higher is Better)
Real-World FPS
4.13 FPS
Facebook DETR
YOLOv11x
Model
Avg Model Confidence (Certainty)
Avg Confidence (%)
Facebook DETR
YOLOv11x
Model
Total Time to Process 1512 Frames (Lower is Better)
Processing Time (s)
365.9 s
Facebook DETR
YOLOv11x
Model
Detection Sensitivity by Object Class
Count
Car
Person
Cell Phone
Backpack
Motorcycle
Class
Model
DETR
YOLOv11x