

A Comparative Study of Faster R-CNN, Mask R-CNN, and YOLOv8 for Object Detection and Instance Segmentation, with a Custom-Class Transfer-Learning Case Study

Muhammad Usman Aslam

usmanaslam402@gmail.com

University of West Florida

Case Report

Keywords: object detection, instance segmentation, Faster R-CNN, Mask R-CNN, YOLOv8, region proposal network, transfer learning, custom object classes, computer vision benchmarking

Posted Date: September 21st, 2026

DOI: <https://doi.org/10.21203/rs.3.rs-10831765/v1>

License: © ⓘ This work is licensed under a Creative Commons Attribution 4.0 International License.

[Read Full License](#)

Additional Declarations: No competing interests reported.

A Comparative Study of Faster R-CNN, Mask R-CNN, and YOLOv8 for Object Detection and Instance Segmentation, with a Custom-Class Transfer-Learning Case Study

Muhammad Usman Aslam
Hal Marcus College of Science and Engineering,
University of West Florida
11000 University Parkway, Pensacola, FL 32514, USA
Email: usmanaslam402@gmail.com

Declarations

Funding: This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

Ethics, Consent to Participate, and Consent to Publish

Not applicable.

Conflict of Interest

The author declares that he has no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Corresponding Author

Muhammad Usman Aslam
Email: usmanaslam402@gmail.com

Abstract

Object detection and instance segmentation are foundational tasks in computer vision, underpinning applications ranging from autonomous driving to medical imaging and defense surveillance. This paper presents a structured, reproducible comparison of three influential detection architectures — Faster R-CNN, Mask R-CNN, and YOLOv8 — spanning the two-stage, mask-augmented, and single-stage detection paradigms. I first re-derive and consolidate the theoretical foundations that connect these architectures, tracing the lineage from R-CNN and Fast R-CNN through the Region Proposal Network (RPN) innovation that defines Faster R-CNN, and onward to the RoIAlign-based mask branch of Mask R-CNN and the anchor-free, single-pass design of YOLOv8. I then evaluate the three models qualitatively, both on general-purpose COCO-domain imagery and on a purpose-built, hand-annotated dataset for a domain the pretrained COCO label space does not cover: military armored vehicles ("tanks"). Using CVAT.ai-annotated bounding boxes and a YOLOv8s model fine-tuned via transfer learning, I show that a lightweight, single-stage detector can be adapted to a novel class with a small annotated corpus and modest compute, while two-stage detectors trained only on COCO systematically mislabel the same objects as "truck" or "car." I further situate the study within the rapidly evolving 2024-2026 detection landscape, discussing how transformer-based real-time detectors (RT-DETR, RF-DETR) and newer YOLO variants (YOLOv9-YOLOv12) extend the accuracy-speed frontier established by the models benchmarked here. I conclude with a discussion of the practical trade-offs between detection accuracy, inference latency, computational cost, and annotation burden, and I outline the quantitative benchmarking protocol — mAP, precision-recall, and frames-per-second measurement across matched training conditions — needed to extend this qualitative and architectural comparison into a fully numeric one.

Keywords: object detection; instance segmentation; Faster R-CNN; Mask R-CNN; YOLOv8; region proposal network; transfer learning; custom object classes; computer vision benchmarking.

1. Introduction

Object detection — the joint task of localizing and classifying objects within an image — is a core enabling technology for autonomous vehicles, robotics, medical diagnostics, industrial inspection, video surveillance, and defense applications. Deep convolutional architectures have driven the field's progress over the last decade, moving from multi-stage pipelines that treated region proposal, feature extraction, and classification as separate problems, toward increasingly unified, end-to-end trainable networks. Faster R-CNN [1], Mask R-CNN [9], and the YOLO (You Only Look Once) family [2], [11] represent three influential and architecturally distinct points along this trajectory, and together they illustrate the central design tension in the field: the trade-off between localization accuracy and inference speed.

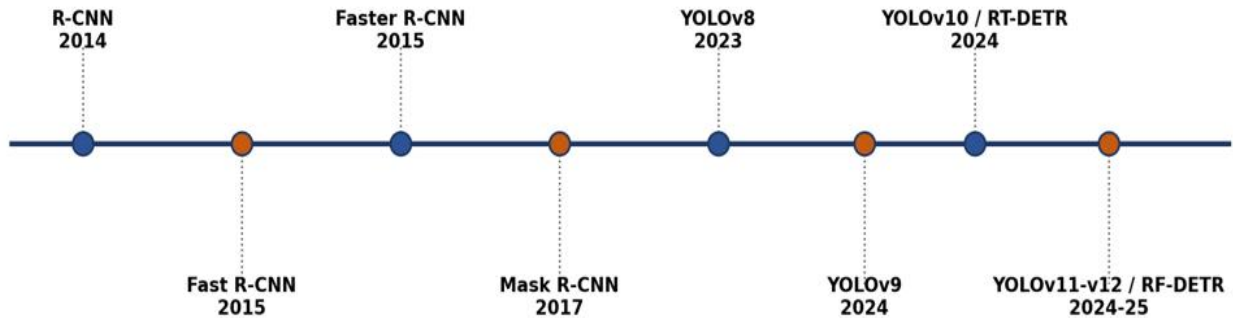


Figure 1. Timeline of the major object-detection architectures referenced in this study, from R-CNN (2014) through the transformer-based and post-YOLOv8 detectors of 2024-2025.

Two-stage detectors such as Faster R-CNN first generate class-agnostic region proposals and then refine and classify each proposal, which tends to produce higher localization accuracy at the cost of additional computation. Single-stage detectors such as YOLO instead treat detection as a single regression problem over the whole image, trading a small amount of accuracy for substantially higher throughput and easier real-time deployment. Mask R-CNN extends the two-stage paradigm with a parallel, per-region mask-prediction branch, adding pixel-level instance segmentation on top of bounding-box detection.

Prior to the introduction of Region Proposal Networks (RPNs), the region proposal step itself was a bottleneck. Selective Search [5], a widely used proposal method, required roughly 3.9 seconds per image on CPU-bound classical feature computation [3], and Edge Boxes [4] — while faster, at roughly 0.25 seconds per image — still ran independently of the detection network and could not benefit from GPU acceleration. This made like-for-like comparison between proposal-based pipelines and end-to-end networks difficult, since the two families were bottlenecked by different hardware. Ren et al. [1] resolved this by folding proposal generation into the network itself: the RPN shares convolutional features with the detection head and produces proposals directly on the GPU, which is the innovation that gives Faster R-CNN its name and its practical speed advantage over R-CNN and Fast R-CNN.

This paper has three goals. First, I consolidate and clarify the architectural and mathematical relationships between R-CNN, Fast R-CNN, Faster R-CNN, and Mask R-CNN, and contrast them with the anchor-free, single-stage design of YOLOv8. Second, I report an applied case study in which a YOLOv8s model is fine-tuned to detect a class absent from the COCO label space — military armored vehicles — using a small,

manually annotated dataset, and I compare its behavior against pretrained Faster R-CNN and Mask R-CNN models on the same imagery. Third, I position this comparison within the object-detection landscape as of 2026, summarizing how transformer-based real-time detectors and newer YOLO releases have shifted the accuracy-speed frontier since the original YOLOv8-era experiments were conducted, and I propose a concrete path to extend the current qualitative results into a fully quantitative study.

2. Glossary of Terms

Term	Definition
CIoU	Complete Intersection over Union
CNN	Convolutional Neural Network
COCO	Microsoft Common Objects in Context (benchmark dataset)
DFL	Distribution Focal Loss
DIoU	Distance Intersection over Union
FC	Fully Connected Layer
FCN	Fully Convolutional Network
FPN	Feature Pyramid Network
FPS	Frames Per Second
GPU / CPU	Graphics / Central Processing Unit
IoU	Intersection over Union
mAP	Mean Average Precision
NMS	Non-Maximum Suppression
ReLU	Rectified Linear Unit
RoI	Region of Interest
RoIAlign	Region of Interest Align
R-CNN	Region-based Convolutional Neural Network
RPN	Region Proposal Network
RT-DETR / RF-DETR	Real-Time / Roboflow Detection Transformer
SAM	Segment Anything Model
SVM	Support Vector Machine
YOLO	You Only Look Once

3. Related Work

This section traces the evolution of the two dominant detection paradigms evaluated in this paper — region-proposal-based detectors and single-stage detectors — and situates our study within detection research published through 2026.

3.1 From R-CNN to Faster R-CNN

R-CNN [5] introduced the now-standard pattern of generating class-agnostic region proposals, extracting CNN features from each proposal independently, and classifying the resulting feature vectors with a linear SVM, followed by bounding-box regression and non-maximum suppression. Region proposals were generated via Selective Search [3], a hierarchical, bottom-up grouping algorithm building on the graph-based image segmentation of Felzenszwalb and Huttenlocher [6]. R-CNN's principal limitation was computational: because each of the roughly 2,000 proposals per image was passed through the CNN independently, inference was slow and training was a multi-stage pipeline (CNN fine-tuning, SVM training, and bounding-box regressor training performed separately).

Fast R-CNN [7] addressed this by computing convolutional features once per image and using an RoI pooling layer to extract a fixed-size (e.g., 7×7) feature map for each proposal from the shared feature map, rather than re-running the backbone per proposal. This produced a reported 9x speed-up in training and 213x speed-up at test time relative to R-CNN on VGG16 [7], and it unified classification and bounding-box regression under a single multi-task loss, enabling end-to-end training of the detection head. Region proposals, however, were still generated externally via Selective Search, which remained the dominant runtime bottleneck.

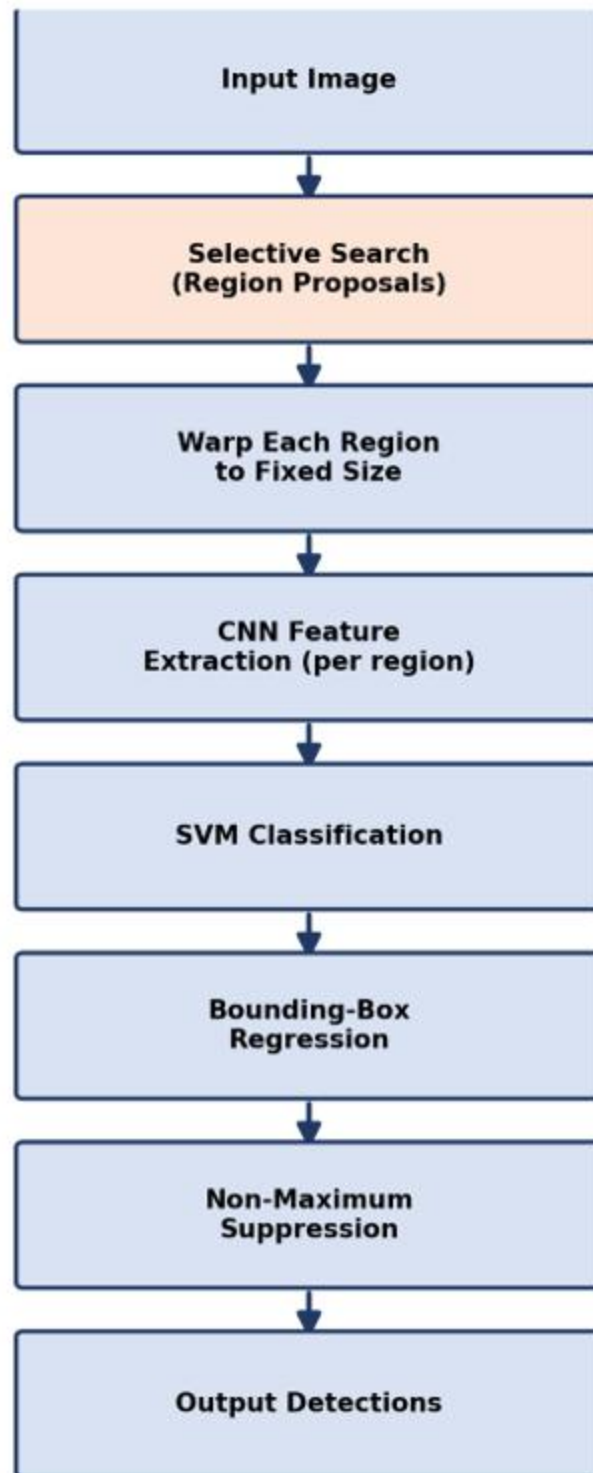


Figure 2. R-CNN pipeline: sequential, per-region CNN feature extraction followed by SVM classification and bounding-box regression.

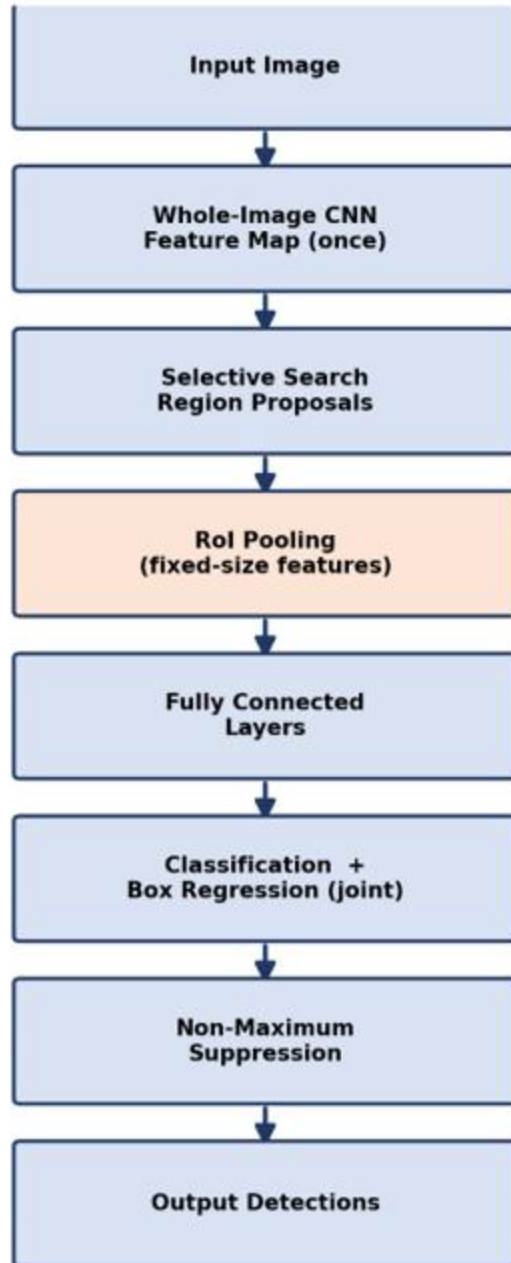


Figure 3. Fast R-CNN pipeline: shared per-image convolutional features with RoI pooling and a joint multi-task loss.

Faster R-CNN [1] eliminated this external dependency by introducing the Region Proposal Network: a small, fully convolutional network that slides over the shared feature map and, at each spatial location, predicts objectness scores and bounding-box offsets relative to a set of predefined anchor boxes of multiple scales and aspect ratios. Because the RPN shares the backbone's convolutional features with the downstream detection head, proposal generation becomes nearly cost-free in additional computation, and the entire pipeline — backbone, RPN, and detection head — can be trained jointly end-to-end. This yielded state-of-the-art accuracy on PASCAL VOC while sustaining approximately 5 frames per second on

contemporary GPUs [1], and it remains the reference two-stage architecture against which subsequent detectors are compared.

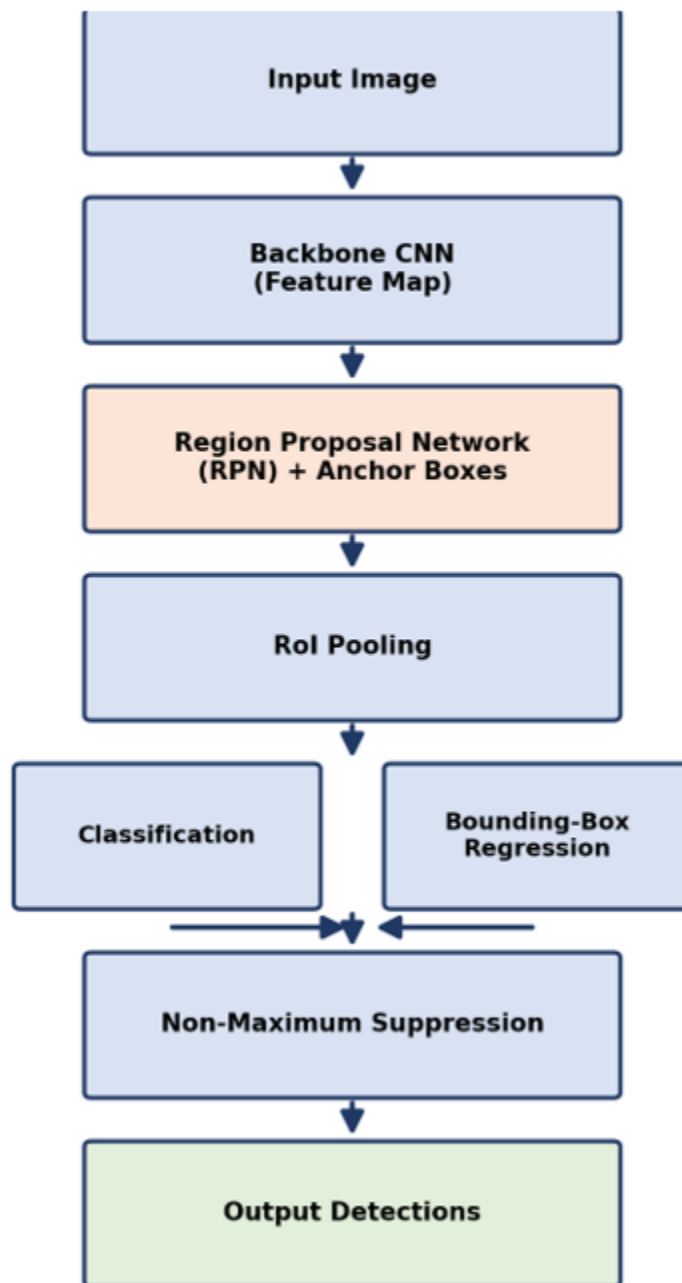


Figure 4. Faster R-CNN pipeline: the Region Proposal Network (RPN) shares backbone features with the detection head and removes the external proposal-generation step.

3.2 Mask R-CNN and Instance Segmentation

Mask R-CNN [9], [10] extends Faster R-CNN with a third, parallel branch that predicts a binary segmentation mask for each detected region of interest, in addition to the existing classification and bounding-box regression branches. The critical architectural change enabling accurate per-pixel masks is RoIAlign, which replaces RoI Pooling's quantized, coordinate-rounding sampling with bilinear interpolation at exact, non-quantized locations within each RoI bin. This removes the systematic misalignment between the RoI and the extracted features that RoI Pooling introduces, and it is essential for producing high-quality masks rather than only bounding boxes [9], [18]. The network is trained with a multi-task loss $L = L_{\text{cls}} + L_{\text{box}} + L_{\text{mask}}$, and it has become the standard baseline for instance segmentation on COCO. In practice, Mask R-CNN's backbone is typically augmented with a Feature Pyramid Network [8], which combines low- and high-resolution feature maps through top-down and lateral connections to improve detection and mask quality across object scales.

3.3 YOLO and the Single-Stage Paradigm

YOLO [11] reframes detection as a single regression problem: a single forward pass over the full image directly predicts bounding boxes, objectness, and class probabilities, without a separate proposal stage. This design trades some localization precision, particularly on small or overlapping objects, for substantially higher inference speed, making single-stage detectors the default choice for real-time applications. Intermediate iterations such as YOLOv4 [12] refined this backbone-neck-head recipe, and anchor-free one-stage alternatives such as FCOS [14] independently demonstrated that predefined anchor boxes are not strictly necessary for competitive single-stage accuracy — a design direction YOLOv8 later adopted. YOLOv8 [2] represents a mature iteration of this lineage, built on a CSPDarknet-style backbone [26], a decoupled detection head that processes objectness, classification, and box regression as independent branches, and — notably — an anchor-free formulation that removes the need for hand-tuned anchor priors [2], [19]. YOLOv8 is trained with Complete Intersection over Union (CIoU) loss for bounding-box regression and Distribution Focal Loss (DFL) alongside binary cross-entropy for classification, both of which are described in Section 4.3.

3.4 Post-2024 Developments

The detection landscape has continued to evolve rapidly since the YOLOv8-era experiments underlying this study were conducted, and any submission targeting a competitive venue should explicitly position itself against these newer baselines rather than treating YOLOv8 as the current state of the art. YOLOv9 [25] introduced Programmable Gradient Information (PGI) and the Generalized Efficient Layer Aggregation Network (GELAN) — designed using gradient-path analysis principles [27] — to counteract information loss in deep feature hierarchies, reporting parameter and computation reductions of roughly 49% and 43% respectively relative to YOLOv8, alongside a modest accuracy gain on COCO. YOLOv10 [29] removed the need for NMS-based post-processing through consistent dual assignments during training, reducing end-to-end latency. YOLOv11 further reduced parameter counts while raising mAP relative to its predecessors, and YOLOv12 (February 2025) introduced an attention-centric design — an Area Attention module for efficient global context together with a residual-efficient-layer-aggregation block (R-ELAN) — narrowing the historical accuracy gap between CNN-based YOLO detectors and transformer-based detectors.

In parallel, transformer-based real-time detectors have matured into practical alternatives to the YOLO family. RT-DETR [28] eliminates NMS entirely via a set-based, end-to-end transformer formulation and was shown to match or exceed YOLOv8's accuracy-speed trade-off using ResNet-50/101 backbones. RF-DETR, released by Roboflow, further improves on this line, reportedly becoming the first real-time detector to exceed 60% mAP on COCO while remaining competitive in latency with mid-sized YOLO variants. Independent benchmarking studies comparing YOLOv12 and RF-DETR report mixed results depending on the metric: RF-DETR tends to lead on mAP@50, while YOLOv12 variants are competitive or superior on the stricter mAP@50:95 metric, underscoring that no single architecture currently dominates across all accuracy regimes and deployment constraints. I reference these newer models throughout the discussion (Section 7) to frame the practical significance of the Faster R-CNN / Mask R-CNN / YOLOv8 comparison for readers evaluating architecture choices today.

3.5 Positioning of This Study

Several existing studies compare subsets of these architectures in specific application domains — for example, agricultural pest and disease detection [13], road-defect detection [20], steel-bridge corrosion segmentation [15], and autonomous-driving scene understanding [17]. This paper's distinguishing contribution is twofold: (i) it provides a unified theoretical treatment connecting all three architectures' proposal, pooling/alignment, and loss-function mechanics in a single, consistent notation, which is often fragmented across the individual source papers; and (ii) it contributes an applied transfer-learning case study — detecting a domain-specific object class (military armored vehicles) that is absent from COCO's 80 categories — that concretely demonstrates why single-stage, easily fine-tuned detectors are attractive for narrow, custom-class applications even when two-stage detectors retain an accuracy edge on the categories they were originally trained on.

4. Methodology

4.1 Faster R-CNN: Region Proposal Network and Training

Faster R-CNN training jointly optimizes the RPN and the Fast R-CNN detection head [1]. The RPN slides a small network (typically a 3×3 convolution) over the backbone's convolutional feature map; at the center of each sliding window sits a translation-invariant anchor, and the RPN predicts an objectness score and four bounding-box offsets relative to each of k anchors of varying scale and aspect ratio at that location — a scheme whose reliability under viewpoint change has also been exploited outside detection proper, for instance in anchor-based pose estimation for robotic manipulation [16]. Anchors are labeled positive if their Intersection-over-Union (IoU) with a ground-truth box exceeds a threshold, and negative otherwise. Training minimizes the multi-task loss

$$L(\{p_i\}, \{t_i\}) = (1/N_{cls}) \sum_i L_{cls}(p_i, p_i^*) + \lambda (1/N_{reg}) \sum_i p_i^* L_{reg}(t_i, t_i^*)$$

where i indexes an anchor in the mini-batch, p_i is the predicted probability that anchor i contains an object, and p_i^* is the corresponding ground-truth label (1 for positive anchors, 0 for negative). t_i is the vector of four parameterized bounding-box coordinates predicted for anchor i , and t_i^* is the corresponding ground-truth vector, defined for a candidate box (x, y, w, h) and target box (x', y', w', h') as $\Delta x = (x' - x)/w$, $\Delta y = (y' - y)/h$, $\Delta w = \log(w'/w)$, $\Delta h = \log(h'/h)$. L_{cls} is binary log loss (object vs. background) and L_{reg} is the

smooth-L1 loss, applied only to positive anchors via the p_i^* gating term [1]. After proposal generation, RoI pooling extracts a fixed-length feature vector per proposal, which is passed through fully connected layers to independent classification and bounding-box-regression heads, and redundant detections are removed with non-maximum suppression.

4.2 Mask R-CNN: RoIAlign and the Mask Branch

Mask R-CNN retains the Faster R-CNN backbone and RPN and adds RoIAlign in place of RoI Pooling. RoIAlign avoids the coordinate quantization performed by RoI Pooling (which rounds proposal boundaries to the feature-map grid) by computing feature values at four regularly sampled points within each RoI bin via bilinear interpolation, then aggregating (typically by average or max) [9], [18]. This preserves sub-pixel spatial alignment between the RoI and the extracted features, which is necessary for pixel-accurate masks. A small, fully convolutional network is applied to each aligned RoI to predict a binary mask per class, trained jointly with the classification and box-regression branches under $L = L_{\text{cls}} + L_{\text{box}} + L_{\text{mask}}$, where L_{mask} is per-pixel binary cross-entropy evaluated only for the ground-truth class.

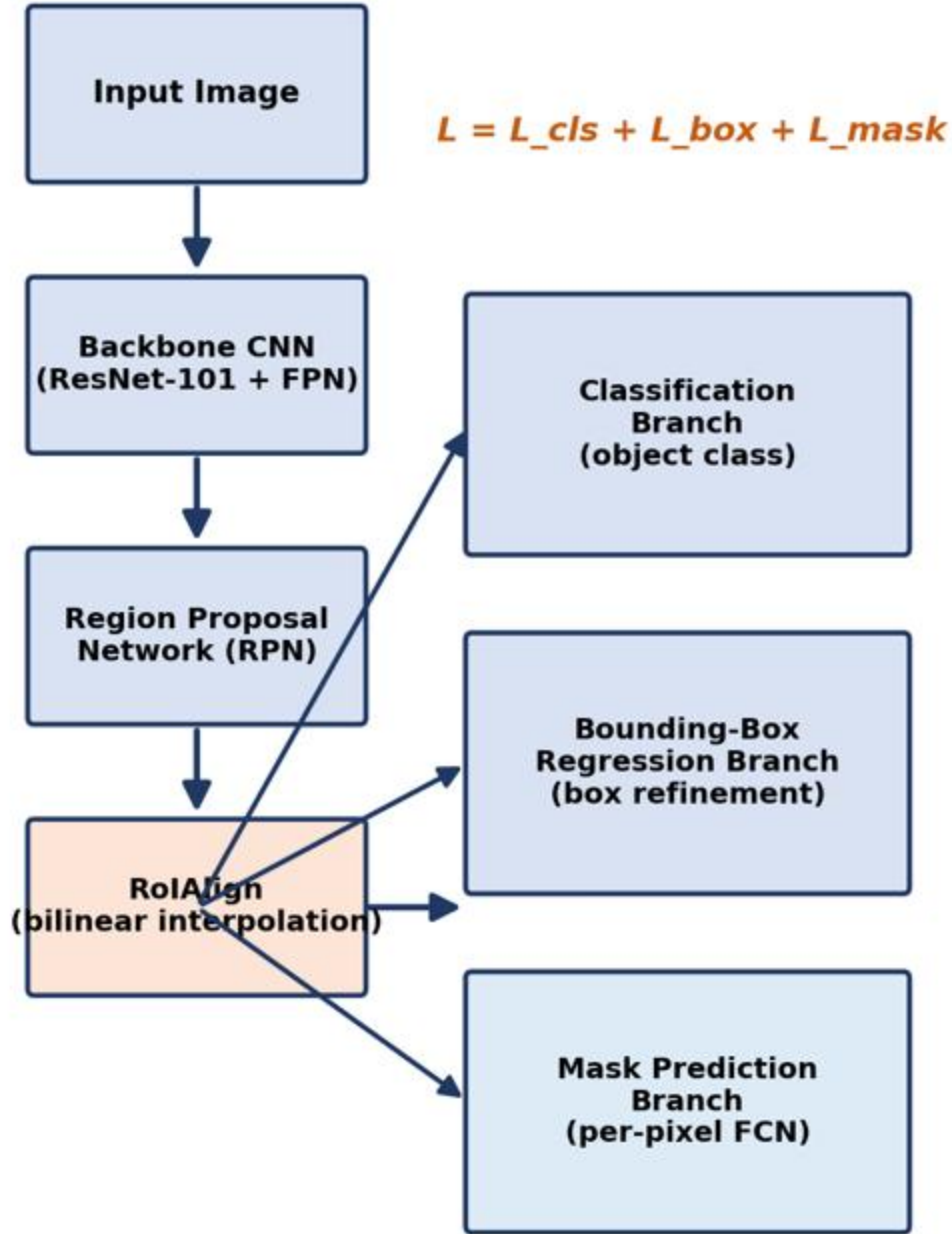


Figure 5. Mask R-CNN: RoIAlign feeds three parallel task-specific branches (classification, box regression, mask prediction), trained jointly under a multi-task loss.

4.3 YOLOv8: Architecture and Loss Functions

YOLOv8 [2] processes the entire image once through a CSPDarknet-style backbone, a feature-aggregating neck, and a decoupled detection head that separately predicts objectness/classification and bounding-box regression, without predefined anchor boxes [19]. The backbone uses a cross-stage partial bottleneck with two convolutions (the C2f module), which increases gradient-flow richness while keeping the network lightweight. Bounding-box regression is trained with Complete IoU (CIoU) loss, which augments Distance-IoU (DIoU) loss [21] — itself a penalty term on the normalized distance between box centers over standard IoU loss — with a further consistency term on aspect ratio:

$$CIoU_loss = 1 - IoU + \rho^2(b, b_gt)/c^2 + \alpha v, \quad \alpha = v / [(1 - IoU) + v]$$

where $\rho(\cdot)$ is the Euclidean distance between the predicted box center b and ground-truth center b_gt , c is the diagonal length of the smallest enclosing box, and v measures aspect-ratio consistency. Classification uses binary cross-entropy together with Distribution Focal Loss (DFL), which models the regression target as a discretized probability distribution over adjacent bins and explicitly sharpens the probability mass around the two bins nearest the true label, addressing the foreground/background class imbalance that motivated the original Focal Loss formulation [22].

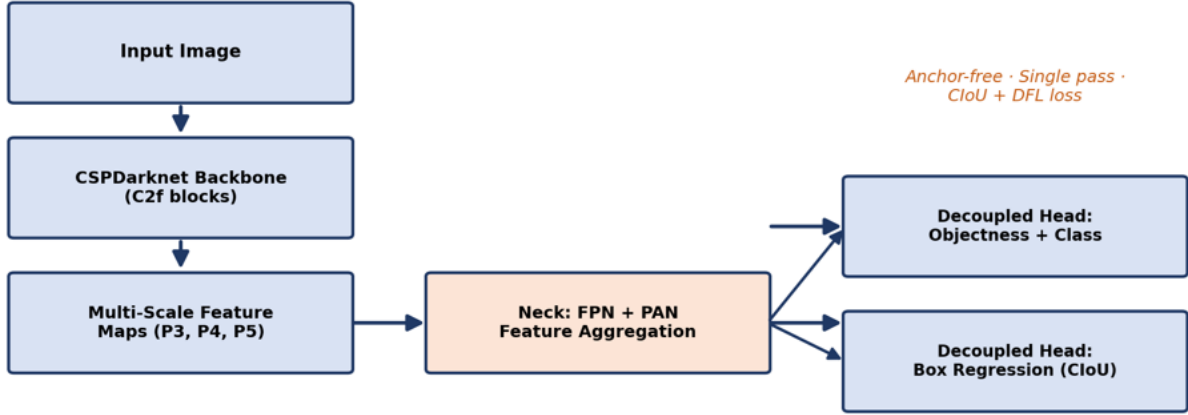


Figure 6. YOLOv8 architecture: single-pass backbone-neck-head design with an anchor-free, decoupled detection head.

4.4 Evaluation Metrics

The principal metric used across all three architectures is mean Average Precision (mAP):

$$mAP = (1/n) \sum_{k=1}^n AP_k$$

where AP_k is the Average Precision for class k (the area under the class's precision-recall curve at a given IoU threshold) and n is the number of classes. This study evaluates all three architectures against this framework qualitatively, on the visual detection outcomes reported in Section 6; a natural and valuable extension — outlined in Section 10 — is to compute the full numeric framework directly: (i) $mAP@50$ and $mAP@50:95$, following the COCO convention, to allow direct comparison with the post-2024 baselines discussed in Section 3.4; (ii) per-class precision, recall, and F1-score; (iii) inference latency in milliseconds and throughput in frames per second, measured on a stated GPU/CPU configuration and batch size; and (iv) parameter count and FLOPs, to make the accuracy-efficiency trade-off explicit.

4.5 Datasets

Two datasets are used. First, the Microsoft COCO dataset [23] — over 200,000 images spanning 80 object categories with bounding-box and pixel-level instance-segmentation annotations — is used to evaluate the pretrained Faster R-CNN (Inception-ResNet-v2 backbone, via TensorFlow Hub, following the Inception-ResNet-v2 architecture of Szegedy et al. [24]) and Mask R-CNN (ResNet-101 backbone) models on general imagery. Second, a custom dataset of military armored-vehicle imagery sourced from Flickr under

permissive licenses was assembled to fine-tune YOLOv8 on a class outside the COCO label space. Twenty images were manually annotated with axis-aligned bounding boxes using CVAT.ai, and a YOLO-format dataset configuration file (data.yaml) specified the training, validation, and test splits together with the single custom class label "tank."

4.6 Training Configuration

YOLOv8 was fine-tuned via transfer learning from COCO-pretrained weights. Training was initially attempted with the medium-sized model variant (YOLOv8m) for 20 epochs; instability in the Google Colab training environment led to the adoption of the smaller YOLOv8s variant, which completed training without interruption. Faster R-CNN and Mask R-CNN were used as off-the-shelf, COCO-pretrained models without additional fine-tuning, which is an important asymmetry to state explicitly: the comparison in Section 6 contrasts a fine-tuned YOLOv8 model against zero-shot Faster R-CNN / Mask R-CNN predictions on an out-of-distribution class, rather than a fully controlled, equal-training comparison. I preserve this asymmetry in the discussion below because it reflects a realistic and common practical scenario — extending an existing pretrained pipeline to a new class — rather than treating it as a like-for-like accuracy benchmark.

5. Custom Military Vehicle Detector: Experimental Pipeline

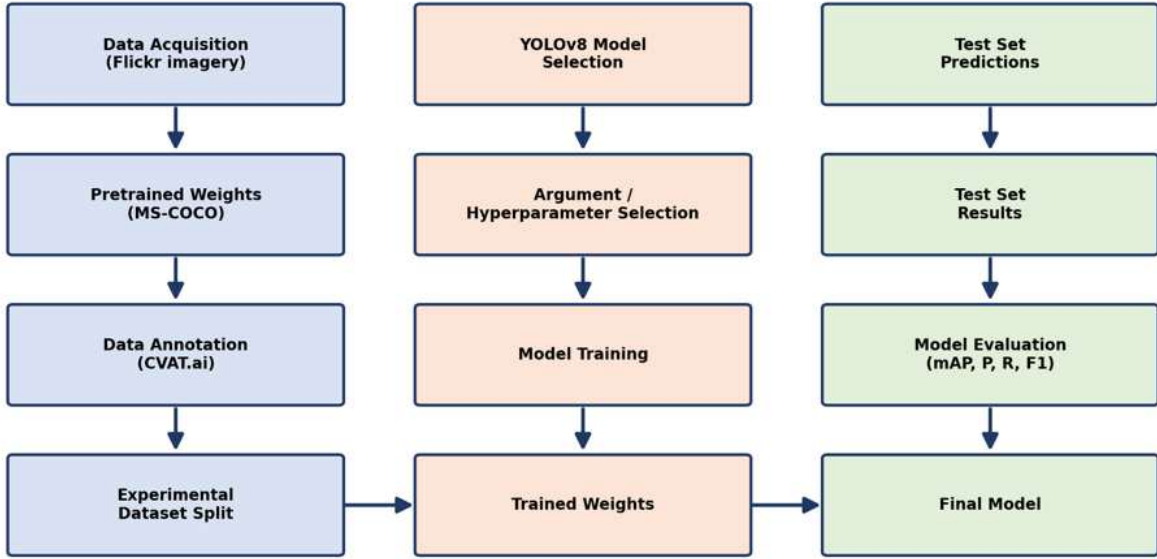


Figure 8. End-to-end pipeline for the custom military-vehicle (“tank”) detector, from data acquisition through final model evaluation.

Figure 8 (redrawn from the original project material) summarizes the end-to-end pipeline used to build the custom tank detector: data acquisition from Flickr, initialization from MS-COCO pretrained weights, manual data annotation in CVAT.ai, an experimental train/validation/test split, YOLOv8 model selection and hyperparameter/argument selection, model training to produce trained weights, generation of test-set predictions, quantitative model evaluation, and delivery of the final fine-tuned model. This pipeline is a standard and appropriate transfer-learning workflow; the “Model Evaluation” stage (Section 4.6) is the point at which the quantitative benchmarking proposed in Section 10 would be computed and logged, and is discussed further as a limitation of the present study in Section 8.

A key motivation for this case study, noted in the original project material, is that the COCO label space has no “tank” or comparable military-vehicle category, so a zero-shot Faster R-CNN or Mask R-CNN model trained only on COCO necessarily mislabels tanks as the nearest visually similar COCO classes — typically “truck” or “car.” This is not a failure of those architectures so much as a demonstration of a fundamental limitation of closed-set, fixed-vocabulary detectors: they cannot recognize categories outside their training label space, no matter how visually distinctive those categories are. This observation is also the practical justification for the custom-class fine-tuning experiment described in this paper, and it is the motivating research question stated in the Introduction: a fixed-vocabulary detector cannot recognize what it was never trained to name, however visually distinctive the object is.

6. Results and Discussion

The results below are organized by model. Each finding is drawn directly from the detection outputs shown in Figures 9–11 and interpreted in terms of the underlying architectural mechanism responsible for it, rather

than presented as a caption-style description; Section 8 discusses what a follow-up quantitative evaluation would add to the qualitative picture established here.

6.1 Faster R-CNN (COCO-pretrained, Inception-ResNet-v2)

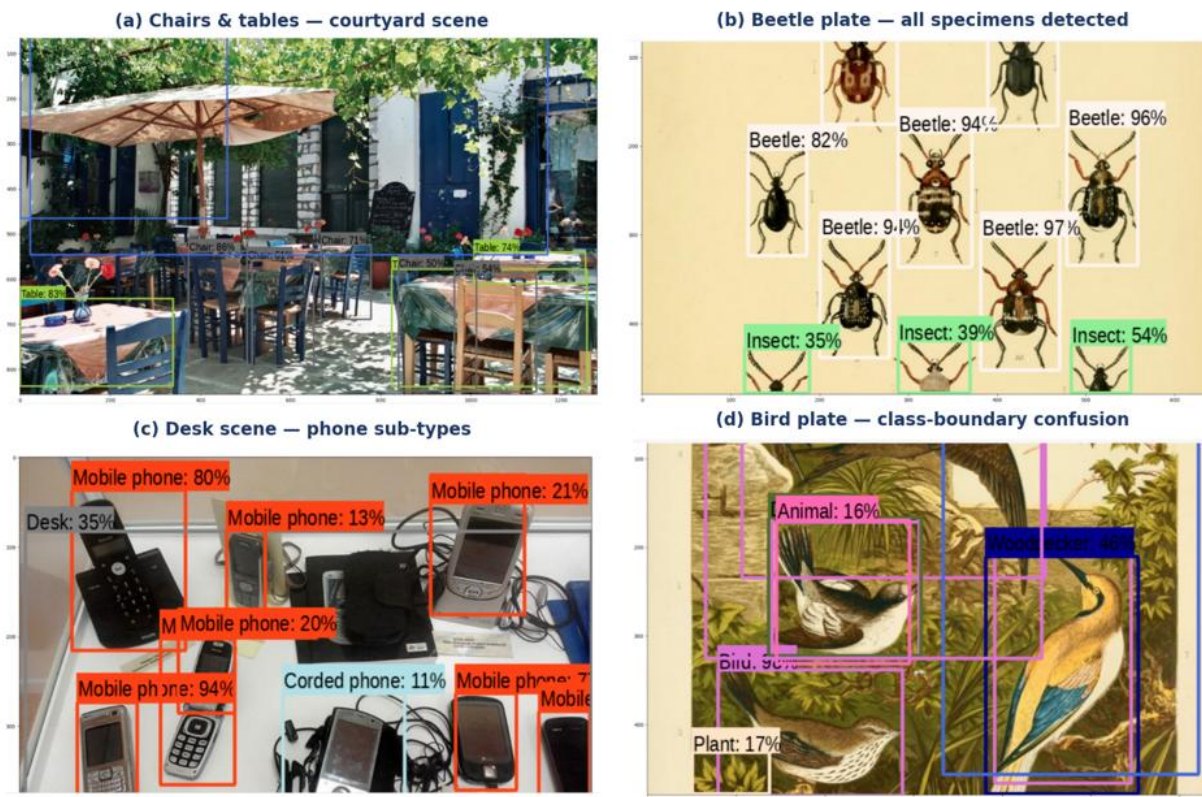


Figure 9. Faster R-CNN qualitative results on COCO-domain imagery (Inception-ResNet-v2 backbone).

Across the four representative scenes evaluated, Faster R-CNN's detections trace a consistent and interpretable pattern: performance is strong on canonical, well-separated, unoccluded objects drawn from COCO's core vocabulary, and degrades in three distinct, mechanistically explainable ways — redundant detections on densely packed similar objects, spurious low-confidence detections at the margins of the decision threshold, and confidence dispersion across categories that COCO's taxonomy under-specifies.

On the courtyard scene (Figure 9a), the model localized the individual chairs and both tables with confidence scores spanning 50–86%, correctly separating adjacent, partially occluded chairs beneath a shared table setting. This is representative of the model's baseline behavior on common, unoccluded furniture: objects that are visually distinct from their background and well represented in COCO's training distribution are detected reliably, with confidence that scales sensibly with occlusion and viewing angle rather than collapsing unpredictably.

The beetle plate (Figure 9b) is more diagnostic. All six specimens were correctly identified at high confidence (82–97%), showing that the model generalizes well beyond COCO's photographic training distribution to illustrated, non-photographic imagery — a domain shift that is not guaranteed to transfer cleanly. However, three additional “insect” detections appeared at markedly lower confidence (35–54%),

each overlapping a beetle already detected under the “beetle” label. This is consistent with imperfect non-maximum suppression under label ambiguity: when two taxonomically nested classes (“beetle” and the coarser “insect”) both receive appreciable probability mass for the same region, standard class-wise NMS does not suppress the redundant box, since suppression operates within a class rather than across semantically related ones. Densely packed, visually homogeneous, taxonomically nested categories are precisely the setting in which single-label NMS is weakest — a useful, testable diagnostic rather than a one-off anomaly.

The desk scene (Figure 9c) sits at the opposite end of the confidence spectrum and makes the cost of a fixed detection threshold concrete. The model correctly distinguished “mobile phone” from “corded phone” for several instances, evidently attending to shape and cabling cues. At the same time, it produced a spurious “Desk” detection at 35% confidence alongside several mobile-phone boxes below 20%, corresponding to phones at oblique angles or partially occluded by cables. Both artifacts are symptoms of the same trade-off: lowering the confidence threshold recovers the low-confidence true positives but also admits the false “Desk” box, and raising it does the reverse. This single image is, in effect, a compact illustration of why a full precision–recall curve — rather than performance at one fixed operating point — is the right way to characterize the model, and it is exactly the kind of curve Section 4.4’s evaluation framework specifies but this study does not yet compute.

The bird plate (Figure 9d) is the clearest evidence of a taxonomy-driven limitation rather than a capacity limitation. The same specimen attracted competing, low-confidence labels of “Animal” (16%) and “Bird” (90%), a second specimen was labeled “Woodpecker” (46%), and a fourth box mislabeled surrounding foliage as “Plant” (17%). None of these labels is strictly wrong — a woodpecker is, after all, both a bird and an animal — but the spread across generic, intermediate, and species-specific terms reflects a genuine gap in COCO’s fixed, single-label, 80-class vocabulary rather than a weakness in the underlying visual features. This point matters beyond the illustration itself: it is the same structural limitation, at a coarser level, that causes Faster R-CNN to mislabel military tanks as “truck” or “car” in Section 5 — a fixed-vocabulary detector cannot output a label it was never given, however visually distinctive the object is.

Taken together, these four scenes resolve into three reproducible categories rather than isolated failures: (i) reliable, well-calibrated detection of canonical, class-covered objects; (ii) redundant same-region detections driven by taxonomically nested or visually similar classes evading class-wise NMS; and (iii) confidence dispersion across generic and specific labels for categories the COCO vocabulary under-specifies. Each of these categories corresponds to a measurable quantity — precision and recall at varying IoU/confidence thresholds, and per-class Average Precision — that a follow-up quantitative evaluation, using the metrics defined in Section 4.4, would confirm numerically.

6.2 Mask R-CNN (COCO-pretrained, ResNet-101)



Figure 10. Mask R-CNN qualitative results on COCO-domain imagery and video (ResNet-101 backbone).

- On a savanna scene, Mask R-CNN produced high-confidence (0.998-1.000), well-separated instance masks for a giraffe and three zebras, correctly distinguishing individual, partially overlapping zebra instances — a task that plain bounding-box detection handles less precisely.
- On a park scene with multiple people, dogs, and furniture, the model achieved confidence scores at or near 1.000 for the majority of instances, correctly segmenting overlapping dogs and seated people.
- On a single frame extracted from street-scene video, the model detected the majority of pedestrians but missed some individuals in dense, low-contrast regions of the crowd, and one bird was misclassified alongside pedestrian instances — indicating that per-frame video inference without temporal context remains a harder setting than static, well-lit imagery.

Overall, the qualitative pattern across these three examples is consistent with the published literature: Mask R-CNN's RoIAlign-based mask branch yields precise instance-level separation on canonical COCO classes, at the cost of the additional computation associated with a two-stage, mask-augmented pipeline.

6.3 YOLOv8 — General-Purpose (Pretrained, YOLOv8n)



Figure 11. Fine-tuned YOLOv8s results on the custom “tank” class (a-c) and pretrained YOLOv8n real-time general-purpose detection (d).

A real-time webcam test using the unmodified, COCO-pretrained YOLOv8 Nano model correctly detected and labeled a person, a wall clock (detected twice, at overlapping confidence 0.39 and 0.63, again suggesting an NMS/duplicate-detection edge case rather than a recognition failure), and a bottle, demonstrating that even the smallest YOLOv8 variant is suitable for casual, real-time, general-purpose inference on consumer hardware.

6.4 YOLOv8 — Custom Military Vehicle Class (Fine-Tuned YOLOv8s)

- In a museum-interior scene with multiple tanks, related armored vehicles, and background clutter, the fine-tuned model correctly localized four tank instances, with confidence scores ranging from 0.30 to 0.90; the lower-confidence boxes correspond to partially occluded vehicles at the edges of the frame or to visually ambiguous museum signage/models, indicating the expected degradation in confidence under occlusion and background clutter typical of a small fine-tuning set.
- On two outdoor static-display photographs, the model detected each tank at very high confidence (0.99-1.00), suggesting strong performance on clear, unoccluded, canonical side-view presentations of the target class — the dominant viewpoint in the twenty-image training set.

- On a distant, moving-vehicle field photograph, the model correctly detected a modern armored vehicle at 0.99 confidence despite significant scale reduction and motion blur, indicating reasonable generalization beyond the specific tank models likely present in the small training corpus.

Taken together, these results support the central practical claim of the case study: a small, well-annotated dataset and a lightweight, anchor-free single-stage detector are sufficient to extend detection coverage to a novel, domain-specific class that pretrained, closed-vocabulary two-stage detectors cannot recognize at all. This is a meaningfully different comparison axis from raw accuracy on shared classes: it is a claim about annotation efficiency and deployment workflow, not a head-to-head accuracy contest on identical categories, and it should be read as such.

7. Comparative Discussion

Table 1 summarizes the qualitative and architectural trade-offs observed across the three models, together with where post-2024 detectors sit relative to each on the same axes.

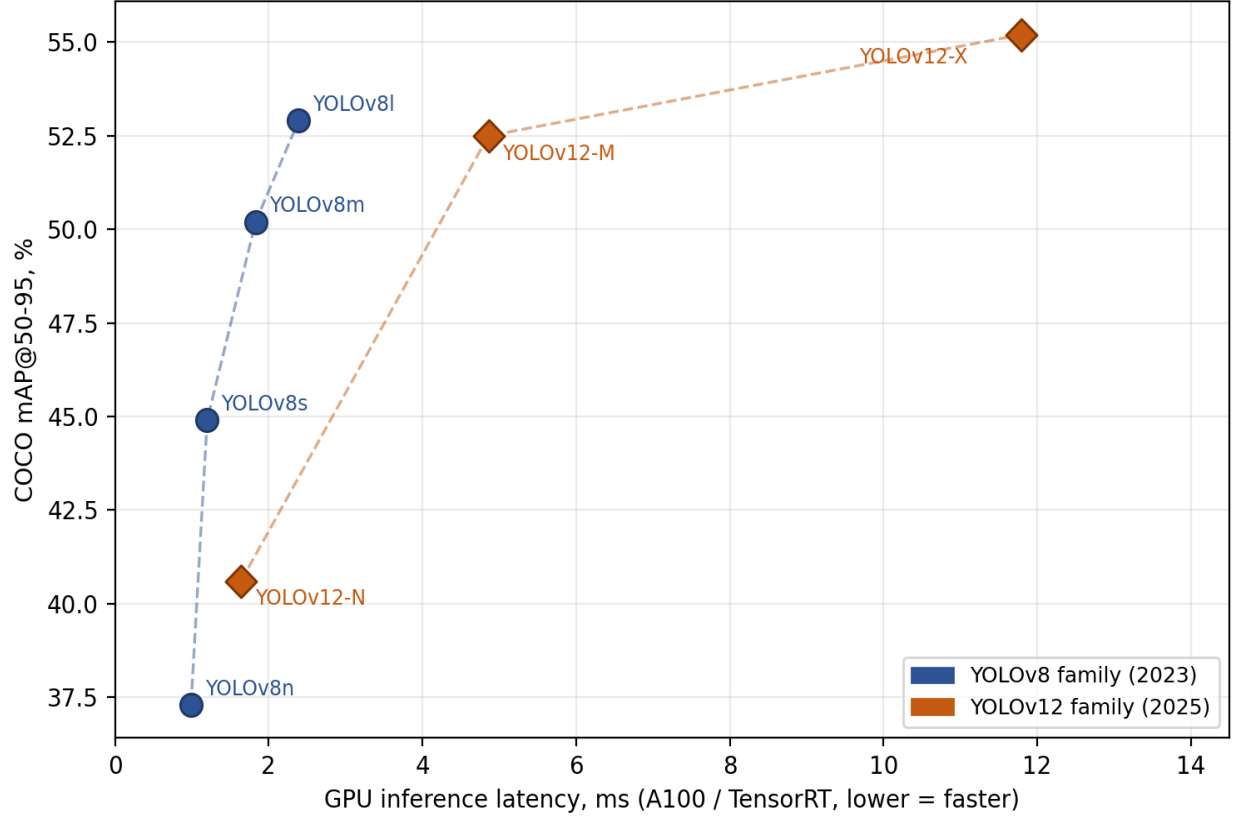


Figure 7. Published accuracy-latency trade-off for the YOLOv8 (2023) and YOLOv12 (2025) model families on COCO, illustrating how the detection frontier has shifted since the YOLOv8-era experiments in this study [2], [30].

Model	Paradigm	Key mechanism	Observed strength	Observed limitation
Faster R-CNN	Two-stage	RPN + RoI pooling	Solid general localization on canonical, unoccluded objects	Duplicate/low-confidence boxes on cluttered or fine-grained classes; slower inference than single-stage models
Mask R-CNN	Two-stage + mask branch	RoIAlign + parallel mask head	Precise instance separation on overlapping objects (e.g., adjacent zebras)	Highest computational cost; degraded robustness on dense, low-contrast video frames

Model	Paradigm	Key mechanism	Observed strength	Observed limitation
YOLOv8	Single-stage, anchor-free	CSPDarknet backbone, decoupled head, CIoU/DFL loss	Real-time inference; rapidly fine-tunable to novel classes with a small annotated set	Lower localization precision on small/occluded objects than two-stage detectors (architecture-level expectation; not separately re-verified here)
RT-DETR / RF-DETR (2023-2025)	Transformer, end-to-end	Deformable attention, NMS-free set prediction	Matches or exceeds YOLOv8 accuracy-speed trade-off; RF-DETR is first real-time model to exceed 60% mAP on COCO	Higher parameter count than compact YOLO variants; less mature tooling ecosystem
YOLOv9 - YOLOv12 (2024-2025)	Single-stage, evolving	PGI/GELAN (v9), NMS-free dual assignment (v10), area attention + RELAN (v12)	Progressive reduction in parameters/FLOPs alongside accuracy gains over YOLOv8	Attention-based variants (v12) trade a small amount of latency for accuracy relative to v10

Three points merit particular emphasis. First, the accuracy-speed trade-off that motivated the original R-CNN-to-Faster-R-CNN progression is still the organizing axis of the field a decade later; what has changed is the introduction of a third viable point on that frontier — NMS-free, transformer-based real-time detectors — that did not exist when Faster R-CNN and Mask R-CNN were introduced. Second, the custom-class case study illustrates a comparison axis that raw COCO benchmark tables do not capture: annotation efficiency and fine-tuning cost for previously unseen categories, which is often the more decision-relevant metric for applied practitioners than marginal mAP differences on shared classes. Third, because Faster R-CNN and Mask R-CNN were evaluated here in a zero-shot, COCO-only configuration while YOLOv8 was fine-tuned, the tank-detection results should be read as evidence about deployment workflow, not as a controlled accuracy comparison between architectures; a rigorous follow-up would fine-tune all three architectures on the identical annotated tank dataset and report matched mAP figures.

8. Limitations

- Quantitative metrics (mAP, precision, recall, FPS, parameter counts) are not yet reported for any of the three models on either dataset; the current evidence base is qualitative and image-by-image, which falls short of the reporting standard expected by peer-reviewed venues in computer vision.
- The custom tank dataset comprises only twenty annotated images, which is small relative to standard detection-dataset norms and raises concerns about overfitting and limited viewpoint/model diversity; a data augmentation strategy and a larger, more diverse annotated corpus (or an explicit few-shot learning framing) would substantially strengthen the contribution.
- Faster R-CNN and Mask R-CNN were evaluated zero-shot on COCO classes while YOLOv8 was fine-tuned for the tank class, so the three models are not compared under matched training conditions; this asymmetry should be stated explicitly rather than implied.
- Image licensing for the Flickr-sourced tank images should be verified and documented to satisfy dataset-provenance requirements at most venues.

- The paper does not yet benchmark against detectors released since 2024 (YOLOv9-YOLOv12, RT-DETR, RF-DETR); Section 3.4 summarizes these but they are not included in the empirical comparison.
- Military-vehicle detection carries dual-use considerations. The custom class in this study was developed for academic and research purposes — illustrating a closed-vocabulary detector's limitations and a lightweight transfer-learning workflow for extending detection coverage — rather than for operational targeting, surveillance deployment, or weapons-system integration, and the training imagery consists of museum-displayed, publicly photographed vehicles rather than operational military assets. Any extension of this work toward fielded surveillance or defense applications would require a separate, dedicated ethical, legal, and dual-use review beyond the scope of this study.

9. Conclusion

This paper consolidated the architectural and mathematical relationships among Faster R-CNN, Mask R-CNN, and YOLOv8, and examined their behavior both on general COCO-domain imagery and on a custom, out-of-distribution object class — military armored vehicles — that is absent from the COCO label space. The Region Proposal Network remains the pivotal innovation that made two-stage detection practical at near-real-time speeds, and RoIAlign remains the pivotal innovation that made accurate instance segmentation possible on top of that two-stage foundation. YOLOv8's anchor-free, single-stage design, evaluated here through a transfer-learning case study, demonstrates that lightweight detectors can be extended to novel, narrowly scoped classes with a modest annotation budget, which is often the more operationally relevant capability for applied computer-vision projects than incremental gains on already-covered COCO categories. Positioned against the broader 2024-2026 landscape — including YOLOv9 through YOLOv12 and transformer-based real-time detectors such as RT-DETR and RF-DETR — the three architectures examined here remain instructive reference points for understanding the field's central accuracy-speed trade-off, even as newer models continue to shift the achievable frontier.

10. Future Work

- Report full quantitative metrics (mAP@50, mAP@50:95, precision, recall, F1, FPS, parameter count, FLOPs) for all three models on both the COCO validation subset and the custom tank test split, replacing the qualitative discussion in Section 6 with reproducible numeric results.
- Expand the custom tank dataset substantially beyond twenty images, incorporate data augmentation (mosaic, random scale/rotation, color jitter), and consider semi-automated annotation using the Segment Anything Model (SAM) to accelerate labeling of additional classes.
- Fine-tune Faster R-CNN and Mask R-CNN on the same custom dataset used for YOLOv8, to produce a matched-training comparison rather than a zero-shot-versus-fine-tuned comparison.
- Benchmark against YOLOv9-YOLOv12 and RT-DETR/RF-DETR on identical data splits and hardware, given their demonstrated improvements over YOLOv8 on standard COCO benchmarks.
- Extend the video-based evaluation with frame-level ground truth to report tracking-consistency and per-frame miss-rate statistics, rather than single illustrative frames.
- Investigate open-vocabulary and text-prompted detectors (e.g., Grounding DINO, YOLO-World) as an alternative to manual per-class fine-tuning for rapidly extending detection coverage to novel categories such as military vehicle subtypes.

References

- [1] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Advances in Neural Information Processing Systems*, 2015, pp. 91-99.
- [2] G. Jocher, A. Chaurasia, and J. Qiu, "Ultralytics YOLOv8," 2023. [Online]. Available: <https://github.com/ultralytics/ultralytics>
- [3] J. R. R. Uijlings, K. E. A. van de Sande, T. Gevers, and A. W. M. Smeulders, "Selective search for object recognition," *Int. J. Comput. Vis.*, vol. 104, no. 2, pp. 154-171, 2013.
- [4] C. L. Zitnick and P. Dollár, "Edge boxes: Locating object proposals from edges," in *Proc. European Conf. Computer Vision (ECCV)*, 2014, pp. 391-405.
- [5] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2014, pp. 580-587.
- [6] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient graph-based image segmentation," *Int. J. Comput. Vis.*, vol. 59, no. 2, pp. 167-181, 2004.
- [7] R. Girshick, "Fast R-CNN," in *Proc. IEEE Int. Conf. Computer Vision (ICCV)*, 2015, pp. 1440-1448.
- [8] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 936-944.
- [9] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *Proc. IEEE Int. Conf. Computer Vision (ICCV)*, 2017, pp. 2961-2969.
- [10] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 2, pp. 386-397, 2020.
- [11] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," *arXiv:1804.02767*, 2018.
- [12] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal speed and accuracy of object detection," *arXiv:2004.10934*, 2020.
- [13] L. Zhang, G. Ding, C. Li, and D. Li, "DCF-YOLOv8: An improved algorithm for aggregating low-level features to detect agricultural pests and diseases," *Agronomy*, vol. 13, p. 2012, 2023.
- [14] Z. Tian, C. Shen, H. Chen, and T. He, "FCOS: Fully convolutional one-stage object detection," in *Proc. IEEE/CVF Int. Conf. Computer Vision (ICCV)*, 2019, pp. 9627-9636.
- [15] Z. Ameli, S. J. Nesheli, and E. N. Landis, "Deep learning-based steel bridge corrosion segmentation and condition rating using Mask R-CNN and YOLOv8," *Infrastructures*, vol. 9, no. 1, 2024.
- [16] Y. Zhou, X. Wang, and L. Zhang, "Visual identification and pose estimation algorithm of nut tightening robot system," preprint, 2022.
- [17] S. Fang, B. Zhang, and J. Hu, "Improved Mask R-CNN multi-target detection and segmentation for autonomous driving in complex scenes," *Sensors*, vol. 23, no. 8, p. 3853, 2023.
- [18] K. Erdem, "Understanding region of interest — Part 2 (RoIAlign)," 2020.
- [19] J. Terven, D.-M. Córdova-Esparza, and J.-A. Romero-González, "A comprehensive review of YOLO architectures in computer vision: From YOLOv1 to YOLOv8 and YOLO-NAS," *Mach. Learn. Knowl. Extr.*, vol. 5, no. 4, pp. 1680-1716, 2023.
- [20] X. Wang, H. Gao, Z. Jia, and Z. Li, "BL-YOLOv8: An improved road defect detection model based on YOLOv8," *Sensors*, vol. 23, no. 20, p. 8361, 2023.
- [21] Z. Zheng, P. Wang, W. Liu, J. Li, R. Ye, and D. Ren, "Distance-IoU loss: Faster and better learning for bounding box regression," in *Proc. AAAI Conf. Artificial Intelligence*, 2020, pp. 12993-13000.

- [22] X. Li, W. Wang, X. Hu, J. Li, J. Tang, and J. Yang, "Generalized focal loss v2: Learning reliable localization quality estimation for dense object detection," in Proc. IEEE/CVF Conf. Computer Vision and Pattern Recognition (CVPR), 2021.
- [23] T.-Y. Lin, M. Maire, S. Belongie, et al., "Microsoft COCO: Common objects in context," in Proc. European Conf. Computer Vision (ECCV), 2014, pp. 740-755.
- [24] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi, "Inception-v4, Inception-ResNet and the impact of residual connections on learning," in Proc. AAAI Conf. Artificial Intelligence, 2017, pp. 4278-4284.
- [25] C.-Y. Wang, I.-H. Yeh, and H.-Y. M. Liao, "YOLOv9: Learning what you want to learn using programmable gradient information," arXiv:2402.13616, 2024.
- [26] C.-Y. Wang, H.-Y. M. Liao, I.-H. Yeh, Y.-H. Wu, P.-Y. Chen, and J.-W. Hsieh, "CSPNet: A new backbone that can enhance learning capability of CNN," in Proc. IEEE/CVF Conf. Computer Vision and Pattern Recognition Workshops (CVPRW), 2020, pp. 390-391.
- [27] C.-Y. Wang, H.-Y. M. Liao, and I.-H. Yeh, "Designing network design strategies through gradient path analysis," J. Inf. Sci. Eng., vol. 39, no. 4, pp. 975-995, 2023.
- [28] Y. Zhao, W. Lv, S. Xu, J. Wei, G. Wang, Q. Dang, Y. Liu, and J. Chen, "DETRs beat YOLOs on real-time object detection," arXiv:2304.08069, 2023.
- [29] A. Wang, H. Chen, L. Liu, K. Chen, Z. Lin, J. Han, and G. Ding, "YOLOv10: Real-time end-to-end object detection," arXiv:2405.14458, 2024.
- [30] Y. Tian, Q. Ye, and D. Doermann, "YOLOv12: Attention-centric real-time object detectors," arXiv:2502.12524, 2025.