ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

NGHIÊN CỨU VÀ ĐÁNH GIÁ HIỆU NĂNG MÔ HÌNH RT-DETR TRONG BÀI TOÁN PHÁT HIỆN ĐỐI TƯỢNG THỜI GIAN THỰC

Nguyễn Huy Hoàn - 250101022
Lớp: CS2205.SEP2025

Thông tin Học Viên

Họ và tên: NGUYỄN HUY HOÀN

MSHV: 250101022

Email: hoannh.20@grad.uit.edu.vn

Lĩnh vực quan tâm:

- Computer Vision (Thị giác máy tính)
- Real-time Object Detection (Phát hiện đối tượng thời gian thực)
- Transformer Models in Vision

Kỹ năng & Công cụ:

- Python, PyTorch, Ultralytics
- Academic Research & Writing

Tài nguyên dự án:

- **GitHub:** https://github.com/hoannh-uitgrad/Research-Methodology
- **YouTube:** https://youtu.be/qCSqRuyEheQ

Giới thiệu

- **Bối cảnh:** Dòng YOLO thống trị mảng phát hiện đối tượng thời gian thực nhờ cân bằng tốc độ - chính xác.
- **Vấn đề (Problem):** YOLO bị phụ thuộc vào **NMS (Non-Maximum Suppression).**
    - NMS làm chậm tốc độ suy luận.
    - Tạo ra độ trễ không ổn định do phụ thuộc vào siêu tham số (Hyperparameters).
- **Hạn chế của giải pháp cũ:** Các mô hình DETR (Transformer) bỏ được NMS nhưng chi phí tính toán quá cao, không chạy được thời gian thực.
- **Giải pháp: RT-DETR** - Mô hình phát hiện đối tượng End-to-End thời gian thực đầu tiên giải quyết được cả 2 vấn đề trên.

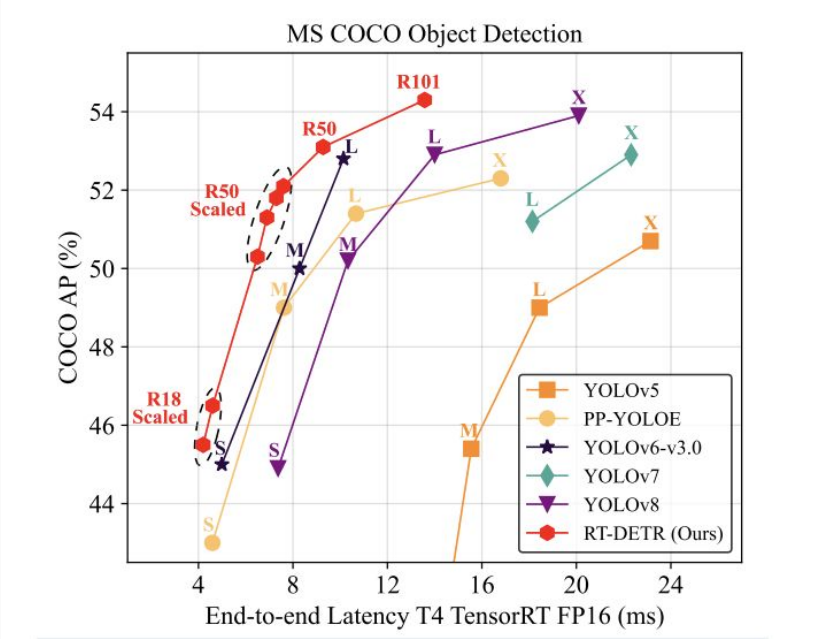


Figure 1 (Biểu đồ so sánh Accuracy/Latency).

MÔ TẢ BÀI TOÁN & TỔNG QUAN

Đầu vào (Input):

- Hình ảnh đầu vào được chuẩn hóa kích thước **(640, 640)** để đảm bảo công bằng khi so sánh với YOLO.
- Sử dụng các đặc trưng từ 3 tầng cuối của Backbone (ResNet/HGNet) là {S3, S4, S5}.

Quy trình xử lý (Pipeline):

1. **Efficient Hybrid Encoder:** Xử lý các đặc trưng đa quy mô, tách biệt tương tác nội bộ (intra-scale) và kết hợp chéo (cross-scale).
2. **Uncertainty-minimal Query Selection:** Chọn ra một số lượng cố định các đặc trưng tốt nhất từ Encoder để làm truy vấn khởi tạo (Object Queries).
3. **Transformer Decoder:** Tối ưu hóa các truy vấn này qua các lớp để sinh ra kết quả cuối cùng.

Đầu ra (Output):

- Dự đoán trực tiếp tập hợp đối tượng (One-to-one set prediction).
- Bao gồm: Nhãn lớp (Category) và Tọa độ hộp (Bounding Box).
- **Quan trọng:** Kết quả là End-to-End, không cần bước hậu xử lý NMS.

KIẾN TRÚC MÔ HÌNH

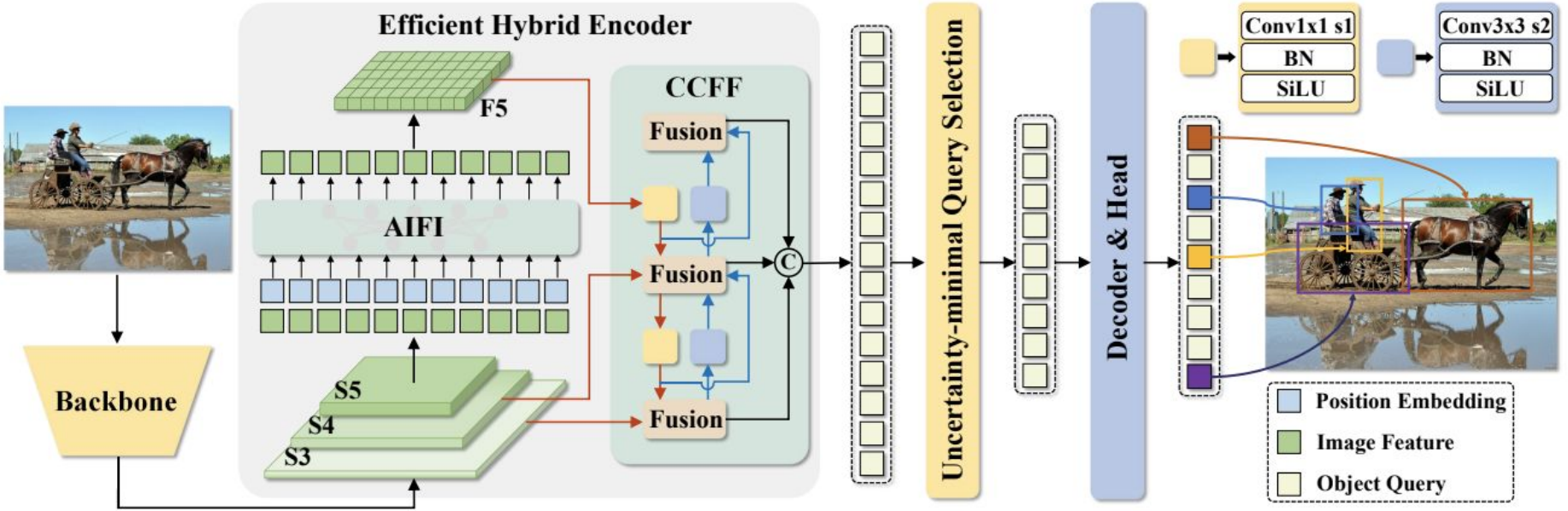


Figure 4 (Overview of RT-DETR) - Sơ đồ kiến trúc tổng quan.

Mục tiêu

Mục tiêu 1: Phân tích hiệu quả của việc loại bỏ NMS

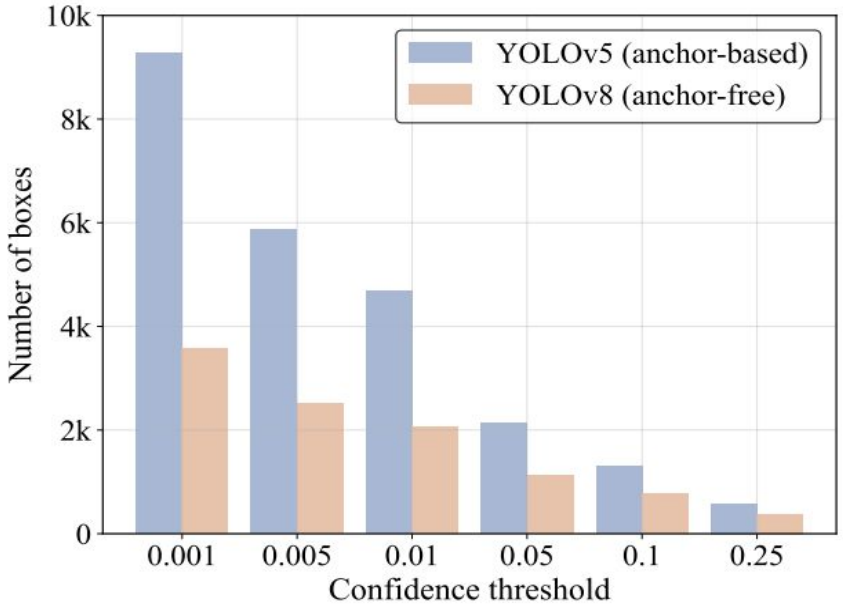
- Nghiên cứu sâu cơ chế hoạt động của **RT-DETR** (đặc biệt là *Hybrid Encoder* và *Query Selection*) để chứng minh khả năng hoạt động End-to-End mà không cần thuật toán NMS, giúp khắc phục độ trễ biến thiên của YOLO.

Mục tiêu 2: Tái hiện thực nghiệm (Reproduction)

- Triển khai mã nguồn và huấn luyện/kiểm thử mô hình trên tập dữ liệu chuẩn **COCO val2017**.
- Thiết lập môi trường đánh giá công bằng (Benchmark) giữa RT-DETR và YOLOv8 trên cùng điều kiện phần cứng (T4 GPU/TensorRT).

Mục tiêu 3: Đánh giá định lượng (Quantitative Evaluation)

- So sánh trực tiếp hai chỉ số hiệu năng chính: **Độ chính xác (AP)** và **Tốc độ suy luận (FPS)**.
- Kiểm chứng tuyên bố của bài báo: RT-DETR vượt trội hơn các mô hình YOLO cùng kích thước (Scale).



Hình 2: Sự phụ thuộc của số lượng bounding box vào ngưỡng Confidence trong YOLO, minh chứng cho sự bất ổn định của NMS

Nội dung và Phương pháp

Phương pháp nghiên cứu (Methodology):

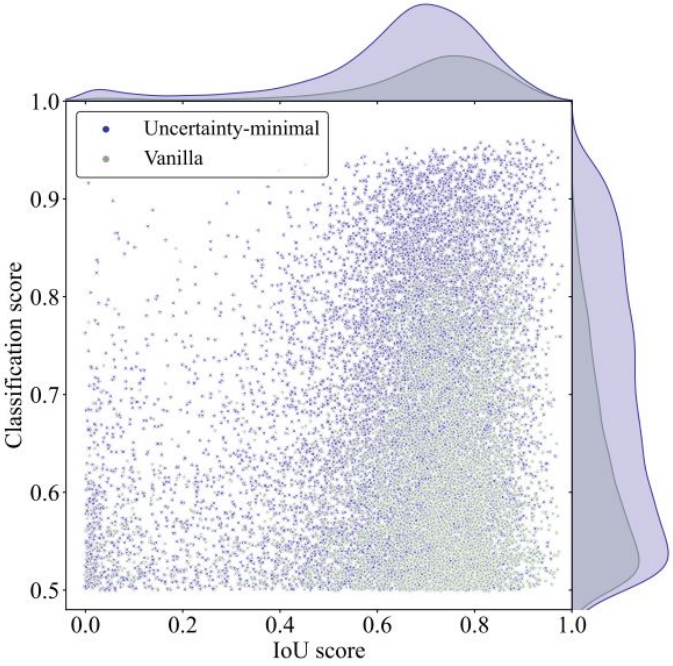
- Sử dụng phương pháp **Thực nghiệm (Empirical Study)** và **So sánh (Comparative Analysis)** trên tập dữ liệu chuẩn.

Nội dung kỹ thuật (Technical Focus):

- **Efficient Hybrid Encoder:** Nghiên cứu kiến trúc lai tách biệt xử lý tương tác nội bộ (AIFI) và kết hợp đa tầng (CCFF) để tối ưu chi phí tính toán.
- **Uncertainty-minimal Query Selection:** Phân tích cơ chế chọn lọc truy vấn dựa trên sự khớp nhau giữa phân loại và định vị để giảm thiểu độ không chắc chắn.

Thiết lập thực nghiệm (Experimental Setup):

- **Dataset:** COCO val2017.
- **Môi trường:** PyTorch, TensorRT FP16 trên GPU T4.
- **Đối sánh:** So sánh RT-DETR-R50/R101 với YOLOv8-L/X về mAP và Latency.



Hình 6: Phân tích hiệu quả của cơ chế chọn lọc truy vấn (Query Selection) trên tập COCO.

Cơ sở Toán học

1. Hybrid Encoder: Tách biệt Tương tác

Thay vì dùng Transformer cho tất cả (gây chậm), RT-DETR tách đôi quy trình:

- **AIFI (Attention):** Chỉ áp dụng cho tầng cao nhất S5 (giàu ngữ nghĩa).
- **CCFF (Fusion):** Dùng CNN để gộp các tầng còn lại (tăng tốc).

$$\mathcal{O} = CCFF(\{\mathcal{S}_3, \mathcal{S}_4, \underbrace{\text{AIFI}(\mathcal{S}_5)}_{\text{Attention}}\})$$

(Ý nghĩa: Giảm khối lượng tính toán nhưng giữ được độ chính xác cao)

2. Query Selection: Giảm thiểu độ Không chắc chắn

Định nghĩa độ không chắc chắn U là sự lệch pha giữa **Phân loại** C và **Định vị** P:

$$\mathcal{U}(\hat{X}) = ||\mathcal{P}(\hat{X}) - \mathcal{C}(\hat{X})||$$

- **Mục tiêu:** Tối thiểu hóa U trong hàm Loss.
- **Hệ quả:** Chỉ chọn các đặc trưng có cả **vị trí chuẩn** VÀ **nhãn đúng** làm đầu vào cho Decoder.

KẾT QUẢ THỰC NGHIỆM VÀ KẾT LUẬN

Kết quả Thực nghiệm:

- **RT-DETR-R50:** Đạt **53.1% AP** với tốc độ **108 FPS** trên T4 GPU.
- **So sánh với YOLOv8-L:** RT-DETR-R50 nhanh hơn **52.1%** và chính xác hơn **0.2% AP**.
- **RT-DETR-R101:** Đạt **54.3% AP**, vượt trội hoàn toàn so với YOLOv8-X cả về tốc độ lẫn độ chính xác.

Kết luận:

- RT-DETR đã giải quyết thành công bài toán loại bỏ NMS, mang lại quy trình End-to-End thực sự.
- Kiến trúc *Hybrid Encoder* và cơ chế *Query Selection* chứng minh được hiệu quả vượt trội so với các mô hình DETR và YOLO trước đây.

Model	Backbone	#Epochs	#Params (M)	GFLOPs	$FPS_{bs=1}$	AP^{val}	AP^{val}_{50}	AP^{val}_{75}	AP^{val}_{S}	AP^{val}_{M}	AP^{val}_{L}
Real-time Object Detectors											
YOLOv5-L [11]	-	300	46	109	54	49.0	67.3	-	-	-	-
YOLOv5-X [11]	-	300	86	205	43	50.7	68.9	-	-	-	-
PPYOLOE-L [40]	-	300	52	110	94	51.4	68.9	55.6	31.4	55.3	66.1
PPYOLOE-X [40]	-	300	98	206	60	52.3	69.9	56.5	33.3	56.3	66.4
YOLOv6-L [16]	-	300	59	150	99	52.8	70.3	57.7	34.4	58.1	70.1
YOLOv7-L [38]	-	300	36	104	55	51.2	69.7	55.5	35.2	55.9	66.7
YOLOv7-X [38]	-	300	71	189	45	52.9	71.1	57.4	36.9	57.7	68.6
YOLOv8-L [12]	-	-	43	165	71	52.9	69.8	57.5	35.3	58.3	69.8
YOLOv8-X [12]	-	-	68	257	50	53.9	71.0	58.7	35.7	59.3	70.7
End-to-end Object Detectors											
DETR-DC5 [4]	R50	500	41	187	-	43.3	63.1	45.9	22.5	47.3	61.1
DETR-DC5 [4]	R101	500	60	253	-	44.9	64.7	47.7	23.7	49.5	62.3
Anchor-DETR-DC5 [39]	R50	50	39	172	-	44.2	64.7	47.5	24.7	48.2	60.6
Anchor-DETR-DC5 [39]	R101	50	-	-	-	45.1	65.7	48.8	25.8	49.4	61.6
Conditional-DETR-DC5 [27]	R50	108	44	195	-	45.1	65.4	48.5	25.3	49.0	62.2
Conditional-DETR-DC5 [27]	R101	108	63	262	-	45.9	66.8	49.5	27.2	50.3	63.3
Efficient-DETR [42]	R50	36	35	210	-	45.1	63.1	49.1	28.3	48.4	59.0
Efficient-DETR [42]	R101	36	54	289	-	45.7	64.1	49.5	28.2	49.1	60.2
SMCA-DETR [9]	R50	108	40	152	-	45.6	65.5	49.1	25.9	49.3	62.6
SMCA-DETR [9]	R101	108	58	218	-	46.3	66.6	50.2	27.2	50.5	63.2
Deformable-DETR [45]	R50	50	40	173	-	46.2	65.2	50.0	28.8	49.2	61.7
DAB-Deformable-DETR [23]	R50	50	48	195	-	46.9	66.0	50.8	30.1	50.4	62.5
DAB-Deformable-DETR++ [23]	R50	50	47	-	-	48.7	67.2	53.0	31.4	51.6	63.9
DN-Deformable-DETR [17]	R50	50	48	195	-	48.6	67.4	52.7	31.0	52.0	63.7
DN-Deformable-DETR++ [17]	R50	50	47	-	-	49.5	67.6	53.8	31.3	52.6	65.4
DINO-Deformable-DETR [44]	R50	36	47	279	5	50.9	69.0	55.3	34.6	54.1	64.6
Real-time End-to-end Object Detector (ours)											
RT-DETR	R50	72	42	136	**108**	**53.1**	**71.3**	**57.7**	34.8	58.0	70.0
RT-DETR	R101	72	76	259	**74**	**54.3**	**72.7**	58.6	36.0	58.8	**72.1**

Bảng 1: So sánh hiệu năng giữa RT-DETR và các mô hình SOTA trên tập COCO

Tài liệu tham khảo

[1] Wenyu Lv, Shangliang Xu, Yian Zhao, Guanzhong Wang, Jinman Wei, Cheng Cui, Yuning Du, Qingqing Dang, Yi Liu: **DETRs Beat YOLOs on Real-time Object Detection.** CVPR 2024: 16965-16974

[2] Glenn Jocher, Ayush Chaurasia, Jing Qiu: **Ultralytics YOLO.** CoRR abs/2304.00577 (2023)

[3] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, Sergey Zagoruyko: **End-to-End Object Detection with Transformers.** ECCV (1) 2020: 213-229

[4] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, Jifeng Dai: **Deformable DETR: Deformable Transformers for End-to-End Object Detection.** ICLR 2021