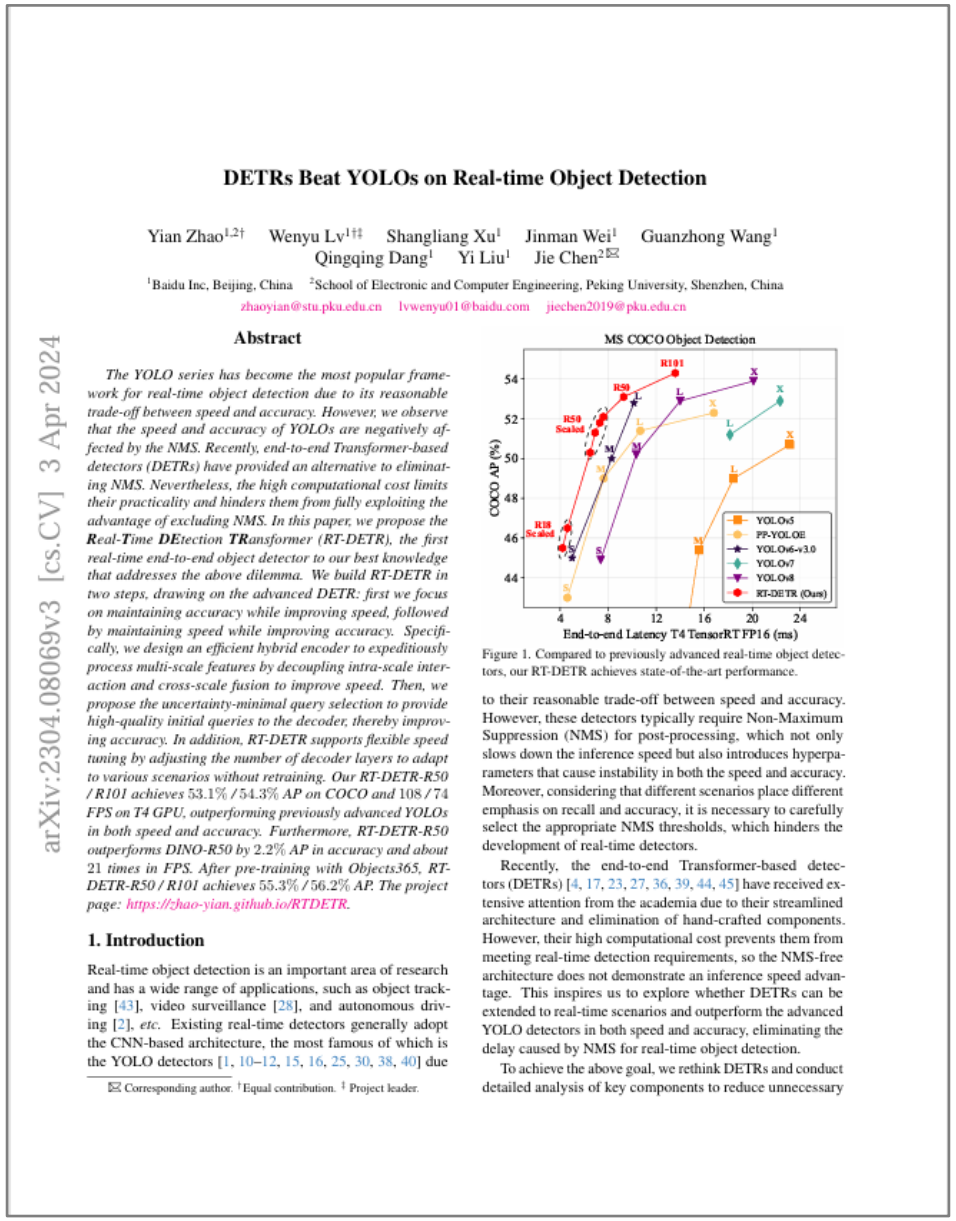
DETRs Beat YOLOs on Real-time Object Detection

July 30, 2024

DETRs Beat YOLOs on Real-time Object Detection

Yian Zhao et al., Baidu+

https://arxiv.org/abs/2304.08069



DETRs Beat YOLOs on Real-time Object Detection

Yian Zhao[1,2†] Wenyu Lv[1†‡] Shangliang Xu[1] Jinman Wei[1] Guanzhong Wang[1]
Qingqing Dang[1] Yi Liu[1] Jie Chen[2✉]

[1]Baidu Inc, Beijing, China [2]School of Electronic and Computer Engineering, Peking University, Shenzhen, China
zhaoyian@stu.pku.edu.cn lvwenyu01@baidu.com jiechen2019@pku.edu.cn

Abstract

*The YOLO series has become the most popular framework for real-time object detection due to its reasonable trade-off between speed and accuracy. However, we observe that the speed and accuracy of YOLOs are negatively affected by the NMS. Recently, end-to-end Transformer-based detectors (DETRs) have provided an alternative to eliminating NMS. Nevertheless, the high computational cost limits their practicality and hinders them from fully exploiting the advantage of excluding NMS. In this paper, we propose the **R**eal-**T**ime **DE**tection **TR**ansformer (RT-DETR), the first real-time end-to-end object detector to our best knowledge that addresses the above dilemma. We build RT-DETR in two steps, drawing on the advanced DETR: first we focus on maintaining accuracy while improving speed, followed by maintaining speed while improving accuracy. Specifically, we design an efficient hybrid encoder to expeditiously process multi-scale features by decoupling intra-scale interaction and cross-scale fusion to improve speed. Then, we propose the uncertainty-minimal query selection to provide high-quality initial queries to the decoder, thereby improving accuracy. In addition, RT-DETR supports flexible speed tuning by adjusting the number of decoder layers to adapt to various scenarios without retraining. Our RT-DETR-R50 / R101 achieves 53.1% / 54.3% AP on COCO and 108 / 74 FPS on T4 GPU, outperforming previously advanced YOLOs in both speed and accuracy. Furthermore, RT-DETR-R50 outperforms DINO-R50 by 2.2% AP in accuracy and about 21 times in FPS. After pre-training with Objects365, RT-DETR-R50 / R101 achieves 55.3% / 56.2% AP. The project page: https://zhao-yian.github.io/RTDETR.*

Figure 1. Compared to previously advanced real-time object detectors, our RT-DETR achieves state-of-the-art performance.

1. Introduction

Real-time object detection is an important area of research and has a wide range of applications, such as object tracking [43], video surveillance [28], and autonomous driving [2], *etc*. Existing real-time detectors generally adopt the CNN-based architecture, the most famous of which is the YOLO detectors [1, 10–12, 15, 16, 25, 30, 38, 40] due to their reasonable trade-off between speed and accuracy. However, these detectors typically require Non-Maximum Suppression (NMS) for post-processing, which not only slows down the inference speed but also introduces hyperparameters that cause instability in both the speed and accuracy. Moreover, considering that different scenarios place different emphasis on recall and accuracy, it is necessary to carefully select the appropriate NMS thresholds, which hinders the development of real-time detectors.

Recently, the end-to-end Transformer-based detectors (DETRs) [4, 17, 23, 27, 36, 39, 44, 45] have received extensive attention from the academia due to their streamlined architecture and elimination of hand-crafted components. However, their high computational cost prevents them from meeting real-time detection requirements, so the NMS-free architecture does not demonstrate an inference speed advantage. This inspires us to explore whether DETRs can be extended to real-time scenarios and outperform the advanced YOLO detectors in both speed and accuracy, eliminating the delay caused by NMS for real-time object detection.

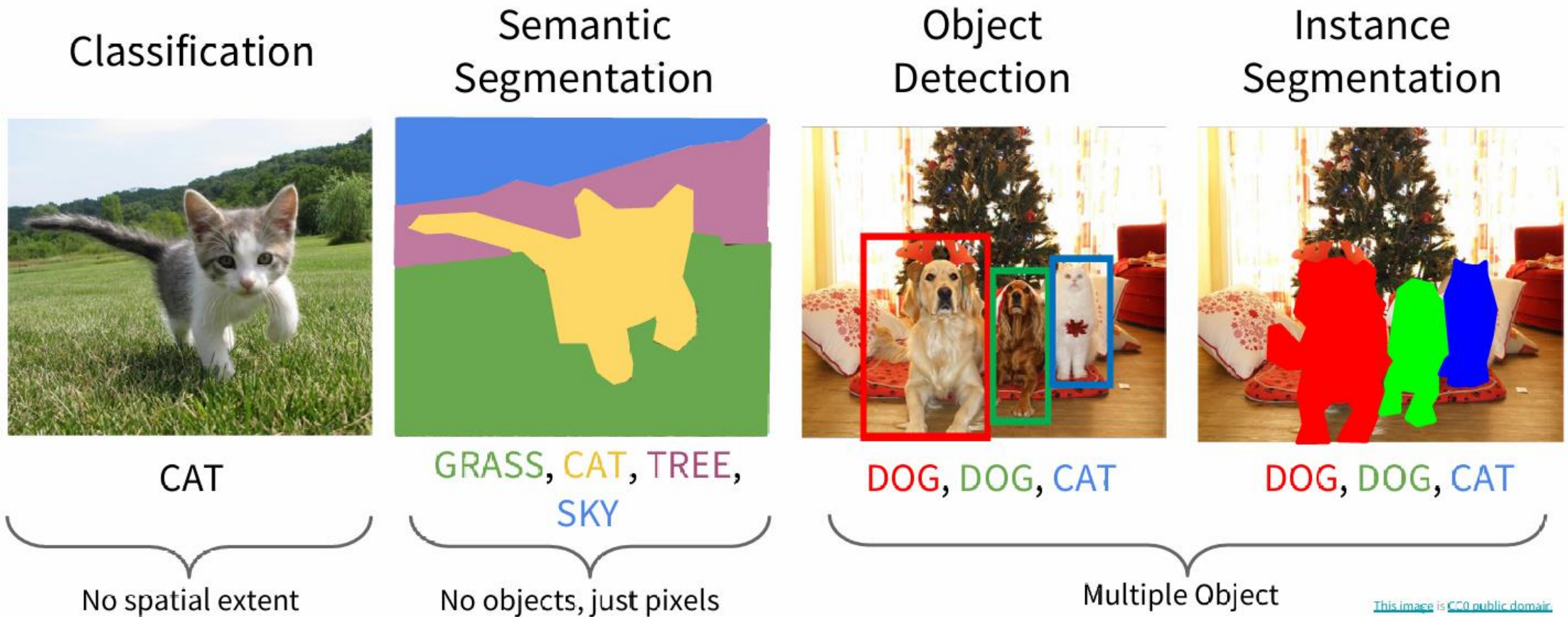
To achieve the above goal, we rethink DETRs and conduct detailed analysis of key components to reduce unnecessary

✉ Corresponding author. †Equal contribution. ‡ Project leader.

DETRs Beat YOLOs overview

- The YOLO family of models are popular for real-time object detection because they run fast while maintaining good accuracy
  - YOLO models require a form of deduplication of similar predictions called Non-Maximum Suppression (NMS) which adds to the compute cost
  - NMS requires two hyperparameters for confidence and IOU thresholds
- The DETR family of models is newer and uses transformers, generally achieving higher accuracy but at a much greater amount of compute
- This work modifies the DETR architecture to get it to run much faster
- Their RT-DETR model can run as fast as YOLOs but better accuracy

Computer vision tasks



http://vision.stanford.edu/teaching/cs231n/slides/2024/lecture_1_part_2.pdf

The original YOLO architecture

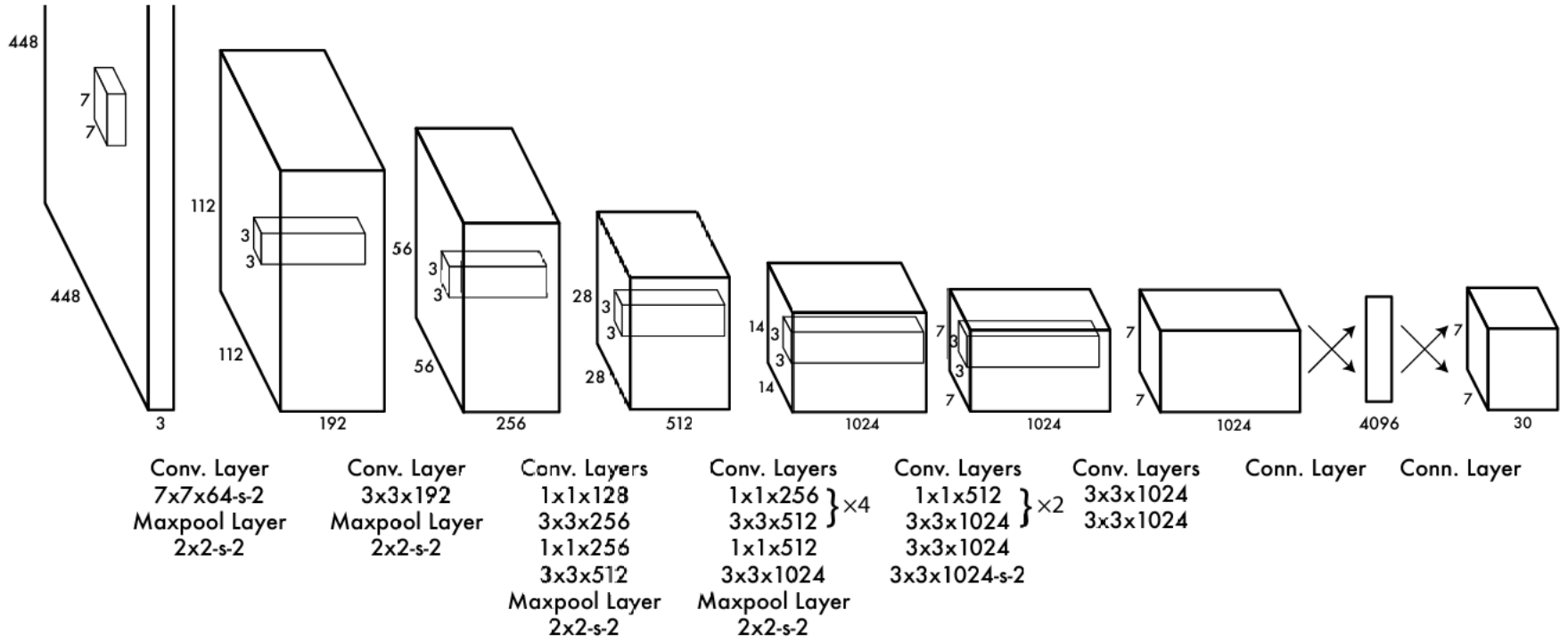
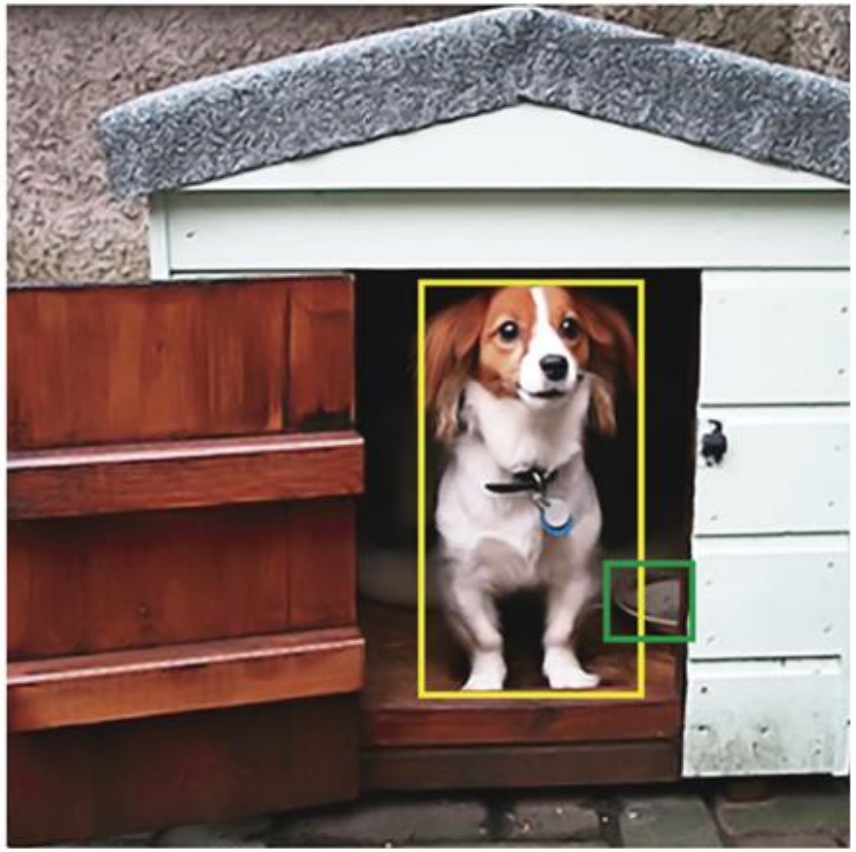


Figure 3: The Architecture. Our detection network has 24 convolutional layers followed by 2 fully connected layers. Alternating 1×1 convolutional layers reduce the features space from preceding layers. We pretrain the convolutional layers on the ImageNet classification task at half the resolution (224×224 input image) and then double the resolution for detection.

Non-Maximum Suppression

NMS



https://arxiv.org/abs/2304.00501

NMS Examples

Conf_thr = 0.001
IoU_thr = 0.3

Conf_thr = 0.001
IoU_thr = 0.7

Conf_thr = 0.25
IoU_thr = 0.7

Figure A. Visualization of YOLOv8-L [12] predictions with different NMS thresholds.

The original DETR architecture

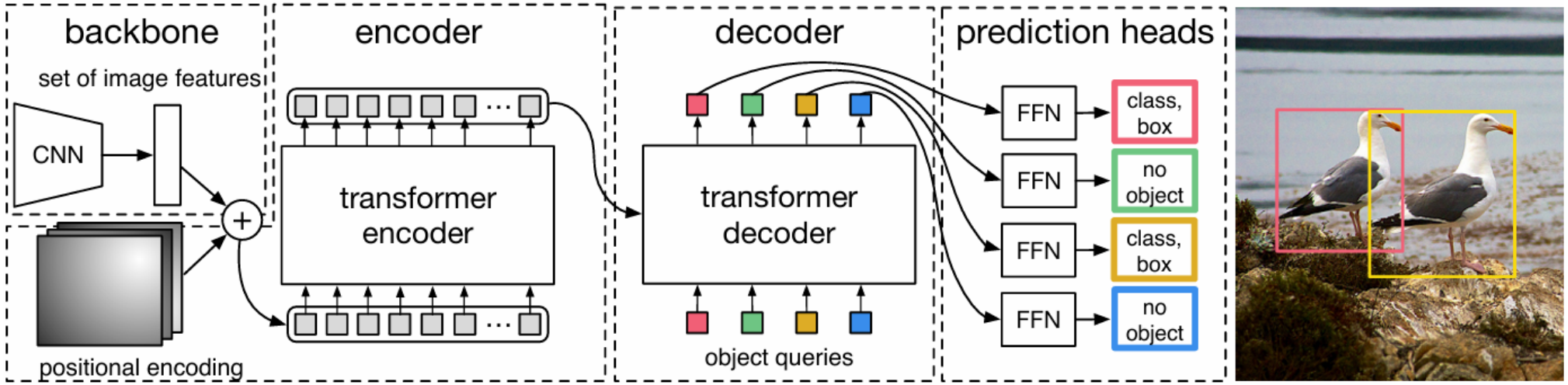


Fig. 2: DETR uses a conventional CNN backbone to learn a 2D representation of an input image. The model flattens it and supplements it with a positional encoding before passing it into a transformer encoder. A transformer decoder then takes as input a small fixed number of learned positional embeddings, which we call *object queries*, and additionally attends to the encoder output. We pass each output embedding of the decoder to a shared feed forward network (FFN) that predicts either a detection (class and bounding box) or a "`no object`" class.

Hungarian algorithm

- The bipartite matching problem has an equivalent matrix form
- The algorithm involves subtracting the smallest value from each row and column, seeking all-zero assignments, and repeating subtraction

In this simple example, there are three workers: Alice, Bob and Carol. One of them has to clean the bathroom, another sweep the floors and the third washes the windows, but they each demand different pay for the various tasks. The problem is to find the lowest-cost way to assign the jobs. The problem can be represented in a matrix of the costs of the workers doing the jobs. For example:

Worker \ Task	Clean bathroom	Sweep floors	Wash windows
Alice	**$8**	$4	$7
Bob	$5	$2	**$3**
Carol	$9	**$4**	$8

https://en.wikipedia.org/wiki/Hungarian_algorithm

Encoder changes

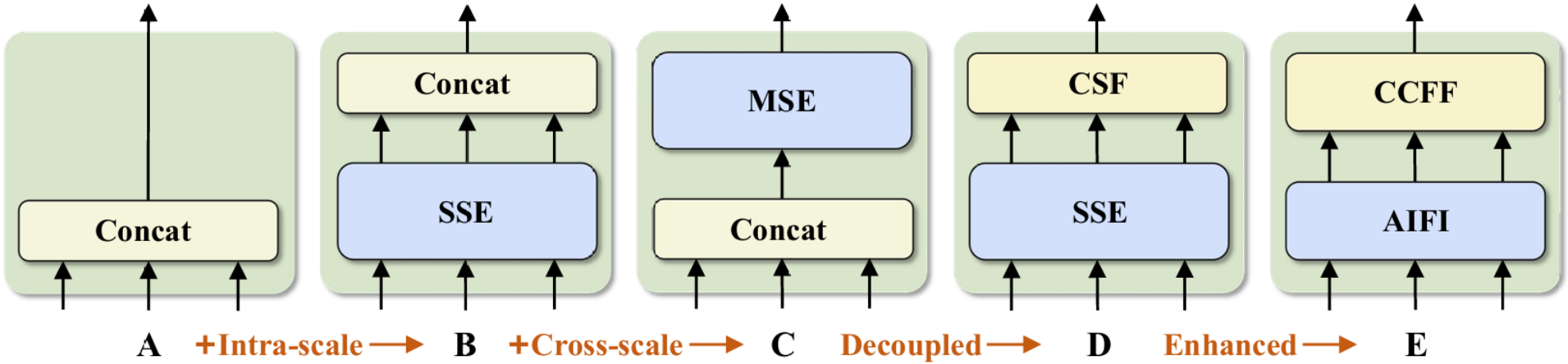


Figure 3. The encoder structure for each variant. **SSE** represents the single-scale Transformer encoder, **MSE** represents the multi-scale Transformer encoder, and **CSF** represents cross-scale fusion. **AIFI** and **CCFF** are the two modules designed into our hybrid encoder.

Variant	AP (%)	#Params (M)	Latency (ms)
A	43.0	31	7.2
B	44.9	32	11.1
C	45.6	32	13.3
D	46.4	35	12.2
D_{S_5}	46.8	35	7.9
E	47.9	42	9.3

Table 3. The indicators of the set of variants illustrated in Figure 3.

- A→B proves that the intra-scale feature interaction is significant, but the single-scale Transformer encoder is expensive
- B→C shows that the cross-scale feature fusion is also necessary but the multi-scale Transformer encoder requires higher compute
- C→D suggests that decoupling intra-scale interaction and cross-scale fusion not only reduces compute but also improves accuracy
- D→D_{S_5} demonstrates that intra-scale interactions of lower-level features are not required

RT-DETR encoder

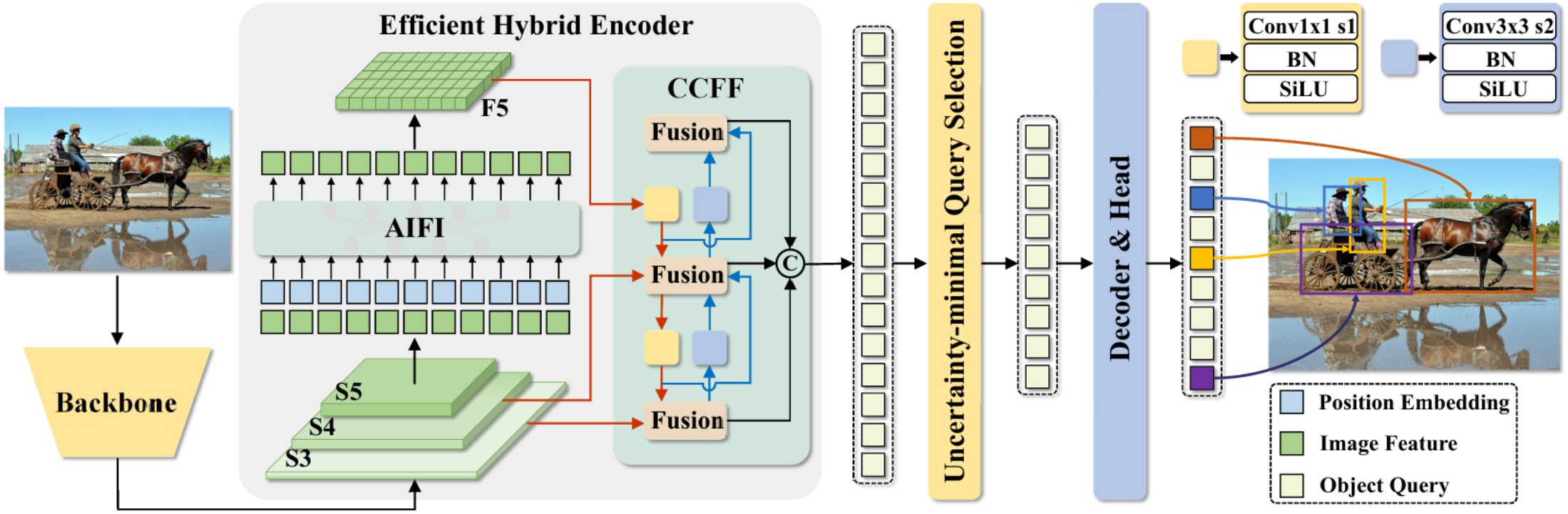


Figure 4. Overview of RT-DETR. We feed the features from the last three stages of the backbone into the encoder. The efficient hybrid encoder transforms multi-scale features into a sequence of image features through the Attention-based Intra-scale Feature Interaction (AIFI) and the CNN-based Cross-scale Feature Fusion (CCFF). Then, the uncertainty-minimal query selection selects a fixed number of encoder features to serve as initial object queries for the decoder. Finally, the decoder with auxiliary prediction heads iteratively optimizes object queries to generate categories and boxes.

CCFF fusion block

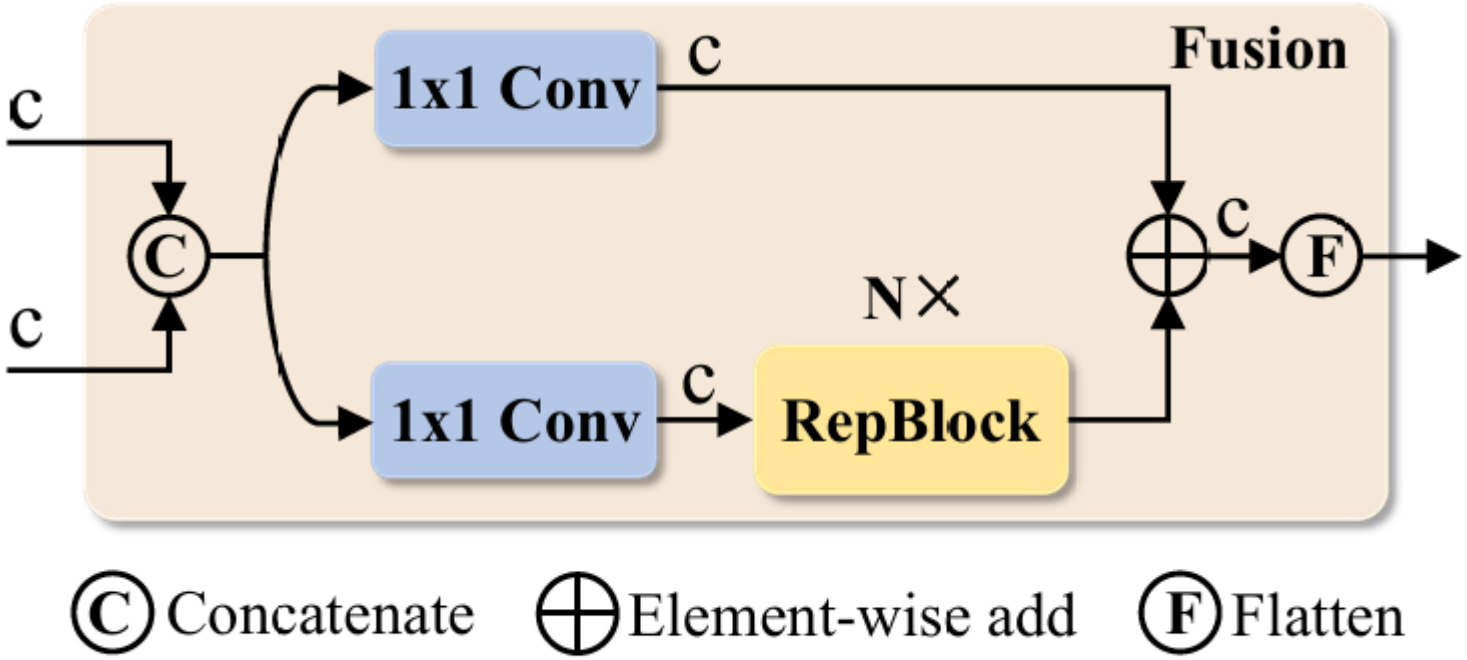


Figure 5. The fusion block in CCFF.

Uncertainty-minimal query selection

- The feature uncertainty is defined as the discrepancy between the predicted distributions of localization and classification

$$\mathcal{U}(\hat{\boldsymbol{\mathcal{X}}}) = \|\mathcal{P}(\hat{\boldsymbol{\mathcal{X}}}) - \mathcal{C}(\hat{\boldsymbol{\mathcal{X}}})\|, \hat{\boldsymbol{\mathcal{X}}} \in \mathbb{R}^D$$

- To minimize the uncertainty of the queries, we integrate the uncertainty into the loss function for the gradient-based optimization

$$\mathcal{L}(\hat{\boldsymbol{\mathcal{X}}}, \hat{\boldsymbol{\mathcal{Y}}}, \boldsymbol{\mathcal{Y}}) = \mathcal{L}_{box}(\hat{\mathbf{b}}, \mathbf{b}) + \mathcal{L}_{cls}(\mathcal{U}(\hat{\boldsymbol{\mathcal{X}}}), \hat{\mathbf{c}}, \mathbf{c})$$

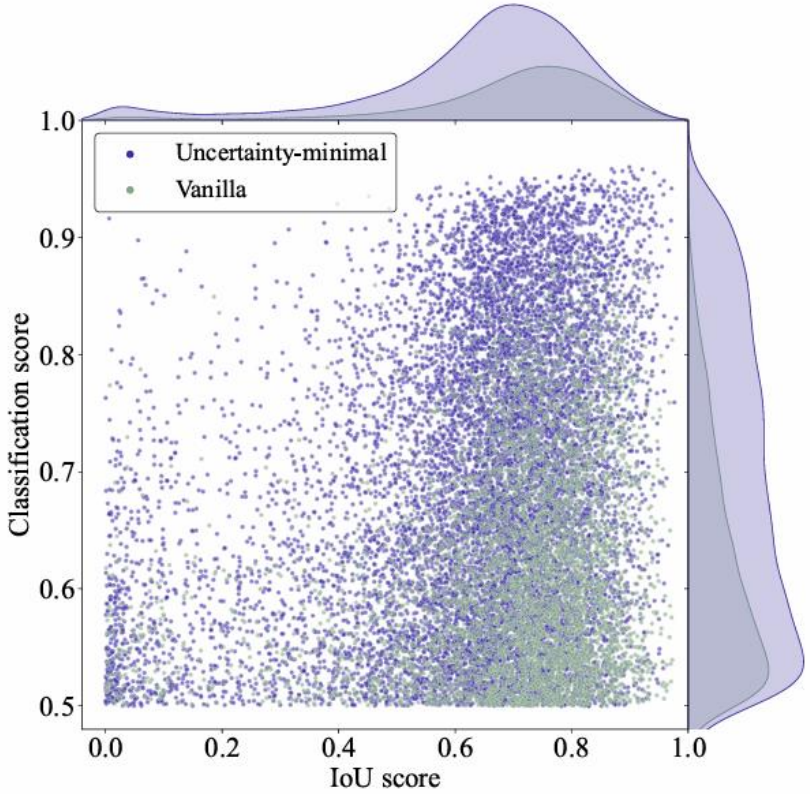
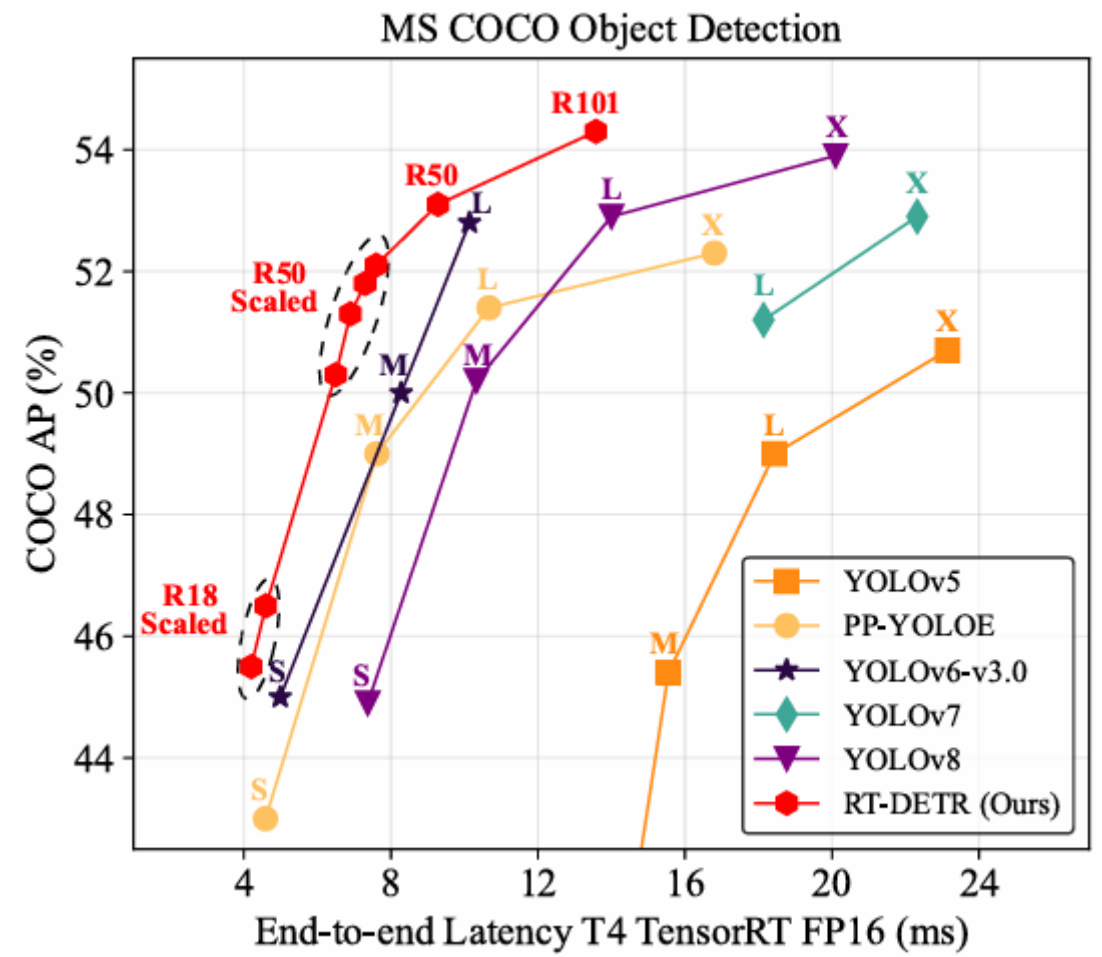


Figure 6. Classification and IoU scores of the selected encoder features. Purple and Green dots represent the selected features from model trained with uncertainty-minimal query selection and vanilla query selection, respectively.

RT-DETR results [1]

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**

RT-DETR results [2]



RT-DETR complex scenarios

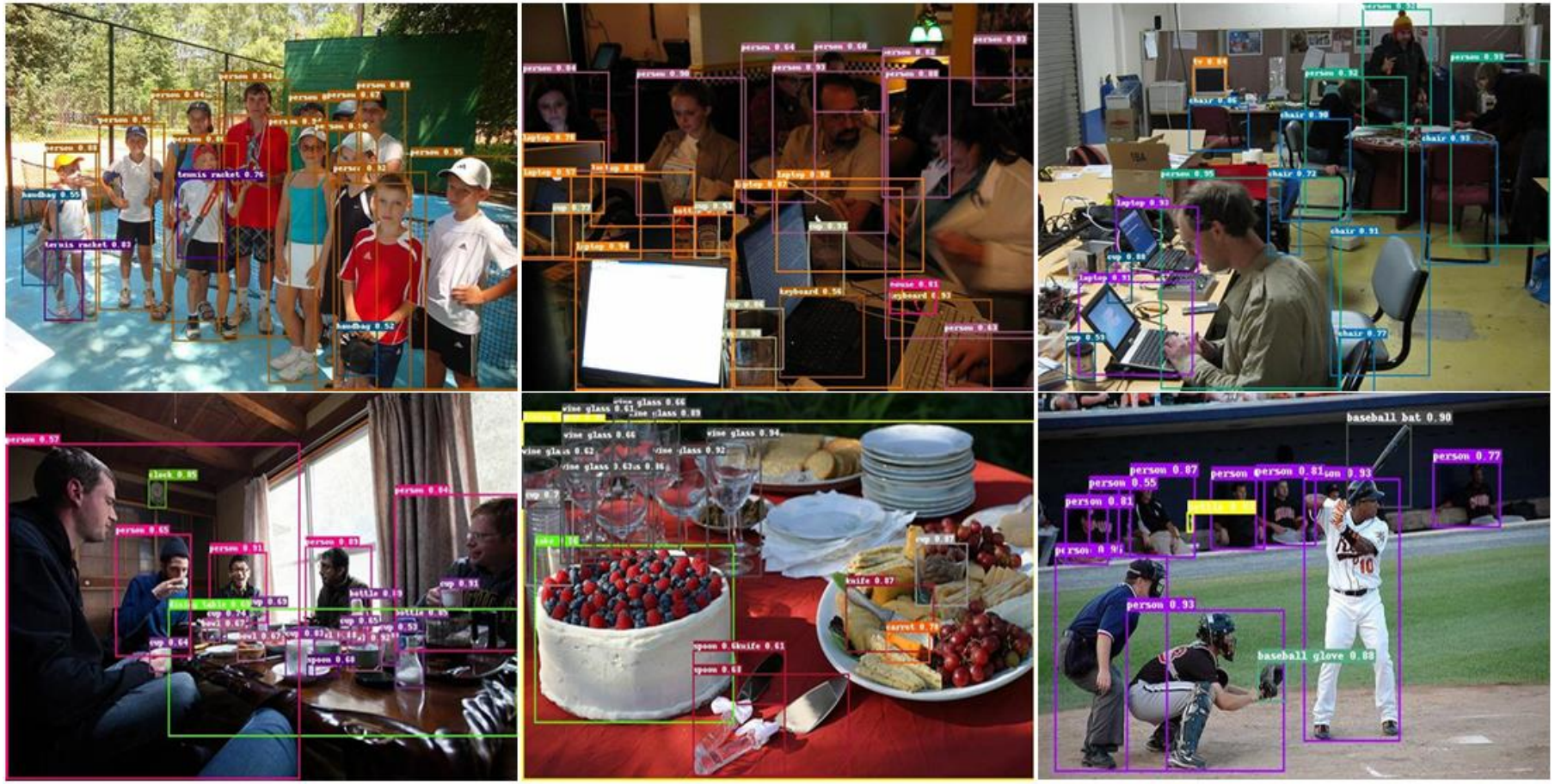


Figure C. Visualization of RT-DETR-R101 predictions in complex scenarios (score threshold=0.5).

RT-DETR difficult conditions

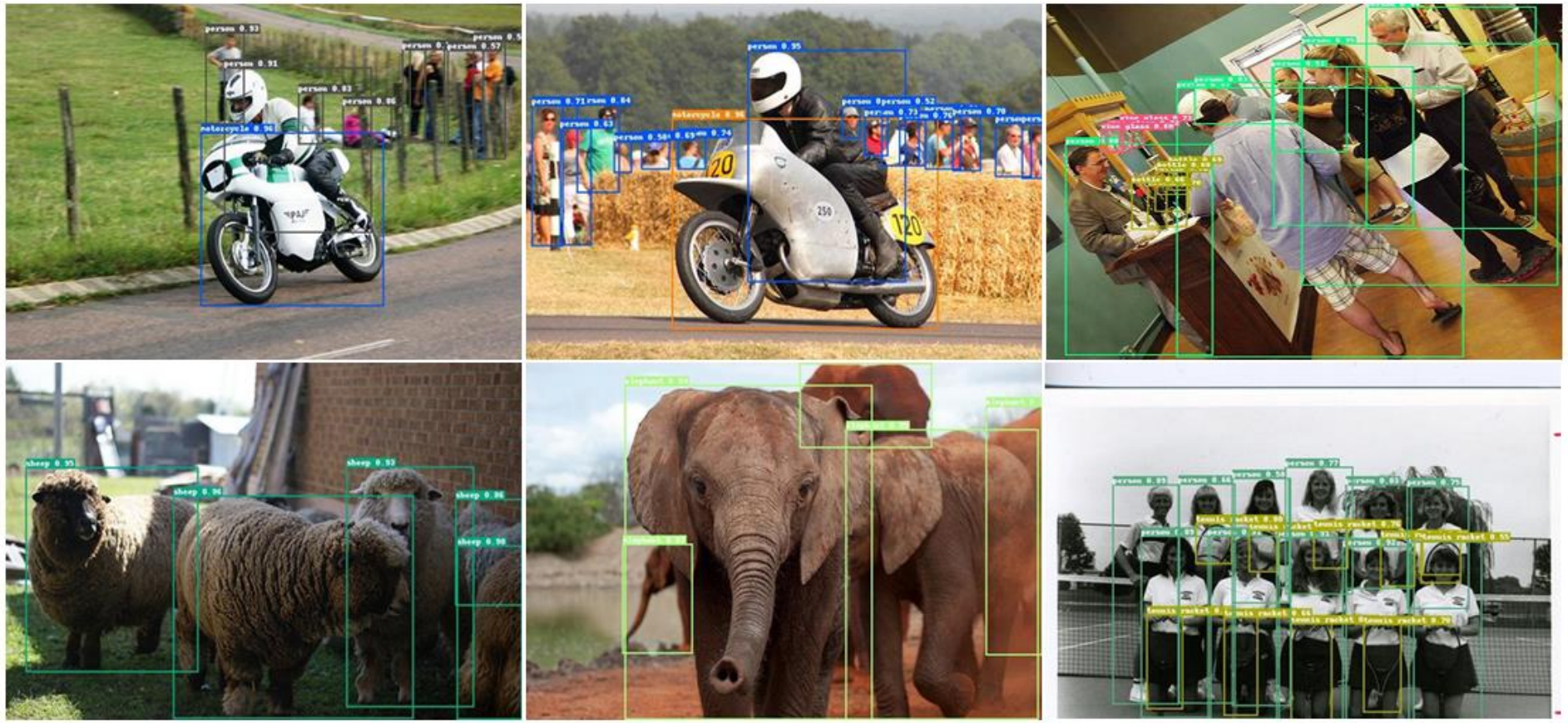


Figure D. Visualization of RT-DETR-R101 predictions under difficult conditions, including motion blur, rotation, and occlusion (score threshold=0.5).

DETRs beat YOLOs conclusion

- Analyzed the compute of YOLO models including cost of NMS
- Eliminated redundancy in the DETR transformer encoder, using a hybrid design that decouples intra-scale interaction from cross-scale fusion:
  - Attention-based Intra-scale Feature Interaction (AIFI)
  - CNN-based Cross-scale Feature Fusion (CCFF)
- Uncertainty-minimal Query Selection for high quality decoder queries
- For similar params and GFLOPs, had higher FPS and better AP scores
  - Performed better on large objects but worse on small objects
- Speed can be tuned by adjusting number of decoder layers
- CVPR poster video: https://cvpr.thecvf.com/virtual/2024/poster/31301

References

- You Only Look Once: Unified, Real-Time Object Detection
  Joseph Redmon et al. (2015)
  https://arxiv.org/abs/1506.02640
- End-to-End Object Detection with Transformers
  Nicolas Carion et al. (2020)
  https://arxiv.org/abs/2005.12872
- Deformable DETR: Deformable Transformers for End-to-End Object Detection
  Xizhou Zhu et al. (2020)
  https://arxiv.org/abs/2010.04159