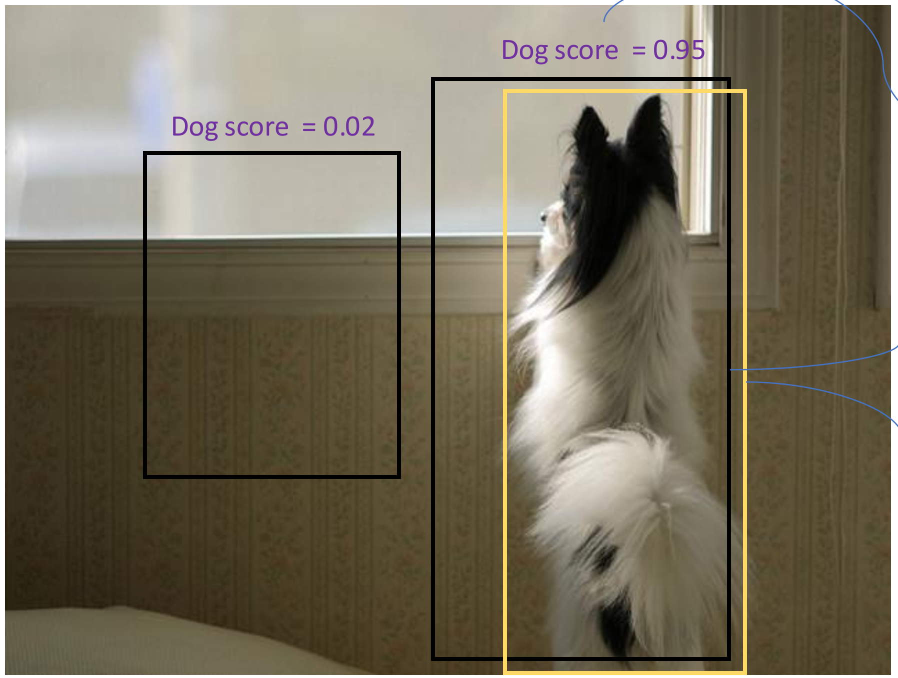
DEIM: DETR with Improved Matching for Fast Convergence

Shihua Huang[1], Zhichao Lu[2], Xiaodong Cun[3], Yongjun Yu[1], Xiao Zhou[4], Xi Shen[1]

1. Intellindust AI Lab
2. City University of Hong Kong
3. Great Bay University
4. Hefei Normal University

Object detection – fundamental CV task



Classification: category and confidence score

Location: accurate bounding box

Ground Truth

Auto-Driving

- Object detection paradigms:

    1. Region proposal: R-CNNs
    2. Pixel anchor: YOLOs
    3. Learnable query: DETRs

Object detection -- NMS

Highly-overlapped predictions

IoU thr. (Conf=0.001)	AP (%)	NMS (ms)
0.5	52.1	2.24
0.6	52.6	2.29
0.8	52.8	2.46

Conf thr. (IoU=0.7)	AP (%)	NMS (ms)
0.001	52.9	2.36
0.01	52.4	1.73
0.05	51.2	1.06

From: Zhao et. al. DETRs Beat YOLOs on Real-time Object Detection. CVPR, 2024.

NMS tuning

- Observations:

  1. Both region-based and anchor-based existing methods require NMS for post-processing
  2. NMS is unstable and introduces latency

***NMS**: Non-Maximum Suppression*

DEtection with Transformer -- DETR

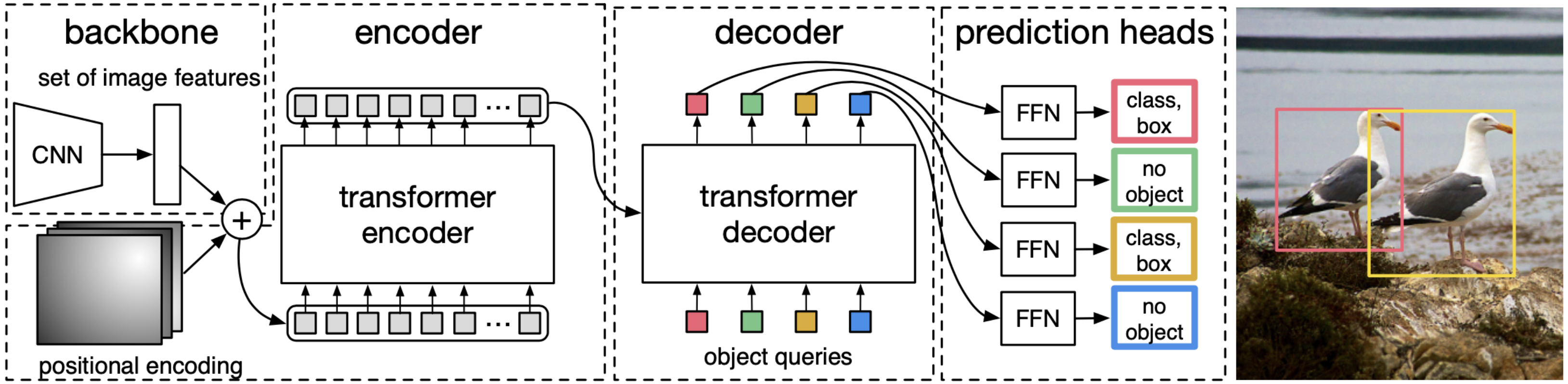


Fig. 2: DETR uses a conventional CNN backbone to learn a 2D representation of an input image. The model flattens it and supplements it with a positional encoding before passing it into a transformer encoder. A transformer decoder then takes as input a small fixed number of learned positional embeddings, 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.

- Advantages:

  1. The Transformer can extract global semantic context
  2. One-to-one assignment eliminates the hand-crafted NMS, an end-to-end detector

Nicolas Carion et.al. End-to-end object detection with Transformer. ECCV, 2020.

Challenges in DETR

- Challenges

  1. **Slow convergence**
  2. **High computation cost**
  3. **Poor performance over small objects**

Reasons behind slow convergence

- Hard optimization

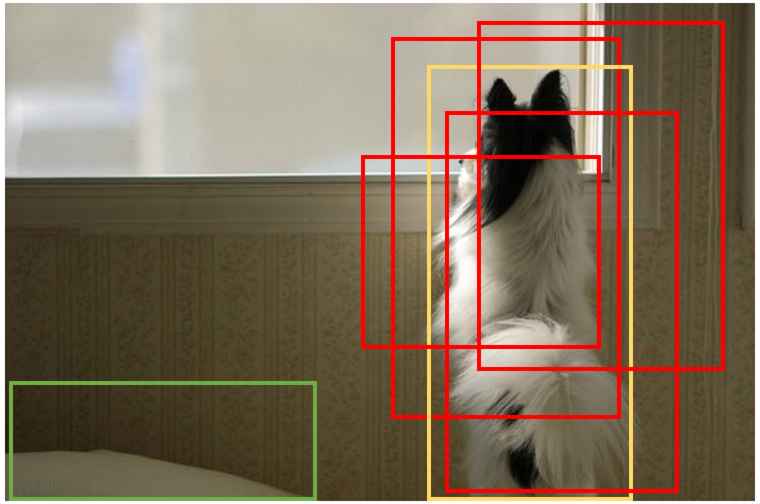
1. **Sparse supervision:** less positive queries
2. **Sparse queries:** low-quality matching

Supervision – O2M vs. O2O

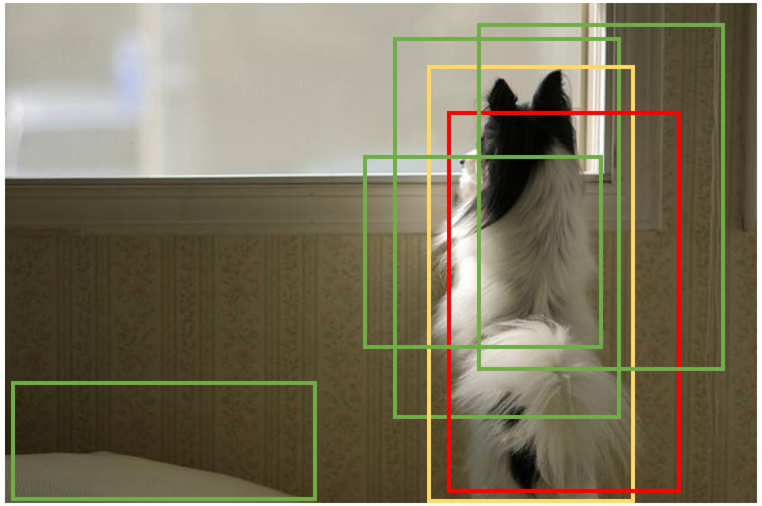
- Assignments

1. **One-to-many (O2M):** Multiple queries are assigned to each GT, and NMS is necessary for duplicate queries.
2. **One-to-one (O2O):** Only assign the best query to the GT, which works end-to-end.

Toy examples -- O2M and O2O for an image with single GT (yellow – GT, red – pos. queries, and green -- neg. queries)

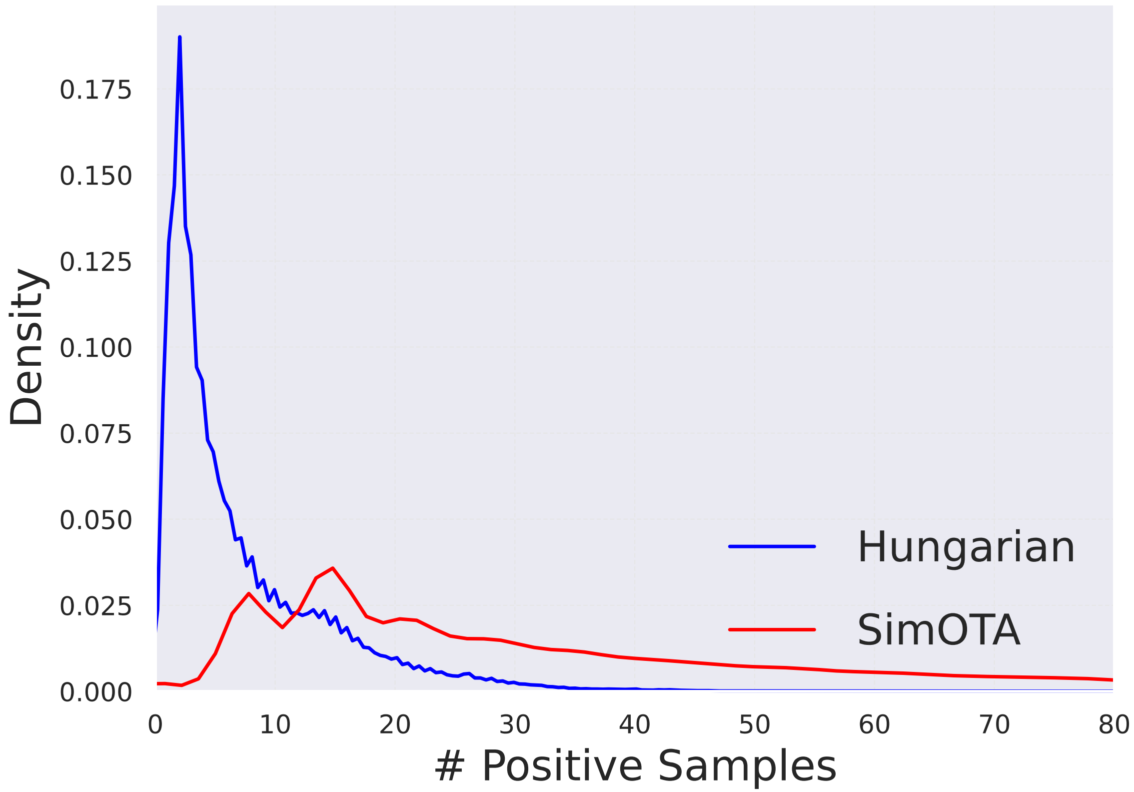


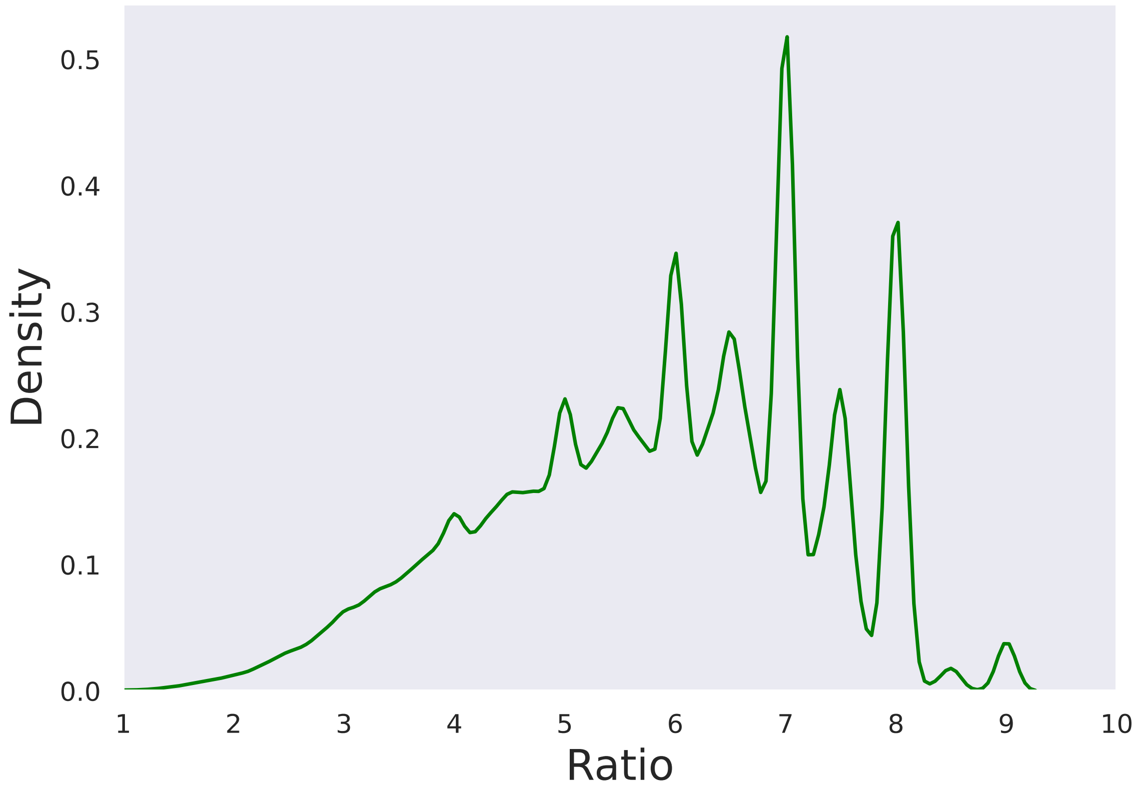
O2M: 1 target & 4 pos.



O2O: 1 target & 1 pos.

Supervision – O2M vs. O2O





- Comparison between O2M (SimOTA) and O2O (Hungarian):

  1. Less than 10 matched queries for most training images in O2O
  2. O2M has several times of matched queries over O2O

Dense supervision – increasing matched queries

- Works on increasing matched queries

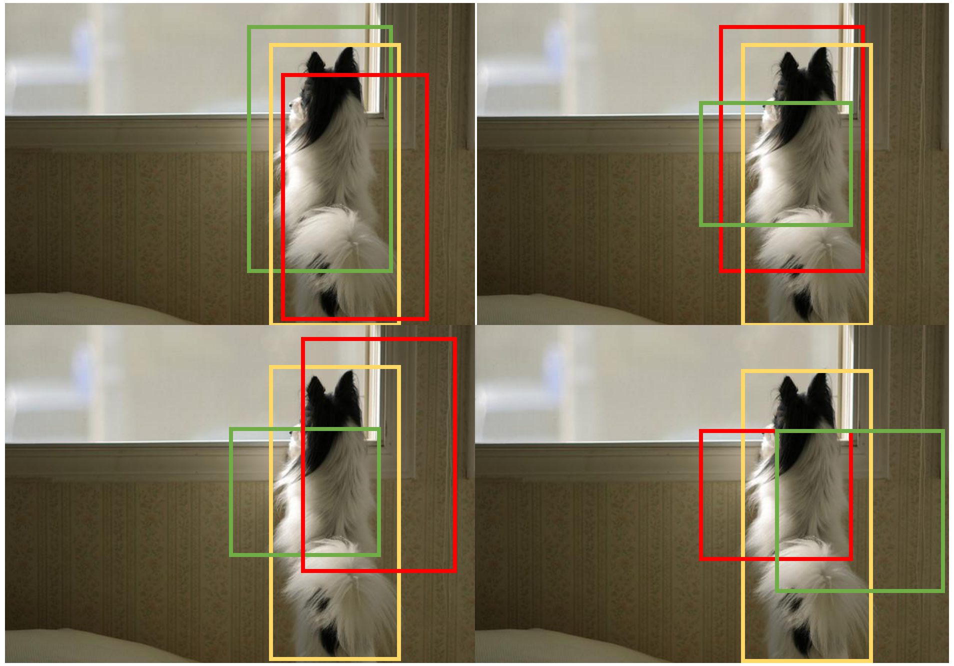
1. **Group DETR** (ICCV 2023): use multiple groups of queries and perform the O2O assignment in each group separately.
2. **Co-DETR** (ICCV 2023): introduce conventional O2Ms as the auxiliary training, including Faster R-CNN, FCOS et. al.

- Limitations

1. **Cost:** auxiliary decoders and additional training cost
2. **Extra Losses:** balance them with the main one carefully
3. **Potential side-effect:** increase high-quality duplicate queries

Dense supervision – Dense O2O

Toy example – by stitching simply



Dense O2O by stitching: 4 targets & 4 pos.

- Advantages:

1. **Conceptually simple and general**
2. **Come from free:** neglectable cost in data transformation

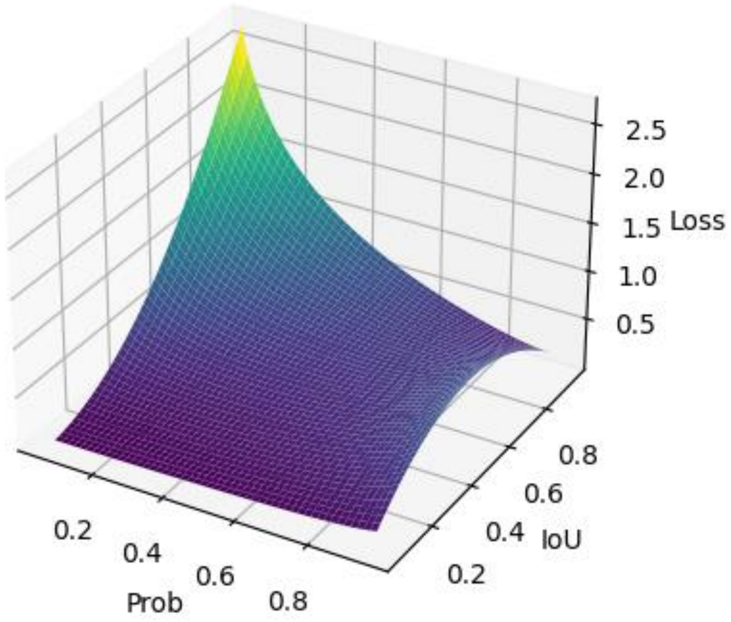
Sparse queries – query initialization

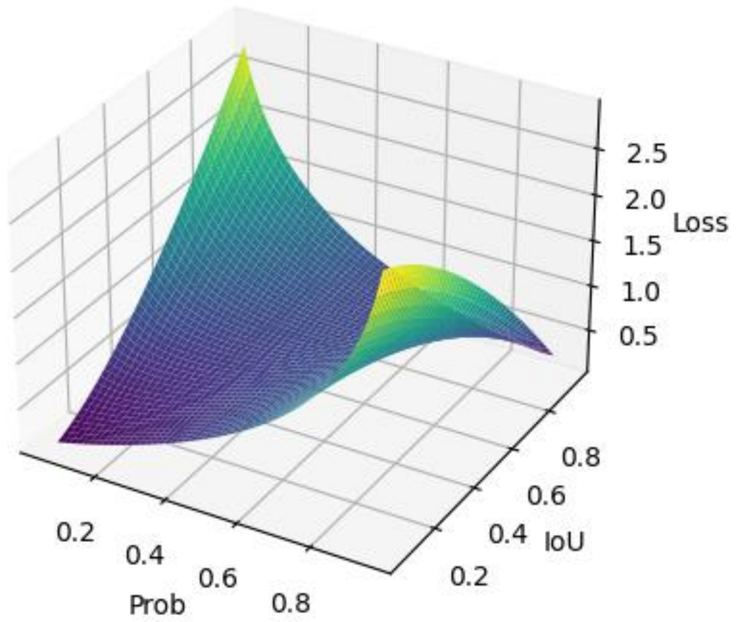
- Works on query initializations

  1. **DETR** (ECCV 2020): set to zero initially
  2. **Deformable DETR** (ICLR 2021): two-stage refinement inspired by R-CNN detectors
  3. **DN-DETR** (CVPR 2022) **and DINO** (ICLR 2023): initialize several auxiliary queries around GTs
  4. **DINO** (ICLR 2023) **and RT-DETR** (CVPR 2024): select top-k queries from the encoder

Introducing priors on query initializations can **alleviate** this but it still **exists** in most cases, particularly in images with more than one object.

Optimization – VFL vs. MAL





$$\mathrm{VFL}(p,q,y)=\begin{cases}-q(q\log(p)+(1-q)\log(1-p)) & q>0\\ -\alpha p^{\gamma}\log(1-p) & q=0,\end{cases}$$

$$\mathrm{MAL}(p,q,y)=\begin{cases}-q^{\gamma}\log(p)+(1-q^{\gamma})\log(1-p) & y=1\\ -p^{\gamma}\log(1-p) & y=0.\end{cases}$$

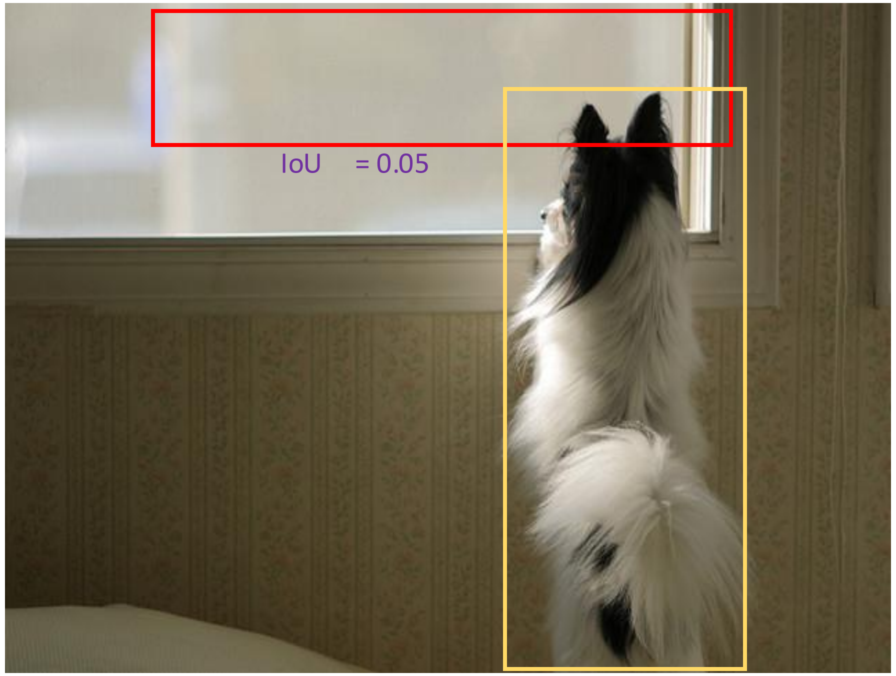
- Comparison between VFL and our MAL:

1. For low-quality matched queries, MAL will punish them harder with higher confidence
2. VFL takes those queries which have IoU = 0 as negative examples
3. MAL is a simpler equation than VFL and has no alpha

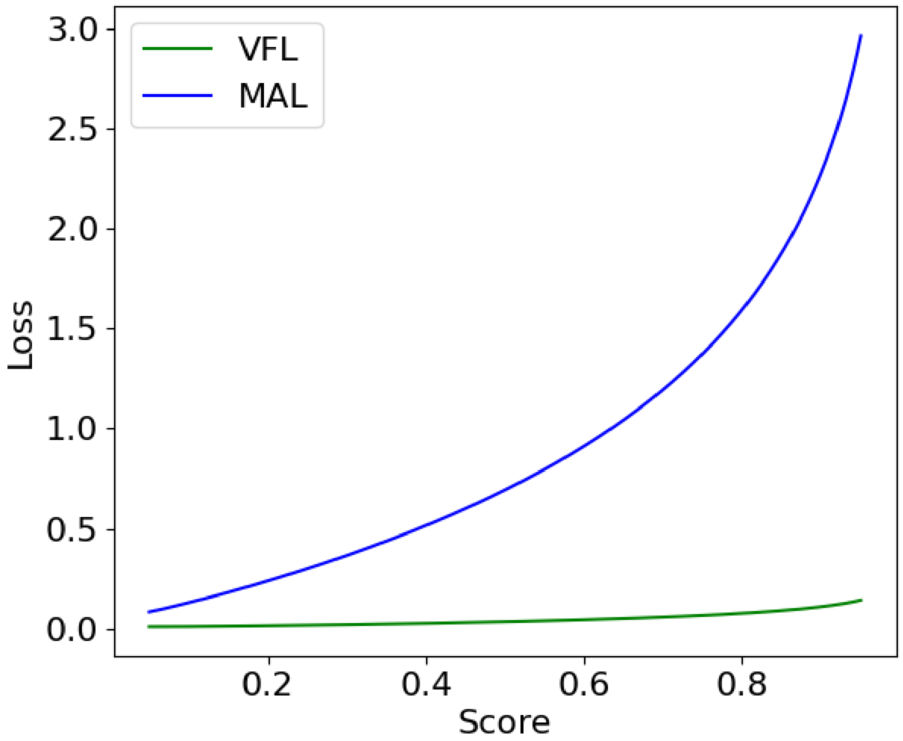
Note: p is the confidence probability, q is the IoU between query and GT, y is the class label; alpha@0.75 and gamma@1.5.

Optimization – VFL vs. MAL

Toy example – low-quality matching



Low-quality matching: IoU@0.05

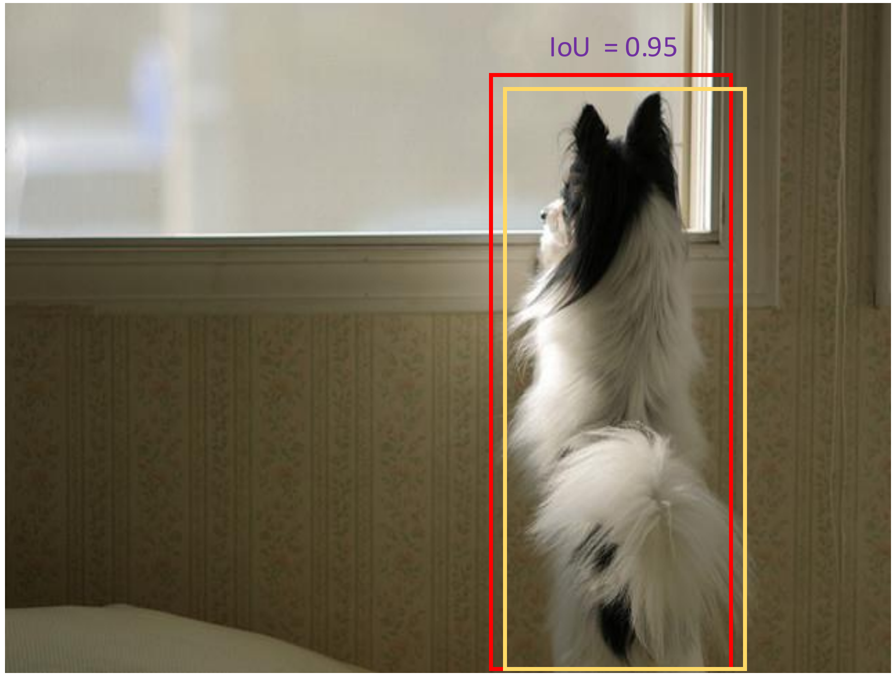


- Comparison between VFL and our MAL:

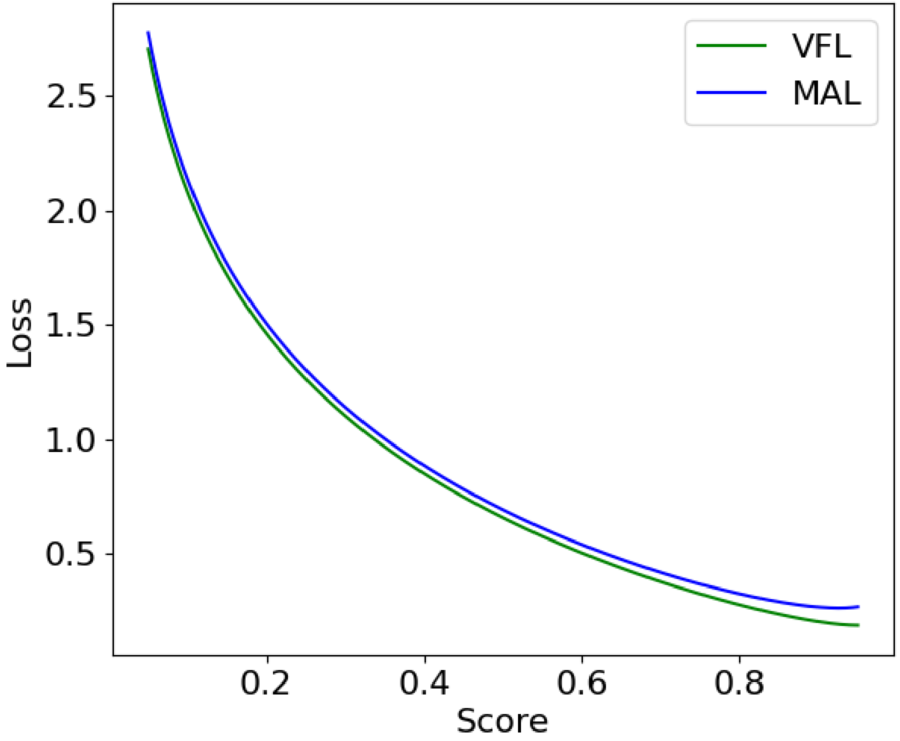
1. MAL punishes the low-quality matched queries a lot

Optimization – VFL vs. MAL

Toy example – high-quality matching



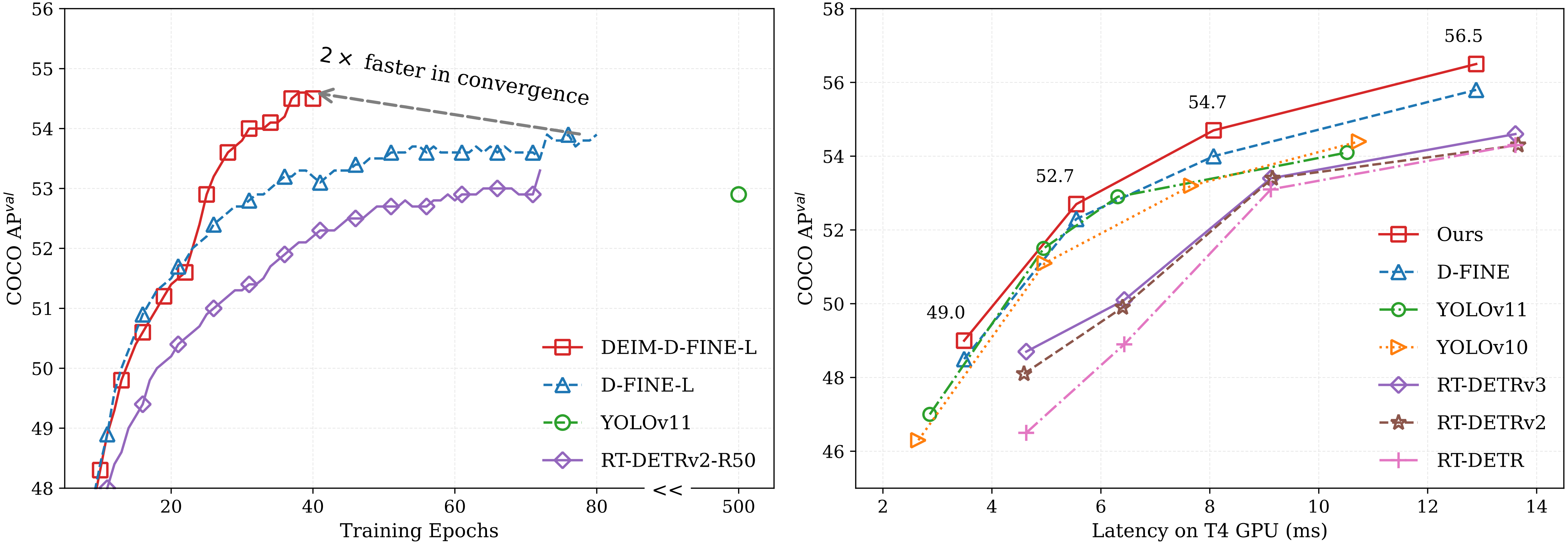
High-quality matching: IoU@0.95



- Comparison between VFL and MAL:

    1. MAL and VFL perform similarly for high-quality matched queries

Main results -- overview



Main results – real-time detectors

Model	#Epochs	#Params	GFLOPs	Latency (ms)	AP^{val}	AP^{val}_{50}	AP^{val}_{75}	AP^{val}_{S}	AP^{val}_{M}	AP^{val}_{L}
YOLO-based Real-time Object Detectors										
YOLOv8-L [12]	500	43	165	12.31	52.9	69.8	57.5	35.3	58.3	69.8
YOLOv8-X [12]	500	68	257	16.59	53.9	71.0	58.7	35.7	59.3	70.7
YOLOv9-C [33]	500	25	102	10.66	53.0	70.2	57.8	36.2	58.5	69.3
YOLOv9-E [33]	500	57	189	20.53	55.6	72.8	60.6	40.2	61.0	71.4
Gold-YOLO-L [32]	300	75	152	9.21	53.3	70.9	-	33.8	58.9	69.9
YOLOv10-L* [31]	500	24	120	7.66	53.2	70.1	58.1	35.8	58.5	69.4
YOLOv10-X* [31]	500	30	160	10.74	54.4	71.3	59.3	37.0	59.8	70.9
YOLO11-L* [13]	500	25	87	6.31	52.9	69.4	57.7	35.2	58.7	68.8
YOLO11-X* [13]	500	57	195	10.52	54.1	70.8	58.9	37.0	59.2	69.7
DETR-based Real-time Object Detectors										
RT-DETR-HG-L [42]	72	32	107	8.77	53.0	71.7	57.3	34.6	57.4	71.2
RT-DETR-HG-X [42]	72	67	234	13.51	54.8	73.1	59.4	35.7	59.6	72.9
D-FINE-L [27]	72	31	91	8.07	54.0	71.6	58.4	36.5	58.0	71.9
DEIM-D-FINE-L	**50**	31	91	8.07	**54.7**	72.4	59.4	36.9	59.6	71.8
D-FINE-X [27]	72	62	202	12.89	55.8	73.7	60.2	37.3	60.5	73.4
DEIM-D-FINE-X	**50**	62	202	12.89	**56.5**	74.0	61.5	38.8	61.4	74.2

- Comparisons with real-time detectors:

  1. Paired with D-FINE, DEIMs exceed all real-time detectors in the trade-off accuracy and latency

Main results – small-sized real-time detectors

Model	#Epochs	#Params.	GFLOPs	Latency (ms)	AP^{val}	AP^{val}_{50}	AP^{val}_{75}	AP^{val}_{S}	AP^{val}_{M}	AP^{val}_{L}
YOLO-based Real-time Object Detectors										
YOLOv8-S [12]	500	11	29	6.96	44.9	61.8	48.6	25.7	49.9	61.0
YOLOv8-M [12]	500	26	79	9.66	50.2	67.2	54.6	32.0	55.7	66.4
YOLOv9-S [33]	500	7	26	8.02	46.8	61.8	48.6	25.7	49.9	61.0
YOLOv9-M [33]	500	20	76	10.15	51.4	67.2	54.6	32.0	55.7	66.4
Gold-YOLO-S [32]	300	22	46	2.01	46.4	63.4	-	25.3	51.3	63.6
Gold-YOLO-M [32]	300	41	88	3.21	51.1	68.5	-	32.3	56.1	68.6
YOLOv10-S [31]	500	7	22	2.65	46.3	63.0	50.4	26.8	51.0	63.8
YOLOv10-M [31]	500	15	59	4.97	51.1	68.1	55.8	33.8	56.5	67.0
YOLO11-S* [13]	500	9	22	2.86	47.0	63.9	50.7	29.0	51.7	64.4
YOLO11-M* [13]	500	20	68	4.95	51.5	68.5	55.7	33.4	57.1	67.9
DETR-based Real-time Object Detectors										
RT-DETR-R18 [42]	72	20	61	4.63	46.5	63.8	50.4	28.4	49.8	63.0
RT-DETR-R34 [42]	72	31	93	6.43	48.9	66.8	52.9	30.6	52.4	66.3
RT-DETRv2-S [24]	120	20	60	4.59	48.1	65.1	57.4	36.1	57.9	70.8
DEIM-RT-DETRv2-S	120	20	60	4.59	49.0	66.1	53.3	32.6	52.5	64.1
RT-DETRv2-M [24]	120	31	92	6.40	49.9	67.5	58.6	35.8	58.6	72.1
DEIM-RT-DETRv2-M	120	31	92	6.40	50.9	68.6	55.2	34.3	54.4	67.1
RT-DETRv2-M* [24]	72	33	100	6.90	51.9	69.9	56.5	33.5	56.8	69.2
DEIM-RT-DETRv2-M*	60	33	100	6.90	53.2	71.2	57.8	35.3	57.6	70.2
D-FINE-S [27]	120	10	25	3.49	48.5	65.6	52.6	29.1	52.2	65.4
DEIM-D-FINE-S	120	10	25	3.49	49.0	65.9	53.1	30.4	52.6	65.7
D-FINE-M [27]	120	19	57	5.55	52.3	69.8	56.4	33.2	56.5	70.2
DEIM-D-FINE-M	90	19	57	5.55	52.7	70.0	57.3	35.3	56.7	69.5

- Comparisons with real-time detectors:

  1. Paired with D-FINE, DEIMs exceed all real-time detectors in the trade-off accuracy and latency

Main results – ResNet-based DETRs

Model	#Epochs	#Params	GFLOPs	AP^{val}	AP^{val}_{50}	AP^{val}_{75}	AP^{val}_{S}	AP^{val}_{M}	AP^{val}_{L}
ResNet50 [14]-based									
DETR-DC5 [3]	500	41	187	43.3	63.1	45.9	22.5	47.3	61.1
Anchor-DETR-DC5 [35]	50	39	172	44.2	64.7	47.5	24.7	48.2	60.6
Conditional-DETR-DC5 [26]	108	44	195	45.1	65.4	48.5	25.3	49.0	62.2
Efficient-DETR [36]	36	35	210	45.1	63.1	49.1	28.3	48.4	59.0
SMCA-DETR [11]	108	40	152	45.6	65.5	49.1	25.9	49.3	62.6
Deformable-DETR [45]	50	40	173	46.2	65.2	50.0	28.8	49.2	61.7
DAB-Deformable-DETR [21]	50	48	195	46.9	66.0	50.8	30.1	50.4	62.5
DAB-Deformable-DETR++ [21]	50	47	-	48.7	67.2	53.0	31.4	51.6	63.9
DN-Deformable-DETR [18]	50	48	195	48.6	67.4	52.7	31.0	52.0	63.7
DN-Deformable-DETR++ [18]	50	47	-	49.5	67.6	53.8	31.3	52.6	65.4
DINO-Deformable-DETR [39]	36	47	279	50.9	69.0	55.3	34.6	54.1	64.6
RT-DETR [43]	72	42	136	53.1	71.3	57.7	34.8	58.0	70.0
RT-DETRv2 [24]	72	42	136	53.4	71.6	57.4	36.1	57.9	70.8
DEIM-RT-DETRv2	**36**	42	136	53.9	71.7	58.6	36.7	58.9	70.9
DEIM-RT-DETRv2	**60**	42	136	**54.3**	72.3	58.8	37.5	58.7	70.8
ResNet101 [14]-based									
DETR-DC5 [3]	500	60	253	44.9	64.7	47.7	23.7	49.5	62.3
Anchor-DETR-DC5 [35]	50	-	-	45.1	65.7	48.8	25.8	49.4	61.6
Conditional-DETR-DC5 [26]	108	63	262	45.9	66.8	49.5	27.2	50.3	63.3
Efficient-DETR [36]	36	54	289	45.7	64.1	49.5	28.2	49.1	60.2
SMCA-DETR [11]	108	58	218	46.3	66.6	50.2	27.2	50.5	63.2
RT-DETR [43]	72	76	259	54.3	72.7	58.6	36.0	58.8	72.1
RT-DETRv2 [24]	72	76	259	54.3	72.8	58.8	35.8	58.8	72.1
DEIM-RT-DETRv2	**36**	76	259	55.2	73.3	59.9	37.8	59.6	72.8
DEIM-RT-DETRv2	**60**	76	259	**55.5**	73.5	60.3	37.9	59.9	73.0

- Comparisons with ResNet-based DETR:

1. DEIMs consistently outperform all DETRs, in particular RT-DETRv2 by ~1 AP
2. DEIMs achieve much better performance on small objects than any DETRs

Main results – CrowdHuman

Method	AP	AP_{50}	AP_{75}	AP_s	AP_m	AP_l
D-FINE-L	56.0	87.2	59.4	29.0	46.1	54.6
w/ DEIM	**57.5**	**87.6**	**62.9**	**33.2**	**48.7**	**55.7**

- Comparisons on CrowdHuman:

1. CrowdHuman is a more challenging dataset that contains dense crowd scenarios
2. DEIM shows 1.5 AP improvement over D-FINE-L, especially APs and AP75
3. Demonstrate the strong generalization capability of DEIM

Ablation study – Dense O2O with Mosaic

- Observations:

  1. The number of GT in 'an' image increases by times
  2. More small objects by zoom-out

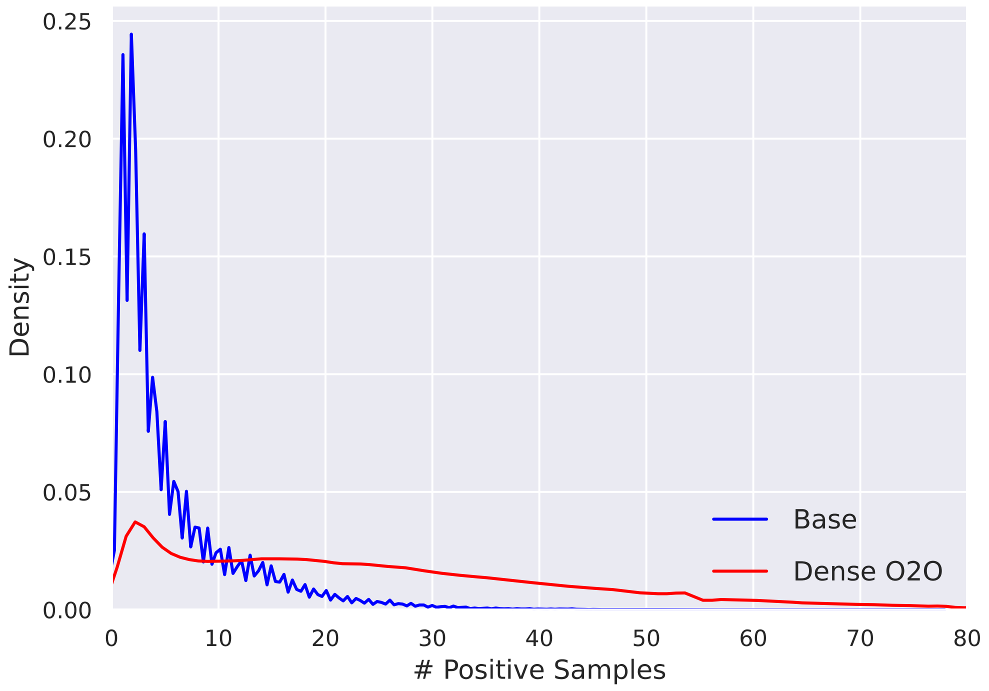
Ablation study – Dense O2O with MixUp

- Observations:

    1. The number of GT in ‘an’ image also increases by times.

Ablation study – Dense O2O

Mosaic Prob.	Mixup Prob.	AP	AP_{50}	AP_{75}	AP_s	AP_m	AP_l
Training 12 Epochs							
0.0	0.0	49.6	67.1	53.6	31.3	54.2	67.8
0.5	0.0	**50.4**	**68.4**	**54.5**	**32.7**	54.6	68.1
0.0	0.5	50.1	67.7	54.0	31.1	54.5	**68.7**
0.5	0.5	**50.4**	68.1	54.2	**32.7**	**54.7**	68.2
Training 24 Epochs							
0.0	0.0	51.7	69.5	55.8	32.8	56.4	69.7
0.5	0.0	51.9	70.1	55.9	**34.9**	56.1	69.3
0.0	0.5	51.5	69.4	55.5	33.2	56.3	69.3
0.5	0.5	**52.5**	**70.6**	**56.7**	**34.9**	**57.1**	**70.1**



- Methods for Dense O2O:

  1. Both mosaic and mixUp can improve training convergence, and they are complementary
  2. Mosaic improves the performance of small objects by a large margin
  3. Dense O2O by Mosaic and Mixup increases # positive samples in training, enhancing supervision

Ablation study – Dense O2O & MAL

Epochs	Dense O2O	MAL	AP	AP_{50}	AP_{75}
RT-DETRv2-R50 [24]					
72			53.4	71.6	57.4
36	✓		53.6	71.9	58.2
	✓	✓	**53.9**	**71.7**	**58.6**
D-FINE-L [27]					
72			54.0	71.6	58.4
36	✓		54.2	72.1	58.9
	✓	✓	**54.6**	**72.2**	**59.5**

- Effectiveness of Dense O2O & MAL:
  1. Dense O2O significantly accelerates model convergence
  2. Our MAL further improves the model performance

Visualizations

In each paired image: **D-FINE-L** (left); **DEIM-D-FINE-L** (right). Confidence threshold@0.5.

- Observations:

  1. D-FINE-L faces highly-overlapped predictions (top) and false positives (bottom).
  2. By training with our DEIM, those problems can be mitigated.

Conclusion

1. DEIM is a simple and flexible training framework for real-time object detection.

2. DEIM accelerates the convergence by improving the quantity and quality of matching with Dense O2O and MAL.

3. With our DEIM, existing real-time DETRs achieve better performance while saving training costs.

Paper

Code

Thanks!

Attention: Our Intellindust AI Lab is seeking self-motivated and passionate researchers and interns to join our team and drive cutting-edge advancements in artificial intelligence for industrial applications. Contact me with shihuahuang95@gmail.com