A. Appendix

A.1. Implementation details

A.1.1 Architectures

Figure A1 and Figure A2 show the architectures of the proposed YOLOv7 P5 and YOLOv7 P6, respectively.

	YOLOv7	YOLOv7x	YOLOv7-tiny
Stage 0	3x3/1 Conv, 32	3x3/1 Conv, 40	
Stage 1	3x3/2 Conv, 64 3x3/1 Conv, 64	3x3/2 Conv, 80 3x3/1 Conv, 80	3x3/2 Conv, 32
Stage 2	3x3/2 Conv, 128 2-4 ELAN, 256	3x3/2 Conv, 160 2-6 ELAN, 320	3x3/2 Conv, 64 1-2 ELAN, 64
Stage 3	/2 Down, 256 2-4 ELAN, 512	/2 Down, 320 2-6 ELAN, 640	/2 MaxPool 1-2 ELAN, 128
Stage 4	/2 Down, 512 2-4 ELAN, 1024	/2 Down, 640 2-6 ELAN, 1280	/2 MaxPool 1-2 ELAN, 256
Stage 5	/2 Down, 1024 2-4 ELAN, 1024	/2 Down, 1280 2-6 ELAN, 1280	/2 MaxPool 1-2 ELAN, 512
Stage 5	CSPSPP, 512	CSPSPP, 640	CSPSPP, 256
Stage 4	*2 Up, 512 1-4 ELAN, 256	*2 Up, 640 2-6 ELAN, 320	*2 Up, 256 1-2 ELAN, 128
Stage 3	*2 Up, 256 1-4 ELAN, 128	*2 Up, 320 2-6 ELAN, 160	*2 Up, 128 1-2 ELAN, 64
Stage 4	/2 Down, 512 1-4 ELAN, 256	/2 Down, 640 2-6 ELAN, 320	/2 Down, 256 1-2 ELAN, 128
Stage 5	/2 Down, 1024 1-4 ELAN, 512	/2 Down, 1280 2-6 ELAN, 640	/2 Down, 512 1-2 ELAN, 256

Figure A1. Architectures of YOLOv7 P5 models.

	YOLOv7-W6	YOLOv7-E6	YOLOv7-D6	YOLOv7-E6E
Stage 1	/2 ReOrg 3x3/1 Conv, 64	/2 ReOrg 3x3/1 Conv, 80	/2 ReOrg 3x3/1 Conv, 96	/2 ReOrg 3x3/1 Conv, 80
Stage 2	3x3/2 Conv, 128 2-4 ELAN, 128	/2 Down, 160 2-6 ELAN, 160	/2 Down, 192 2-8 ELAN, 192	/2 Down, 160 2-6 E-ELAN, 160
Stage 3	3x3/2 Conv, 256 2-4 ELAN, 256	/2 Down, 320 2-6 ELAN, 320	/2 Down, 384 2-8 ELAN, 384	/2 Down, 320 2-6 E-ELAN, 320
Stage 4	3x3/2 Conv, 512 2-4 ELAN, 512	/2 Down, 640 2-6 ELAN, 640	/2 Down, 768 2-8 ELAN, 768	/2 Down, 640 2-6 E-ELAN, 640
Stage 5	3x3/2 Conv, 768 2-4 ELAN, 768	/2 Down, 960 2-6 ELAN, 960	/2 Down, 1152 2-8 ELAN, 1152	/2 Down, 960 2-6 E-ELAN, 960
Stage 6	3x3/2 Conv, 1024 2-4 ELAN, 1024	/2 Down, 1280 2-6 ELAN, 1280	/2 Down, 1536 2-8 ELAN, 1536	/2 Down, 1280 2-6 E-ELAN, 1280
Stage 6	CSPSPP, 512	CSPSPP, 640	CSPSPP, 768	CSPSPP, 640
Stage 5	*2 Up, 768 1-4 ELAN, 384	*2 Up, 960 1-6 ELAN, 480	*2 Up, 1152 1-8 ELAN, 576	*2 Up, 960 1-6 E-ELAN, 480
Stage 4	*2 Up, 512 1-4 ELAN, 256	*2 Up, 640 1-6 ELAN, 320	*2 Up, 768 1-8 ELAN, 384	*2 Up, 640 1-6 E-ELAN, 320
Stage 3	*2 Up, 256 1-4 ELAN, 128	*2 Up, 320 1-6 ELAN, 160	*2 Up, 384 1-8 ELAN, 192	*2 Up, 320 1-6 E-ELAN, 160
Stage 4	/2 Down, 512 1-4 ELAN, 256	/2 Down, 640 1-6 ELAN, 320	/2 Down, 768 1-8 ELAN, 384	/2 Down, 640 1-6 E-ELAN, 320
Stage 5	/2 Down, 768 1-4 ELAN, 384	/2 Down, 960 1-6 ELAN, 480	/2 Down, 1152 1-8 ELAN, 576	/2 Down, 960 1-6 E-ELAN, 480
Stage 6	/2 Down, 1024 1-4 ELAN, 512	/2 Down, 1280 1-6 ELAN, 640	/2 Down, 1536 1-8 ELAN, 768	/2 Down, 1280 1-6 E-ELAN, 640

Figure A2. Architectures of YOLOv7 P6 models.

For E-ELAN architecture, since our edge device do not support group convolution and shuffle operation, we are forced to implement it as an equivalence architecture, which is shown in Figure A3.

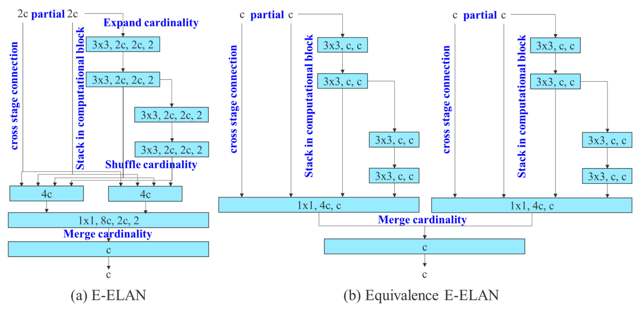
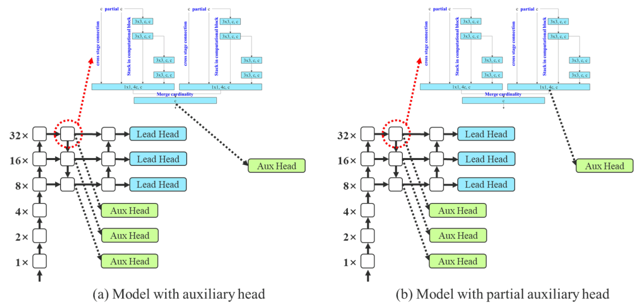


Figure A3. Equivalence E-ELAN.

The designed equivalence E-ELAN makes it easier for us to implement partial auxiliary head. E-ELAN with normal auxiliary head and partial auxilary head are shown in Figure A4.



32×
16×
8×
4×
2×
1×
Lead Head
Aux Head
(a) Model with auxiliary head
(b) Model with partial auxiliary head

Figure A4. Partial auxilary head on E-ELAN.

Here we show how we make coarse-to-fine constraint lead head guided label assigner in Figure A5. We make a dynamic constraint by limiting the decoder of two additional candidate positive grids (yellow grids in the figure). Theoretically, yellow grids need to predict the range in [-1, 2] to fit ground truth bounding box, and we make the decoder can only predict in the range of [-0.5, 1.5]. This constraint makes the model can automatically learn the pink grids and yellow grid at different levels.

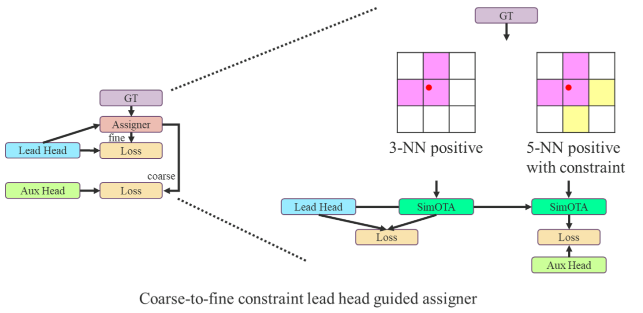


Figure A5. Coarse-to-fine lead head guided label assigner.

A.1.2 Hyper-parameters

We have three different training hyper-parameters settings. One is for YOLOv7-tiny, one is for YOLOv7 and YOLOv7x, and the last one is for YOLOv7-W6, YOLOv7-E6, YOLOv7-D6, and YOLOv7-E6E. The hyper-parameter setting are follow updated YOLOR github code.

An additional training hyper-parameter is top k of simOTA. To train 640 $\times$ 640 models, we follow YOLOX to use $k = 10$. For 1280 $\times$ 1280 models, as described in previous section, 3-NN candidate postive grids are used in our training. That is to say, when input resolution grows, the candidate positive grids will grow along two directions, not four directions. Therefore, we set $k = 20$ on simOTA to train 1280 $\times$ 1280 models.

A.1.3 Re-parameterization

Merging Convolution-BatchNorm-Activation into Convolution-Activation at inference time is a well-known technique, and its cprresponding formulas are shown in Figure A6. RepConv has become a popular re-parameterization method in recent years. Here we show how the YOLOR implicit knowledge can be merged into convolutional layer when addition or multiplication is used to combine implicit knowledge and representation. Figure A7 shows the formula to re-parameterize the implicit knowledge of YOLOR and the convolutional layer.

convolution → batch normalization → activation function

$$\begin{aligned} &(((wx + b) - m) / v) \\ &= (wx + (b-m)) / v \\ &= (w/v)x + (b-m)/v \\ &= w'x + b' \end{aligned}$$

Figure A6. Batch normalization as trainable BoF.

YOLOR+ → convolution → YOLOR+

$$\begin{aligned} &w(x + g_1(z_1)) + b + g_2(z_2) \\ &= wx + (b + wg_1(z_1) + g_2(z_2)) \\ &= wx + b' \end{aligned}$$

YOLOR* → convolution → YOLOR*

$$\begin{aligned} &(w\,(g_1(z_1)x) + b)g_2(z_2) \\ &= g_1(z_1)g_2(z_2)wx + bg_2(z_2) \\ &= w'x + b' \end{aligned}$$

Figure A7. YOLOR implicit knowledge as trainable BoF.

A.2. More experiments

In Table A1 and Table A2 we respectively show the full ablation studies of proposed auxiliary head learning strategies and ablation studies of proposed compound scaling method and extended ELAN.

Table A1. Ablation study on auxiliary head.

Model	AP^{val}	AP^{val}_{50}	AP^{val}_{75}
base (v7-E6)	55.6%	73.2%	60.7%
independent	55.8%	73.4%	60.9%
lead guided	55.9%	73.5%	61.0%
c2f lead guided without constraint	55.9%	73.5%	61.0%
c2f lead guided with constraint	**55.9%**	**73.5%**	**61.1%**
improvement	+0.3	+0.3	+0.4
base (v7-E6E)	56.3%	74.0%	61.5%
c2f lead guided with constraint	56.5%	74.0%	61.6%
partial c2f lead guided with constraint	**56.8%**	**74.4%**	**62.1%**
improvement	+0.3	+0.4	+0.5

Table A2. Ablation study on compond scaling and E-ELAN.

Model	#Param.	FLOPs	AP^{val}	AP^{val}_{50}	AP^{val}_{75}
base (v7-E6)	97.2M	515.2G	55.9%	73.5%	61.1%
compond scaling (v7-D6)	154.7M	806.8G	56.3%	73.8%	61.4%
extended ELAN (v7-E6E)	151.7M	843.2G	**56.8%**	**74.4%**	**62.1%**
improvement	-	-	+0.5	+0.6	+0.7

In Figure A8 we show the objectness map predicted by different methods at auxiliary head and lead head. From Figure A8 we find that if auxiliary head learns lead guided soft label, it will indeed help lead head to extract the residual information from the consistant targets.

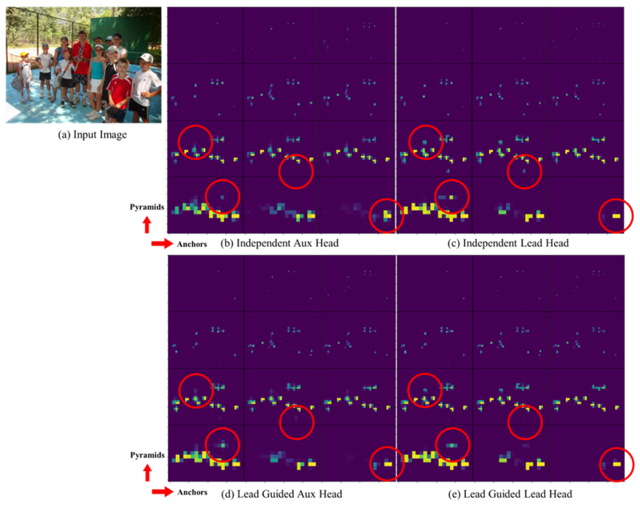


Figure A8. Objectness map predicted by different methods at auxiliary head and lead head.

Table A3. More comparison (batch=1, no-TRT, without extra object detection training data, sorted by FLOPs)

Model	**#Param.**	**FLOPs**	**Size**	**FPS^{V100}**	**AP^{test} / AP^{val}**	**AP$_{50}^{test}$**	**AP$_{75}^{test}$**
YOLOv5-N (r6.1) [1]	1.9M	4.5G	640	159	- / **28.0%**	-	-
YOLOv7-tiny-SiLU	6.2M	13.8G	640	286	**38.7% / 38.7%**	**56.7%**	**41.7%**
YOLOv5-S (r6.1) [1]	7.2M	16.5G	640	156	- / 37.4%	-	-
PPYOLOE-S [2]	7.9M	17.4G	640	208	**43.1% / 42.7%**	**60.5%**	**46.6%**
YOLOv5-N6 (r6.1) [1]	3.2M	18.4G	1280	123	- / 36.0%	-	-
YOLOX-S [3]	9.0M	26.8G	640	102	40.5% / 40.5%	-	-
YOLOv5-M (r6.1) [1]	21.2M	49.0G	640	122	- / **45.4%**	-	-
PPYOLOE-M [2]	23.4M	49.9G	640	123	**48.9% / 48.6%**	**66.5%**	**53.0%**
YOLOv5-S6 (r6.1) [1]	12.6M	67.2G	1280	122	- / 44.8%	-	-
YOLOX-M [3]	25.3M	73.8G	640	81	47.2% / 46.9%	-	-
YOLOv7	36.9M	104.7G	640	161	**51.4% / 51.2%**	**69.7%**	**55.9%**
YOLOv5-L (r6.1) [1]	46.5M	109.1G	640	99	- / 49.0%	-	-
PPYOLOE-L [2]	52.2M	110.1G	640	78	51.4% / 50.9%	68.9%	55.6%
YOLOR-CSP [4]	52.9M	120.4G	640	106	51.1% / 50.8%	69.6%	55.7%
YOLOX-L [3]	54.2M	155.6G	640	69	50.1% / 49.7%	-	-
Deformable DETR [5]	40.0M	173.0G	-	19	- / 46.2%	-	-
YOLOv7-X	71.3M	189.9G	640	114	**53.1% / 52.9%**	**71.2%**	**57.8%**
YOLOv5-M6 (r6.1) [1]	35.7M	200.0G	1280	90	- / 51.3%	-	-
YOLOv5-X (r6.1) [1]	86.7M	205.7G	640	83	- / 50.7%	-	-
PPYOLOE-X [2]	98.4M	206.6G	640	45	52.2% / 51.9%	69.9%	56.5%
YOLOR-CSP-X [4]	96.9M	226.8G	640	87	53.0% / 52.7%	**71.4%**	**57.9%**
F-RCNN-R101-FPN+ [6]	60.0M	246.0G	1333	20	- / 44.0%	-	-
DETR DC5-R101 [6]	60.0M	253.0G	1333	10	- / 44.9%	-	-
YOLOX-X [3]	99.1M	281.9G	640	58	51.5% / 51.1%	-	-
YOLOR-P6 [4]	37.2M	325.6G	1280	76	**53.9% / 53.5%**	71.4%	**58.9%**
YOLOv7-W6	70.4M	360.0G	1280	84	**54.9% / 54.6%**	**72.6%**	**60.1%**
EfficientDet-D7x [7]	77.0M	410.0G	1536	6.5	**55.1%** / 54.4%	72.4%	58.4%
YOLOv5-L6 (r6.1) [1]	76.8M	445.6G	1280	63	- / 53.7%	-	-
YOLOR-W6 [4]	79.8G	453.2G	1280	66	**55.2% / 54.8%**	**72.7%**	**60.5%**
YOLOv7-E6	97.2M	515.2G	1280	56	**56.0% / 55.9%**	**73.5%**	**61.2%**
YOLOR-E6 [4]	115.8M	683.2G	1280	45	55.8% / 55.7%	73.4%	61.1%
YOLOv7-D6	154.7M	806.8G	1280	44	**56.6% / 56.3%**	**74.0%**	**61.8%**
Dual-Swin-T (C-M-RCNN) [8]	113.8M	836.0G	1333	6.5	- / 53.6%	-	-
YOLOv5-X6 (r6.1) [1]	140.7M	839.2G	1280	38	- / 55.0%	-	-
YOLOv7-E6E	151.7M	843.2G	1280	36	**56.8% / 56.8%**	**74.4%**	**62.1%**
YOLOR-D6 [4]	151.7M	935.6G	1280	34	56.5% / 56.1%	**74.1%**	**61.9%**
Swin-B (C-M-RCNN) [9]	145.0M	975.0G	1333	5.9	- / 51.9%	-	-
Swin-B (C-M-RCNN) [9]	145.0M	982.0G	1333	11.6	- / 51.9%	-	-
ViT-Adapter-B [10]	122.0M	997.0G	-	4.4	- / 50.8%	-	-
Dual-Swin-B (HTC) [8]	235.0M	-	1600	2.5	**58.7% / 58.4%**	-	-
Dual-Swin-L (HTC) [8]	453.0M	-	1600	1.5	**59.4% / 59.1%**	-	-

Model	**#Param.**	**FLOPs**	**Size**	**FPS^{A100}**	**AP^{test} / AP^{val}**	**AP$_{50}^{test}$**	**AP$_{75}^{test}$**
DN-Deformable-DETR [11]	48.0M	265.0G	1333	23.0	- / 48.6%	-	-
DINO-5scale (R50) [12]	47.0M	860.0G	1333	10.0	- / 51.0%	-	-
ConvNeXt-B (C-M-RCNN) [13]	-	964.0G	1280	11.5	- / 54.0%	73.1%	58.8%
Swin-B (C-M-RCNN) [9]	-	982.0G	1280	10.7	- / 53.0%	71.8%	57.5%
ConvNeXt-L (C-M-RCNN) [13]	-	1354.0G	1280	10.0	- / 54.8%	73.8%	59.8%
Swin-L (C-M-RCNN) [9]	-	1382.0G	1280	9.2	- / 53.9%	72.4%	58.8%
ConvNeXt-XL (C-M-RCNN) [13]	-	1898.0G	1280	8.6	- / 55.2%	74.2%	59.9%

B. More comparison

As shown in Table A3 and Figure A9, YOLOv7 surpasses all known object detectors in both speed and accuracy in the range from 5 FPS to 160 FPS and has the highest accuracy 56.8% AP test-dev / 56.8% AP min-val among all known real-time object detectors with 30 FPS or higher on GPU V100. YOLOv7-E6 object detector (56 FPS V100, 55.9% AP) outperforms both transformer-based detector SWIN-L Cascade-Mask R-CNN (9.2 FPS A100, 53.9% AP) by 509% in speed and 2% in accuracy, and convolutional-based detector ConvNeXt-XL Cascade-Mask R-CNN (8.6 FPS A100, 55.2% AP) by 551% in speed and 0.7% AP in accuracy, as well as YOLOv7 outperforms: YOLOR, YOLOX, Scaled-YOLOv4, YOLOv5, DETR, Deformable DETR, DINO-5scale-R50, ViT-Adapter-B and many other object detectors in speed and accuracy.

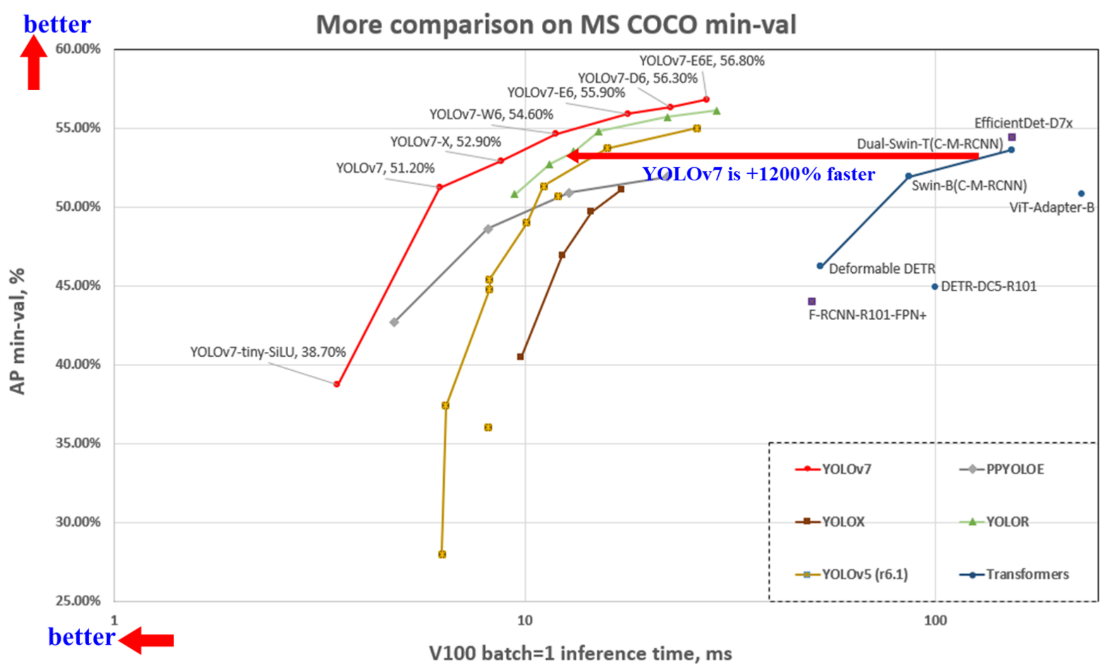
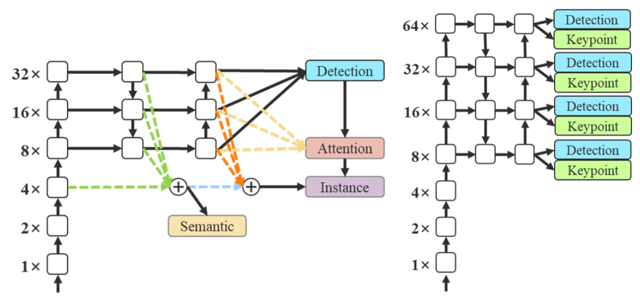


Figure A9. Comparison with other object detectors.

B.1. More results



(a) YOLOv7-mask (b) YOLOv7-pose

Figure A10. Architectures of YOLOv7-mask and YOLOv7-pose.

B.1.1 YOLOv7-mask

We integrate YOLOv7 with BlendMask [14] to do instance segmentation. We simply fine-tune YOLOv7 object detection model on MS COCO instance segmentation dataset and trained for 30 epochs. It achieves state-of-the-art real-time instance segmentation result. The architecture of YOLOv7-mask and the corresponding results are shown in Figure A10 (a) and Figure A11, respectively.

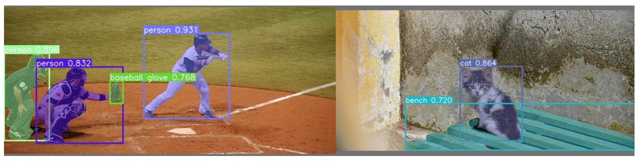


Figure A11. Sample results after applying YOLOv7-mask.

B.1.2 YOLOv7-pose

We integrate YOLOv7 with YOLO-Pose [15] to do keypoint detection. We follow the same setting as [15] to fine-tune YOLOv7-W6 people detection model on MS COCO keypoint detection dataset. YOLOv7-W6-pose achieves state-of-the-art real-time pose estimation result. The architecture and sample results are shown in Figure A10 (b) and Figure A12, respectively.

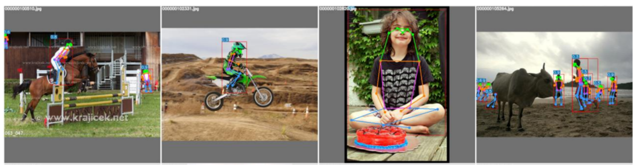


Figure A12. Sample results after applying YOLOv7-pose.

B.1.3 YOLOv7-AF

We integrate YOLOv7 with YOLOv6 (v1.0) [16] and YOLOv5 (r7.0) [18] to do anchor-free object detection and make comparison to methods released after CVPR submission in Table A4.

Table A4. YOLOv7-AF (anchor free).

Model	date	#Param.	FLOPs	AP^{val}	AP^{val}_{50}	AP^{val}_{75}
PPYOLOEX [2]	(2022.03)	98.4M	206.6G	52.3%	69.5%	56.8%
YOLOv6L (v2.0) [17]	(2022.09)	58.5M	144.0G	51.0%	-	-
YOLOv5X (r7.0) [18]	(2022.11)	86.7M	205.7G	50.7%	68.9%	-
YOLOv7-AF	(2022.11)	43.6M	130.5G	53.0%	70.2%	57.5%
RTMDetX [19]	(2022.12)	94.9M	141.7G	52.8%	70.4%	-
YOLOv8L [20]	(2023.01)	43.7M	165.2G	52.9%	-	-
YOLOv6L (v3.0) [21]	(2023.01)	59.6M	150.7G	51.8%	69.2%	-

[1] All models are trained from scratch and without knowledge distillation.

References

[1] Jocher Glenn and others. YOLOv5 release v6.1. https://github.com/ultralytics/yolov5/releases/tag/v6.1, 2022. 3

[2] Shangliang Xu, Xinxin Wang, Wenyu Lv, Qinyao Chang, Cheng Cui, Kaipeng Deng, Guanzhong Wang, Qingqing Dang, Shengyu Wei, Yuning Du, et al. PP-YOLOE: An evolved version of YOLO. *arXiv preprint arXiv:2203.16250*, 2022. 3, 4

[3] Zheng Ge, Songtao Liu, Feng Wang, Zeming Li, and Jian Sun. YOLOX: Exceeding YOLO series in 2021. *arXiv preprint arXiv:2107.08430*, 2021. 3

[4] Chien-Yao Wang, I-Hau Yeh, and Hong-Yuan Mark Liao. You only learn one representation: Unified network for multiple tasks. *arXiv preprint arXiv:2105.04206*, 2021. 3

[5] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable DETR: Deformable transformers for end-to-end object detection. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021. 3

[6] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 213–229, 2020. 3

[7] Mingxing Tan, Ruoming Pang, and Quoc V Le. EfficientDet: Scalable and efficient object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10781–10790, 2020. 3

[8] Tingting Liang, Xiaojie Chu, Yudong Liu, Yongtao Wang, Zhi Tang, Wei Chu, Jingdong Chen, and Haibin Ling. CBNetV2: A composite backbone network architecture for object detection. *arXiv preprint arXiv:2107.00420*, 2021. 3

[9] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 10012–10022, 2021. 3

[10] Zhe Chen, Yuchen Duan, Wenhai Wang, Junjun He, Tong Lu, Jifeng Dai, and Yu Qiao. Vision transformer adapter for dense predictions. *arXiv preprint arXiv:2205.08534*, 2022. 3

[11] Feng Li, Hao Zhang, Shilong Liu, Jian Guo, Lionel M Ni, and Lei Zhang. DN-DETR: Accelerate detr training by introducing query denoising. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13619–13627, 2022. 3

[12] Hao Zhang, Feng Li, Shilong Liu, Lei Zhang, Hang Su, Jun Zhu, Lionel M Ni, and Heung-Yeung Shum. DINO: DETR with improved denoising anchor boxes for end-to-end object detection. *arXiv preprint arXiv:2203.03605*, 2022. 3

[13] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A ConvNet for the 2020s. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11976–11986, 2022. 3

[14] Chen *et al*. BlendMask: Top-down meets bottom-up for instance segmentation. *CVPR*, 2020. 4

[15] Maji *et al*. YOLO-Pose: Enhancing YOLO for Multi Person Pose Estimation Using Object Keypoint Similarity Loss. *CVPRW*, 2022. 4

[16] Chilicyy and others. YOLOv6. https://github.com/meituan/YOLOv6/releases/tag/0.1.0, 2022. 4

[17] Chuyi Li, Lulu Li, Hongliang Jiang, Kaiheng Weng, Yifei Geng, Liang Li, Zaidan Ke, Qingyuan Li, Meng Cheng, Weiqiang Nie, and others. YOLOv6: A single-stage object detection framework for industrial applications. *arXiv preprint arXiv:2209.02976*, 2022. 4

[18] Jocher Glenn and others. YOLOv5 release v7.0. https://github.com/ultralytics/yolov5/releases/tag/v7.0, 2022. 4

[19] Chengqi Lyu, Wenwei Zhang, Haian Huang, Yue Zhou, Yudong Wang, Yanyi Liu, Shilong Zhang, and Kai Chen. RTMDet: An Empirical Study of Designing Real-Time Object Detectors. *arXiv preprint arXiv:2212.07784*, 2022. 4

[20] Jocher Glenn and others. YOLOv8. https://github.com/ultralytics/ultralytics, 2023. 4

[21] Chuyi Li, Lulu Li, Yifei Geng, Hongliang Jiang, Meng Cheng, Bo Zhang, Zaidan Ke, Xiaoming Xu, and Xiangxiang Chu. YOLOv6 v3.0: A Full-Scale Reloading. *arXiv preprint arXiv:2301.05586*, 2023. 4