Supplementary of YOLOv9

Chien-Yao Wang[1,2], I-Hau Yeh[2], and Hong-Yuan Mark Liao[1,2,3]

[1] Institute of Information Science, Academia Sinica, Taiwan
[2] National Taipei University of Technology, Taiwan
[3] Department of Information and Computer Engineering,
Chung Yuan Christian University, Taiwan
kinyiu@iis.sinica.edu.tw, ihyeh@emc.com.tw, liao@iis.sinica.edu.tw

A Implementation Details

Table 1: Hyper parameter settings of YOLOv9.

hyper parameter	**value**
epochs	500
optimizer	SGD
initial learning rate	0.01
finish learning rate	0.0001
learning rate decay	linear
momentum	0.937
weight decay	0.0005
warm-up epochs	3
warm-up momentum	0.8
warm-up bias learning rate	0.1
box loss gain	7.5
class loss gain	0.5
DFL loss gain	1.5
HSV saturation augmentation	0.7
HSV value augmentation	0.4
translation augmentation	0.1
scale augmentation	0.9
mosaic augmentation	1.0
MixUp augmentation	0.15
copy & paste augmentation	0.3
close mosaic epochs	15

The training parameters of YOLOv9 are shown in Table 1. We fully follow the settings of YOLOv7 AF [10], which is to use SGD optimizer to train 500 epochs. We first warm-up for 3 epochs and only update the bias during the warm-up stage. Next we step down from the initial learning rate 0.01 to 0.0001 in linear decay manner, and the data augmentation settings are listed in the bottom part of Table 1. We shut down mosaic data augmentation operations on the last 15 epochs.

Table 2: Network configurations of YOLOv9.

Index	Module	Route	Filters	Depth	Size	Stride
0	Conv	–	64	–	3	2
1	Conv	0	128	–	3	2
2	CSP-ELAN	1	256, 128, 64	2, 1	–	1
3	DOWN	2	256	–	3	2
4	CSP-ELAN	3	512, 256, 128	2, 1	–	1
5	DOWN	4	512	–	3	2
6	CSP-ELAN	5	512, 512, 256	2, 1	–	1
7	DOWN	6	512	–	3	2
8	CSP-ELAN	7	512, 512, 256	2, 1	–	1
9	SPP-ELAN	8	512, 256, 256	3, 1	–	1
10	Up	9	512	–	–	2
11	Concat	10, 6	1024	–	–	1
12	CSP-ELAN	11	512, 512, 256	2, 1	–	1
13	Up	12	512	–	–	2
14	Concat	13, 4	1024	–	–	1
15	CSP-ELAN	14	256, 256, 128	2, 1	–	1
16	DOWN	15	256	–	3	2
17	Concat	16, 12	768	–	–	1
18	CSP-ELAN	17	512, 512, 256	2, 1	–	1
19	DOWN	18	512	–	3	2
20	Concat	19, 9	1024	–	–	1
21	CSP-ELAN	20	512, 512, 256	2, 1	–	1
22	Predict	15, 18, 21	–	–	–	–

The network topology of YOLOv9 completely follows YOLOv7 AF [10], that is, we replace ELAN with the proposed CSP-ELAN block. As listed in Table 2, the depth parameters of CSP-ELAN are represented as ELAN depth and CSP depth, respectively. As for the parameters of CSP-ELAN filters, they are represented as ELAN output filter, CSP output filter, and CSP inside filter. In the down-sampling module part, we simplify CSP-DOWN module to DOWN module. DOWN module is composed of a pooling layer with size 2 and stride 1, and a Conv layer with size 3 and stride 2. Finally, we optimized the prediction layer and replaced top, left, bottom, and right in the regression branch with decoupled branch.

Table 3: Comparison of object detectors with different training settings.

	Model	#Param.	FLOPs	$AP_{50:95}$	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
From-scratch	Dy-YOLOv7 [5]	–	181.7	53.9	72.2	58.7	35.3	57.6	66.4
	Dy-YOLOv7-X [5]	–	307.9	55.0	73.2	60.0	36.6	58.7	68.5
	YOLOv9-S (Ours)	7.1	26.4	46.8	63.4	50.7	26.6	56.0	64.5
	YOLOv9-M (Ours)	20.0	76.3	51.4	68.1	56.1	33.6	57.0	68.0
	YOLOv9-C (Ours)	25.3	102.1	53.0	70.2	57.8	36.2	58.5	69.3
	YOLOv9-E (Ours)	34.7	147.1	54.5	71.7	59.2	38.1	59.9	70.3
	YOLOv9-E (Ours)	44.0	183.9	55.1	72.3	60.7	38.7	60.6	71.4
	YOLOv9-E (Ours)	57.3	189.0	55.6	72.8	60.6	40.2	61.0	71.4
ImageNet Pretrained	RTMDet-T [7]	4.8	12.6	41.1	57.9	–	–	–	–
	RTMDet-S [7]	9.0	25.6	44.6	61.9	–	–	–	–
	RTMDet-M [7]	24.7	78.6	49.4	66.8	–	–	–	–
	RTMDet-L [7]	52.3	160.4	51.5	68.8	–	–	–	–
	RTMDet-X [7]	94.9	283.4	52.8	70.4	–	–	–	–
	PPYOLOE-S [13]	7.9	14.4	43.0	60.5	46.6	23.2	46.4	56.9
	PPYOLOE-M [13]	23.4	49.9	49.0	66.5	53.0	28.6	52.9	63.8
	PPYOLOE-L [13]	52.2	110.1	51.4	68.9	55.6	31.4	55.3	66.1
	PPYOLOE-X [13]	98.4	206.6	52.3	69.5	56.8	35.1	57.0	68.6
	RT DETR-L [6]	32	110	53.0	71.6	57.3	34.6	57.3	71.2
	RT DETR-X [6]	67	234	54.8	73.1	59.4	35.7	59.6	72.9
	RT DETR-R18 [6]	20	60	46.5	63.8	–	–	–	–
	RT DETR-R34 [6]	31	92	48.9	66.8	–	–	–	–
	RT DETR-R50M [6]	36	100	51.3	69.6	–	–	–	–
	RT DETR-R50 [6]	42	136	53.1	71.3	57.7	34.8	58.0	70.0
	RT DETR-R101 [6]	76	259	54.3	72.7	58.6	36.0	58.8	72.1
	Gold YOLO-S [9]	21.5	46.0	45.5	62.2	–	–	–	–
	Gold YOLO-M [9]	41.3	57.5	50.2	67.5	–	–	–	–
	Gold YOLO-L [9]	75.1	151.7	52.3	69.6	–	–	–	–
Distillation	YOLOv6-N v3.0 [4]	4.7	11.4	37.5	53.1	–	–	–	–
	YOLOv6-S v3.0 [4]	18.5	45.3	45.0	61.8	–	–	–	–
	YOLOv6-M v3.0 [4]	34.9	85.8	50.0	66.9	–	–	–	–
	YOLOv6-L v3.0 [4]	59.6	150.7	52.8	70.3	–	–	–	–
	DAMO YOLO-T [14]	8.5	18.1	43.6	59.4	46.6	23.3	47.4	61.0
	DAMO YOLO-S [14]	16.3	37.8	47.7	63.5	51.1	26.9	51.7	64.9
	DAMO YOLO-M [14]	28.2	61.8	50.4	67.2	55.1	31.6	55.3	67.1
	DAMO YOLO-L [14]	42.1	97.3	51.9	68.5	56.7	33.3	57.0	67.6
	Gold YOLO-N [9]	5.6	12.1	39.9	55.9	–	–	–	–
	Gold YOLO-S [9]	21.5	46.0	46.1	63.3	–	–	–	–
	Gold YOLO-M [9]	41.3	57.5	50.9	68.2	–	–	–	–
	Gold YOLO-L [9]	75.1	151.7	53.2	70.5	–	–	–	–
Complex Setting	Gold YOLO-S [9]	21.5	46.0	46.4	63.4	–	–	–	–
	Gold YOLO-M [9]	41.3	57.5	51.1	68.5	–	–	–	–
	Gold YOLO-L [9]	75.1	151.7	53.3	70.9	–	–	–	–
	YOLOR-CSP [12]	52.9	120.4	52.8	71.2	57.6	–	–	–
	YOLOR-CSP-X [12]	96.9	226.8	54.8	73.1	59.7	–	–	–
	PPYOLOE+-S [13]	7.9	14.4	43.7	60.6	47.9	23.2	46.4	56.9
	PPYOLOE+-M [13]	23.4	49.9	49.8	67.1	54.5	31.8	53.9	66.2
	PPYOLOE+-L [13]	52.2	110.1	52.9	70.1	57.9	35.2	57.5	69.1
	PPYOLOE+-X [13]	98.4	206.6	54.7	72.0	59.9	37.9	59.3	70.4

[1] Param. (M); FLOPs (G); APs (%).

B More Comparison

We compare YOLOv9 to state-of-the-art real-time object detectors trained with different methods. It mainly includes four different training methods: (1) train-from-scratch: we have completed most of the comparisons in the text. Here are only list of additional data of DynamicDet [5] for comparisons; (2) Pretrained by ImageNet: this includes two methods of using ImageNet for supervised pretrain and self-supervised pretrain; (3) knowledge distillation: a method to perform additional self-distillation after training is completed; and (4) a more complex training process: a combination of steps including pretrained by ImageNet, knowledge distillation, DAMO-YOLO and even additional pretrained large object detection dataset. We show the results in Table 3. From this table, we can see that our proposed YOLOv9 performed better than all other methods. Compared with PPYOLOE+-X trained using ImageNet and Objects365, our method still reduces the number of parameters by 55% and the amount of computation by 11%, and improving 0.4% AP.

Table 4: Comparison of state-of-the-art object detectors with different training settings (sorted by number of parameters).

Model	#Param.	FLOPs	$AP^{val}_{50:95}$	AP^{val}_{50}	AP^{val}_{75}	AP^{val}_{S}	AP^{val}_{M}	AP^{val}_{L}
YOLOv6-N v3.0 [4] (D)	4.7	11.4	**37.5**	53.1	–	–	–	–
RTMDet-T [7] (I)	4.8	12.6	**41.1**	57.9	–	–	–	–
Gold YOLO-N [9] (D)	5.6	12.1	39.9	55.9	–	–	–	–
YOLOv9-S (S)	7.1	26.4	**46.8**	63.4	50.7	26.6	56.0	64.5
PPYOLOE+-S [13] (C)	7.9	14.4	43.7	60.6	47.9	23.2	46.4	56.9
PPYOLOE-S [13] (I)	7.9	14.4	43.0	60.5	46.6	23.2	46.4	56.9
DAMO YOLO-T [14] (D)	8.5	18.1	43.6	59.4	46.6	23.3	47.4	61.0
RTMDet-S [7] (I)	9.0	25.6	44.6	61.9	–	–	–	–
DAMO YOLO-S [14] (D)	16.3	37.8	**47.7**	63.5	51.1	26.9	51.7	64.9
YOLOv6-S v3.0 [4] (D)	18.5	45.3	45.0	61.8	–	–	–	–
RT DETR-R18 [6] (I)	20	60	46.5	63.8	–	–	–	–
YOLOv9-M (S)	20.0	76.3	**51.4**	68.1	56.1	33.6	57.0	68.0
Gold YOLO-S [9] (C)	21.5	46.0	46.4	63.4	–	–	–	–
Gold YOLO-S [9] (D)	21.5	46.0	46.1	63.3	–	–	–	–
Gold YOLO-S [9] (I)	21.5	46.0	45.5	62.2	–	–	–	–
PPYOLOE+-M [13] (C)	23.4	49.9	49.8	67.1	54.5	31.8	53.9	66.2
PPYOLOE-M [13] (I)	23.4	49.9	49.0	66.5	53.0	28.6	52.9	63.8
RTMDet-M [7] (I)	24.7	78.6	49.4	66.8	–	–	–	–
YOLOv9-C (S)	25.3	102.1	**53.0**	70.2	57.8	36.2	58.5	69.3
DAMO YOLO-M [14] (D)	28.2	61.8	50.4	67.2	55.1	31.6	55.3	67.1
RT DETR-R34 [6] (I)	31	92	48.9	66.8	–	–	–	–
RT DETR-L [6] (I)	32	110	53.0	71.6	57.3	34.6	57.3	71.2
YOLOv9-E (S)	34.7	147.1	**54.5**	71.7	59.2	38.1	59.9	70.3
YOLOv6-M v3.0 [4] (D)	34.9	85.8	50.0	66.9	–	–	–	–
RT DETR-R50M [6] (I)	36	100	51.3	69.6	–	–	–	–
Gold YOLO-M [9] (C)	41.3	57.5	51.1	68.5	–	–	–	–
Gold YOLO-M [9] (D)	41.3	57.5	50.9	68.2	–	–	–	–
Gold YOLO-M [9] (I)	41.3	57.5	50.2	67.5	–	–	–	–
RT DETR-R50 [6] (I)	42	136	53.1	71.3	57.7	34.8	58.0	70.0
DAMO YOLO-L [14] (D)	42.1	97.3	51.9	68.5	56.7	33.3	57.0	67.6
YOLOv9-E (S)	44.0	183.9	**55.1**	72.3	60.7	38.7	60.6	71.4
PPYOLOE+-L [13] (C)	52.2	110.1	52.9	70.1	57.9	35.2	57.5	69.1
PPYOLOE-L [13] (I)	52.2	110.1	51.4	68.9	55.6	31.4	55.3	66.1
RTMDet-L [7] (I)	52.3	160.4	51.5	68.8	–	–	–	–
YOLOR-CSP [12] (C)	52.9	120.4	52.8	71.2	57.6	–	–	–
YOLOv9-E (S)	57.3	189.0	**55.6**	72.8	60.6	40.2	61.0	71.4
YOLOv6-L v3.0 [4] (D)	59.6	150.7	52.8	70.3	–	–	–	–
RT DETR-X [6] (I)	67	234	54.8	73.1	59.4	35.7	59.6	72.9
Gold YOLO-L [9] (C)	75.1	151.7	53.3	70.9	–	–	–	–
Gold YOLO-L [9] (D)	75.1	151.7	53.2	70.5	–	–	–	–
Gold YOLO-L [9] (I)	75.1	151.7	52.3	69.6	–	–	–	–
RT DETR-R101 [6] (I)	76	259	54.3	72.7	58.6	36.0	58.8	72.1
RTMDet-X [7] (I)	94.9	283.4	52.8	70.4	–	–	–	–
YOLOR-CSP-X [12] (C)	96.9	226.8	54.8	73.1	59.7	–	–	–
PPYOLOE+-X [13] (C)	98.4	206.6	54.7	72.0	59.9	37.9	59.3	70.4
PPYOLOE-X [13] (I)	98.4	206.6	52.3	69.5	56.8	35.1	57.0	68.6

[1] (S), (I), (D), (C) indicate from-scratch, ImageNet pretrained, distillation, and complex setting, respectively.

Table 4 shows the performance of all models sorted by parameter size. Our proposed YOLOv9 is Pareto optimal in all models of different sizes. Among them, we found no other method for Pareto optimal in models with more than 20M parameters. The above experimental data shows that our YOLOv9 has excellent parameter usage efficiency.

Table 5: Comparison of state-of-the-art object detectors with different training settings (sorted by amount of computation).

Model	#Param.	FLOPs	$AP^{val}_{50:95}$	AP^{val}_{50}	AP^{val}_{75}	AP^{val}_{S}	AP^{val}_{M}	AP^{val}_{L}
YOLOv6-N v3.0 [4] (D)	4.7	11.4	**37.5**	53.1	–	–	–	–
Gold YOLO-N [9] (D)	5.6	12.1	**39.9**	55.9	–	–	–	–
RTMDet-T [7] (I)	4.8	12.6	**41.1**	57.9	–	–	–	–
PPYOLOE+-S [13] (C)	7.9	14.4	**43.7**	60.6	47.9	23.2	46.4	56.9
PPYOLOE-S [13] (I)	7.9	14.4	43.0	60.5	46.6	23.2	46.4	56.9
DAMO YOLO-T [14] (D)	8.5	18.1	43.6	59.4	46.6	23.3	47.4	61.0
RTMDet-S [7] (I)	9.0	25.6	**44.6**	61.9	–	–	–	–
YOLOv9-S (S)	7.1	26.4	**46.8**	63.4	50.7	26.6	56.0	64.5
DAMO YOLO-S [14] (D)	16.3	37.8	**47.7**	63.5	51.1	26.9	51.7	64.9
YOLOv6-S v3.0 [4] (D)	18.5	45.3	45.0	61.8	–	–	–	–
Gold YOLO-S [9] (C)	21.5	46.0	46.4	63.4	–	–	–	–
Gold YOLO-S [9] (D)	21.5	46.0	46.1	63.3	–	–	–	–
Gold YOLO-S [9] (I)	21.5	46.0	45.5	62.2	–	–	–	–
PPYOLOE+-M [13] (C)	23.4	49.9	**49.8**	67.1	54.5	31.8	53.9	66.2
PPYOLOE-M [13] (I)	23.4	49.9	49.0	66.5	53.0	28.6	52.9	63.8
Gold YOLO-M [9] (C)	41.3	57.5	**51.1**	68.5	–	–	–	–
Gold YOLO-M [9] (D)	41.3	57.5	50.9	68.2	–	–	–	–
Gold YOLO-M [9] (I)	41.3	57.5	50.2	67.5	–	–	–	–
RT DETR-R18 [6] (I)	20	60	46.5	63.8	–	–	–	–
DAMO YOLO-M [14] (D)	28.2	61.8	50.4	67.2	55.1	31.6	55.3	67.1
YOLOv9-M (S)	20.0	76.3	**51.4**	68.1	56.1	33.6	57.0	68.0
RTMDet-M [7] (I)	24.7	78.6	49.4	66.8	–	–	–	–
YOLOv6-M v3.0 [4] (D)	34.9	85.8	50.0	66.9	–	–	–	–
RT DETR-R34 [6] (I)	31	92	48.9	66.8	–	–	–	–
DAMO YOLO-L [14] (D)	42.1	97.3	**51.9**	68.5	56.7	33.3	57.0	67.6
RT DETR-R50M [6] (I)	36	100	51.3	69.6	–	–	–	–
YOLOv9-C (S)	25.3	102.1	**53.0**	70.2	57.8	36.2	58.5	69.3
RT DETR-L [6] (I)	32	110	53.0	71.6	57.3	34.6	57.3	71.2
PPYOLOE+-L [13] (C)	52.2	110.1	52.9	70.1	57.9	35.2	57.5	69.1
PPYOLOE-L [13] (I)	52.2	110.1	51.4	68.9	55.6	31.4	55.3	66.1
YOLOR-CSP [12] (C)	52.9	120.4	52.8	71.2	57.6	–	–	–
RT DETR-R50 [6] (I)	42	136	**53.1**	71.3	57.7	34.8	58.0	70.0
YOLOv9-E (S)	34.7	147.1	**54.5**	71.7	59.2	38.1	59.9	70.3
YOLOv6-L v3.0 [4] (D)	59.6	150.7	52.8	70.3	–	–	–	–
Gold YOLO-L [9] (C)	75.1	151.7	53.3	70.9	–	–	–	–
Gold YOLO-L [9] (D)	75.1	151.7	53.2	70.5	–	–	–	–
Gold YOLO-L [9] (I)	75.1	151.7	52.3	69.6	–	–	–	–
RTMDet-L [7] (I)	52.3	160.4	51.5	68.8	–	–	–	–
Dy-YOLOv7 [5] (S)	–	181.7	53.9	72.2	58.7	35.3	57.6	66.4
YOLOv9-E (S)	44.0	183.9	**55.1**	72.3	60.7	38.7	60.6	71.4
YOLOv9-E (S)	57.3	189.0	**55.6**	72.8	60.6	40.2	61.0	71.4
PPYOLOE+-X [13] (C)	98.4	206.6	54.7	72.0	59.9	37.9	59.3	70.4
PPYOLOE-X [13] (I)	98.4	206.6	52.3	69.5	56.8	35.1	57.0	68.6
YOLOR-CSP-X [12] (C)	96.9	226.8	54.8	73.1	59.7	–	–	–
RT DETR-X [6] (I)	67	234	54.8	73.1	59.4	35.7	59.6	72.9
RT DETR-R101 [6] (I)	76	259	54.3	72.7	58.6	36.0	58.8	72.1
RTMDet-X [7] (I)	94.9	283.4	52.8	70.4	–	–	–	–
Dy-YOLOv7-X [5] (S)	–	307.9	55.0	73.2	60.0	36.6	58.7	68.5

[1] (S), (I), (D), (C) indicate from-scratch, ImageNet pretrained, distillation, and complex setting, respectively.

Shown in Table 5 is the performance of all participating models sorted by the amount of computation. Our proposed YOLOv9 is Pareto optimal in all models with different scales. Among models with more than 60 GFLOPs, only ELAN-based DAMO-YOLO and DETR-based RT DETR can rival the proposed YOLOv9. The above comparison results show that YOLOv9 has the most outstanding performance in the trade-off between computation complexity and accuracy.

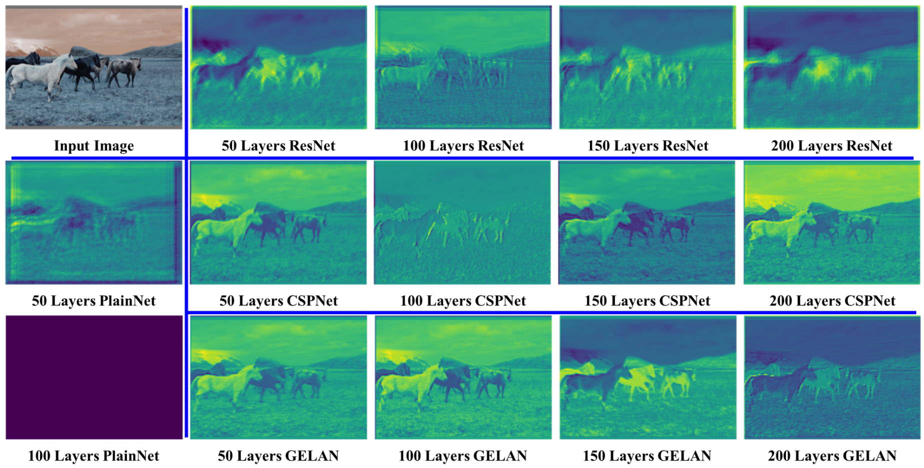

Fig. 1: Feature maps (visualization results) output by random initial weights of PlainNet, ResNet, CSPNet, and GELAN at different depths. After 100 layers, ResNet begins to produce feedforward output that is enough to obfuscate object information. Our proposed GELAN can still retain quite complete information up to the 150^{th} layer, and is still sufficiently discriminative up to the 200^{th} layer.

In Figure 1 we show the visualization results of feature maps obtained by different networks. We can see that as the number of layers increases, the original information of all networks gradually decreases. For example, at the 50^{th} layer of the PlainNet, it is difficult to see the location of objects, and all distinguishable features will be lost at the 100^{th} layer. As for ResNet, the boundary information has been lost at the 50^{th} layer, and becomes blurry when the depth reached to the 100^{th} layer. Both CSPNet and the proposed GELAN maintain sufficient information for object detection until the 200^{th} layer, and GELAN has more stable results and clearer boundary information.

Table 6: Comparison with state-of-the-art instance segmentors.

Model	#Param.	FLOPs	$\mathrm{AP}^{box}_{50:95}$	$\mathrm{AP}^{mask}_{50:95}$
YOLOv9-C-seg (Ours)	**27.4M**	**144.6G**	**53.3%**	43.5%
YOLOv5-L-seg [1]	47.9M	147.7G	49.1%	40.0%
FastInst [3]	∼35M	∼151G	–	40.1%
YOLOR-ELAN-AF [11]	45.9M	172.6G	53.0%	43.3%
RTMDet-L-inst [7]	57.4M	213.2G	51.2%	**43.7%**
YOLOv8-L-seg [2]	46.0M	220.5G	52.3%	42.6%
CondInst [8]	33.9M	240.8G	42.6%	38.2%

Shown in Table 6 is the comparison of state-of-the-art real-time instance segmentors. Our proposed YOLOv9 has the best parameter utilization and the lowest computation cost. Compare with YOLOR-ELAN-AF [11], the proposed YOLOv9-C-seg gets on par accuracy with only 60% number of parameters and 84% computation cost.

References

1. Glenn, J.: YOLOv5 release v7.0. https://github.com/ultralytics/yolov5/releases/tag/v7.0 (2022)
2. Glenn, J.: YOLOv8 release v8.1.0. https://github.com/ultralytics/ultralytics/releases/tag/v8.1.0 (2024)
3. He, J., Li, P., Geng, Y., Xie, X.: FastInst: A simple query-based model for real-time instance segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 23663–23672 (2023)
4. Li, C., Li, L., Geng, Y., Jiang, H., Cheng, M., Zhang, B., Ke, Z., Xu, X., Chu, X.: YOLOv6 v3.0: A full-scale reloading. arXiv preprint arXiv:2301.05586 (2023)
5. Lin, Z., Wang, Y., Zhang, J., Chu, X.: DynamicDet: A unified dynamic architecture for object detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 6282–6291 (2023)
6. Lv, W., Xu, S., Zhao, Y., Wang, G., Wei, J., Cui, C., Du, Y., Dang, Q., Liu, Y.: DETRs beat YOLOs on real-time object detection. arXiv preprint arXiv:2304.08069 (2023)
7. Lyu, C., Zhang, W., Huang, H., Zhou, Y., Wang, Y., Liu, Y., Zhang, S., Chen, K.: RTMDet: An empirical study of designing real-time object detectors. arXiv preprint arXiv:2212.07784 (2022)
8. Tian, Z., Shen, C., Chen, H.: Conditional convolutions for instance segmentation. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 282–298 (2020)
9. Wang, C., He, W., Nie, Y., Guo, J., Liu, C., Han, K., Wang, Y.: Gold-YOLO: Efficient object detector via gather-and-distribute mechanism. Advances in Neural Information Processing Systems (NeurIPS) (2023)
10. Wang, C.Y., Bochkovskiy, A., Liao, H.Y.M.: YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 7464–7475 (2023)
11. Wang, C.Y., Liao, H.Y.M., Yeh, I.H.: Designing network design strategies through gradient path analysis. Journal of Information Science and Engineering (JISE) (2023)
12. Wang, C.Y., Yeh, I.H., Liao, H.Y.M.: You only learn one representation: Unified network for multiple tasks. Journal of Information Science & Engineering (JISE) (2023)
13. Xu, S., Wang, X., Lv, W., Chang, Q., Cui, C., Deng, K., Wang, G., Dang, Q., Wei, S., Du, Y., et al.: PP-YOLOE: An evolved version of YOLO. arXiv preprint arXiv:2203.16250 (2022)
14. Xu, X., Jiang, Y., Chen, W., Huang, Y., Zhang, Y., Sun, X.: DAMO-YOLO: A report on real-time object detection design. arXiv preprint arXiv:2211.15444 (2022)