DynamicDet: A Unified Dynamic Architecture for Object Detection

Zhihao Lin Yongtao Wang[†] Jinhe Zhang Xiaojie Chu
Wangxuan Institute of Computer Technology, Peking University
linzhihao@stu.pku.edu.cn, wyt@pku.edu.cn
jinhezhang17@gmail.com, chuxiaojie@stu.pku.edu.cn

A. Pseudo code for dynamic detector

We present the pseudo code of training the adaptive router on Algorithm 1, and the dynamic inference on Algorithm 2.

Input: The dynamic detector constructed by the first backbone $\mathcal{B}_1$, the first neck and head $\mathcal{D}_1$, the second backbone $\mathcal{B}_2$, the second neck and head $\mathcal{D}_2$, the composite connection module $\mathcal{G}$, and the adaptive router $\mathcal{R}$. The median of the training loss difference between two detectors Δ. Input images $\mathbf{x}_i \in \mathbf{X}$ and the corresponding ground truths $\overline{\mathbf{y}}_i \in \overline{\mathbf{Y}}$. Training iteration T.

for $i = 1, \ldots, T$ **do**

$F_1 = \mathcal{B}_1(\mathbf{x}_i)$; // Extract the first multi-scale features.

$\mathbf{y}_1 = \mathcal{D}_1(F_1)$; // Predict the detection results by the first detector.

$\phi = \mathcal{R}(F_1)$;// Predict the difficulty score.

$H = \mathcal{G}(F_1)$; // Embed the first multi-scale features.

$F_2 = \mathcal{B}_2(\mathbf{x}_i, H)$; // Extract the enhanced multi-scale features based on the input image and the embedding of previous multi-scale features.

$\mathbf{y}_2 = \mathcal{D}_2(F_2)$; // Predict the detection results by the second detector.

$\mathcal{L} = ((1-\phi)(\mathcal{L}_{det}(\mathbf{y}_1, \overline{\mathbf{y}}_i) - \Delta/2) + \phi(\mathcal{L}_{det}(\mathbf{y}_2, \overline{\mathbf{y}}_i) + \Delta/2))$;// Loss.

update the parameters of adaptive router based on the gradient from loss $\mathcal{L}$.

end

Algorithm 1: Pseudo code of training the adaptive router on DynamicDet.

Input: The dynamic detector constructed by the first backbone $\mathcal{B}_1$, the first neck and head $\mathcal{D}_1$, the second backbone $\mathcal{B}_2$, the second neck and head $\mathcal{D}_2$, the composite connection module $\mathcal{G}$, and the adaptive router $\mathcal{R}$. Input image $\mathbf{x}$. Threshold τ.

Output: Predicted detection results $\mathbf{y}$

$F_1 = \mathcal{B}_1(\mathbf{x})$; // Extract the first multi-scale features.

$\phi = \mathcal{R}(F_1)$;// Predict the difficulty score.

if $\phi \leq \tau$ **then**

// Easy image.

$\mathbf{y} = \mathcal{D}_1(F_1)$; // Predict the detection results by the first detector.

else

// Hard image.

$H = \mathcal{G}(F_1)$; // Embed the first multi-scale features.

$F_2 = \mathcal{B}_2(\mathbf{x}, H)$; // Extract the enhanced multi-scale features based on the input image and the embedding of previous multi-scale features.

$\mathbf{y} = \mathcal{D}_2(F_2)$; // Predict the detection results by the second detector.

end

Algorithm 2: Pseudo code of dynamic inference on DynamicDet.

B. Additional results

B.1. More comparison on real-time object detection

We present more precision results (*e.g.*, AP_{50}) to compare with other real-time object detectors in Tab. 4.

[†]Corresponding author.

Model	Size	FLOPs	FPS	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
EAutoDet-S [6]	640	24.9G	120†	40.1	58.7	43.5	21.7	43.8	50.5
EAutoDet-M [6]	640	60.8G	70†	45.2	63.5	49.1	25.7	49.1	57.3
EAutoDet-L [6]	640	115.4G	59†	47.9	66.3	52.0	28.3	52.0	59.9
EAutoDet-X [6]	640	225.3G	41†	49.2	67.5	53.6	30.4	53.4	61.5
EfficientDet-D0 [4]	512	2.5G	98†	34.6	53.0	37.1	-	-	-
EfficientDet-D1 [4]	640	6.1G	74†	40.5	59.1	43.7	-	-	-
YOLOX-S [1]	640	26.8G	102†	40.5	-	-	-	-	-
YOLOX-M [1]	640	73.8G	81†	47.2	-	-	-	-	-
YOLOX-L [1]	640	155.6G	69†	50.1	-	-	-	-	-
YOLOX-X [1]	640	281.9G	58†	51.5	-	-	-	-	-
YOLOv5-N (r6.2) [2]	640	4.5G	200	28.1	46.2	29.4	12.8	31.3	35.4
YOLOv5-S (r6.2) [2]	640	16.5G	196	37.7	57.3	40.5	19.8	41.7	47.4
YOLOv5-M (r6.2) [2]	640	49.0G	137	45.4	64.3	49.2	26.3	49.9	56.4
YOLOv5-L (r6.2) [2]	640	109.1G	114	49.0	67.5	53.1	29.8	53.4	61.2
YOLOv5-X (r6.2) [2]	640	205.7G	100	50.9	69.2	55.1	31.9	55.2	63.6
YOLOv6-N [3]	640	11.1G	216	36.4	51.9	39.2	15.5	39.5	50.6
YOLOv6-T [3]	640	36.7G	206	41.2	57.9	44.6	19.9	45.0	56.0
YOLOv6-S [3]	640	44.2G	184	43.9	60.9	47.5	22.2	47.9	58.9
YOLOv6-M [3]	640	82.2G	109	49.8	67.0	54.3	28.5	54.6	65.4
YOLOv6-L [3]	640	144.0G	76	52.3	69.9	56.8	31.6	57.2	67.8
PP-YOLOE+-S [7]	640	17.4G	208†	43.9	-	-	-	-	-
PP-YOLOE+-M [7]	640	49.9G	123†	50.0	-	-	-	-	-
PP-YOLOE+-L [7]	640	110.1G	78†	53.3	-	-	-	-	-
PP-YOLOE+-X [7]	640	206.6G	45†	54.9	-	-	-	-	-
YOLOv7 [5]	640	104.7G	114	51.4	69.7	55.9	31.8	55.5	65.0
Dy-YOLOv7 / 10	**640**	**112.4G**	**110**	**52.1**	**70.5**	**56.8**	**33.3**	**55.9**	**64.7**
Dy-YOLOv7 / 50	**640**	**143.2G**	**96**	**53.3**	**71.7**	**58.1**	**34.9**	**57.0**	**65.4**
Dy-YOLOv7 / 90	**640**	**174.0G**	**85**	**53.8**	**72.2**	**58.7**	**35.3**	**57.5**	**66.3**
Dy-YOLOv7 / 100	**640**	**181.7G**	**83**	**53.9**	**72.2**	**58.7**	**35.3**	**57.6**	**66.4**
YOLOv7-X [5]	640	189.9G	105	53.1	71.2	57.8	33.8	57.1	67.4
Dy-YOLOv7-X / 10	**640**	**201.7G**	**98**	**53.3**	**71.6**	**58.0**	**34.2**	**57.1**	**67.1**
Dy-YOLOv7-X / 50	**640**	**248.9G**	**78**	**54.4**	**72.7**	**59.3**	**36.0**	**58.0**	**67.7**
Dy-YOLOv7-X / 90	**640**	**296.1G**	**65**	**55.0**	**73.2**	**59.9**	**36.6**	**58.6**	**68.2**
Dy-YOLOv7-X / 100	**640**	**307.9G**	**64**	**55.0**	**73.2**	**60.0**	**36.6**	**58.7**	**68.5**
EfficientDet-D2 [4]	768	11.0G	56†	43.9	62.7	47.6	-	-	-
EfficientDet-D3 [4]	896	25.0G	34†	47.2	65.9	51.2	-	-	-
EfficientDet-D4 [4]	1024	55.0G	23†	49.7	68.4	53.9	-	-	-
EfficientDet-D5 [4]	1280	135.0G	14†	51.5	70.5	56.1	-	-	-
EfficientDet-D6 [4]	1280	226.0G	11†	52.6	71.5	57.2	-	-	-
EfficientDet-D7 [4]	1536	325.0G	8†	53.7	72.4	58.4	-	-	-
EfficientDet-D7X [4]	1536	410.0G	7†	55.1	74.3	59.9	-	-	-
YOLOv5-N6 (r6.2) [2]	1280	18.4G	161	36.2	55.0	39.0	19.4	39.3	45.2
YOLOv5-S6 (r6.2) [2]	1280	67.2G	152	44.6	63.9	48.6	26.4	48.3	55.1
YOLOv5-M6 (r6.2) [2]	1280	200.0G	96	51.4	69.7	56.0	33.3	55.2	62.5
YOLOv5-L6 (r6.2) [2]	1280	445.6G	65	53.8	71.8	58.5	36.3	57.6	65.0
YOLOv5-X6 (r6.2) [2]	1280	839.2G	39	55.0	72.8	59.8	37.3	58.5	66.8
YOLOv7-W6 [5]	1280	360.0G	78	54.9	72.6	60.1	37.3	58.7	67.1
YOLOv7-E6 [5]	1280	515.2G	52	56.0	73.5	61.2	38.0	59.9	68.4
YOLOv7-D6 [5]	1280	806.8G	41	56.6	74.0	61.8	38.8	60.1	69.5
YOLOv7-E6E [5]	1280	843.2G	33	56.8	74.4	62.1	39.3	60.5	69.0
Dy-YOLOv7-W6 / 10	**1280**	**384.2G**	**74**	**55.2**	**73.0**	**60.4**	**37.9**	**58.4**	**66.6**
Dy-YOLOv7-W6 / 50	**1280**	**480.8G**	**58**	**56.1**	**73.8**	**61.4**	**39.3**	**59.3**	**66.9**
Dy-YOLOv7-W6 / 90	**1280**	**577.4G**	**48**	**56.7**	**74.3**	**62.1**	**39.5**	**59.9**	**67.8**
Dy-YOLOv7-W6 / 100	**1280**	**601.6G**	**46**	**56.8**	**74.4**	**62.1**	**39.6**	**59.9**	**68.3**

[1] The FPS marked with † are from the corresponding papers, and others are measured on the same machine with 1 NVIDIA V100 GPU.

Table 4. Comparison of the state-of-the-art real-time object detectors on COCO *test-dev*.

Model	Size	FLOPs	FPS	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Dy-YOLOv7 / 0	640	104.7G	114	51.1	69.5	55.6	31.5	55.2	64.5
Dy-YOLOv7 / 10	640	112.4G	110	52.1	70.5	56.8	33.3	55.9	64.7
Dy-YOLOv7 / 20	640	120.1G	106	52.5	71.0	57.3	34.1	56.2	64.9
Dy-YOLOv7 / 30	640	127.8G	102	52.9	71.3	57.6	34.5	56.5	65.0
Dy-YOLOv7 / 40	640	135.5G	99	53.1	71.6	57.9	34.7	56.8	65.2
Dy-YOLOv7 / 50	640	143.2G	96	53.3	71.7	58.1	34.9	57.0	65.4
Dy-YOLOv7 / 60	640	150.9G	93	53.5	71.9	58.3	35.1	57.2	65.5
Dy-YOLOv7 / 70	640	158.6G	91	53.6	72.0	58.5	35.2	57.4	65.7
Dy-YOLOv7 / 80	640	166.3G	88	53.7	72.1	58.6	35.3	57.5	66.0
Dy-YOLOv7 / 90	640	174.0G	85	53.8	72.2	58.7	35.3	57.5	66.3
Dy-YOLOv7 / 100	640	181.7G	83	53.9	72.2	58.7	35.3	57.6	66.4
Dy-YOLOv7-X / 0	640	189.9G	105	52.6	70.7	57.2	32.9	56.6	67.1
Dy-YOLOv7-X / 10	640	201.7G	98	53.3	71.6	58.0	34.2	57.1	67.1
Dy-YOLOv7-X / 20	640	213.5G	93	53.7	71.9	58.5	34.8	57.3	67.4
Dy-YOLOv7-X / 30	640	225.3G	86	53.9	72.2	58.8	35.3	57.5	67.4
Dy-YOLOv7-X / 40	640	237.1G	82	54.1	72.5	59.0	35.6	57.8	67.4
Dy-YOLOv7-X / 50	640	248.9G	78	54.4	72.7	59.3	36.0	58.0	67.7
Dy-YOLOv7-X / 60	640	260.7G	75	54.6	72.8	59.5	36.3	58.2	67.8
Dy-YOLOv7-X / 70	640	272.5G	70	54.7	72.9	59.6	36.4	58.3	67.8
Dy-YOLOv7-X / 80	640	284.3G	68	54.8	73.0	59.8	36.6	58.4	68.0
Dy-YOLOv7-X / 90	640	296.1G	65	55.0	73.2	59.9	36.6	58.6	68.2
Dy-YOLOv7-X / 100	640	307.9G	64	55.0	73.2	60.0	36.6	58.7	68.5
Dy-YOLOv7-W6 / 0	1280	360.0G	78	54.7	72.4	59.8	36.6	58.1	66.5
Dy-YOLOv7-W6 / 10	1280	384.2G	74	55.2	73.0	60.4	37.9	58.4	66.6
Dy-YOLOv7-W6 / 20	1280	408.3G	69	55.5	73.3	60.8	38.5	58.7	66.7
Dy-YOLOv7-W6 / 30	1280	432.5G	66	55.8	73.5	61.1	38.8	58.9	66.7
Dy-YOLOv7-W6 / 40	1280	456.6G	62	55.9	73.7	61.2	39.1	59.1	66.8
Dy-YOLOv7-W6 / 50	1280	480.8G	58	56.1	73.8	61.4	39.3	59.3	66.9
Dy-YOLOv7-W6 / 60	1280	505.0G	56	56.2	73.9	61.6	39.4	59.4	67.0
Dy-YOLOv7-W6 / 70	1280	529.1G	53	56.3	74.0	61.7	39.4	59.5	67.1
Dy-YOLOv7-W6 / 80	1280	553.3G	51	56.5	74.2	61.9	39.4	59.7	67.5
Dy-YOLOv7-W6 / 90	1280	577.4G	48	56.7	74.3	62.1	39.5	59.9	67.8
Dy-YOLOv7-W6 / 100	1280	601.6G	46	56.8	74.4	62.1	39.6	59.9	68.3

Table 5. Detailed results of dynamic YOLOv7 models on COCO *test-dev*.

B.2. More results for dynamic detectors

We present more results for our dynamic detector (*i.e.*, Dy-YOLOv7 with / 0, / 10, . . . , / 100) in Tab. 5. It is observed that our dynamic detectors can obtain a wide range of trade-offs of different precision and speed by proposed variable-speed inference strategy. For instance, using the same weight with different thresholds for inference, our Dy-YOLOv7-W6 can achieve 54.7%∼56.8% mAP with 78∼46 FPS.

C. Additional analyses

C.1. Consistency of the inference time

To further demonstrate the consistency of the inference time between the validation set and the test set, we compare the inference time of Dy-YOLOv7-W6 on these two sets. Specifically, we calculate the thresholds on the validation set of different sizes (*i.e.*, 0.5k, 2k, 5k) and measure their inference time on the validation set and the test set. As shown in Fig. 11, it is observed that the inference time is consistent between these two sets when calculating the thresholds by 5k validation images, and is very close to the ideal case. Moreover, when the validation set's size decreases, the inference time consistency becomes slightly worse but is still acceptable.

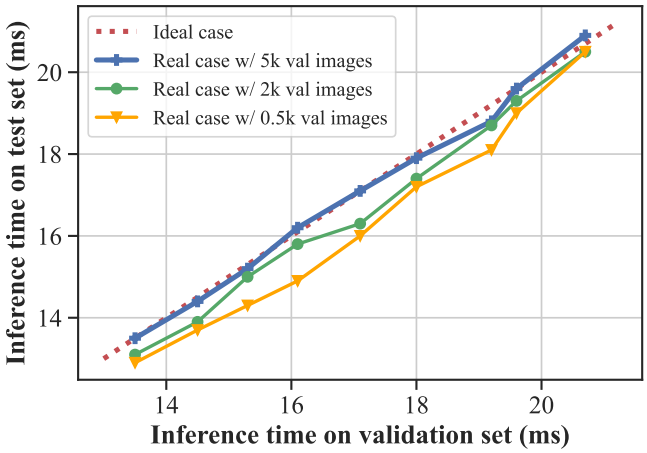

Figure 11. Comparison of the inference time on the validation set and the test set under the different thresholds obtained from the validation set with different size.

References

[1] Zheng Ge, Songtao Liu, Feng Wang, Zeming Li, and Jian Sun. Yolox: Exceeding yolo series in 2021. *arXiv preprint arXiv:2107.08430*, 2021. 2

[2] Jocher Glenn. Yolov5 release v6.2, 2022. https://github.com/ultralytics/yolov5/releases/tag/v6.2. 2

[3] Chuyi Li, Lulu Li, Hongliang Jiang, Kaiheng Weng, Yifei Geng, Liang Li, Zaidan Ke, Qingyuan Li, Meng Cheng, Weiqiang Nie, et al. Yolov6: a single-stage object detection framework for industrial applications. *arXiv preprint arXiv:2209.02976*, 2022. 2

[4] Mingxing Tan, Ruoming Pang, and Quoc V Le. Efficientdet: Scalable and efficient object detection. In *CVPR*, pages 10781–10790, 2020. 2

[5] Chien-Yao Wang, Alexey Bochkovskiy, and Hong-Yuan Mark Liao. Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. *arXiv preprint arXiv:2207.02696*, 2022. 2

[6] Xiaoxing Wang, Jiale Lin, Juanping Zhao, Xiaokang Yang, and Junchi Yan. Eautodet: efficient architecture search for object detection. In *ECCV*, pages 668–684. Springer, 2022. 2

[7] 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. 2