DETR++: Taming Your Multi-Scale Detection Transformer

Chi Zhang[1,2,★] Lijuan Liu[2] Xiaoxue Zang[2,★] Frederick Liu[2] Hao Zhang[2] Xinying Song[2] Jindong Chen[2]

Google Research

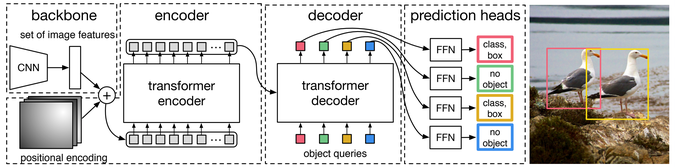
[1] Department of Computer Science, University of California, Los Angeles [2] Google Research

★ work done while at Google

chi.zhang@ucla.edu, lijuanliu@google.com, zxx1204007@gmail.com, {frederickliu,haozhangthu,xysong,jdchen}@google.com

CVPR JUNE 19-24 2022 NEW ORLEANS • LOUISIANA

Motivation



Model	GFLOPS/FPS	#params	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Faster RCNN-DC5	320/16	166M	39.0	60.5	42.3	21.4	43.5	52.5
Faster RCNN-FPN	180/26	42M	40.2	61.0	43.8	24.2	43.5	52.0
Faster RCNN-R101-FPN	246/20	60M	42.0	62.5	45.9	25.2	45.6	54.6
Faster RCNN-DC5+	320/16	166M	41.1	61.4	44.3	22.9	45.9	55.0
Faster RCNN-FPN+	180/26	42M	42.0	62.1	45.5	26.6	45.4	53.4
Faster RCNN-R101-FPN+	246/20	60M	44.0	63.9	**47.8**	**27.2**	48.1	56.0
DETR	86/28	41M	42.0	62.4	44.2	20.5	45.8	61.1
DETR-DC5	187/12	41M	43.3	63.1	45.9	22.5	47.3	61.1
DETR-R101	152/20	60M	43.5	63.8	46.4	21.9	48.0	61.8
DETR-DC5-R101	253/10	60M	**44.9**	**64.7**	47.7	23.7	**49.5**	**62.3**

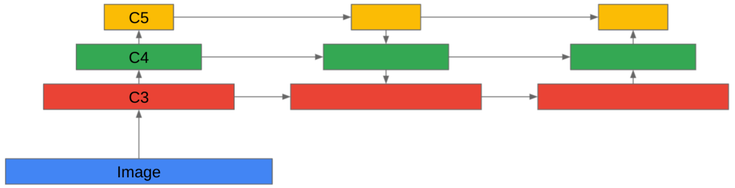
Despite the simpler design in the DETR architecture, earlier experimental results show that the DETR model is inferior to existing convolutional models and slower in training.

- The self-attention mechanism in the encoder is resource-hungry, especially for visual features that could span over thousands of tokens.
- The Hungarian matcher is cubic in time.

These slow operations make the common strategy of adding multi-scale features in a detector to improve performance a non-trivial work: conventional methods are extremely memory- and time-consuming.

Multi-Scale Designs

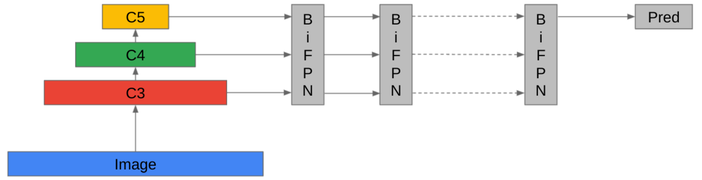
- Removing the Encoder: We consider the following two options.
  - Stack: In this stacking strategy, we consecutively apply three decoders on the image features from C3, C4, and C5. The decoded output from C3 is further processed by the decoder for C4, followed by the C5 decoder.
  - Multi-Head: We use three six-layer decoders for each resolution similarly to the stacking method. However, unlike the stacking method, each decoder independently produces the box proposals from a single scale.
- Shifted Windows: We apply the Transformer detection head on each shifted window.
- Specialized Heads: We use three detection heads for small objects, medium objects, and large objects, respectively. All the detection heads operate on the C5 features.
- Bi-directional Feature Pyramid: We aggregate multiple features using a BiFPN.



Remaining Issues

- There is still room for improvement small object detection. These gaps further motivate us in pursuing this direction to make Transformer-based detectors a first-class citizen.
- Convergence speed of Transformer detectors is slower than existing baselines. This slow-down significantly impacts model iteration. And we notice few of the recent acceleration methods bring the model to the optimal point from the plain one.

DETR++



Benchmarking

Performance on MS COCO 2017

Method	AP	AP@0.5	AP@0.75	AP^L	AP^M	AP^S
DETR-NoEnc-Stack	37.3	56.8	39.7	54.4	40.6	16.9
DETR-NoEnc-MHead	35.0	54.9	36.3	52.0	37.5	14.6
DETR-Swin	39.9	59.8	42.2	57.9	43.6	18.4
DETR-SHead	36.4	54.0	39.2	54.7	39.5	15.1
DETR++	**41.8**	**60.1**	**44.6**	**58.6**	**45.0**	22.1
DETR	39.9	59.8	42.4	57.2	43.3	18.8
CenterNet	41.6	59.4	44.2	54.1	43.1	**22.5**

Performance on RICO icon detection

Method	AP	AP@0.5	AP@0.75	AP^L	AP^M	AP^S
DETR++	**48.1**	**89.8**	**45.3**	**52.9**	**49.6**	**43.6**
DETR	47.4	89.4	44.3	52.0	48.8	43.1
IconNet	36.6	79.3	26.8	15.1	35.6	36.8

Performance on RICO layout detection

Method	AP	AP@0.5	AP@0.75	AP^L	AP^M	AP^S
DETR++	**25.3**	**43.6**	**24.2**	**28.6**	**9.7**	1.4
DETR	24.7	42.5	23.4	28.0	8.1	1.1
IconNet	16.2	30.5	15.9	19.8	8.6	**5.6**