Paper review

DE:TR: End-to-End Object Detection with Transformers

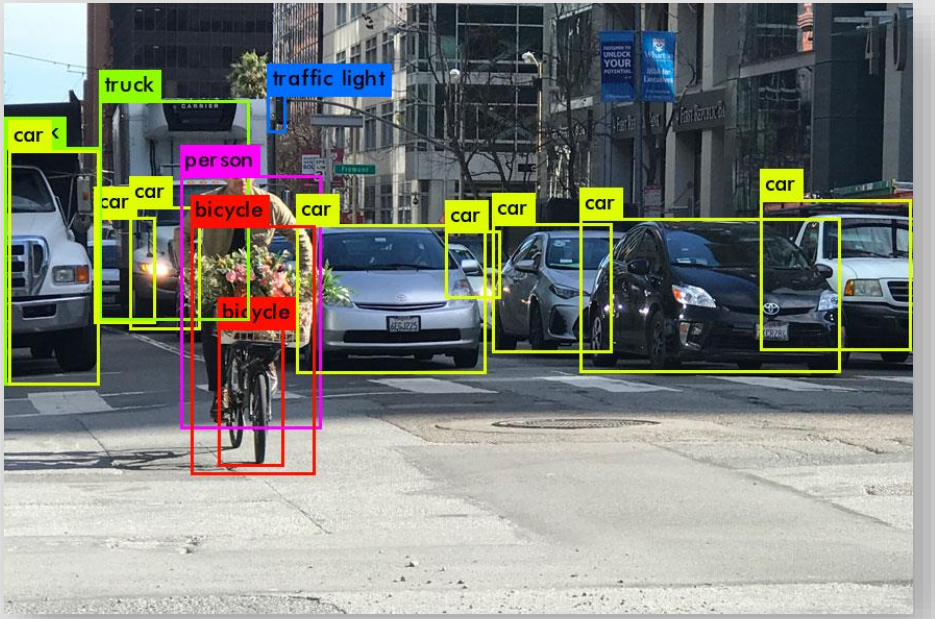
Facebook AI (ECCV 2020)

01

Quick View

Object Detection: Direct set prediction problem

- Predict a set of bounding boxes
- Category labels for each object of interest



Object Detection
기존 객체 탐지

1) 복잡하며 다양한 라이브러리 사용
2) 사전지식(prior knowledge) 요구
   * 앵커 박스 구조를 이용한 관심 영역 추출
   * 예측한 bounding box에 대한 Non-Maximum Suppression(NMS)

Contributes
이 논문이 기여한 바

1) 모델의 간소화(CNN Feature 추출 → Transformer)
2) 휴리스틱한(heuristic) 부분의 제거
3) End-to-End 방식

DETR(**De**tection **TR**ansformer):

1) 이분 매칭 손실 함수

2) Transformer

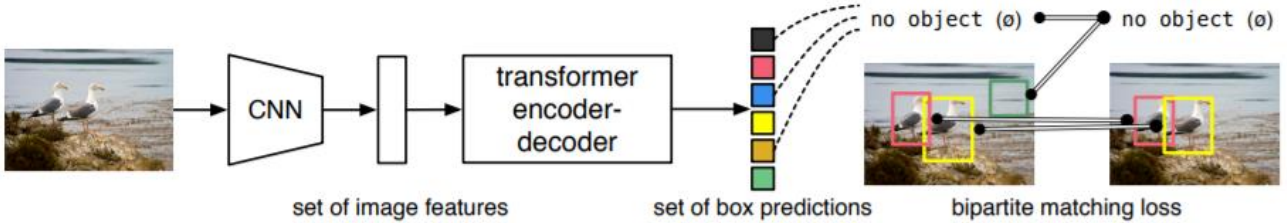


Fig. 1: DETR directly predicts (in parallel) the final set of detections by combining a common CNN with a transformer architecture. During training, bipartite matching uniquely assigns predictions with ground truth boxes. Prediction with no match should yield a "`no object`" (∅) class prediction.

02

Prior Knowledge

<Object Detection>2 stage detector

Input Image

Region Proposal

- Selective Search
- Region Proposal Network
- etc.

Classification

For Each Proposed Region

Multi-Class Classification

Bounding Box Regression

Output

<Object Detection>1 stage detector

Input Image

Conv Layers

Feature Extraction

Feature Maps

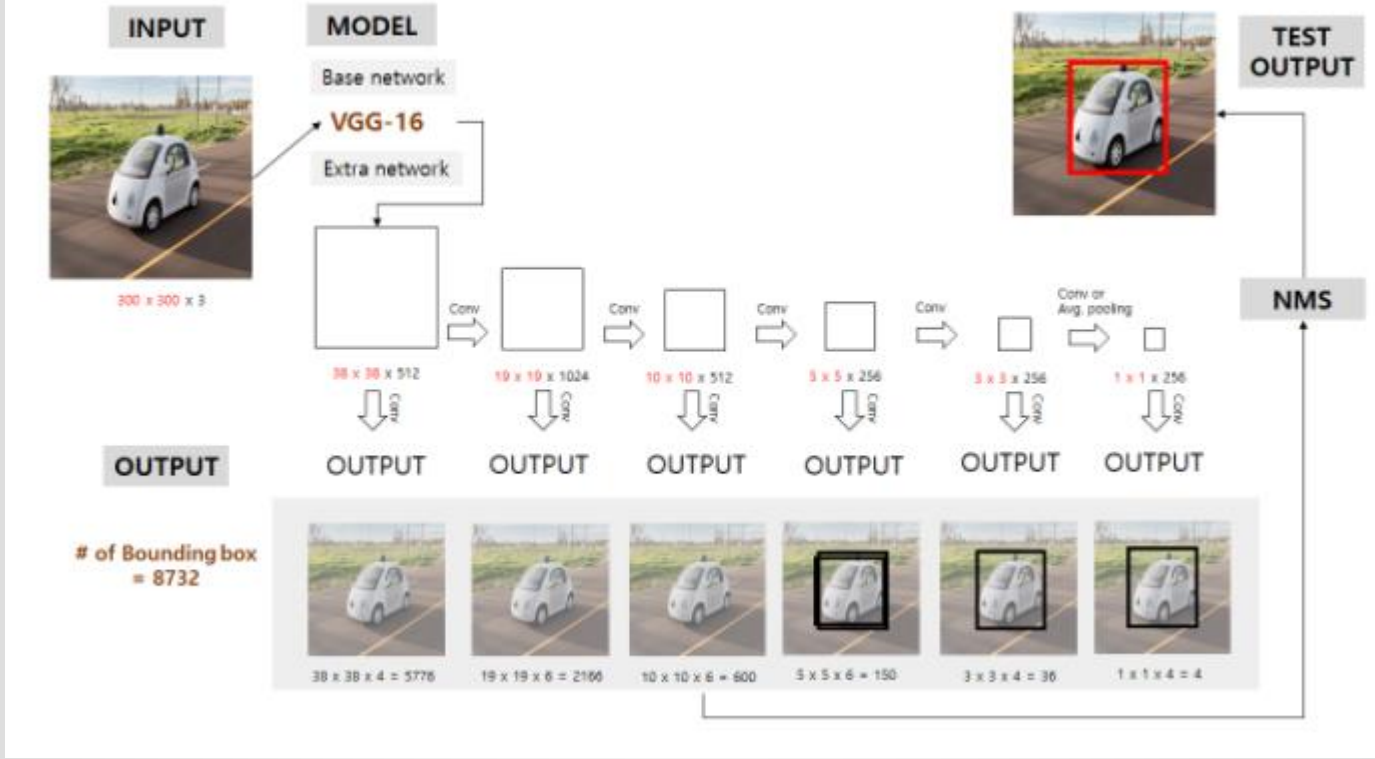
For Each Grid or Spatial Location

Multi-Class Classification

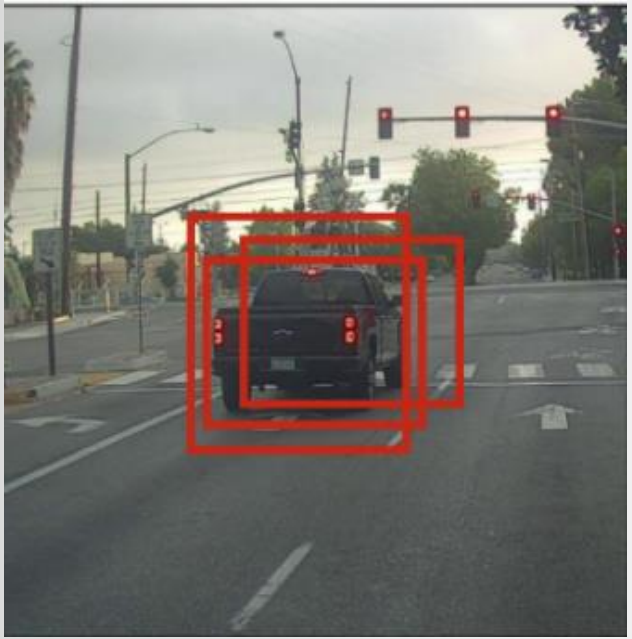
Bounding Box Regression

Output

<Object Detection>SSD pipeline

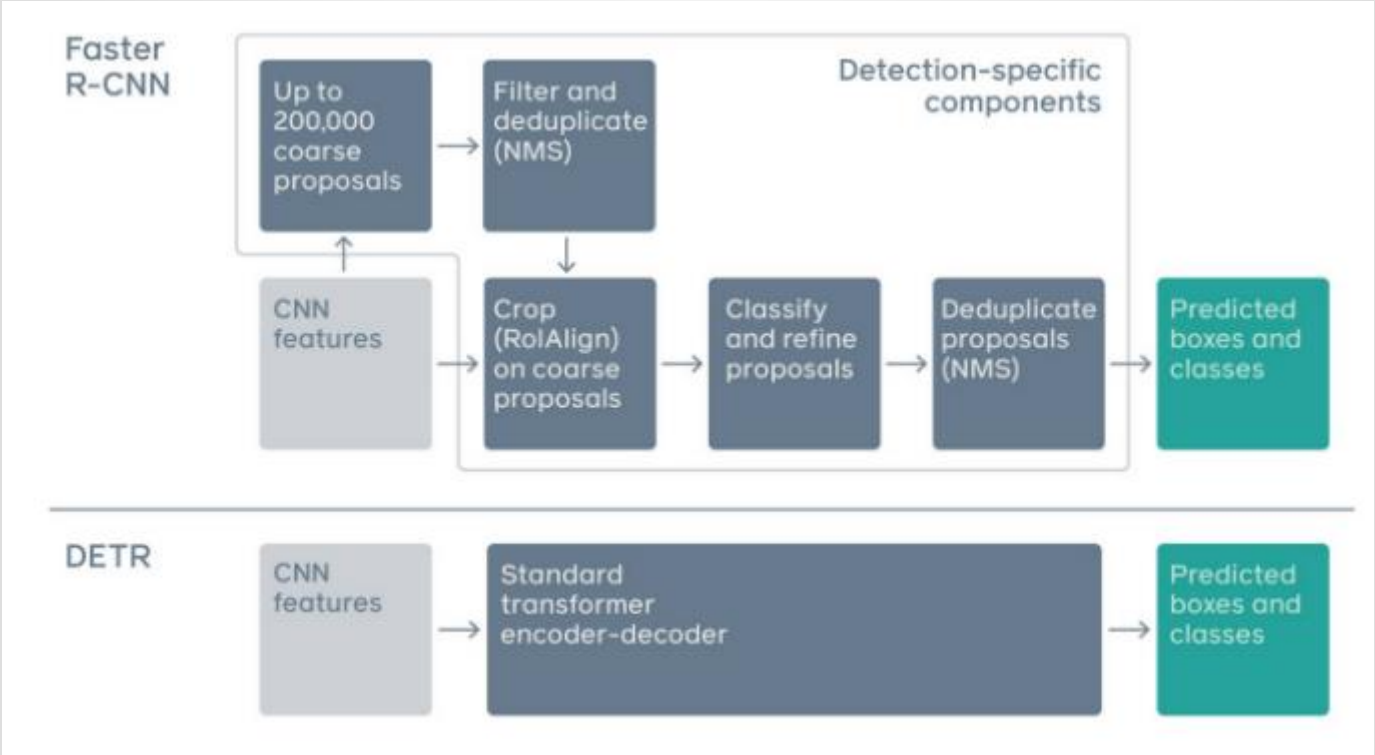


Before non-maximum suppression



Non-Maximum
Suppression

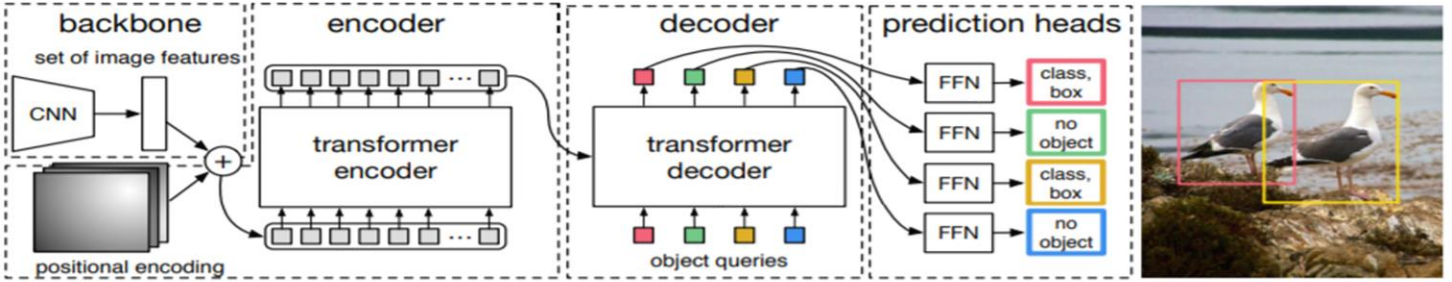
After non-maximum suppression


Faster
R-CNN
Detection-specific
components
Up to
200,000
coarse
proposals
Filter and
deduplicate
(NMS)
CNN
features
Crop
(RoIAlign)
on coarse
proposals
Classify
and refine
proposals
Deduplicate
proposals
(NMS)
Predicted
boxes and
classes
DETR
CNN
features
Standard
transformer
encoder-decoder
Predicted
boxes and
classes

03

The DETR model

<DETR> model



Bipartite matching

출력 개수 고정: $N = 6$

예측 결과

$(c_0 = \emptyset, b_0)$
$(c_1 = bird, b_1 = (180, 180, 150, 240))$
$(c_2 = \emptyset, b_2)$
$(c_3 = bird, b_3 = (120, 150, 100, 150))$
$(c_4 = \emptyset, b_4)$
$(c_5 = dog, b_5)$

$(c_0 = bird, b_0 = (122, 151, 100, 150))$
$(c_1 = bird, b_1 = (182, 180, 148, 238))$
$(c_2 = \emptyset, b_2)$
$(c_3 = \emptyset, b_3)$
$(c_4 = \emptyset, b_4)$
$(c_5 = \emptyset, b_5)$

실제 값

<Bipartite matching>

$$\hat{\sigma} = \arg\min_{\sigma \in \mathfrak{S}_N} \sum_{i}^{N} \mathcal{L}_{\text{match}}(y_i, \hat{y}_{\sigma(i)})$$

<Matching cost>

$$-\mathbb{1}_{\{c_i \neq \varnothing\}} \hat{p}_{\sigma(i)}(c_i) + \mathbb{1}_{\{c_i \neq \varnothing\}} \mathcal{L}_{\text{box}}(b_i, \hat{b}_{\sigma(i)})$$

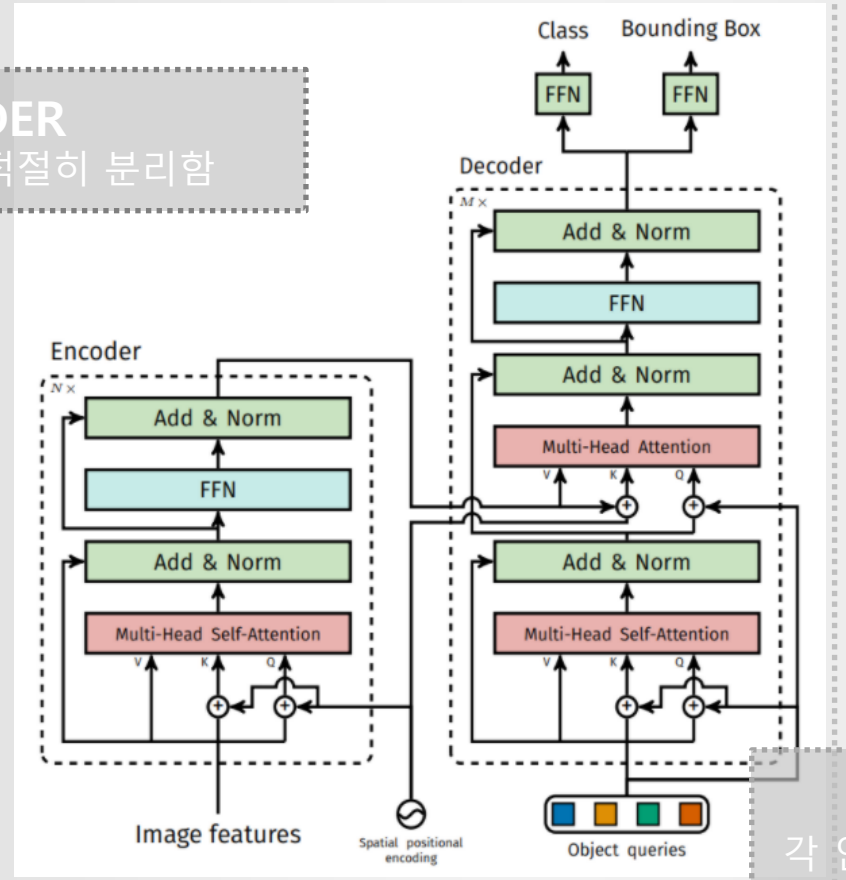
<Hungarian matching>

$$\mathcal{L}_{\text{Hungarian}}(y, \hat{y}) = \sum_{i=1}^{N} \left[-\log \hat{p}_{\hat{\sigma}(i)}(c_i) + \mathbb{1}_{\{c_i \neq \varnothing\}} \mathcal{L}_{\text{box}}(b_i, \hat{b}_{\hat{\sigma}}(i)) \right]$$

<Bounding box cost>

$$\lambda_{\text{iou}} \mathcal{L}_{\text{iou}}(b_i, \hat{b}_{\sigma(i)}) + \lambda_{\text{L1}} ||b_i - \hat{b}_{\sigma(i)}||_1$$

ENCODER
개별 인스턴스를 적절히 분리함

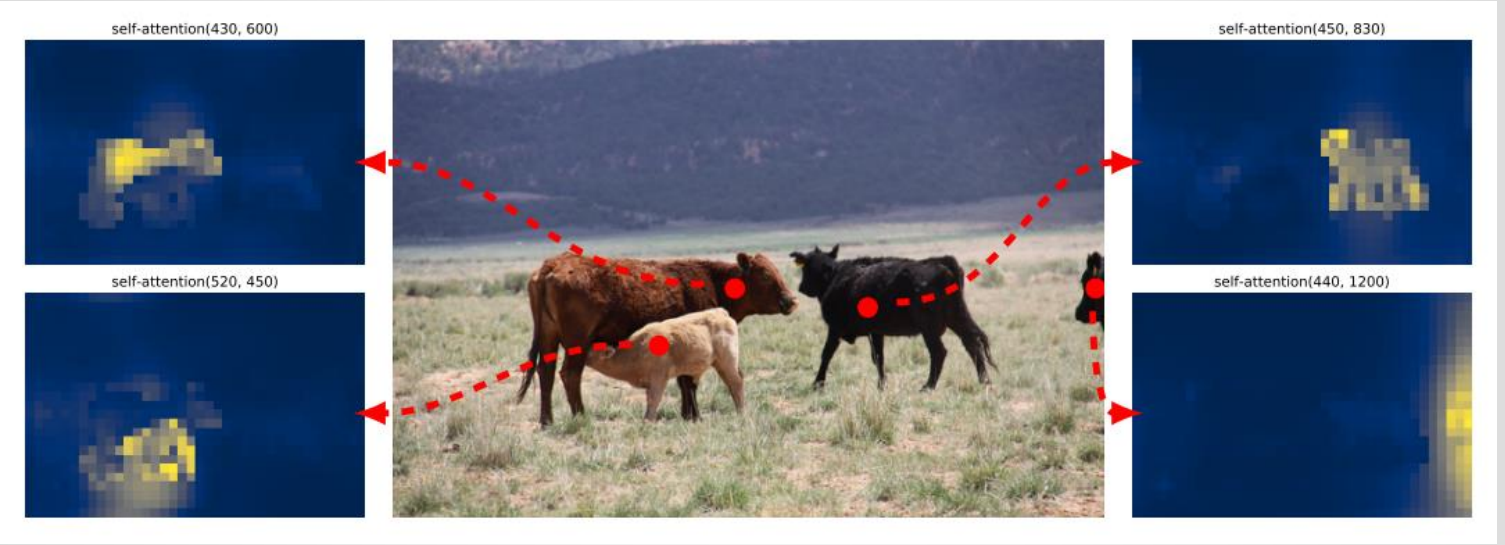


DECODER
각 인스턴스의 클래스와 경계선을 추출

Encoder은 multi-head self-attention과 FFN 으로 구성

Encoder의 self-attention map을 시각화해보면 개별 인스턴스를 적절히 분리하는 것을 확인할 수 있다.

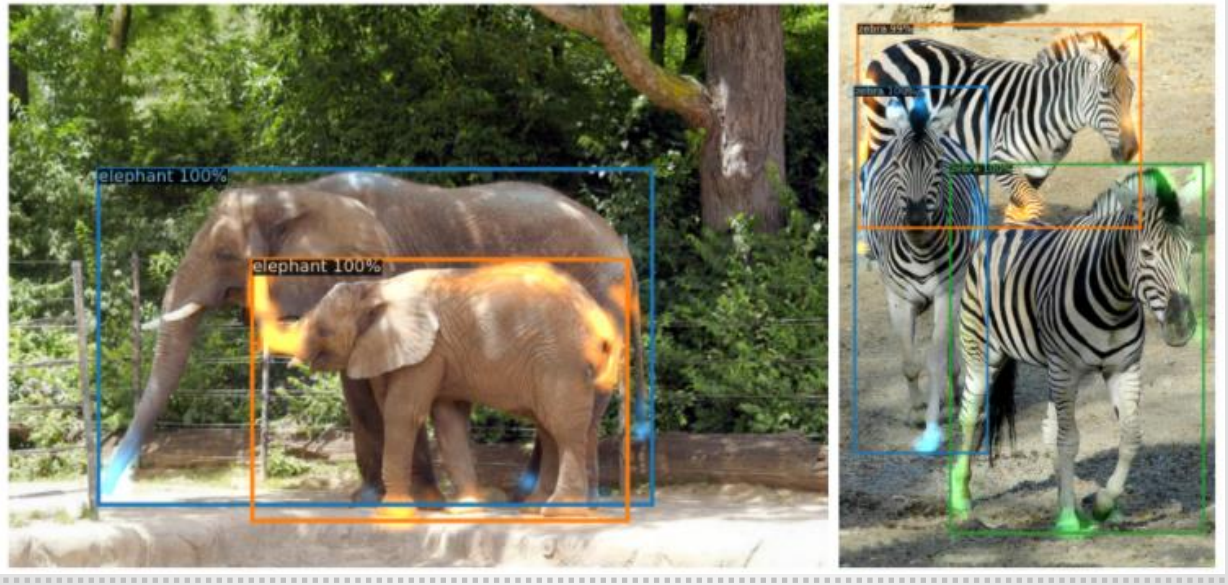
<transformer> Encoder Self Attention



Decoder는 N개의 object query(학습된 위치 임베딩)를 초기 입력으로 사용

Decoder는 각 인스턴스의 **클래스**와 **경계선**을 추출한다.
(Encoder는 global attention을 통해 **인스턴스를 분리**)

<transformer> Decoder Self Attention



04

Experiments

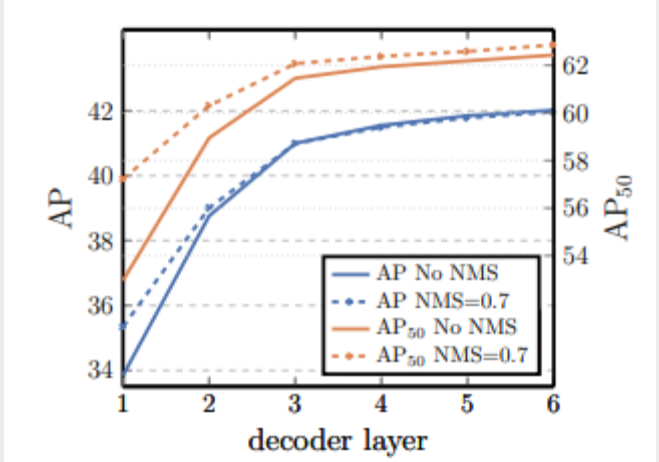
Base model: Faster R-CNN
Dataset: COCO minival
Optimizer: AdamW
Backbone: ResNet50, ResNet 101(DETR, DETR-R101)
Featrure resolution을 증가시키기 위해 dilate를 적용한
DETR-DC5, DETR DC5-R101 진행

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**

<Encoder> # of layers

#layers	GFLOPS/FPS	#params	AP	AP_{50}	AP_S	AP_M	AP_L
0	76/28	33.4M	36.7	57.4	16.8	39.6	54.2
3	81/25	37.4M	40.1	60.6	18.5	43.8	58.6
6	86/23	41.3M	40.6	61.6	19.9	44.3	60.2
12	95/20	49.2M	41.6	62.1	19.8	44.9	61.9

<Decoder> # of layers



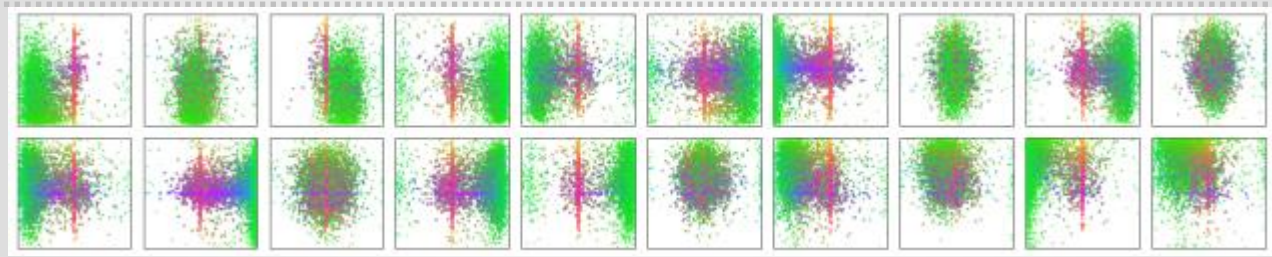
<Encoder> positional Encoding

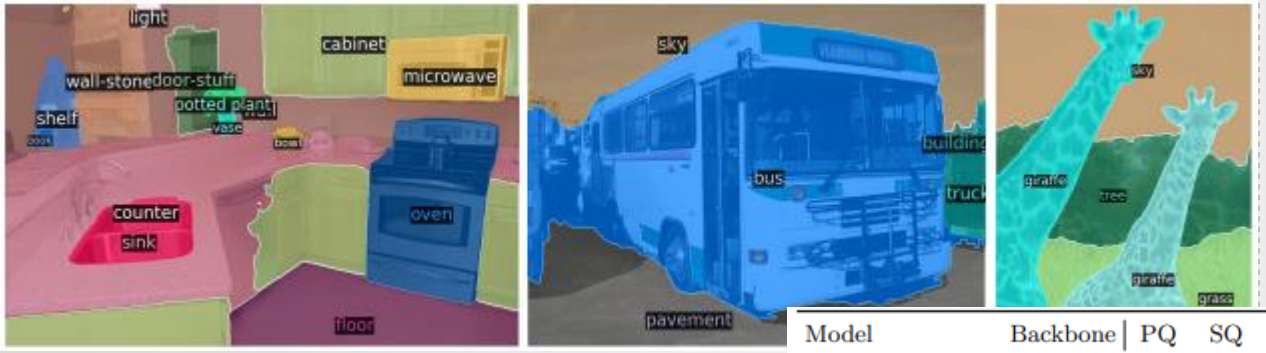
spatial pos. enc. encoder	spatial pos. enc. decoder	output pos. enc. decoder	AP	Δ	AP_{50}	Δ
none	none	learned at input	32.8	-7.8	55.2	-6.5
sine at input	sine at input	learned at input	39.2	-1.4	60.0	-1.6
learned at attn.	learned at attn.	learned at attn.	39.6	-1.0	60.7	-0.9
none	sine at attn.	learned at attn.	39.3	-1.3	60.3	-1.4
sine at attn.	sine at attn.	learned at attn.	**40.6**	-	**61.6**	-

Loss

class	ℓ_1	GIoU	AP	Δ	AP_{50}	Δ	AP_S	AP_M	AP_L
✓	✓		35.8	-4.8	57.3	-4.4	13.7	39.8	57.9
✓		✓	39.9	-0.7	**61.6**	0	**19.9**	43.2	57.9
✓	✓	✓	**40.6**	-	**61.6**	-	**19.9**	**44.3**	**60.2**

<Object Queries>

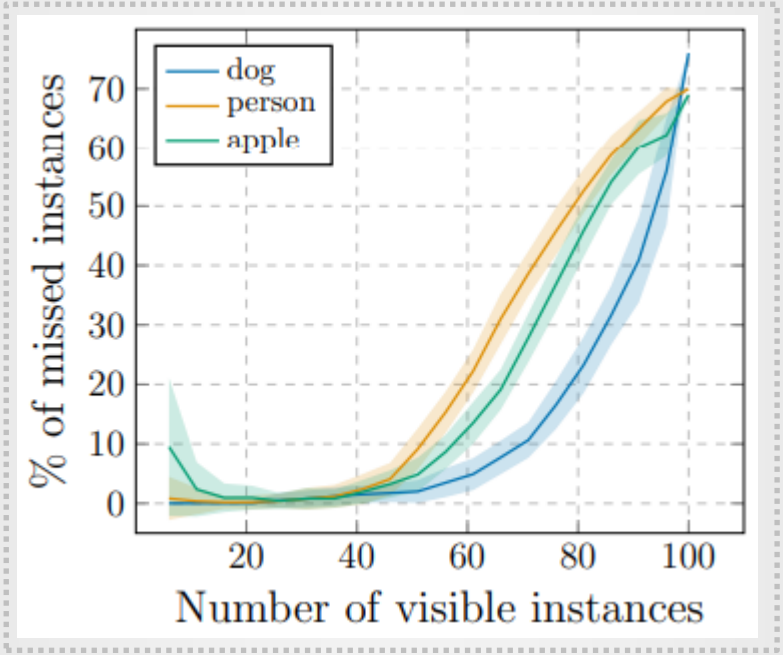




<Panoptic segmentation>

Model	Backbone	PQ	SQ	RQ	PQ^{th}	SQ^{th}	RQ^{th}	PQ^{st}	SQ^{st}	RQ^{st}	AP
PanopticFPN++	R50	42.4	79.3	51.6	49.2	82.4	58.8	32.3	74.8	40.6	37.7
UPSnet	R50	42.5	78.0	52.5	48.6	79.4	59.6	33.4	75.9	41.7	34.3
UPSnet-M	R50	43.0	79.1	52.8	48.9	79.7	59.7	34.1	78.2	42.3	34.3
PanopticFPN++	R101	44.1	79.5	53.3	**51.0**	**83.2**	60.6	33.6	74.0	42.1	**39.7**
DETR	R50	43.4	79.3	53.8	48.2	79.8	59.5	36.3	78.5	45.3	31.1
DETR-DC5	R50	44.6	79.8	55.0	49.4	80.5	60.6	**37.3**	**78.7**	**46.5**	31.9
DETR-R101	R101	**45.1**	**79.9**	**55.5**	50.5	80.9	**61.7**	37.0	78.5	46.0	33.0

<# of instances>



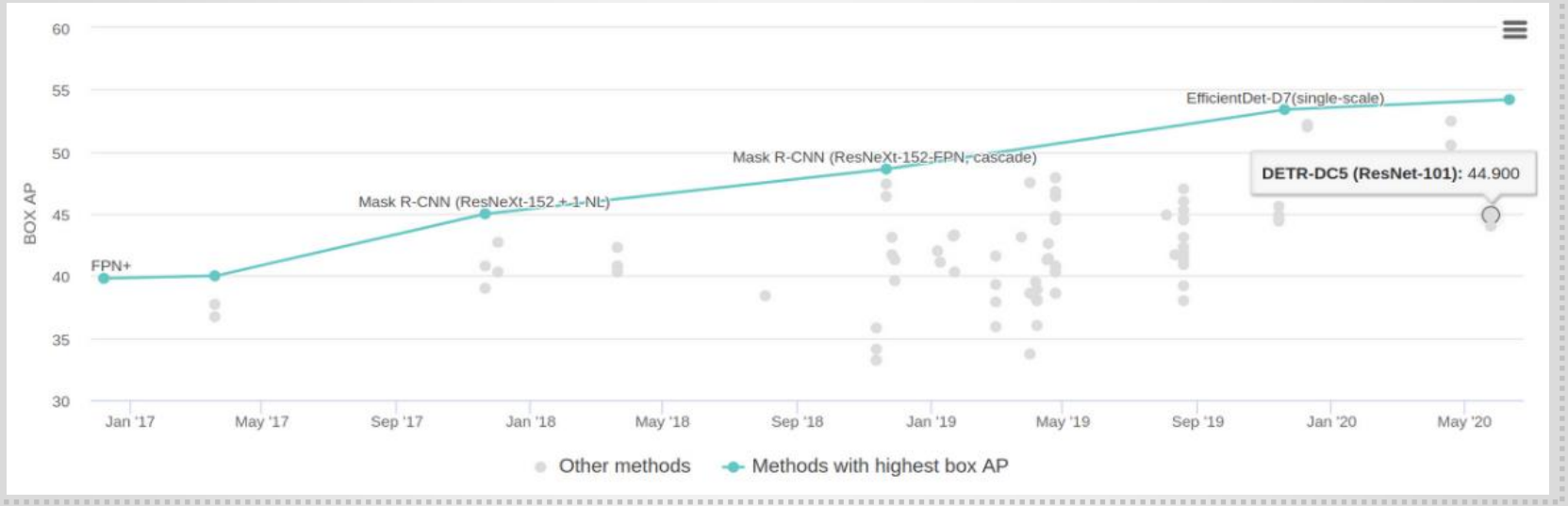
05

Discussion

DETR

new design for object detection systems based on transformers and bipartite matching loss for direct set problem

<SOTA> Object Detection on COCO



THE END

감 사 합 니 다