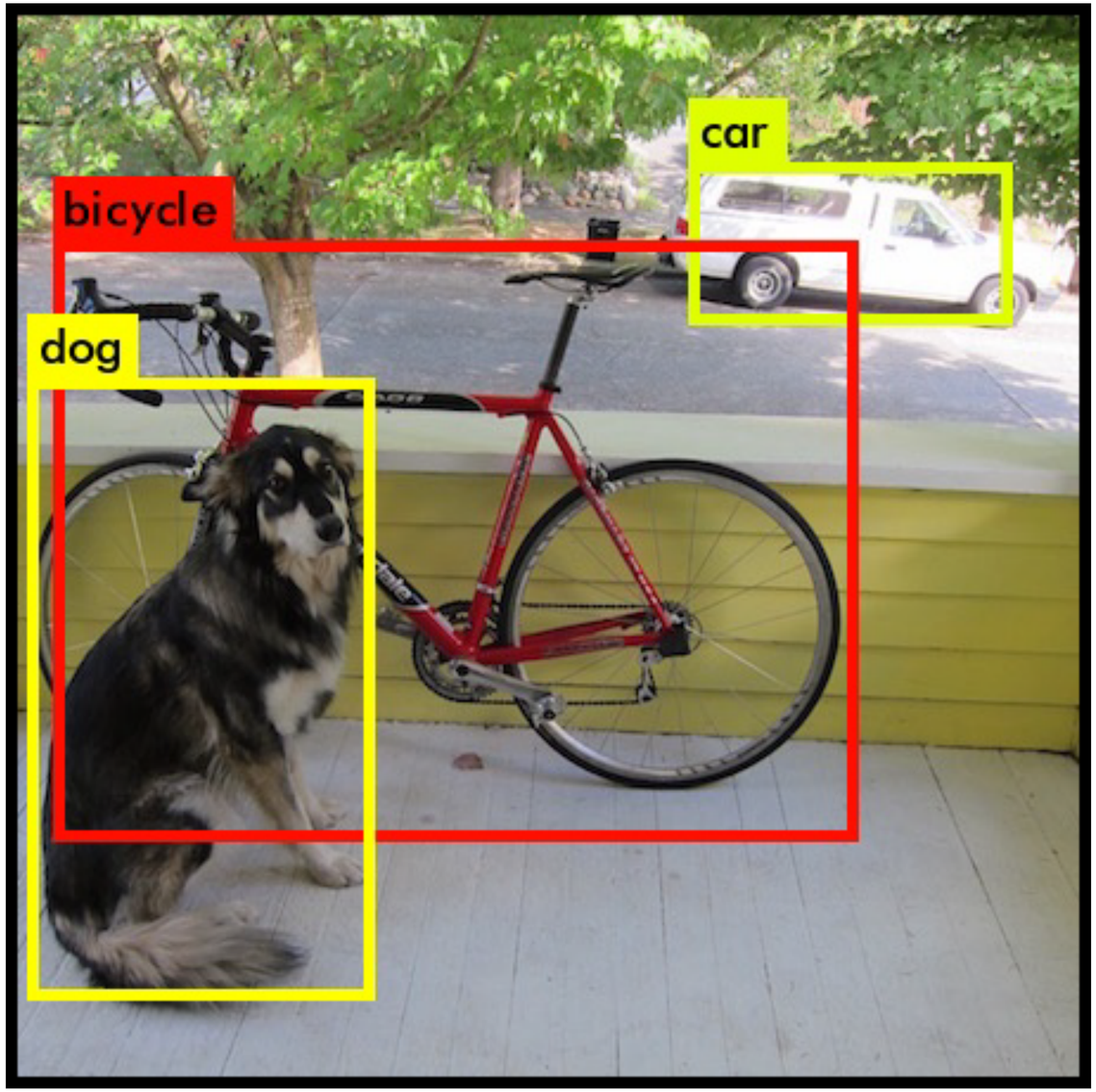
End-to-end Object Detection with Transformers

Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier,
Alexander Kirillov, and Sergey Zagoruyko
Facebook AI
CVPR 2020

The task of object detection

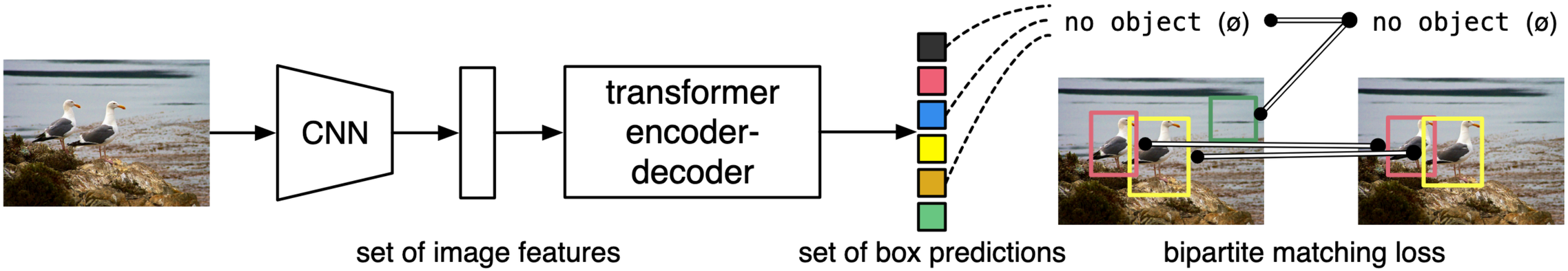
- Given an image, predict
  - A set of bounding boxes
  - And its labels



Classical approach to detection

- Popular approach: detection := classification of boxes
- Requires selecting a subset of candidate boxes

Architecture Overview

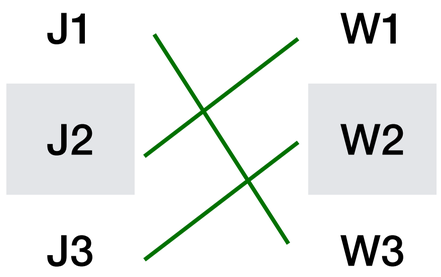


Bipartite Matching Loss

Workers

Jobs	W1	W2	W3
J1	30	20	10
J2	20	40	60
J3	60	5	20

Optimal Matching σ



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

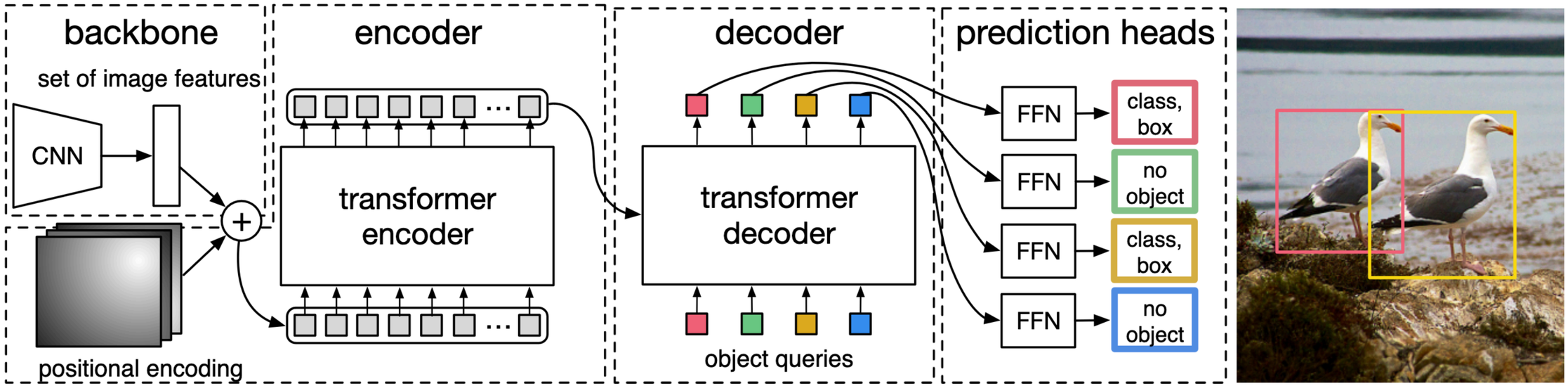
$$\mathcal{L}_{\text{match}}(y_i, \hat{y}_{\sigma(i)}) = -\mathbf{1}_{\{c_i \neq \emptyset\}} \hat{p}_{\sigma(i)}(c_i) + -\mathbf{1}_{\{c_i \neq \emptyset\}} \mathcal{L}_{\text{box}}(b_i, \hat{b}_{\hat{\sigma}(i)})$$

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

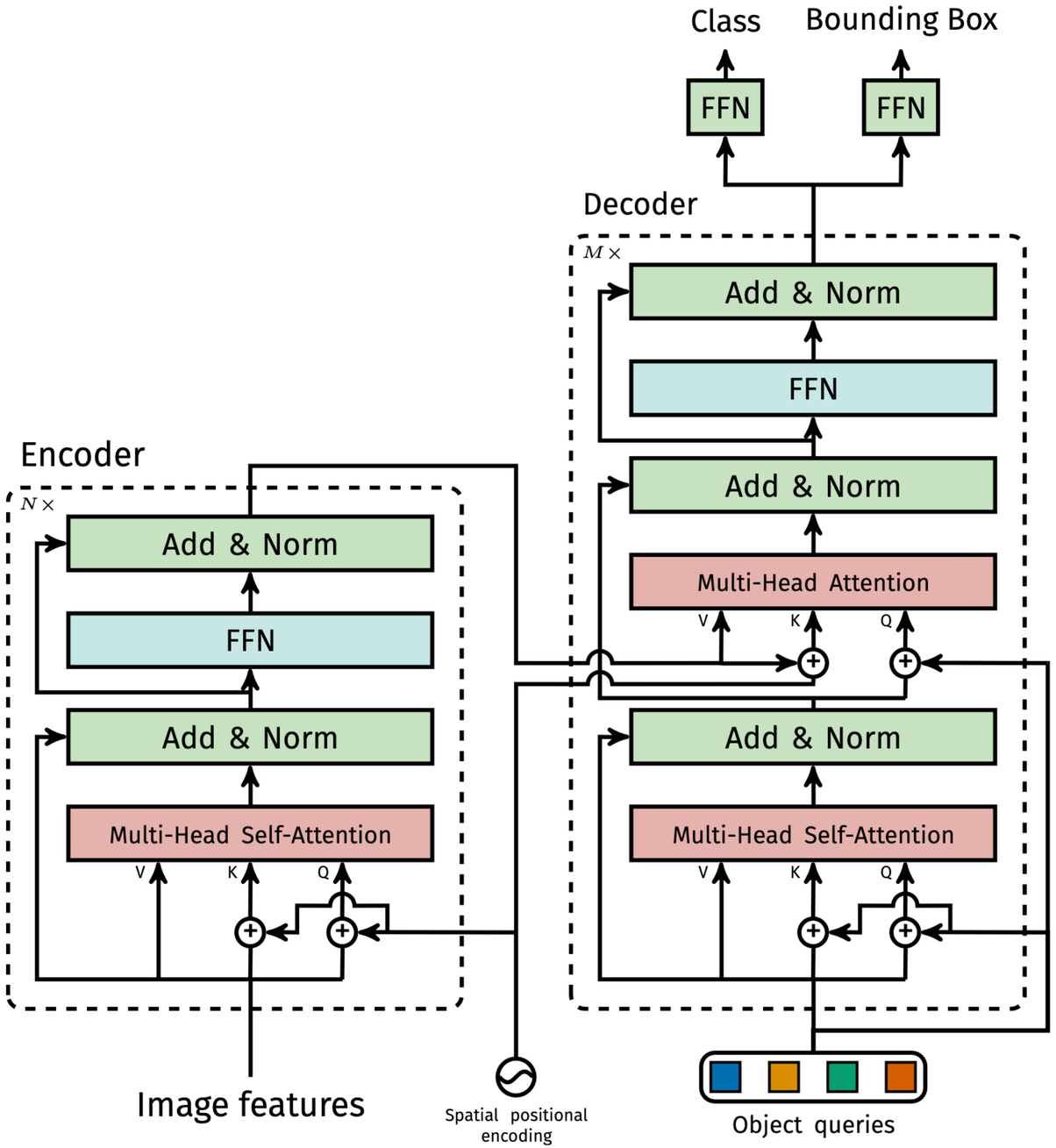
Optimize Object Specific Losses

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

DETR


backbone
set of image features
CNN
positional encoding
encoder
transformer encoder
decoder
transformer decoder
object queries
prediction heads
FFN
FFN
FFN
FFN
class, box
no object
class, box
no object

DETR


Class
Bounding Box
FFN
FFN
Decoder
$M\times$
Add & Norm
FFN
Add & Norm
Multi-Head Attention
V
K
Q
Add & Norm
Multi-Head Self-Attention
V
K
Q
Encoder
$N\times$
Add & Norm
FFN
Add & Norm
Multi-Head Self-Attention
V
K
Q
Image features
Spatial positional encoding
Object queries

Results

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**

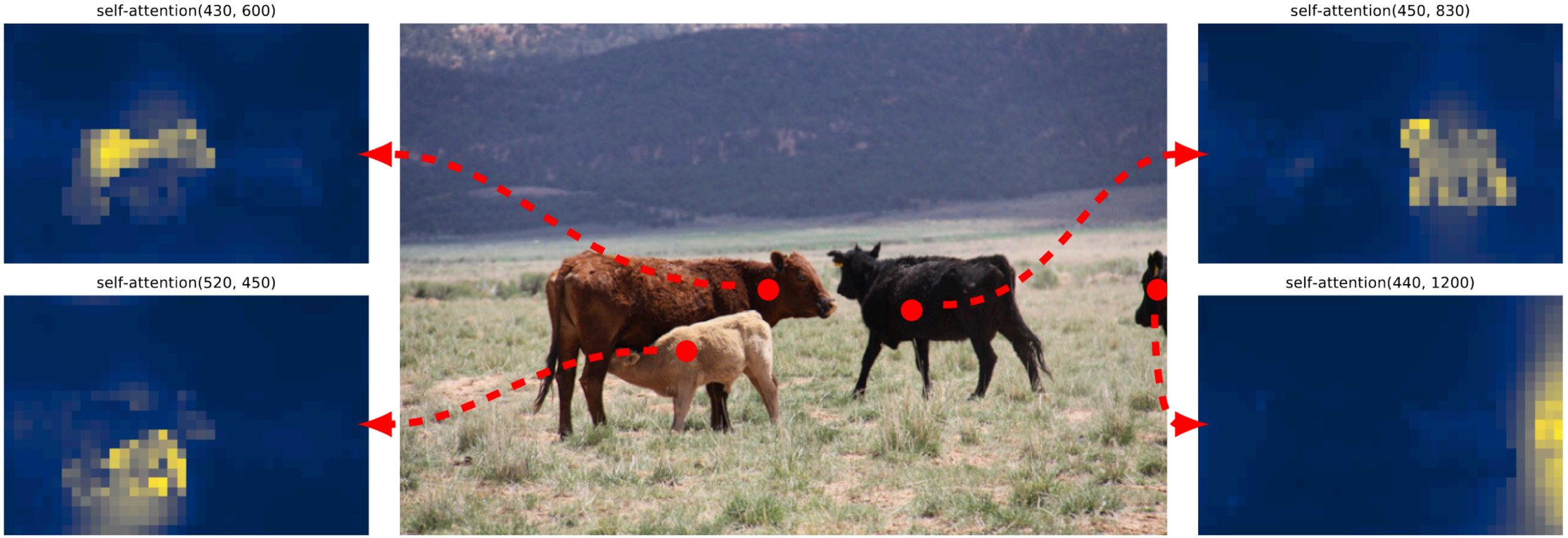
Comparison with Faster R-CNN with a ResNet-50 and ResNet-101 backbones
on the COCO validation set.

Results

#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

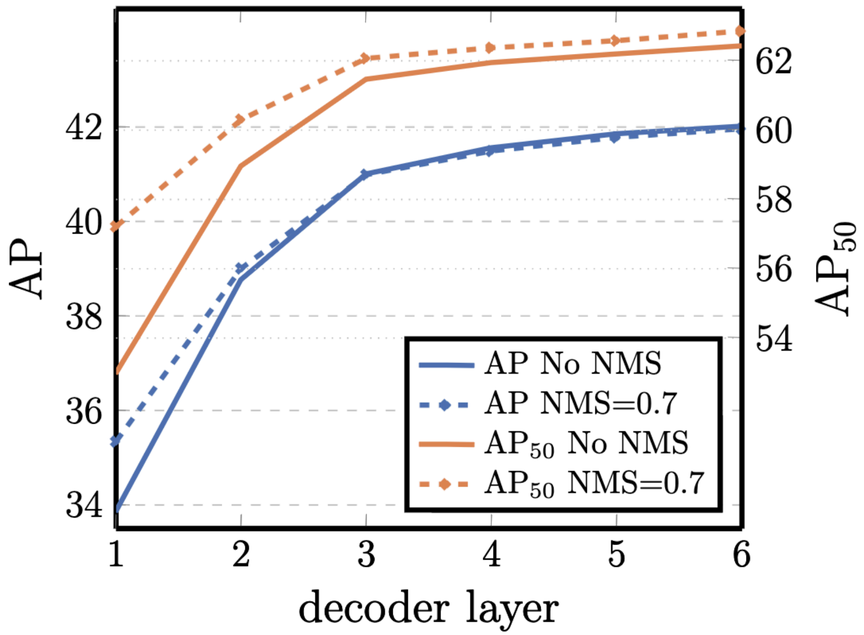
Effect of encoder size

Results



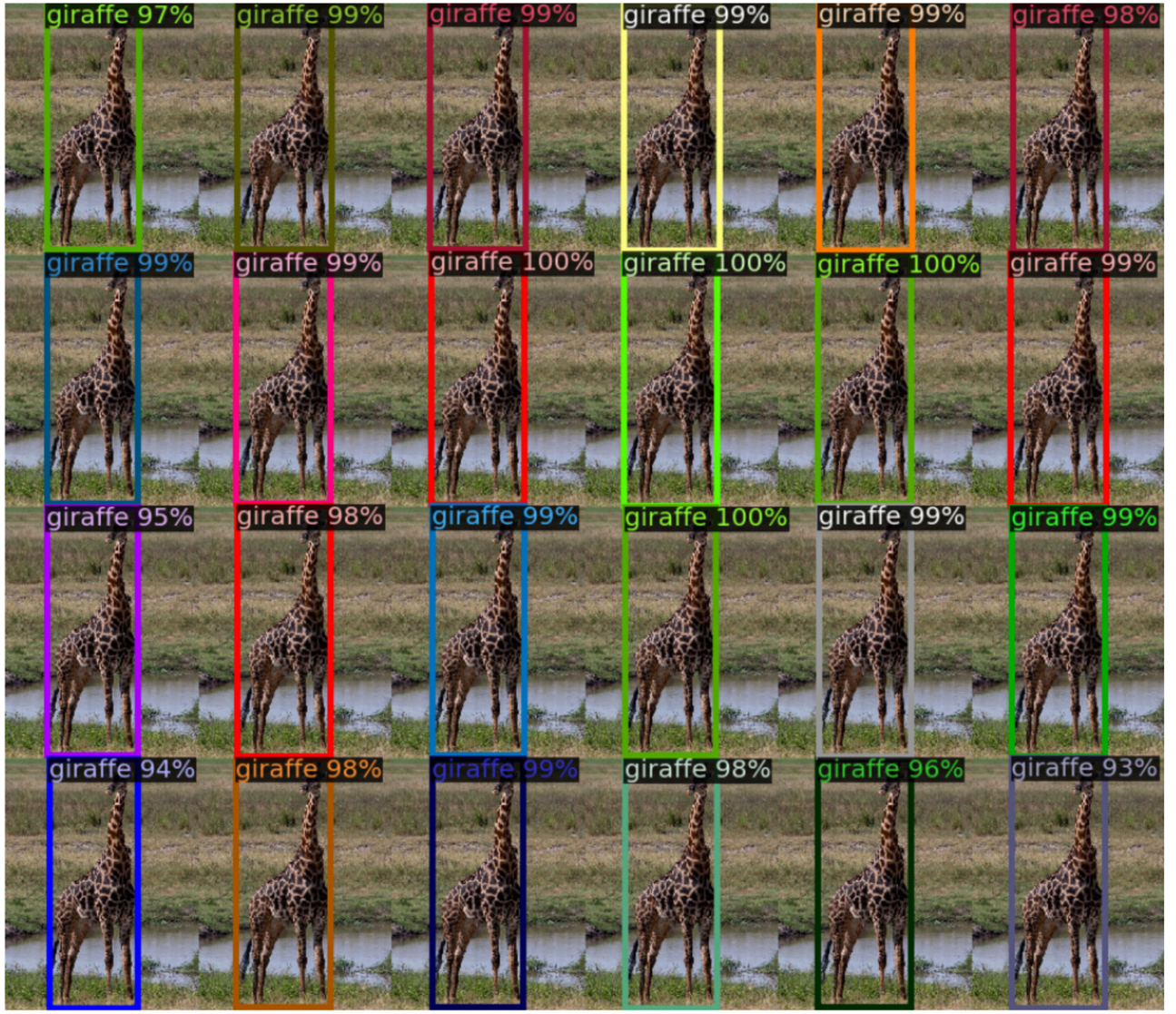
Encoder self-attention for a set of reference points

Results



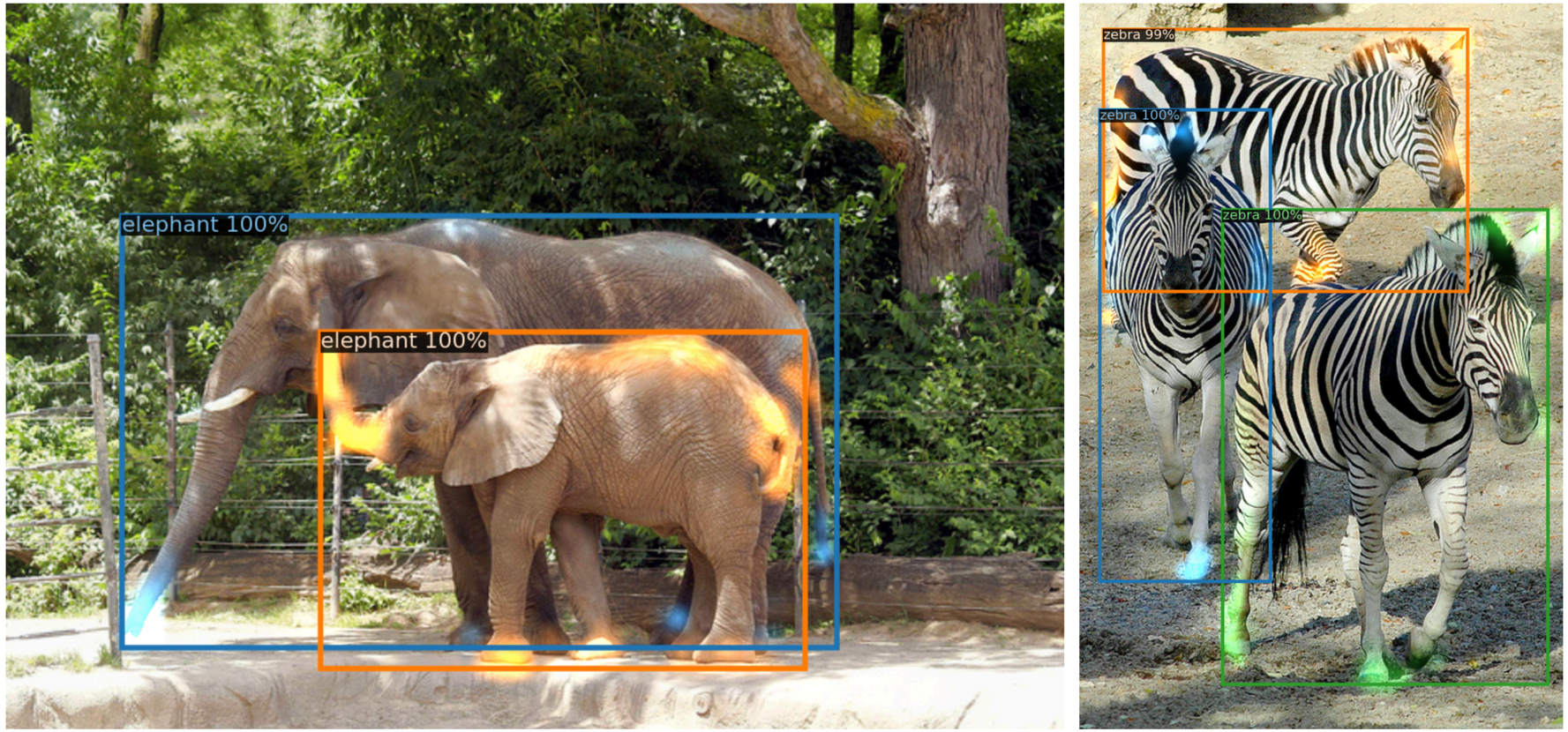
AP and AP50 performance after each decoder layer

Results



Out of distribution generalization for rare classes

Results



Visualizing decoder attention for every predicted object (images from COCO val set)