Deformable DETR:
Deformable Transformers for End-to-End Object Detection

Jifeng Dai

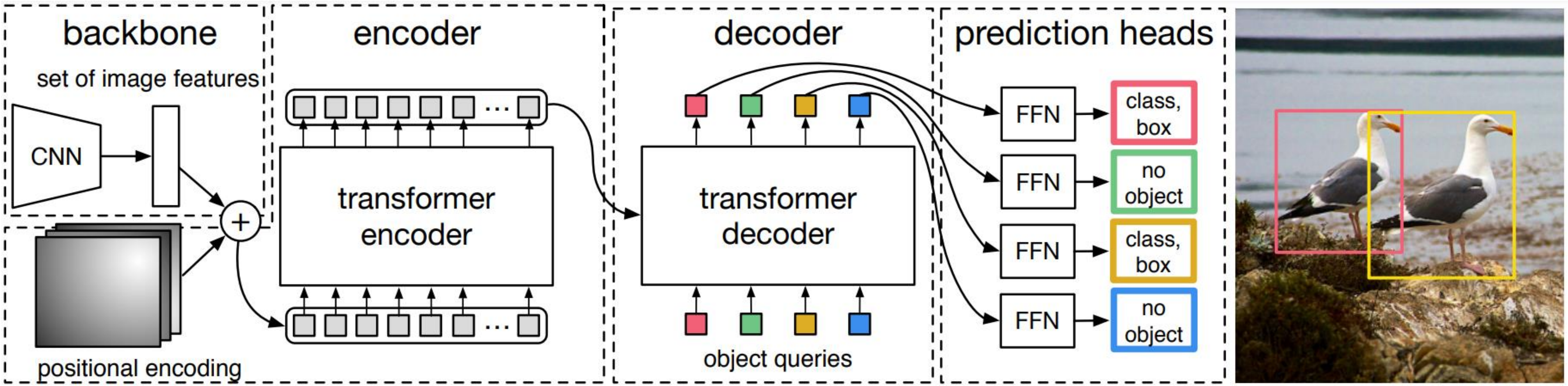
With Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li and Xiaogang Wang

SenseTime Research, USTC, CUHK

Previous Modern Object Detectors

- Rely on Hand-Crafted Components, e.g.,
  - anchor generation
  - rule-based training target assignment
  - non-maximum suppression (NMS) post-processing
- Not Fully End-to-End
  - complex combination of hand-crafted components
  - requiring manually adjustment (e.g., anchor size and NMS threshold) for specific datasets

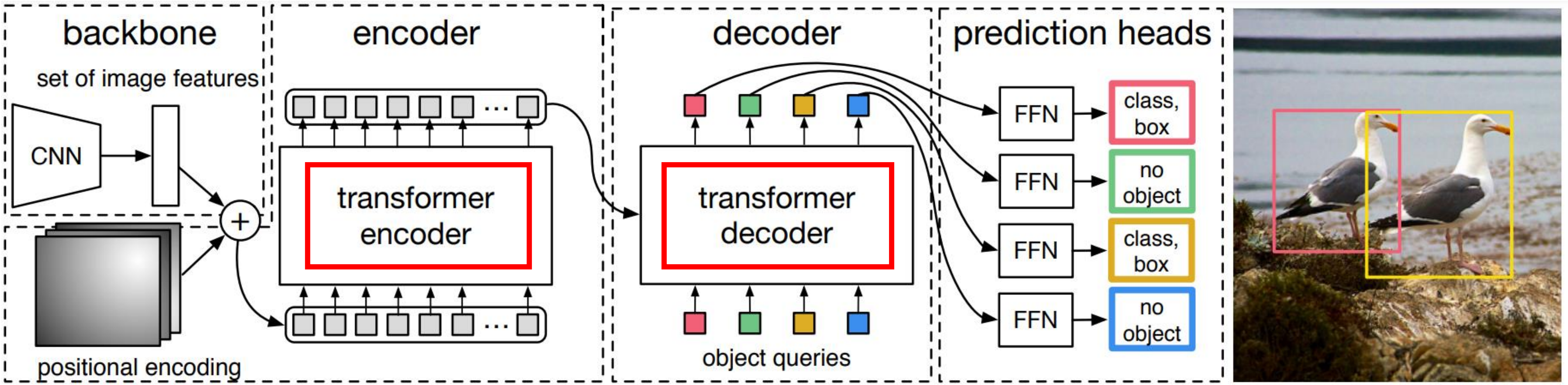
DETR - The First End-to-End Object Detector



- Eliminate the need for hand-crafted components
- Achieve very competitive performance with previous modern object detectors

Carion, Nicolas, et al. "End-to-End Object Detection with Transformers." In ECCV 2020.

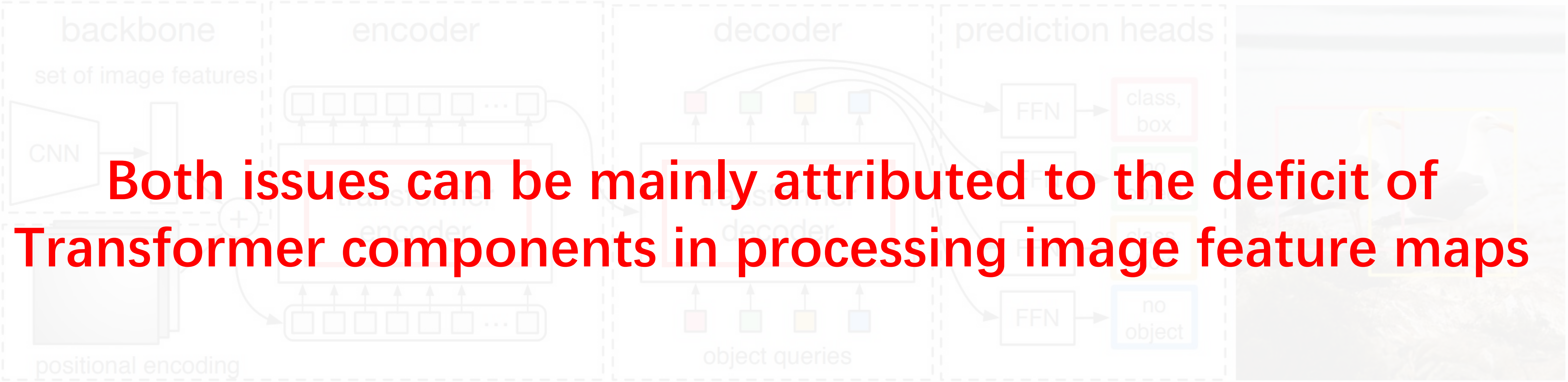
DETR - Issues



- Much longer training epochs to converge
  - 500 epochs on COCO, around 10 to 20 times slower than Faster R-CNN
- Low performance at detecting small objects

Carion, Nicolas, et al. "End-to-End Object Detection with Transformers." In ECCV 2020.

DETR - Issues



Both issues can be mainly attributed to the deficit of Transformer components in processing image feature maps

- Much longer training epochs to converge
  - 500 epochs on COCO, around 10 to 20 times slower than Faster R-CNN
- Low performance at detecting small objects

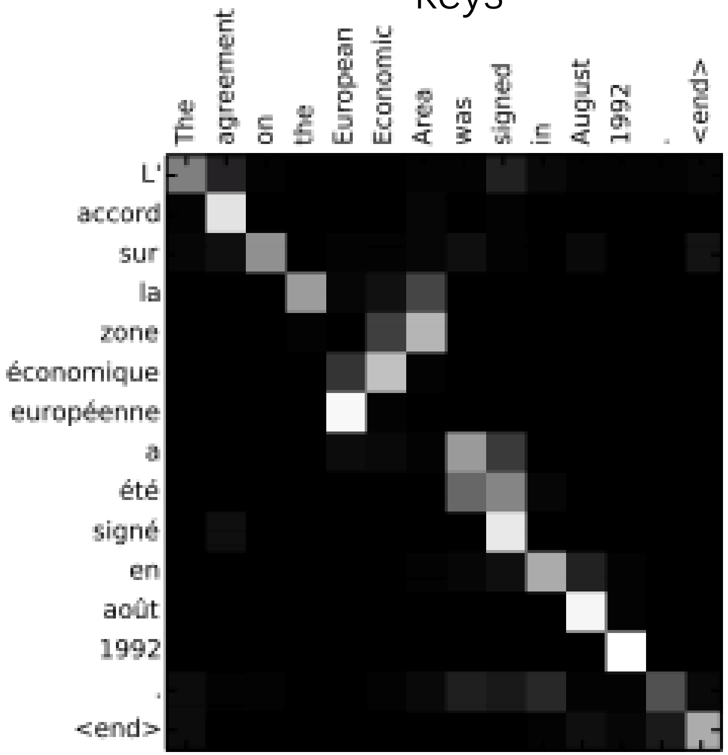
Carion, Nicolas, et al. "End-to-End Object Detection with Transformers." In ECCV 2020.

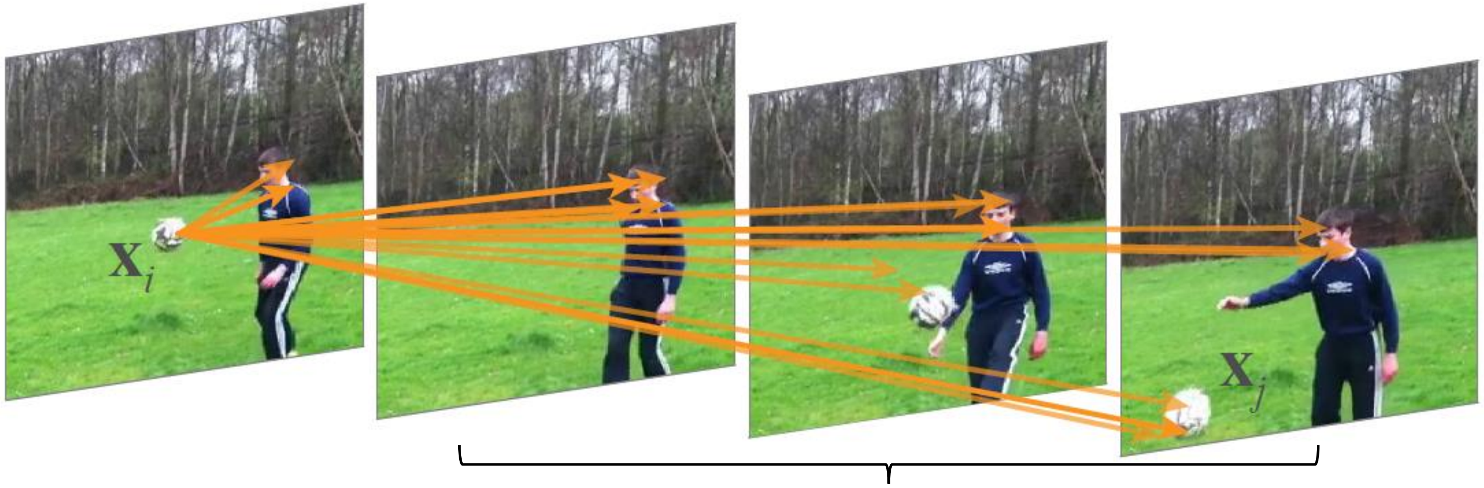
Revisit Multi-Head Attention in Transformers

- Enable neural networks to focus more on relevant elements of the input than on irrelevant parts.

keys

queries





query

keys

Vaswani, Ashish, et al. "Attention is all you need." In NeurIPS 2017

Revisit Multi-Head Attention in Transformers

query element

key elements

Learnable weights

$$\text{MultiHeadAttn}(\boldsymbol{z}_q, \boldsymbol{x}) = \sum_{m=1}^{M} \boldsymbol{W}_m \Big[\sum_{k \in \Omega_k} A_{mqk} \cdot \boldsymbol{W}'_m \boldsymbol{x}_k \Big]$$

Sum over attention heads

Attention weights

Key feature

subjective to $\sum_{k \in \Omega_k} A_{mqk} = 1$

Vaswani, Ashish, et al. "Attention is all you need." In NeurIPS 2017

Issues of Multi-Head Attention in Transformers

$$\text{MultiHeadAttn}(\boldsymbol{z}_q, \boldsymbol{x}) = \sum_{m=1}^{M} \boldsymbol{W}_m \Big[\sum_{k \in \Omega_k} A_{mqk} \cdot \boldsymbol{W}'_m \boldsymbol{x}_k \Big]$$

- Long training schedules are required so that the attention weights can focus on specific keys
  - $A_{mqk} \approx \frac{1}{N_k}$ at initialization, which leads to ambiguous gradients for inputs
    - N_k is the number of key elements
  - In the image domain, where the key elements are usually of image pixels, N_k can be very large and the convergence is tedious
- DETR requires much longer training epochs to converge
  - Attention modules processing image features are difficult to train

Vaswani, Ashish, et al. "Attention is all you need." In NeurIPS 2017

Issues of Multi-Head Attention in Transformers

$$\text{MultiHeadAttn}(\boldsymbol{z}_q, \boldsymbol{x}) = \sum_{m=1}^{M} \boldsymbol{W}_m \Big[\sum_{k \in \Omega_k} A_{mqk} \cdot \boldsymbol{W}'_m \boldsymbol{x}_k \Big]$$

- the computational and memory complexity can be very high
  - Computational complexity $O(N_q C^2 + N_k C^2 + N_q N_k C)$,
    - N_q and N_k are the number of query and key elements,
    - C is the feature dimension
  - In the image domain, where the query and key elements are both of pixels, $N_q = N_k \gg C$, the complexity is dominated by $O(N_q N_k C)$
- DETR delivers low performance at detecting small objects
  - Modern detectors use high-resolution feature maps to better detect small objects
  - high-resolution feature maps lead to unacceptable complexity

Vaswani, Ashish, et al. "Attention is all you need." In NeurIPS 2017

Issues of Multi-Head Attention in Transformers

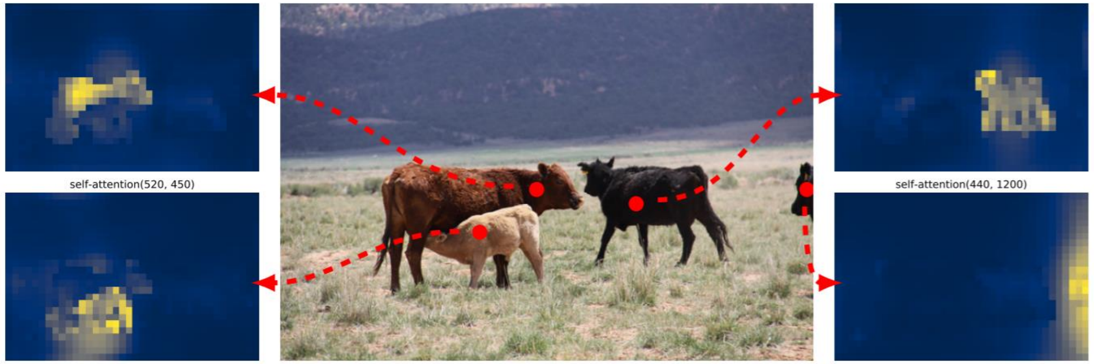
$$\text{MultiHeadAttn}(\boldsymbol{z}_q, \boldsymbol{x}) = \sum_{m=1}^{M} \boldsymbol{W}_m \Big[\sum_{k \in \Omega_k} A_{mqk} \cdot \boldsymbol{W}'_m \boldsymbol{x}_k \Big]$$

- long training schedules are required so that the attention weights can focus on specific keys
- computational and memory complexity can be very high

The core issue is Transformer attention would look over all possible spatial locations

Vaswani, Ashish, et al. "Attention is all you need." In NeurIPS 2017

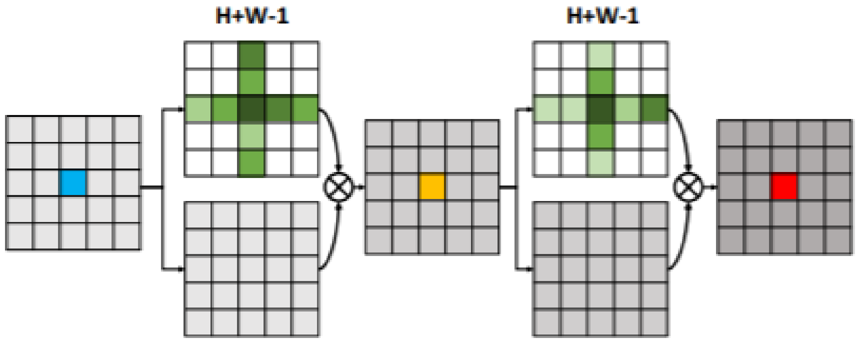
Efficient Sparse Attention in Image Domain



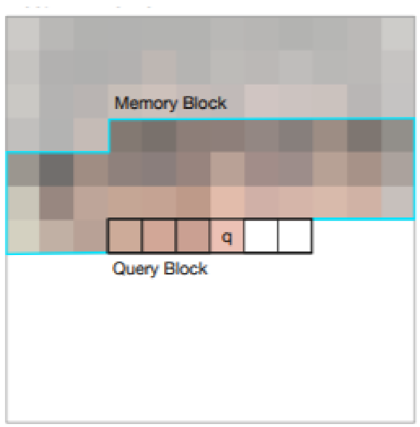
Dense attention (e.g., Transformer[1], Non-Local[2])

look over **all possible** spatial locations

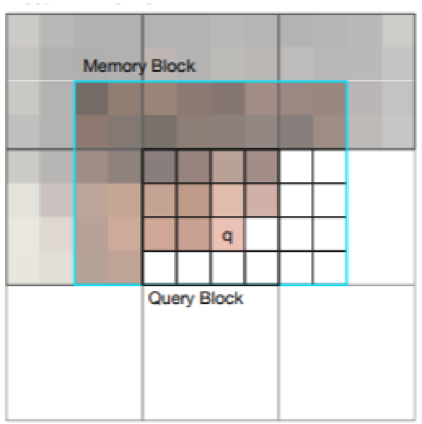
look over **pre-defined sparse** spatial locations



attention along each axis
(e.g., Axial Attention[3], CCNet[4])



1D local attention
(e.g., Image Transformer[5])

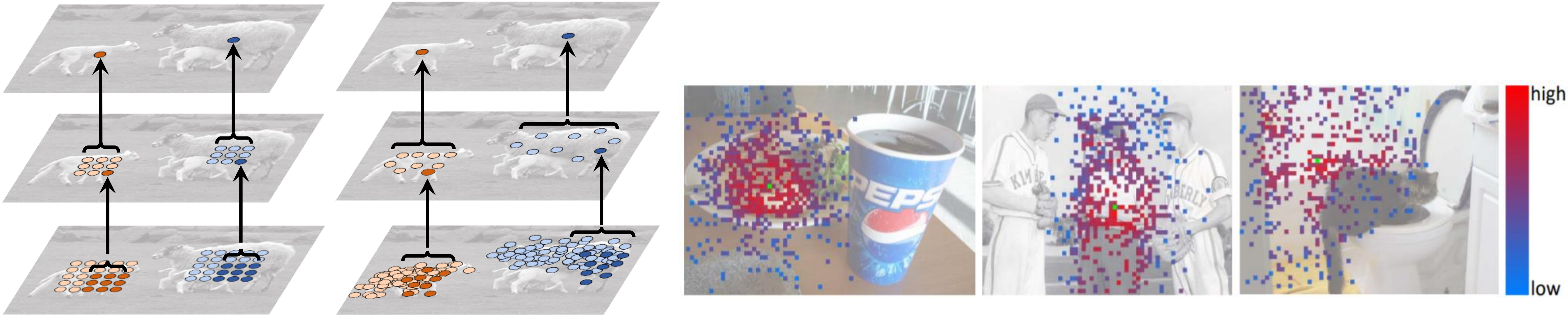


2D local attention
(e.g., Image Transformer[5], Stand-Alone[6], Local Relation[7])

[1] Vaswani, Ashish, et al. “Attention is all you need.” In NeurIPS 2017
[2] Wang, Xiaolong, et al. "Non-local neural networks." In CVPR 2018.
[3] Ho, Jonathan, et al. "Axial Attention in Multidimensional Transformers.“ In ICLR 2020.
[4] Huang, Zilong, et al. "Ccnet: Criss-cross attention for semantic segmentation.“ In ICCV 2019.
[5] Parmar, Niki, et al. "Image transformer." In PMLR 2018.
[6] Ramachandran, Prajit, et al. "Stand-alone self-attention in vision models.“ In NeurIPS 2019.
[7] Hu, Han, et al. "Local relation networks for image recognition.“ In ICCV 2019.

much slower in implementation than traditional convolution with the same FLOPs

Deformable Convolution as Self-Attention



(a) standard convolution (b) deformable convolution (c) effective sampling locations in deformable convolutions

Deformable convolution is effective and efficient on image recognition

However, it lacks the element relation modeling mechanism.

[1] Dai, Jifeng, et al. "Deformable convolutional networks." In ICCV 2017.
[2] Zhu, Xizhou, et al. "Deformable convnets v2: More deformable, better results." In CVPR 2019.

Deformable DETR

- Deformable Attention

query element

key elements

Learnable weights

$$\mathrm{DeformAttn}(\boldsymbol{z}_q, \boldsymbol{p}_q, \boldsymbol{x}) = \sum_{m=1}^{M} \boldsymbol{W}_m \Big[\sum_{k=1}^{K} A_{mqk} \cdot \boldsymbol{W}'_m \boxed{\boldsymbol{x}(\boldsymbol{p}_q + \Delta \boldsymbol{p}_{mqk})}\Big]$$

reference point

Sum over attention heads

Attention weights

Sparsely sampled key feature

  - It only attends to a small set of key sampling points around a reference point, regardless of the spatial size of the feature maps
  - K is the total sampled key number ($K \ll HW$)

Deformable DETR

- Deformable Attention

query element

key elements

Learnable weights

$$\mathrm{DeformAttn}(\boldsymbol{z}_q, \boldsymbol{p}_q, \boldsymbol{x}) = \sum_{m=1}^{M} \boldsymbol{W}_m \Big[\sum_{k=1}^{K} A_{mqk} \cdot \boldsymbol{W}'_m \boxed{\boldsymbol{x}(\boldsymbol{p}_q + \Delta \boldsymbol{p}_{mqk})}\Big]$$

reference point

Sum over attention heads

Attention weights

sparse sampled key feature

- Equivalent to **Deformable Convolution**, when $K = 1$ and W'_m is fixed as an identity matrix
- Equivalent to **Transformer Attention**, when $K = HW$ and the sampling points traverse all possible locations

Deformable DETR

- Deformable Attention

$$\text{DeformAttn}(\boldsymbol{z}_q, \boldsymbol{p}_q, \boldsymbol{x}) = \sum_{m=1}^{M} \boldsymbol{W}_m \left[\sum_{k=1}^{K} A_{mqk} \cdot \boldsymbol{W}'_m \boldsymbol{x}(\boldsymbol{p}_q + \Delta \boldsymbol{p}_{mqk}) \right]$$

- Multi-scale Deformable Attention

$$\text{MSDeformAttn}(\boldsymbol{z}_q, \hat{\boldsymbol{p}}_q, \{\boldsymbol{x}^l\}_{l=1}^{L}) = \sum_{m=1}^{M} \boldsymbol{W}_m \left[\sum_{l=1}^{L} \sum_{k=1}^{K} A_{mlqk} \cdot \boldsymbol{W}'_m \boldsymbol{x}^l(\phi_l(\hat{\boldsymbol{p}}_q) + \Delta \boldsymbol{p}_{mlqk}) \right]$$

  - normalized coordinates $\hat{p}_q \in [0, 1]^2$ for the clarity of scale formulation
  - function $\phi_l(\hat{p}_q)$ re-scales the normalized coordinates $\hat{p}_q$ to the input feature map of the l-th level

Deformable DETR

- Deformable Attention

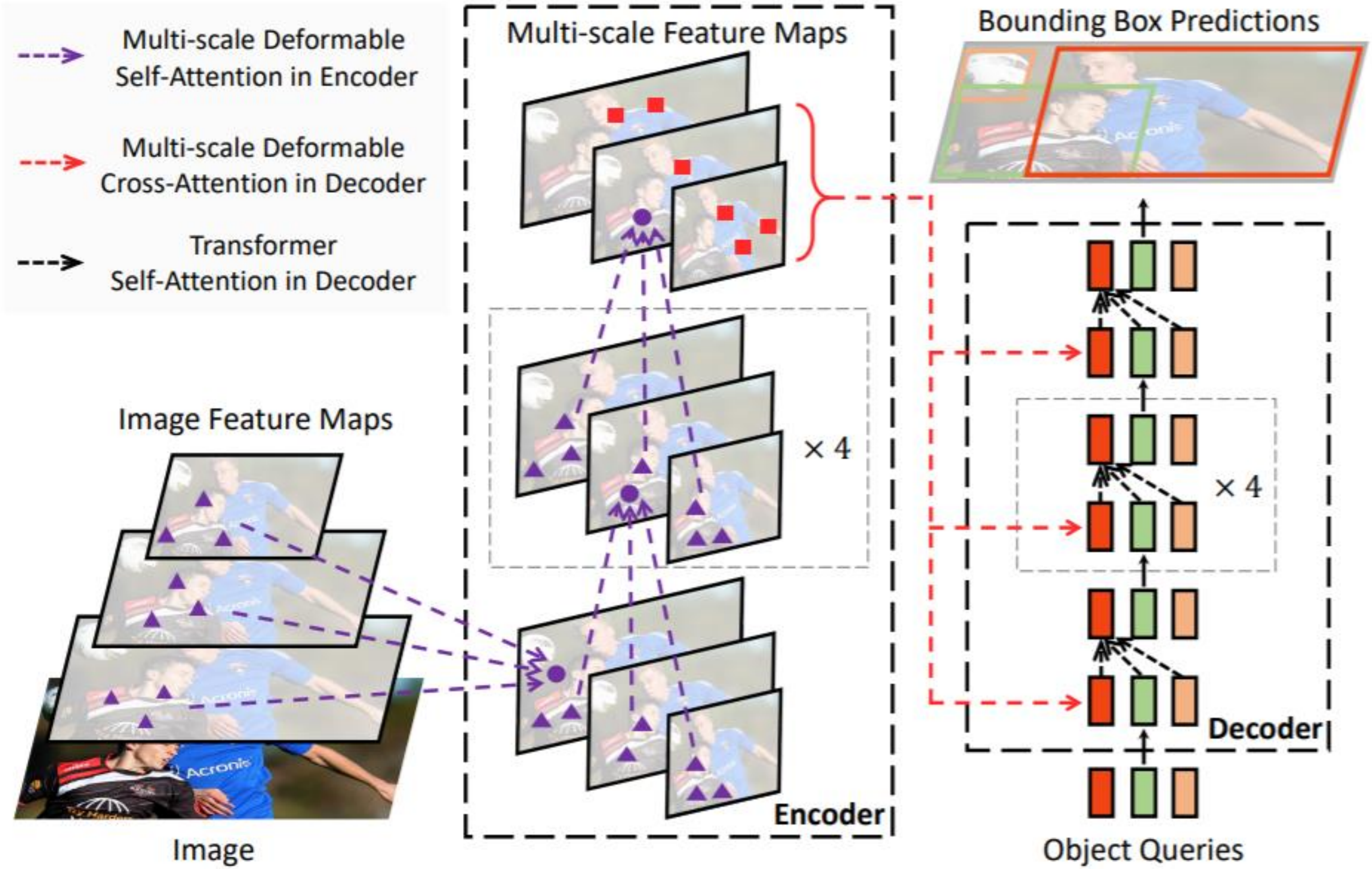
$$\text{DeformAttn}(\boldsymbol{z}_q, \boldsymbol{p}_q, \boldsymbol{x}) = \sum_{m=1}^{M} \boldsymbol{W}_m \left[\sum_{k=1}^{K} A_{mqk} \cdot \boldsymbol{W}'_m \boldsymbol{x}(\boldsymbol{p}_q + \Delta \boldsymbol{p}_{mqk}) \right]$$

- Multi-scale Deformable Attention

$$\text{MSDeformAttn}(\boldsymbol{z}_q, \hat{\boldsymbol{p}}_q, \{\boldsymbol{x}^l\}_{l=1}^{L}) = \sum_{m=1}^{M} \boldsymbol{W}_m \left[\sum_{l=1}^{L} \sum_{k=1}^{K} A_{mlqk} \cdot \boldsymbol{W}'_m \boldsymbol{x}^l(\phi_l(\hat{\boldsymbol{p}}_q) + \Delta \boldsymbol{p}_{mlqk}) \right]$$

  - In Transformer encoder, for each query pixel, the reference point $\hat{p}_q$ is itself
  - In Transformer decoder, the reference point $\hat{p}_q$ is predicted from its object query embedding via a learnable linear projection followed by a sigmoid function

Deformable DETR



Experiments

Table 1: Comparision of Deformable DETR with DETR on COCO 2017 val set. DETR-DC5$^+$ denotes DETR-DC5 with Focal Loss and 300 object queries.

Method	Epochs	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L	params	FLOPs	FPS
Faster R-CNN + FPN	109	42.0	62.1	45.5	26.6	45.4	53.4	42M	180G	26
DETR	500	42.0	62.4	44.2	20.5	45.8	61.1	41M	86G	28
DETR-DC5	500	43.3	63.1	45.9	22.5	47.3	61.1	41M	187G	12
DETR-DC5	50	35.3	55.7	36.8	15.2	37.5	53.6	41M	187G	12
DETR-DC5$^+$	50	36.2	57.0	37.4	16.3	39.2	53.9	41M	187G	12
Deformable DETR	50	43.8	62.6	47.7	26.4	47.1	58.0	40M	173G	19
+ iterative bounding box refinement	50	45.4	64.7	49.0	26.8	48.3	61.7	40M	173G	19
++ two-stage Deformable DETR	50	46.2	65.2	50.0	28.8	49.2	61.7	40M	173G	19

Deformable DETR achieves better performance (especially on small objects) with 10× less training epochs

Experiments

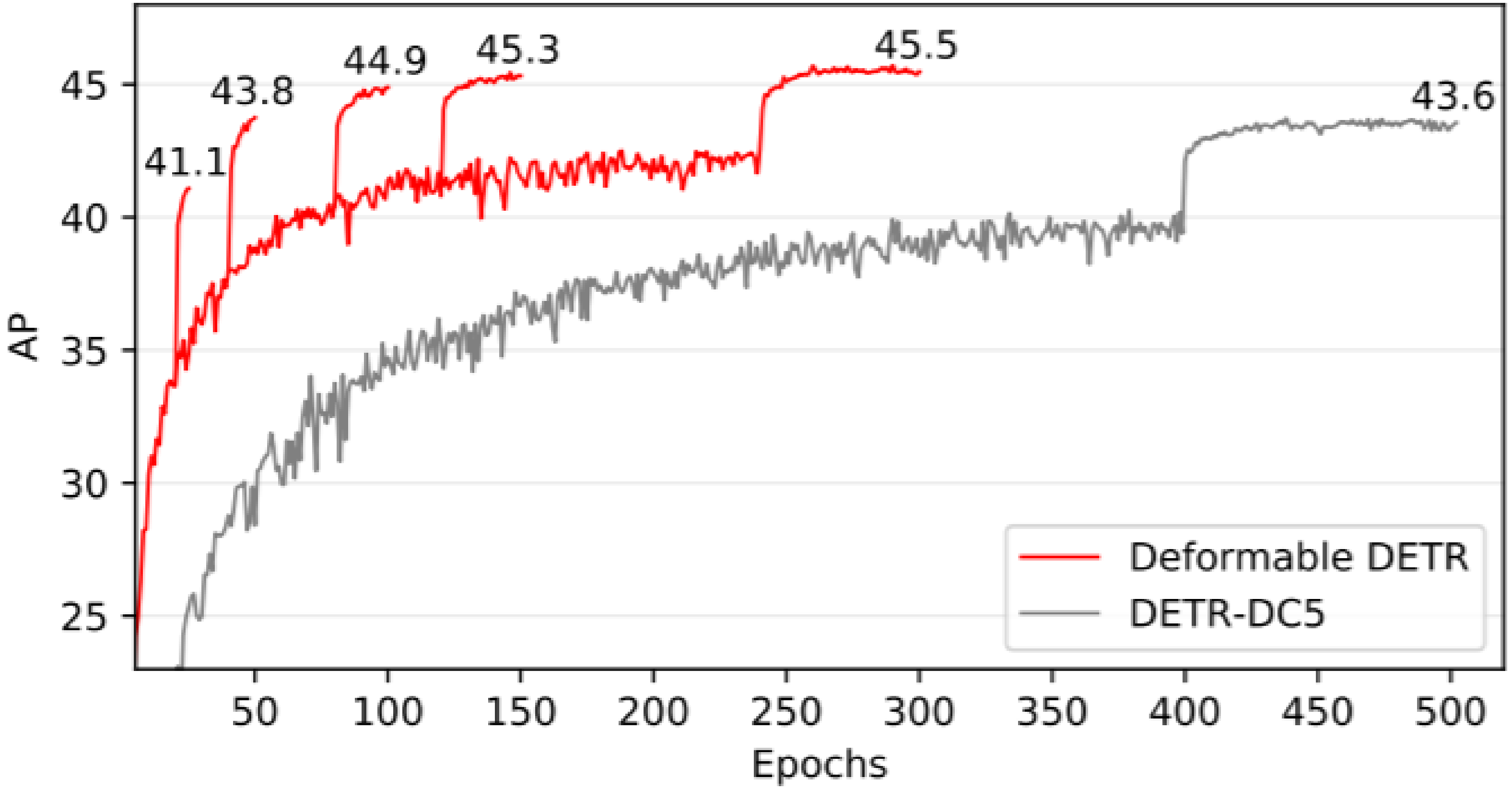


Figure 2: Convergence curves of Deformable DETR and DETR-DC5 on COCO 2017 val set. For Deformable DETR, we explore different training schedules by varying the epochs at which the learning rate is reduced (where the AP score leaps).

Experiments

Table 2: Ablations for deformable attention on COCO 2017 val set. "MS inputs" indicates using multi-scale inputs. "MS attention" indicates using multi-scale deformable attention. K is the number of sampling points for each attention head on each feature level.

MS inputs	MS attention	K	FPNs	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
✓	✓	4	FPN (Lin et al., 2017a)	43.8	62.6	47.8	26.5	47.3	58.1
✓	✓	4	BiFPN (Tan et al., 2020)	43.9	62.5	47.7	25.6	47.4	57.7
		1	w/o	39.7	60.1	42.4	21.2	44.3	56.0
✓		1	w/o	41.4	60.9	44.9	24.1	44.6	56.1
✓		4	w/o	42.3	61.4	46.0	24.8	45.1	56.3
✓	✓	4	w/o	43.8	62.6	47.7	26.4	47.1	58.0

Using multi-scale inputs can effectively improve detection accuracy with 1.7% AP, especially on small objects with 2.9% AP_S

Experiments

Table 2: Ablations for deformable attention on COCO 2017 val set. "MS inputs" indicates using multi-scale inputs. "MS attention" indicates using multi-scale deformable attention. K is the number of sampling points for each attention head on each feature level.

MS inputs	MS attention	K	FPNs	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
✓	✓	4	FPN (Lin et al., 2017a)	43.8	62.6	47.8	26.5	47.3	58.1
✓	✓	4	BiFPN (Tan et al., 2020)	43.9	62.5	47.7	25.6	47.4	57.7
		1	w/o	39.7	60.1	42.4	21.2	44.3	56.0
✓		1	w/o	41.4	60.9	44.9	24.1	44.6	56.1
✓		4	w/o	42.3	61.4	46.0	24.8	45.1	56.3
✓	✓	4	w/o	43.8	62.6	47.7	26.4	47.1	58.0

Increasing the number of sampling points K can further improve 0.9% AP

Experiments

Table 2: Ablations for deformable attention on COCO 2017 val set. "MS inputs" indicates using multi-scale inputs. "MS attention" indicates using multi-scale deformable attention. K is the number of sampling points for each attention head on each feature level.

MS inputs	MS attention	K	FPNs	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
✓	✓	4	FPN (Lin et al., 2017a)	43.8	62.6	47.8	26.5	47.3	58.1
✓	✓	4	BiFPN (Tan et al., 2020)	43.9	62.5	47.7	25.6	47.4	57.7
		1	w/o	39.7	60.1	42.4	21.2	44.3	56.0
✓		1		41.4	60.9	44.9	24.1	44.6	56.1
✓		4		42.3	61.4	46.0	24.8	45.1	56.3
✓	✓	4		43.8	62.6	47.7	26.4	47.1	58.0

Using multi-scale deformable attention, which allows information exchange among different scale levels, can bring additional 1.5% improvement in AP

Experiments

Table 2: Ablations for deformable attention on COCO 2017 val set. "MS inputs" indicates using multi-scale inputs. "MS attention" indicates using multi-scale deformable attention. K is the number of sampling points for each attention head on each feature level.

MS inputs	MS attention	K	FPNs	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
✓	✓	4	FPN (Lin et al., 2017a)	43.8	62.6	47.8	26.5	47.3	58.1
✓	✓	4	BiFPN (Tan et al., 2020)	43.9	62.5	47.7	25.6	47.4	57.7
		1	w/o	39.7	60.1	42.4	21.2	44.3	56.0
✓		1	w/o	41.4	60.9	44.9	24.1	44.6	56.1
✓		4	w/o	42.3	61.4	46.0	24.8	45.1	56.3
✓	✓	4	w/o	43.8	62.6	47.7	26.4	47.1	58.0

Because the cross-level feature exchange is already adopted, adding FPNs will not improve the performance

Experiments

Table 3: Comparison of Deformable DETR with state-of-the-art methods on COCO 2017 test-dev set. "TTA" indicates test-time augmentations including horizontal flip and multi-scale testing.

Method	Backbone	TTA	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
FCOS (Tian et al., 2019)	ResNeXt-101		44.7	64.1	48.4	27.6	47.5	55.6
ATSS (Zhang et al., 2020)	ResNeXt-101 + DCN	✓	50.7	68.9	56.3	33.2	52.9	62.4
TSD (Song et al., 2020)	SENet154 + DCN	✓	51.2	71.9	56.0	33.8	54.8	64.2
EfficientDet-D7 (Tan et al., 2020)	EfficientNet-B6		52.2	71.4	56.3	-	-	-
Deformable DETR	ResNet-50		46.9	66.4	50.8	27.7	49.7	59.9
Deformable DETR	ResNet-101		48.7	68.1	52.9	29.1	51.5	62.0
Deformable DETR	ResNeXt-101		49.0	68.5	53.2	29.7	51.7	62.8
Deformable DETR	ResNeXt-101 + DCN		50.1	69.7	54.6	30.6	52.8	64.7
Deformable DETR	ResNeXt-101 + DCN	✓	52.3	71.9	58.1	34.4	54.4	65.6

Conclusion

- Deformable DETR is an end-to-end object detector, which is efficient and fast-converging.
- Compared with DETR, Deformable DETR can achieve better performance (especially on small objects) with 10× less training epochs.
- It enables us to explore more interesting and practical variants of end-to-end object detectors.

- We hope our work opens up new possibilities in exploring end-to-end object detection.
- Code is released at

  https://github.com/fundamentalvision/Deformable-DETR

We are hiring!

- Email: daijifeng@sensetime.com