DEFORMABLE DETR: DEFORMABLE TRANSFORMERS FOR END-TO-END OBJECT DETECTION

Anonymous authors
Paper under double-blind review

Abstract

DETR has been recently proposed to eliminate the need for many hand-designed components in object detection while demonstrating good performance. However, it suffers from slow convergence and limited feature spatial resolution, due to the limitation of Transformer attention modules in processing image feature maps. To mitigate these issues, we proposed Deformable DETR, whose attention modules only attend to a small set of key sampling points around a reference. Deformable DETR can achieve better performance than DETR (especially on small objects) with $10\times$ less training epochs. Extensive experiments on the COCO benchmark demonstrate the effectiveness of our approach. Code shall be released.

1 Introduction

Modern object detectors employ many hand-crafted components (Liu et al., 2020), e.g., anchor generation, rule-based training target assignment, non-maximum suppression (NMS) post-processing. They are not fully end-to-end. Recently, Carion et al. (2020) proposed DETR to eliminate the need for such hand-crafted components, and built the first fully end-to-end object detector, achieving very competitive performance. DETR utilizes a simple architecture, by combining convolutional neural networks (CNNs) and Transformer (Vaswani et al., 2017) encoder-decoders. They exploit the versatile and powerful relation modeling capability of Transformers to replace the hand-crafted rules, under properly designed training signals.

Despite its interesting design and good performance, DETR has its own issues: (1) It requires much longer training epochs to converge than the existing object detectors. For example, on the COCO (Lin et al., 2014) benchmark, DETR needs 500 epochs to converge, which is around 10 to 20 times slower than Faster R-CNN (Ren et al., 2015). (2) DETR delivers relatively low performance at detecting small objects. Modern object detectors usually exploit multi-scale features, where small objects are detected from high-resolution feature maps. Meanwhile, high-resolution feature maps lead to unacceptable complexities for DETR. The above-mentioned issues can be mainly attributed to the deficit of Transformer components in processing image feature maps. At initialization, the attention modules cast nearly uniform attention weights to all the pixels in the feature maps. Long training epoches is necessary for the attention weights to be learned to focus on sparse meaningful locations. On the other hand, the attention weights computation in Transformer encoder is of quadratic computation with respect to pixel numbers. Thus, it is of very high computational and memory complexities to process high-resolution feature maps.

In the image domain, deformable convolution (Dai et al., 2017) is of a powerful and efficient mechanism to attend to sparse spatial locations. It naturally avoids the above-mentioned issues. While it lacks the element relation modeling mechanism, which is the key for the success of DETR.

In this paper, we propose Deformable DETR, which mitigates the slow convergence and high complexity issues of DETR. It combines the best of the sparse spatial sampling of deformable convolution, and the relation modeling capability of Transformers. We propose the deformable attention module, which attends to a small set of sampling locations as a pre-filter for prominent key elements out of all the feature map pixels. The module can be naturally extended to aggregating multi-scale features, without the help of FPN (Lin et al., 2017a). In Deformable DETR, we utilize (multi-scale) deformable attention modules to replace the Transformer attention modules processing feature maps, as shown in Fig. 1.
Deformable DETR opens up possibilities for us to exploit variants of end-to-end object detectors, thanks to its fast convergence, and computational and memory efficiency. We explore a simple and effective iterative bounding box refinement mechanism to improve the detection performance. We also try a two-stage Deformable DETR, where the region proposals are also generated by a variant of Deformable DETR, which are further fed into the decoder for iterative bounding box refinement.

Extensive experiments on the COCO (Lin et al., 2014) benchmark demonstrate the effectiveness of our approach. Compared with DETR, Deformable DETR can achieve better performance (especially on small objects) with 10× less training epochs. The proposed variant of two-stage Deformable DETR can further improve the performance. Code shall be released.

2 RELATED WORK

Efficient Attention Mechanism. Transformers (Vaswani et al., 2017) involve both self-attention and cross-attention mechanisms. One of the most well-known concern of Transformers is the high time and memory complexity at vast key element numbers, which hinders model scalability in many cases. Recently, many efforts have been made to address this problem (Tay et al., 2020b), which can be roughly divided into three categories in practice.

The first category is to use pre-defined sparse attention patterns on keys. The most straightforward paradigm is restricting the attention pattern to be fixed local windows. Most works (Liu et al., 2018a; Parmar et al., 2018; Child et al., 2019; Huang et al., 2019; Ho et al., 2019; Hu et al., 2019; Parmar et al., 2019; Qiu et al., 2019; Beltagy et al., 2020; Ainslie et al., 2020; Zaheer et al., 2020) follow this paradigm. Although restricting the attention pattern to a local neighborhood can decrease the complexity, it loses global information. To compensate, Child et al. (2019); Huang et al. (2019); Ho et al. (2019) attend key elements at fixed intervals to significantly increase the receptive field on keys. Beltagy et al. (2020); Ainslie et al. (2020); Zaheer et al. (2020) allow a small number of special tokens having access to all key elements. Zaheer et al. (2020); Qiu et al. (2019) also add some pre-fixed sparse attention patterns to attend distant key elements directly.

The second category is to learn data-dependent sparse attention. Kitaev et al. (2020) proposes a locality sensitive hashing (LSH) based attention, which hashes both the query and key elements to different bins. A similar idea is proposed by Roy et al. (2020), where k-means finds out the most related keys. Tay et al. (2020a) learns block permutation for block-wise sparse attention.

The third category is to explore the low-rank property in self-attention. Wang et al. (2020) reduces the number of key elements through a linear projection on the size dimension instead of the channel
dimension. [Katharopoulos et al. (2020); Choromanski et al. (2020)] rewrite the calculation of self-attention through kernelization approximation.

In the image domain, the designs of efficient attention mechanism (e.g., Parmar et al. (2018); Child et al. (2019); Huang et al. (2019); Ho et al. (2019); Hu et al. (2019); Parmar et al. (2019)) are still limited to the first category. Despite the theoretically reduced complexity, [Parmar et al. (2019); Hu et al. (2019)] admit such approaches are much slower in implementation than traditional convolution with the same FLOPs (at least 3x slower), due to the intrinsic limitation in memory access patterns.

On the other hand, as discussed in [Zhu et al. (2019a)], there are variants of convolution, such as deformable convolution (Dai et al. (2017); Zhu et al. (2019b)) and dynamic convolution (Wu et al., 2019), that also can be viewed as self-attention mechanisms. Especially, deformable convolution operates much more effectively and efficiently on image recognition than Transformer self-attention. Meanwhile, it lacks the element relation modeling mechanism.

Our proposed deformable attention module is inspired by deformable convolution, and belongs to the second category. It only focuses on a small fixed set of sampling points predicted from the feature of query elements. Different from Parmar et al. (2019); Hu et al. (2019), deformable attention is just slightly slower than the traditional convolution under the same FLOPs.

**Multi-scale Feature Representation for Object Detection.** One of the main difficulties in object detection is to effectively represent objects at vastly different scales. Modern object detectors usually exploit multi-scale features to accommodate this. As one of the pioneering works, FPN (Lin et al., 2017a) proposes a top-down path to combine multi-scale features. PANet (Liu et al., 2018b) further adds an bottom-up path on the top of FPN. Kong et al. (2018) combines features from all scales with a global attention operation. Zhao et al. (2019) proposes a U-shape module to fuse multi-scale features. Recently, NAS-FPN (Ghiasi et al., 2019) and Auto-FPN (Xu et al., 2019) are proposed to automatically design cross-scale connections via neural architecture search. Tan et al. (2020) proposes the BiFPN, which is a repeated simplified version of PANet. Our proposed multi-scale deformable attention module can naturally aggregate multi-scale feature maps via attention mechanism, without the help of these feature pyramid networks.

**3 Revisiting Transformers and DETR**

**Multi-Head Attention in Transformers.** Transformers (Vaswani et al., 2017) are of a network architecture based on attention mechanisms for machine translation. Given a query element (e.g., a target word in the output sentence) and a set of key elements (e.g., source words in the input sentence), the multi-head attention module adaptively aggregates the key contents according to the attention weights that measure the compatibility of query-key pairs. To allow the model focusing on contents from different representation subspaces and different positions, the outputs of different attention heads are linearly aggregated with learnable weights. Let \( q \in \Omega_q \) indexes a query element with representation feature \( z_q \in \mathbb{R}^C \), and \( k \in \Omega_k \) indexes a key element with representation feature \( x_k \in \mathbb{R}^C \), where \( C \) is the feature dimension, \( \Omega_q \) and \( \Omega_k \) specify the set of query and key elements, respectively. Then the multi-head attention feature is calculated by

\[
\text{MultiHeadAttn}(z_q, x) = \sum_{m=1}^{M} W_m \left[ \sum_{k \in \Omega_k} A_{mqk} \cdot W'_m x_k \right],
\]

where \( m \) indexes the attention head, \( W'_m \in \mathbb{R}^{C_x \times C} \) and \( W_m \in \mathbb{R}^{C \times C_x} \) are of learnable weights (\( C_x = C/M \) by default). The attention weights \( A_{mqk} \propto \exp \left( \frac{z_q \cdot U_m x_k}{\sqrt{C}} \right) \) are normalized as \( \sum_{k \in \Omega_k} A_{mqk} = 1 \), in which \( U_m, V_m \in \mathbb{R}^{C_x \times C} \) are also learnable weights. To disambiguate different spatial positions, the representation features \( z_q \) and \( x_k \) are usually of the concatenation/summation of element contents and positional embeddings.

There are two known issues with Transformers. One is Transformers need long training schedules before convergence. Suppose the number of query and key elements are of \( N_q \) and \( N_k \), respectively. Typically, with proper parameter initialization, \( U_m z_q \) and \( V_m x_k \) follow distribution with mean of 0 and variance of 1, which makes attention weights \( A_{mqk} \approx \frac{1}{N_k} \), when \( N_k \) is large. It will lead to ambiguous gradients for input features. Thus, long training schedules are required so that the
attention weights can focus on specific keys. In the image domain, where the key elements are usually of image pixels, \( N_k \) can be very large and the convergence is tedious.

On the other hand, the computational and memory complexity for multi-head attention can be very high with numerous query and key elements. The computational complexity of Eq. \([1]\) is of \( O(N_qC^2 + N_kC^2 + N_qN_kC) \). 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 the third term, as \( O(N_qN_kC) \). Thus, the multi-head attention module suffers from a quadratic complexity growth with the feature map size.

DETR. DETR (Carion et al., 2020) is built upon the Transformer encoder-decoder architecture, combined with a set-based Hungarian loss that forces unique predictions for each ground-truth bounding box via bipartite matching. We briefly review the network architecture as follows.

Given the input feature maps \( x \in \mathbb{R}^{C \times H \times W} \) extracted by a CNN backbone (e.g., ResNet (He et al., 2016)), DETR exploits a standard Transformer encoder-decoder architecture to transform the input feature maps to be features of a set of object queries. A 3-layer feed-forward neural network (FFN) and a linear projection are added on top of the object query features (produced by the decoder) as the detection head. The FFN acts as the regression branch to predict the bounding box coordinates \( b \in [0, 1]^4 \), where \( b = \{b_x, b_y, b_w, b_h\} \) encodes the normalized box center coordinates, box height and width (relative to the image size). The linear projection acts as the classification branch to produce the classification results.

For the Transformer encoder in DETR, both query and key elements are of pixels in the feature maps. The inputs are of ResNet feature maps (with encoded positional embeddings). Let \( H \) and \( W \) denote the feature map height and width, respectively. The computational complexity of self-attention is of \( O(H^2W^2C) \), which grows quadratically with the spatial size.

For the Transformer decoder in DETR, the input includes both feature maps from the encoder, and \( N \) object queries represented by learnable positional embeddings (e.g., \( N = 100 \)). There are two types of attention modules in the decoder, namely, cross-attention and self-attention modules. In the cross-attention modules, object queries extract features from the feature maps. The query elements are of the object queries, and key elements are of the output feature maps from the encoder. In it, \( N_q = N \), \( N_k = H \times W \) and the complexity of the cross-attention is of \( O(HWC^2 + NHWC) \).

The complexity grows linearly with the spatial size of feature maps. In the self-attention modules, object queries interact with each other, so as to capture their relations. The query and key elements are both of the object queries. In it, \( N_q = N_k = N \), and the complexity of the self-attention module is of \( O(2NC^2 + N^2C) \). The complexity is acceptable with moderate number of object queries.

DETR is an attractive design for object detection, which removes the need for many hand-designed components. However, it also has its own issues. These issues can be mainly attributed to the deficits of Transformer attention in handling image feature maps as key elements: (1) DETR has relatively low performance in detecting small objects. Modern object detectors use high-resolution feature maps to better detect small objects. However, high-resolution feature maps would lead to an unacceptable complexity for the self-attention module in the Transformer encoder of DETR, which has a quadratic complexity with the spatial size of input feature maps. (2) Compared with modern object detectors, DETR requires many more training epochs to converge. This is mainly because the attention modules processing image features are difficult to train. For example, at initialization, the cross-attention modules are almost of average attention on the whole feature maps. While, at the end of the training, the attention maps are learned to be very sparse, focusing only on the object extremities. It seems that DETR requires a long training schedule to learn such significant changes in the attention maps.

4 Method

4.1 Deformable Transformers for End-to-End Object Detection

Deformable Attention Module. The core issue of applying Transformer attention on image feature maps is that it would look over all possible spatial locations. To address this, we present a deformable attention module. Inspired by deformable convolution (Dai et al., 2017; Zhu et al., 2019b), the deformable attention module only attends to a small set of key sampling points around a reference
point, regardless of the spatial size of the feature maps. By assigning only a small fixed number of keys for each query, the issues of convergence and feature spatial resolution can be mitigated.

Given an input feature map \( x \in \mathbb{R}^{C \times H \times W} \), let \( q \) index a query element with content feature \( z_q \) and a 2-d reference point \( p_q \), the deformable attention feature is calculated by

\[
\text{DeformAttn}(z_q, p_q, x) = \sum_{m=1}^{M} W_m \left[ \sum_{k=1}^{K} A_{mqk} \cdot W'_m x(p_q + \Delta p_{mqk}) \right],
\]

where \( m \) indexes the attention head, \( k \) indexes the sampled keys, and \( K \) is the total sampled key number (\( K \ll HW \)). \( \Delta p_{mqk} \) and \( A_{mqk} \) denote the sampling offset and attention weight of the \( k \)-th sampling point in the \( m \)-th attention head, respectively. The scalar attention weight \( A_{mqk} \) lies in the range \([0, 1]\), normalized by \( \sum_{k=1}^{K} A_{mqk} = 1 \). \( \Delta p_{mqk} \) is fractional, bilinear interpolation is applied as in [Dai et al., 2017] in computing \( x(p_q + \Delta p_{mqk}) \). Both \( \Delta p_{mqk} \) and \( A_{mqk} \) are obtained via linear projection over the query feature \( z_q \). In implementation, the query feature \( z_q \) is fed to a linear projection operator of \( 3MK \) channels, where the first \( 2MK \) channels encode the sampling offsets \( \Delta p_{mqk} \) and the remaining \( MK \) channels are fed to a softmax operator to obtain the attention weights \( A_{mqk} \).

The deformable attention module is designed for processing convolutional feature maps as key elements. Let \( N_q \) be the number of query elements, when \( MK \) is relatively small, the complexity of the deformable attention module is of \( O(2N_qC^2 + \min(HWC^2, N_qKC^2)) \) (See Appendix A.1 for details). When it is applied in DETR encoder, where \( N_q = HW \), the complexity becomes \( O(HWC^2) \), which is of linear complexity with the spatial size. When it is applied as the cross-attention modules in DETR decoder, where \( N_q = N \) (\( N \) is the number of object queries), the complexity becomes \( O(NKC^2) \), which is irrelevant to the spatial size \( HW \).

**Multi-scale Deformable Attention Module.** Most modern object detection frameworks benefit from multi-scale feature maps (Liu et al., 2020). Our proposed deformable attention module can be naturally extended for multi-scale feature maps.

Let \( \{x^l\}_{l=1}^{L} \) be the input multi-scale feature maps, where \( x^l \in \mathbb{R}^{C \times H_l \times W_l} \). Let \( \hat{p}_q \in [0, 1]^2 \) be the normalized coordinates of the reference point for each query element \( q \), then the multi-scale deformable attention module is applied as

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

where \( m \) indexes the attention head, \( l \) indexes the input feature level, and \( k \) indexes the sampling point. \( \Delta p_{mlqk} \) and \( A_{mlqk} \) denote the sampling offset and attention weight of the \( k \)-th sampling point in the \( l \)-th feature level and the \( m \)-th attention head, respectively. The scalar attention weight \( A_{mlqk} \) is normalized by \( \sum_{l=1}^{L} \sum_{k=1}^{K} A_{mlqk} = 1 \). Here, we use normalized coordinates \( \hat{p}_q \in [0, 1]^2 \) for the clarity of scale formulation, in which the normalized coordinates \((0, 0)\) and \((1, 1)\) indicate the top-left and the bottom-right image corners, respectively. Function \( \hat{p}_l(\hat{p}_q) \) in Equation 3 re-scales the normalized coordinates \( \hat{p}_q \) to the input feature map of the \( l \)-th level. The multi-scale deformable attention is very similar to the previous single-scale version, except that it samples \( LK \) points from multi-scale feature maps instead of \( K \) points from single-scale feature maps.

The proposed attention module will degenerate to deformable convolution [Dai et al., 2017], when \( L = 1, K = 1 \), and \( W'_m \in \mathbb{R}^{C \times C} \) is fixed as an identity matrix. Deformable convolution is designed for single-scale inputs, focusing only on one sampling point for each attention head. However, our multi-scale deformable attention looks over multiple sampling points from multi-scale inputs. The proposed (multi-scale) deformable attention module can also be perceived as an efficient variant of Transformer attention, where a pre-filtering mechanism is introduced by the deformable sampling locations. When the sampling points traverse all possible locations, the proposed attention module is equivalent to Transformer attention.

**Deformable Transformer Encoder.** We replace the Transformer attention modules processing feature maps in DETR with the proposed multi-scale deformable attention module. Both the input and output of the encoder are of multi-scale feature maps with the same resolutions. In encoder, we extract multi-scale feature maps \( \{x^l\}_{l=1}^{L-1} (L = 4) \) from the output feature maps of stages \( C_3 \) through
C$_5$ in ResNet (He et al., 2016) (transformed by a $1 \times 1$ convolution), where C$_l$ is of resolution $2^l$ lower than the input image. The lowest resolution feature map $x^L$ is obtained via a $3 \times 3$ stride 2 convolution on the final C$_5$ stage, denoted as C$_6$. All the multi-scale feature maps are of C = 256 channels. Note that the top-down structure in FPN (Lin et al., 2017a) is not used, because our proposed multi-scale deformable attention in itself can exchange information among multi-scale feature maps. The constructing of multi-scale feature maps are also illustrated in Appendix A.2. Experiments in Section 5.2 show that adding FPN will not improve the performance.

In application of the multi-scale deformable attention module in encoder, the output are of multi-scale feature maps with the same resolutions as the input. Both the key and query elements are of pixels from the multi-scale feature maps. For each query pixel, the reference point is itself. To identify which feature level each query pixel lies in, we add a scale-level embedding, denoted as $e_l$, to the feature representation, in addition to the positional embedding. Different from the positional embedding with fixed encodings, the scale-level embedding $\{e_l\}_{l=1}^L$ are randomly initialized and jointly trained with the network.

**Deformable Transformer Decoder.** There are cross-attention and self-attention modules in the decoder. The query elements for both types of attention modules are of object queries. In the cross-attention modules, object queries extract features from the feature maps, where the key elements are of the output feature maps from the encoder. In the self-attention modules, object queries interact with each other, where the key elements are of the object queries. Since our proposed deformable attention module is designed for processing convolutional feature maps as key elements, we only replace each cross-attention module to be the multi-scale deformable attention module, while leaving the self-attention modules unchanged. For each object query, the 2-d normalized coordinate of the reference point $\hat{p}_q$ is predicted from its object query embedding via a learnable linear projection followed by a sigmoid function.

Because the multi-scale deformable attention module extracts image features around the reference point, we let the detection head predict the bounding box as relative offsets w.r.t. the reference point to further reduce the optimization difficulty. The reference point is used as the initial guess of the box center. The detection head predicts the relative offsets w.r.t. the reference point. Check Appendix A.3 for the details. In this way, the learned decoder attention will have strong correlation with the predicted bounding boxes, which also accelerates the training convergence.

By replacing Transformer attention modules with deformable attention modules in DETR, we establish an efficient and fast converging detection system, dubbed as Deformable DETR (see Fig. 1).

4.2 **Additional Improvements and Variants for Deformable DETR**

Deformable DETR opens up possibilities for us to exploit various variants of end-to-end object detectors, thanks to its fast convergence, and computational and memory efficiency. Due to limited space, we only introduce the core ideas of these improvements and variants here. The implementation details are given in Appendix A.4.

**Iterative Bounding Box Refinement.** This is inspired by the iterative refinement developed in optical flow estimation (Teed & Deng, 2020). We establish a simple and effective iterative bounding box refinement mechanism to improve detection performance. Here, each decoder layer refines the bounding boxes based on the predictions from the previous layer.

**Two-Stage Deformable DETR.** In the original DETR, object queries in the decoder are irrelevant to the current image. Inspired by two-stage object detectors, we explore a variant of Deformable DETR for generating region proposals as the first stage. The generated region proposals will be fed into the decoder as object queries for further refinement, forming a two-stage Deformable DETR.

In the first stage, to achieve high-recall proposals, each pixel in the multi-scale feature maps would serve as an object query. However, directly setting object queries as pixels will bring unacceptable computational and memory cost for the self-attention modules in the decoder, whose complexity grows quadratically with the number of queries. To avoid this problem, we remove the decoder and form an encoder-only Deformable DETR for region proposal generation. In it, each pixel is assigned as an object query, which directly predicts a bounding box. Top scoring bounding boxes are picked as region proposals. No NMS is applied before feeding the region proposals to the second stage.
5 EXPERIMENT

**Dataset.** We conduct experiments on COCO 2017 dataset \cite{lin2014microsoft}. Our models are trained on the train set, and evaluated on the val set and test-dev set.

**Implementation Details.** ImageNet \cite{deng2009imagenet} pre-trained ResNet-50 \cite{he2016deep} is utilized as the backbone for ablations. Multi-scale feature maps are extracted without FPN \cite{lin2017feature}. \(M = 8\) and \(K = 4\) are set for deformable attentions by default. Parameters of the deformable Transformer encoder are shared among different feature levels. Other hyper-parameter setting and training strategy mainly follow DETR \cite{carion2020end}, except that Focal Loss \cite{lin2017focal} with loss weight of 2 is used for bounding box classification, and the number of object queries is increased from 100 to 300. We also report the performance of DETR-DC5 with these modifications for a fair comparison, denoted as DETR-DC5\(^+\). By default, models are trained for 50 epochs and the learning rate is decayed at the 40-th epoch by a factor of 0.1. Following DETR \cite{carion2020end}, we train our models using Adam optimizer \cite{kingma2015adam} with base learning rate of \(2 \times 10^{-4}\), \(\beta_1 = 0.9\), \(\beta_2 = 0.999\), and weight decay of \(10^{-4}\). Learning rates of the linear projections, used for predicting object query reference points and sampling offsets, are multiplied by a factor of 0.1. Run time is evaluated on NVIDIA Tesla V100 GPU.

5.1 COMPARISON WITH DETR

As shown in Table \(\text{1}\) compared with Faster R-CNN + FPN, DETR requires many more training epochs to converge, and delivers lower performance at detecting small objects. Compared with DETR, Deformable DETR achieves better performance (especially on small objects) with 10\(\times\) less training epochs. Detailed convergence curves are shown in Fig. \(\text{2}\). With the aid of iterative bounding box refinement and two-stage paradigm, our method can further improve the detection accuracy.

Our proposed Deformable DETR has on par FLOPs with Faster R-CNN + FPN and DETR-DC5. But the runtime speed is much faster (1.6\(\times\)) than DETR-DC5, and is just 25\% slower than Faster R-CNN + FPN. The speed issue of DETR-DC5 is mainly due to the large amount of memory access in Transformer attention. Our proposed deformable attention can mitigate this issue, at the cost of unordered memory access. Thus, it is still slightly slower than traditional convolution.

<table>
<thead>
<tr>
<th>Method</th>
<th>Epochs</th>
<th>AP</th>
<th>AP(_{50})</th>
<th>AP(_{75})</th>
<th>AP(_{S})</th>
<th>AP(_{M})</th>
<th>AP(_{L})</th>
<th>params</th>
<th>FLOPs</th>
<th>FPS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Faster R-CNN + FPN</td>
<td>109</td>
<td>42.0</td>
<td>62.1</td>
<td>45.5</td>
<td>26.6</td>
<td>45.4</td>
<td>53.4</td>
<td>42M</td>
<td>180G</td>
<td>26</td>
</tr>
<tr>
<td>DETR</td>
<td>500</td>
<td>42.0</td>
<td>62.4</td>
<td>44.2</td>
<td>20.5</td>
<td>45.8</td>
<td>61.1</td>
<td>41M</td>
<td>86G</td>
<td>28</td>
</tr>
<tr>
<td>DETR-DC5</td>
<td>500</td>
<td>43.3</td>
<td>63.1</td>
<td>45.9</td>
<td>22.5</td>
<td>47.3</td>
<td>61.1</td>
<td>41M</td>
<td>187G</td>
<td>12</td>
</tr>
<tr>
<td>DETR-DC5(^+)</td>
<td>50</td>
<td>35.3</td>
<td>55.7</td>
<td>36.8</td>
<td>15.2</td>
<td>37.5</td>
<td>53.6</td>
<td>41M</td>
<td>187G</td>
<td>12</td>
</tr>
<tr>
<td>Deformable DETR</td>
<td>50</td>
<td>43.8</td>
<td>62.6</td>
<td>47.7</td>
<td>26.4</td>
<td>47.1</td>
<td>58.0</td>
<td>40M</td>
<td>173G</td>
<td>19</td>
</tr>
<tr>
<td>+ iterative bounding box refinement</td>
<td>50</td>
<td>45.4</td>
<td>64.7</td>
<td>49.0</td>
<td>26.8</td>
<td>48.3</td>
<td>61.7</td>
<td>40M</td>
<td>173G</td>
<td>19</td>
</tr>
<tr>
<td>++ two-stage Deformable DETR</td>
<td>50</td>
<td>46.2</td>
<td>65.2</td>
<td>50.0</td>
<td>28.8</td>
<td>49.2</td>
<td>61.7</td>
<td>40M</td>
<td>173G</td>
<td>19</td>
</tr>
</tbody>
</table>

5.2 ABLATION STUDY ON DEFORMABLE ATTENTION

Table\(\text{2}\)presents ablations for various design choices of the proposed deformable attention module. Using multi-scale inputs instead of single-scale inputs can effectively improve detection accuracy with 1.7\% AP, especially on small objects with 2.9\% AP\(_S\). Increasing the number of sampling points \(K\) can further improve 0.9\% AP. Using multi-scale deformable attention, which allows information exchange among different scale levels, can bring additional 1.5\% improvement in AP. Because the cross-level feature exchange is already adopted, adding FPNs will not improve the performance. When multi-scale attention is not applied, and \(K = 1\), our (multi-scale) deformable attention module degenerates to deformable convolution, delivering noticeable lower accuracy.
Deformable DETR is an end-to-end object detector, which is efficient and fast-converging. It enables us to explore more interesting and practical variants of end-to-end object detectors. At the core of Deformable DETR are the (multi-scale) deformable attention modules, which is an efficient attention mechanism in processing image feature maps. We hope our work opens up new possibilities in exploring end-to-end object detection.
REFERENCES


Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In CVPR, 2009.


Shu Liu, Lu Qi, Haifang Qin, Jianping Shi, and Jiaya Jia. Path aggregation network for instance segmentation. In CVPR, 2018b.


