1. DAT and Deformable DETR

In this section, we provide a detailed comparison between our proposed deformable attention and the direct adaptation from the deformable convolution [4], which is also known as the multiscale deformable attention in Deformable DETR [20].

First, our deformable attention serves as a feature extractor in the vision backbones while the one in Deformable DETR which replaces the vanilla attention in DETR [1] with a linear deformable attention, plays the role of the detection head. Second, the \( m \)-th head of query \( q \) in the attention in Deformable DETR with \( M \) heads in a single scale is formulated as

\[
z_q^{(m)} = \sum_{k=1}^{K} A_{qk}^{(m)} W_v \phi(x; p_q + \Delta p_{qk}^{(m)}),
\]

where \( K \) key points are sampled from the input features, mapped by \( W_v \) and then aggregated by attention weights \( A_{qk}^{(m)} \). Compared to our deformable attention (Eq.(9) in the paper), this attention weights is learned from \( x \) by a linear projection, i.e. \( A_{qk}^{(m)} = \sigma(W_{at} x) \), where \( W_{at} \in \mathbb{R}^{C \times MK} \) is the weight matrix to predict the attention weights of each key \( k \) and head \( m \), after which a softmax function \( \sigma \) is applied to the dimensions of \( K \) keys to normalize the attention score. In fact, the attention weights are predicted directly by queries instead of measuring the similarities between queries and keys. If we change the \( \sigma \) function to a sigmoid, this will be a variant of modulated deformable convolution [19], hence this deformable attention is more similar to convolution rather than attention.

Third, the deformable attention in Deformable DETR is not compatible to the dot-product attention for its enormous memory consumption mentioned in Sec.3.2 in the paper. Therefore, the linear predicted attention is used directly by queries instead of measuring the similarities between queries and keys. If we change the \( \sigma \) function to a sigmoid, this will be a variant of modulated deformable convolution [19], hence this deformable attention is more similar to convolution rather than attention.

To experimentally validate our claim, we replace our deformable attention modules in DAT with the modules in [20] to verify that the naive adaptation is inferior for vision backbone. The comparison results are shown in Table 1. To obtain a Deformable DETR under low memory budget, we reduce the number of keys to 16. Comparing the first row and the fourth row, our model achieves 1.2% better performance with similar memory cost. Comparing the third and last row, we can see that the D-DETR attention with the same number of keys as DAT consumes 2.6× memory and 1.3× FLOPs, while the performances are still lower than DAT.

2. Adding Convolutions to DAT

Recent works [2,5,12,13] have proved that adopting convolution layers in the Vision Transformer architecture can further improve model performances. For example, using convolutional patch embedding can generally boost model performances by 0.5% ~ 1.0% on ImageNet classification tasks. It is worth noticing that our proposed DAT can easily combine with these techniques, while we maintain the convolution-free architecture in the main paper to perform fair comparison with baselines.

To fully explore the capacity of DAT, we substitute the patch embedding layers in the original model with strided and overlapped convolutions. The comparison results are shown in Table 2, where baseline models have similar modifications. It is shown that our model with additional convolution modules achieve 0.7% improvement comparing to the original version, and consistently outperform other baselines.
### Table 2. Comparisons of DAT with other vision transformer backbones on FLOPS, parameters, accuracy on the ImageNet-1K classification task. DAT-T refers to the original version. DAT-T* refers to the model with convolutional patch embeddings.

<table>
<thead>
<tr>
<th>Method</th>
<th>FLOPs</th>
<th>#Param</th>
<th>Top-1 Acc.</th>
</tr>
</thead>
<tbody>
<tr>
<td>CvT-T3 [13]</td>
<td>4.5G</td>
<td>20M</td>
<td>81.6</td>
</tr>
<tr>
<td>CoAt-Lite Small [14]</td>
<td>4.0G</td>
<td>20M</td>
<td>81.9</td>
</tr>
<tr>
<td>CeiT-S [15]</td>
<td>4.8G</td>
<td>24M</td>
<td>82.0</td>
</tr>
<tr>
<td>PVTv2-B2 [12]</td>
<td>4.0G</td>
<td>25M</td>
<td>82.0</td>
</tr>
<tr>
<td>CoAt Small [14]</td>
<td>12.6G</td>
<td>22M</td>
<td>82.1</td>
</tr>
<tr>
<td>RegionViT-S [2]</td>
<td>5.3G</td>
<td>31M</td>
<td>82.5</td>
</tr>
<tr>
<td>DAT-T</td>
<td>4.6G</td>
<td>28M</td>
<td>82.0</td>
</tr>
<tr>
<td>DAT-T*</td>
<td>4.8G</td>
<td>30M</td>
<td>82.7</td>
</tr>
</tbody>
</table>

3. More Visualizations

We visualize examples of learned deformed locations in our DAT to verify the effectiveness of our method. As illustrated in Figure 1, the sampling points are depicted on the top of the object detection boxes and instance segmentation masks, from which we can see that the points are shifted to the target objects. In the left column, the deformed points are contracted to two target giraffes, while other points are keeping a nearly uniform grid with small offsets. In the middle column, the deformed points distribute densely among the person’s body and the surfing board both in the two stages. The right column shows the deformed points focus well to each of the six donuts, which shows our model has the ability to better model geometric shapes even with multiple targets. The above visualizations demonstrate that DAT learns meaningful offsets to sample better keys for attention to improve the performances on various vision tasks.

We also provide visualization results of the attention map given specific query tokens, and compare with Swin Transformer [8] in Figure 2. We show key tokens with the highest attention values. It can be observed that our model focus on the more relevant part. As a showcase, our model allocates most attention to foreground objects, e.g., both giraffes in the first row. On the other hand, the region of interests in Swin Transformer is comparably local and fail to distinguish foreground from background, which is depicted in the last surfboard.

4. Training Details of DAT

We use AdamW [9] optimizer to train our models for 300 epochs with a cosine learning rate decay. The basic learning rate for a batch size of 1024 is set to $1 \times 10^{-3}$. 
and then linearly scaled w.r.t. the batch size. To stabilize training procedures, we schedule a linear warm-up of learning rate from $1 \times 10^{-6}$ to the basic learning rate, and for a better convergence the cosine decay rule is applied to gradually decrease the learning rate to $1 \times 10^{-7}$ during training. We follow DeiT [11] to set the advanced data augmentation, including RandAugment [3], Mixup [17] and CutMix [16] to avoid overfitting. In addition, stochastic depth [6] and weight decay of 0.05 are also applied, in which the stochastic depth degree is chosen 0.2, 0.3 and 0.5 for the tiny, small and base model, respectively. We do not adopt EMA [10], random erasing [18] and the vanilla drop out, which does not improve the training of Vision Transformers, as verified in [8, 11]. In terms of larger resolution finetuning, we finetune our DAT-B using AdamW optimizer with a cosine scheduled learning rate $4 \times 10^{-6}$ for 30 epochs. We set the stochastic depth rate to 0.5 and lower the weight decay to $1 \times 10^{-8}$ to keep the regularization.

References

[16] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In ICCV, pages 6023–6032, 2019. 3