

# Less is More: Pay Less Attention in Vision Transformers

Zizheng Pan, Bohan Zhuang<sup>†</sup>, Haoyu He, Jing Liu, Jianfei Cai  
Dept of Data Science and AI, Monash University

## Abstract

Transformers [36] have become one of the dominant architectures in deep learning, particularly as a powerful alternative to convolutional neural networks (CNNs) in computer vision. However, Transformer training and inference in previous works can be prohibitively expensive due to the quadratic complexity of self-attention over a long sequence of representations, especially for high-resolution dense prediction tasks. To this end, we present a novel Less attention vIision Transformer (LIT), building upon the fact that convolutions, fully-connected (FC) layers, and self-attentions have almost equivalent mathematical expressions for processing image patch sequences. Specifically, we propose a hierarchical Transformer where we use pure multi-layer perceptrons (MLPs) to encode rich local patterns in the early stages while applying self-attention modules to capture longer dependencies in deeper layers. Moreover, we further propose a learned deformable token merging module to adaptively fuse informative patches in a non-uniform manner. The proposed LIT achieves promising performance on image recognition tasks, including image classification, dense detection and segmentation, serving as a strong backbone for many vision tasks. Code is available at: <https://github.com/MonashAI/LIT>

## 1 Introduction

Transformers have made substantial strides in natural language processing (NLP) (*e.g.*, [36, 7]) and recently in the computer vision (CV) field (*e.g.*, [8, 33]). Inspired by the pyramid design [19] in CNNs, recent hierarchical vision Transformers (HVTs) [38, 22, 39, 41] divide transformer blocks into several stages and progressively shrink feature maps as the network goes deeper. However, high-resolution feature maps in the early stages result in long token sequences, which brings huge computational cost and memory consumption due to the quadratic complexity of self-attention. For instance, a feature map of size  $56 \times 56 \times 96$  costs 2.0G FLOPs in one Multi-head Self-Attention (MSA) [36], while the entire model of ResNet-18 [11] only requires 1.8G FLOPs. Such a huge computing cost makes it difficult to apply Transformers into broad computer vision tasks.

Several emerging efforts have been made to reduce the computational cost in the early stages of HVTs. For example, some works [39, 35] reduce the number of self-attention heads in an MSA layer or further decreasing the number of Transformer blocks [8]. Another line of works proposes to trade-off accuracy and efficiency for MSA via heuristic approximation, such as spatial reduction attention (SRA) [38] and shifted window based multi-head self-attention (SW-MSA) [22]. There are also studies simply employ convolutional layers [31, 9] when the resolution of feature maps are considerably large. However, how much the early adoption of self-attention layers really contributes to the final performance remains unclear.

In this paper, we present a Less attention vIision Transformer (LIT) to address the aforementioned problem for HVTs. Specifically, we propose to exclusively use MLP layers to capture local patterns

<sup>†</sup>Corresponding author. E-mail: bohan.zhuang@monash.edu

in the early stages while introducing MSA layers with sufficient number of heads to handle long range dependencies in the later stages. The motivation comes from two aspects. First, previous studies in CNNs have shown that shallow layers focus on local patterns and deeper layers tend to produce more discriminative features [40, 14, 44]. Second, from the theoretical perspective, a self-attention layer with sufficient heads applied to images can express any convolutional layer [5], and the size of the receptive field for an MSA layer is positively correlated with the number of heads. Therefore, fewer heads in an MSA layer theoretically result in a smaller receptive field, where the extreme case is as expressive as a  $1 \times 1$  convolution that can be viewed as a standard FC layer applied to each pixel independently. To be emphasized, by exploiting MLP blocks in the early stages, the model avoids the huge computational cost and memory footprint arising from self-attention on high-resolution feature maps. Moreover, applying self-attention in the later stages to capture long range dependencies is quite efficient due to the progressive shrinking pyramid. Our comprehensive results show that such a simple architecture design brings a sweet spot between model performance and efficiency.

Furthermore, recent HVTs either adopt a standard convolutional layer or a linear projection layer to merge nearby tokens [38, 22], aiming to control the scale of feature maps. However, such methods hinder the representational power for vision Transformers to model geometric transformations, considering that not every pixel equally contributes to an output unit [24]. To this end, we propose a Deformable Token Merging (DTM) module, inspired by deformable convolutions [6, 47], where we learn a grid of offsets to adaptively augment the spatial sampling locations for merging neighboring patches from a sub-window in a feature map. In this way, we can obtain more informative downsampled tokens for subsequent processing.

Our contributions can be summarized as follows. First, we propose a simple HVT structure with pure MLP blocks in the early stages, making it tractable for Transformers to process high-resolution images, especially in dense prediction tasks. Second, we propose a deformable token merging module to adaptively merge more informative patches to deliver hierarchical representations, with enhanced transformation modeling capability. Finally, we conduct extensive experiments to show that the proposed LIT performs favorably against other state-of-the-art vision Transformers with similar or even reduced computational complexity and memory consumption.

## 2 Related Work

**Vision Transformers.** Vision Transformers are models which adopt the self-attention mechanism [36] into CV tasks. Recent works towards Vision Transformers either follow a hybrid architecture that combines convolution and self-attention [2, 31], or design a pure self-attention architecture without convolution [26, 15]. More recently, Dosovitskiy *et al.* [8] propose a Vision Transformer (ViT) which achieves promising results on ImageNet. Since then, a few subsequent works have been proposed to improve ViT from different aspects. For example, some works [43, 18] seek to bring locality into ViT as they find ViT failed to model the important local structures (*e.g.*, edges, lines). Another line of works aims to explore deeper architectures [46, 34] by stacking more Transformer blocks.

There is also a prevailing trend to introduce hierarchical representations into ViT [25, 42, 38, 22, 39, 13, 35]. To do so, these works divide Transformer blocks into several stages and downsample feature maps as the network goes deeper. However, high-resolution feature maps in the early stages inevitably result in high computational and memory costs due to the quadratic complexity of the self-attention module. Targeting at this problem, Wang *et al.* [38] propose to reduce the spatial dimensions of attention’s key and value matrices. Liu *et al.* [22] propose to limit self-attention in non-overlapped local windows. However, these replacements seek to shrink the global attention maps for efficiency. In this paper, we elaborately design the shallow layers with pure MLPs, that are powerful enough to encode local patterns. This neat architecture design keeps the capability for modelling global dependencies in the later stages while easing the prohibitively expensive complexity introduced by high-resolution feature maps, especially in the dense prediction tasks.

**Deformable Convolutions.** Deformable convolutions (DC) are initially proposed by Dai *et al.* [6] in object detection and semantic segmentation tasks. Different from the regular sampling grid of a standard convolution, DC adaptively augments the spatial sampling locations with learned offsets. One following work by Zhu *et al.* [47] improves DC with a modulation mechanism, which modulates the input feature amplitudes from different spatial locations. With the advantage on

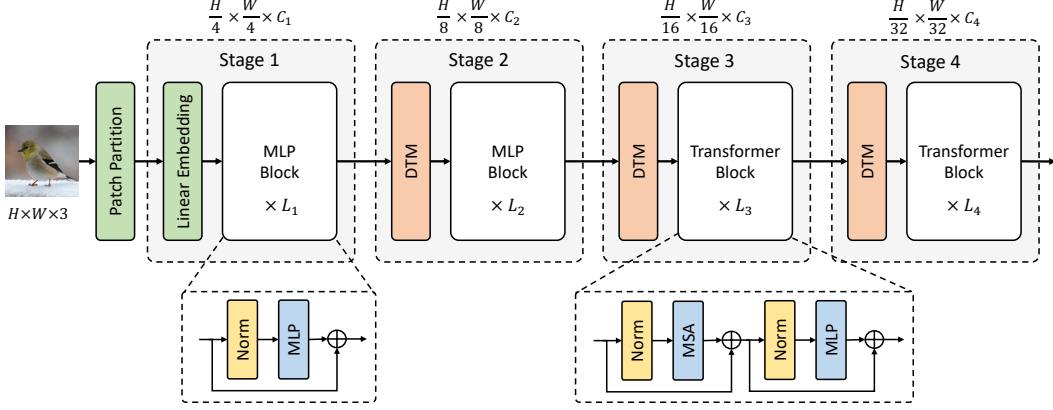


Figure 1: **Overall architecture of LIT.** The model is divided into four stages, where we apply MLP blocks in the first two stages and employ standard Transformer blocks in the last two stages. “DTM” denotes our proposed deformable token merging module.

modeling geometric transformations, many works adopt DC to target various CV problems. For example, Zhu *et al.* [48] propose a deformable attention module for object detection. Shim *et al.* [30] construct a similarity search and extraction network built upon DC layers for single image super-resolution. Thomas *et al.* [32] introduce a deformable kernel point convolution operator for point clouds. In this paper, we propose a deformable token merging module to adaptively merge more informative image tokens. Unlike previous works that merge tokens from a regular grid, DTM introduces better transformation modeling capability for HVTs.

### 3 Proposed Method

#### 3.1 Overall Architecture

The overall architecture of LIT is illustrated in Figure 1. Let  $\mathcal{I} \in \mathbb{R}^{H \times W \times 3}$  be an input RGB image, where  $H$  and  $W$  represent the height and width, respectively. We first split  $\mathcal{I}$  into non-overlapping patches with a patch size of  $4 \times 4$ , and thus the initial feature dimension of each patch is  $4 \times 4 \times 3 = 48$ . Next, a linear embedding layer is exploited to project each patch into dimension  $C_1$ , serving as the initial input for the following pipeline. The entire model is divided into 4 stages. Letting  $s \in [1, 2, 3, 4]$  be the index of a stage, we employ  $L_s$  blocks at each stage, where the first two stages solely utilise MLP blocks to encode local representations and the last two stages employ standard Transformer blocks [8] to handle longer dependencies. At each stage, we scale the input feature maps into  $\frac{H_{s-1}}{P_s} \times \frac{W_{s-1}}{P_s} \times C_s$ , where  $P_s$  and  $C_s$  represent the patch size and the hidden dimension at the  $s$ -th stage, respectively. For the last two stages, we set  $N_s$  self-attention heads in each Transformer block.

#### 3.2 Block Design in LIT

As shown in Figure 1, LIT employs two types of blocks: MLP blocks and Transformer blocks. In the early stages, we apply MLP blocks. Concretely, an MLP block is built upon an MLP which consists of two FC layers with GELU [12] non-linearity in between. For each MLP at the  $s$ -th stage, an expansion ratio of  $E_s$  is used. Specifically, the first FC layer expands the dimension of a token from  $C_s$  to  $E_s \times C_s$ , and the other FC layer reduces the dimension back to  $C_s$ . Formally, letting  $\mathbf{X} \in \mathbb{R}^{(\frac{H_{s-1}}{P_s} \times \frac{W_{s-1}}{P_s}) \times C_s}$  be the input of the  $s$ -th stage and  $l$  be the index of a block, an MLP block can be formulated as

$$\mathbf{X}_l = \mathbf{X}_{l-1} + \text{MLP}(\text{LN}(\mathbf{X}_{l-1})), \quad (1)$$

where LN indicates the layer normalization [1] and MLP denotes an MLP. In the last stages, a Transformer block as described in ViT [8] contains an MSA layer and an MLP, which can be

expressed as

$$\mathbf{X}'_{l-1} = \mathbf{X}_{l-1} + \text{MSA}(\text{LN}(\mathbf{X}_{l-1})), \quad (2)$$

$$\mathbf{X}_l = \mathbf{X}'_{l-1} + \text{MLP}(\text{LN}(\mathbf{X}'_{l-1})). \quad (3)$$

With this architecture, our model benefits from two main advantages: First, we avoid the huge computational costs and memory footprint that are introduced by long sequences in the early stages. Second, unlike recent works that shrink the attention maps using sub-windows [22] or reduce the spatial dimensions of the key and value matrices, we keep standard MSA layers in the last two stages so as to maintain the capability of LIT to handle long range dependencies while keeping mild FLOPs due to the pyramid structure. We will show in the ablation study that our simple architecture design outperforms the state-of-the-art hierarchical ViT variants on ImageNet with comparable FLOPs.

**Remark.** Here we justify the rationality of applying pure MLP blocks in the early stages by considering the relationship among a convolutional layer, an FC layer and an MSA layer. Firstly, we begin with a review of a standard convolutional layer. Let  $\mathbf{X} \in \mathbb{R}^{H \times W \times C_{in}}$  be the input feature map, and let  $\mathbf{W} \in \mathbb{R}^{K \times K \times C_{in} \times C_{out}}$  be the convolutional weight tensor, where  $K$  is the kernel size,  $C_{in}$  and  $C_{out}$  are the input and output channel dimensions, respectively. For simplicity, we omit the bias term and use  $\mathbf{X}_{p,:}$  to represent  $\mathbf{X}_{i,j,:}$ , where  $(i, j)$  denotes the pixel index and  $p \in [H] \times [W]$ . Given a convolutional kernel of  $K \times K$  sampling locations, the output for a pixel  $p$  can be formulated as

$$\text{Conv}(\mathbf{X})_{p,:} = \sum_{k \in [K \times K]} \mathbf{X}_{p+g(k),:} \mathbf{W}_{g(k),:,:}, \quad (4)$$

where  $g : [K \times K] \rightarrow \Delta_K$  is a bijective mapping of sampling indexes onto the pre-specified offsets  $\Delta_K$ . For example, let  $\Delta_K = \{(-1, -1), (-1, 0), \dots, (0, 1), (1, 1)\}$  be a  $3 \times 3$  kernel with dilation 1, then  $g(0) = (-1, -1)$  represents the first sampling offset.

When  $K = 1$ , the weight tensor  $\mathbf{W}$  is equivalent to a matrix, such that  $\mathbf{W} \in \mathbb{R}^{C_{in} \times C_{out}}$ . In this case, Eq. (4) can express an FC layer, and the output for a pixel  $p$  is defined by

$$\text{FC}(\mathbf{X})_{p,:} = \mathbf{X}_{p,:} \mathbf{W}_{:,,:}. \quad (5)$$

Last, let  $N_h$  be the number of heads in an MSA layer and  $\mathbf{W}^{(h)} \in \mathbb{R}^{C_{in} \times C_{out}}$  be learnable parameters of the  $h$ -th head. Under mild conditions, Cordonnier *et al.* [5] prove that the output from an MSA layer at pixel  $p$  is identical to

$$\text{MSA}(\mathbf{X})_p = \sum_{h \in [N_h]} \mathbf{X}_{p+f(h),:} \mathbf{W}^{(h)}, \quad (6)$$

where  $f : [N_h] \rightarrow \Delta_K$  is a bijective mapping of heads onto pixel shifts. In that case, Eq. (6) can be seen to be equivalent to a convolutional layer with a kernel size of  $\sqrt{N_h} \times \sqrt{N_h}$ . We refer detailed explanations to [5].

From Eqs. (4)-(6), we observe almost equivalent mathematical formulations among a convolutional layer, an FC layer and an MSA layer. In fact, the main difference lies in the size of their receptive fields. Specifically, the size of the receptive field in a convolutional layer is  $K \times K$ . For an FC layer,  $K$  is always 1. For an MSA layer,  $K$  is equivalent to  $\sqrt{N_h}$ . The observation tells us that when  $N_h$  is pretty small (*e.g.*, 1 or 2), an MSA layer theoretically has a limited receptive field that is approximately or equal to an FC layer. Considering most recent HVTs [38, 39, 41] adopt very few heads in the early stages, this justifies our method of applying pure MLP blocks in the first two stages without self-attention.

### 3.3 Deformable Token Merging

Previous works on HVTs [38, 22] rely on patch merging to achieve pyramid feature representations. However, they merge patches from a regular grid and neglect the fact that not every patch contributes equally to an output unit [24]. Inspired by deformable convolutions [6, 47], we propose a deformable token merging module to learn a grid of offsets to adaptively sample more informative patches. Formally, a deformable convolution is formulated as

$$\text{DC}(\mathbf{X})_{p,:} = \sum_{k \in [K \times K]} \mathbf{X}_{p+g(k)+\Delta g(k),:} \mathbf{W}_{g(k),:,:} \quad (7)$$

Table 1: Architecture specifications of LIT.  $P$  denotes the patch size and  $C$  is the channel dimension.  $L$  refers to the number of blocks applied at each stage.  $N$  is the number of self-attention heads. As the first two stages do not have self-attention layers, we only show the numbers of  $N_3$  and  $N_4$ . The expansion ratios of LIT-Ti at each stage are [8, 8, 4, 4]. We use an expansion ratio of 4 for all MLP layers in LIT-S, LIT-M and LIT-B. The numbers denoted under each stage index (e.g.,  $28 \times 28$ ) represents the size of the feature maps at that stage, based on the input resolution of  $224 \times 224$ .

Stage	Layer Name	LIT-Ti	LIT-S	LIT-M	LIT-B
Stage 1 $56 \times 56$	Patch Embedding	$P_1 = 4$ $C_1 = 64$	$P_1 = 4$ $C_1 = 96$	$P_1 = 4$ $C_1 = 96$	$P_1 = 4$ $C_1 = 128$
	MLP Block	$L_1 = 3$	$L_1 = 2$	$L_1 = 2$	$L_1 = 2$
Stage 2 $28 \times 28$	DTM	$P_2 = 2$ $C_2 = 128$	$P_2 = 2$ $C_2 = 192$	$P_2 = 2$ $C_2 = 192$	$P_2 = 2$ $C_2 = 256$
	MLP Block	$L_2 = 4$	$L_2 = 2$	$L_2 = 2$	$L_2 = 2$
Stage 3 $14 \times 14$	DTM	$P_3 = 2$ $C_3 = 320$	$P_3 = 2$ $C_3 = 384$	$P_3 = 2$ $C_3 = 384$	$P_3 = 2$ $C_3 = 512$
	Transformer Block	$N_3 = 5$	$N_3 = 12$	$N_3 = 12$	$N_3 = 16$
		$L_3 = 6$	$L_3 = 6$	$L_3 = 18$	$L_3 = 18$
Stage 4 $7 \times 7$	DTM	$P_4 = 2$ $C_4 = 512$	$P_4 = 2$ $C_4 = 768$	$P_4 = 2$ $C_4 = 768$	$P_4 = 2$ $C_4 = 1024$
	Transformer Block	$N_4 = 8$ $L_4 = 3$	$N_4 = 24$ $L_4 = 2$	$N_4 = 24$ $L_4 = 2$	$N_4 = 32$ $L_4 = 2$

Compared to a standard convolution operation as in Eq. (4), DC learns an offset  $\Delta g(k)$  for each pre-specified offset  $g(k)$ . Learning  $\Delta g(k)$  requires a separate convolutional layer, which is also applied over the input feature map  $\mathbf{X}$ . To merge patches in an adaptive manner, we adopt one DC layer in a DTM module, which can be formulated by

$$\text{DTM}(\mathbf{X}) = \text{GELU}(\text{BN}(\text{DC}(\mathbf{X}))), \quad (8)$$

where BN denotes the batch normalization [16] and we employ the GELU non-linearity. We will show in the ablation study that the sampling locations in DTM are adaptively adjusted when objects' scales and shapes change, benefiting from the learned offsets. Also note that our light-weight DTM introduces negligible FLOPs and parameters compared to regular grid sampling in baselines, thus making it a plug-and-play module for recent HVTs.

## 4 Experiments

### 4.1 ImageNet Classification

We conduct experiments on ImageNet (ILSVRC2012) [29] dataset. ImageNet is a large-scale dataset which has  $\sim 1.2\text{M}$  training images from 1K categories and 50K validation images. We compare with CNN-based ResNet [11] and Transformer-based models including DeiT [33], PVT [38] and Swin [22]. For simplicity, we denote them as "Model-Ti/S/M/B" to refer to their tiny, small, medium and base variants. Similarly, we define four variants of our LIT models: LIT-Ti, LIT-S, LIT-M and LIT-B. Detailed architecture settings are shown in Table 1. For better comparison, we design LIT-Ti as a counterpart to PVT-S, where both models adopt the absolute positional encoding. Our LIT-S, LIT-M and LIT-B use the relative positional encoding, and these three models can be seen as competitors to Swin-Ti, Swin-S, Swin-B, respectively.

**Implementation details.** In general, all models are trained on ImageNet with 300 epochs and a total batch size of 1024. For all ImageNet experiments, training images are resized to  $256 \times 256$ , and  $224 \times 224$  patches are randomly cropped from an image or its horizontal flip, with the per-pixel mean subtracted. We use the single-crop setting for testing. We use AdamW optimizer [23] with a cosine decay learning rate scheduler. The initial learning rate is  $1 \times 10^{-3}$ , and the weight decay is set to  $5 \times 10^{-2}$ . The initial values of learnable offsets in DTM are set to 0, and the initial learning rate for

Table 2: Comparisons with state-of-the-art methods on ImageNet. All models are trained and evaluated with the input resolution of  $224 \times 224$ .

Method	Params (M)	FLOPs (G)	Top-1 Acc. (%)
ResNet-18 [11]	12	1.8	68.5
ResNet-50	26	4.1	78.5
ResNet-101	45	7.9	79.8
DeiT-Ti [33]	5	1.3	72.2
DeiT-S	22	4.6	79.8
DeiT-B	86	17.5	81.8
PVT-Ti [38]	13	1.9	75.1
PVT-S	25	3.8	79.8
PVT-M	44	6.7	81.2
PVT-L	61	9.8	81.7
Swin-Ti [22]	29	4.5	81.3
Swin-S	50	8.7	83.0
Swin-B	88	15.4	83.3
LIT-Ti	19	3.6	81.1
LIT-S	27	4.1	81.5
LIT-M	48	8.6	83.0
LIT-B	86	15.0	83.4

offset parameters is set to  $1 \times 10^{-5}$ . For a fair comparison, we adopt the same training strategies as PVT and Swin when comparing our models to each of them.

Table 3: Effect of our architecture design principle. \* denotes the LIT model which adopts the same uniform patch merging strategies as in PVT-S or Swin-Ti.

Model	Params	FLOPs	Top-1 Acc. (%)
PVT-S [38]	25M	3.8G	79.8
<b>LIT-Ti*</b>	<b>19M</b>	<b>3.6G</b>	<b>80.4</b>
Swin-Ti [22]	28M	4.5G	81.3
<b>LIT-S*</b>	<b>27M</b>	<b>4.1G</b>	81.3

Table 4: Effect of the proposed deformable token merging module. We replace the uniform patch merging strategies in PVT-S and Swin-Ti with our DTM.

Model	Params	FLOPs	Top-1 Acc. (%)
PVT-S [38]	25M	3.8G	79.8
<b>PVT-S + DTM</b>	25M	3.8G	<b>80.5</b>
Swin-Ti [22]	28M	4.5G	81.3
<b>Swin-Ti + DTM</b>	28M	4.5G	<b>81.6</b>

**Results on ImageNet.** In Table 2, we compare LIT with several state-of-the-art methods on ImageNet. In general, all LIT models achieve on par or better performance than their counterparts while having fewer parameters and less FLOPs, without applying any existing efficient self-attention mechanisms. Specifically, our LIT-Ti outperforms PVT-S by 1.3% on the Top-1 accuracy while the FLOPs is reduced by 0.2G. LIT-S surpasses Swin-Ti by 0.2% with 0.4G less FLOPs. LIT-M achieves on par performance with Swin-S, whereas the FLOPs of LIT-M is reduced. LIT-B brings 0.1% Top-1 accuracy increase over Swin-B while using 0.4G less FLOPs. For ResNet and DeiT, LIT demonstrates better performance when compared to them with the same magnitude of FLOPs and parameters (*e.g.*, ResNet-50 v.s. LIT-S, DeiT-B v.s. LIT-B).

## 4.2 Ablation Studies

**Effect of the architecture design.** To explore the effect of our architecture design principle in LIT, we conduct experiments on ImageNet and compare the architecture of LIT with two recent HVTs: PVT-S [38] and Swin-Ti [22]. The results are shown in Table 3. In general, our architecture improves baseline PVT-S by 0.6% in Top-1 accuracy while using less FLOPs (3.6G v.s. 3.8G). For Swin-Ti, our method reduces the FLOPs by 0.4G while achieving on par performance. It is also worth noting



that the total number of parameters is reduced for both PVT-S and Swin-Ti. The overall performance demonstrates the effectiveness of the proposed architecture.

**Effect of deformable token merging.** To verify the effectiveness of our proposed DTM strategy, we replace the default patch merging scheme in PVT-S and Swin-Ti with DTM and train the models on ImageNet. The results are shown in Table 4. We observe that for both models, DTM introduces negligible FLOPs and parameters while improving PVT-S and Swin-Ti by 0.7% and 0.3% in terms of the Top-1 accuracy, respectively. Furthermore, we visualize the learned offsets in Figure 2. As it shows, unlike previous uniform patch merging strategy where sampled locations are limited within the green rectangle, our DTM adaptively merges patches according to objects’ scales and shapes (*e.g.*, koala leg, keyboard, cat tail). This again emphasizes the power of LIT to accommodate various geometric transformations.

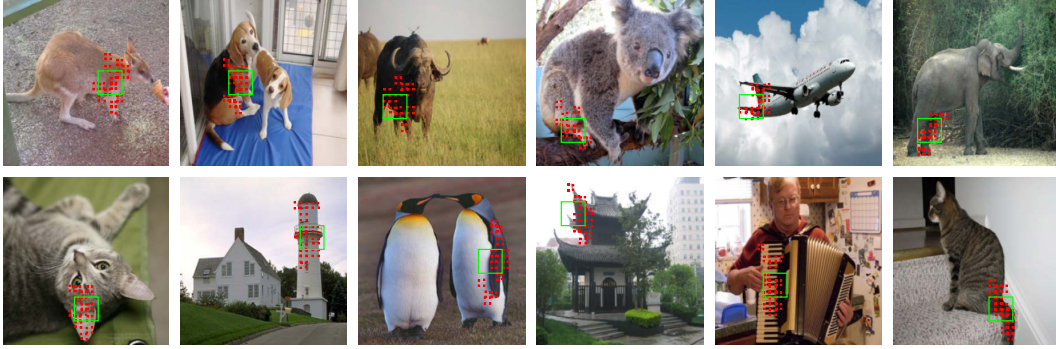


Figure 2: Visualization of learned offsets by the proposed deformable token merging modules. Each image shows  $4^3$  sampling locations (*red dots*) in three DTM modules with  $2 \times 2$  filter. The green rectangle outlines a  $32 \times 32$  patch of the original image, which also indicates a regular sampling field of previous methods. Best viewed in color.

**Effect of MSA in each stage.** To explore the effect of self-attention in recent HVTs, we train PVT-S on ImageNet and gradually remove self-attention layers at each stage. The results are presented in Table 5. First, after replacing the SRA layers in PVT-S with standard MSA layers, we observe 1.1% improvement on the Top-1 accuracy whereas the FLOPs is almost doubled. This indicates that PVT makes a trade-off between performance and efficiency. Next, by gradually removing MSA layers in the first two stages, the Top-1 accuracy only drops by 0.1%, 0.5%, respectively. It implies that the self-attention layers in the early stages of PVT contribute less than expected to the final performance, and they perform not much better than pure MLP layers. It can be attributed to the fact that shallow layers focus more on encoding local patterns. However, we observe a huge performance drop when removing self-attention layers in the last two stages. The results show that the self-attention layers play an important role in the later stages and capturing long range dependencies is essential for well-performed hierarchical vision Transformers.

Table 5: Impact of the MSA layers at each stage. Note that PVT-S has four stages, which adopts SRA at all blocks instead of MSA. Here we denote “w/ MSA” as PVT-S with standard MSA layers. “Removed Stage” refers to the stages where we remove all self-attention layers. For example, “1,2” means we remove the self-attention layers in the first two stages. Note that “0” means a model without removing any self-attention layers.

Model	Removed Stage	Params (M)	FLOPs (G)	Top-1 Acc. (%)
PVT-S [38]	0	25	3.8	79.8
PVT-S w/ MSA	0	20	8.4	80.9
PVT-S w/ MSA	1	20	4.5	80.8
PVT-S w/ MSA	1,2	19	3.6	80.4
PVT-S w/ MSA	1,2,3	17	3.0	75.0
PVT-S w/ MSA	1,2,3,4	14	2.8	66.8

To better understand the phenomenon, we visualize the attention probabilities for PVT-S without removing any MSA layers, which are depicted in Figure 3. First, the attention map at the first stage shows that the query pixel almost pays no attention to other locations. At the second stage, the receptive field of the query pixel is slightly enlarged, but similar to the first stage. Considering that PVT-S only has one head at the first stage and two heads at the second stage, this strongly supports our hypothesis that very few heads in an MSA layer result in a smaller receptive field, such that a self-attention layer is almost equivalent to an FC layer. Furthermore, we observe relatively larger receptive fields from the attention maps of the last two stages. As a large receptive field usually helps to model longer dependencies, this explains the huge performance drop in Table 5 after we remove the MSA layers in the last two stages.

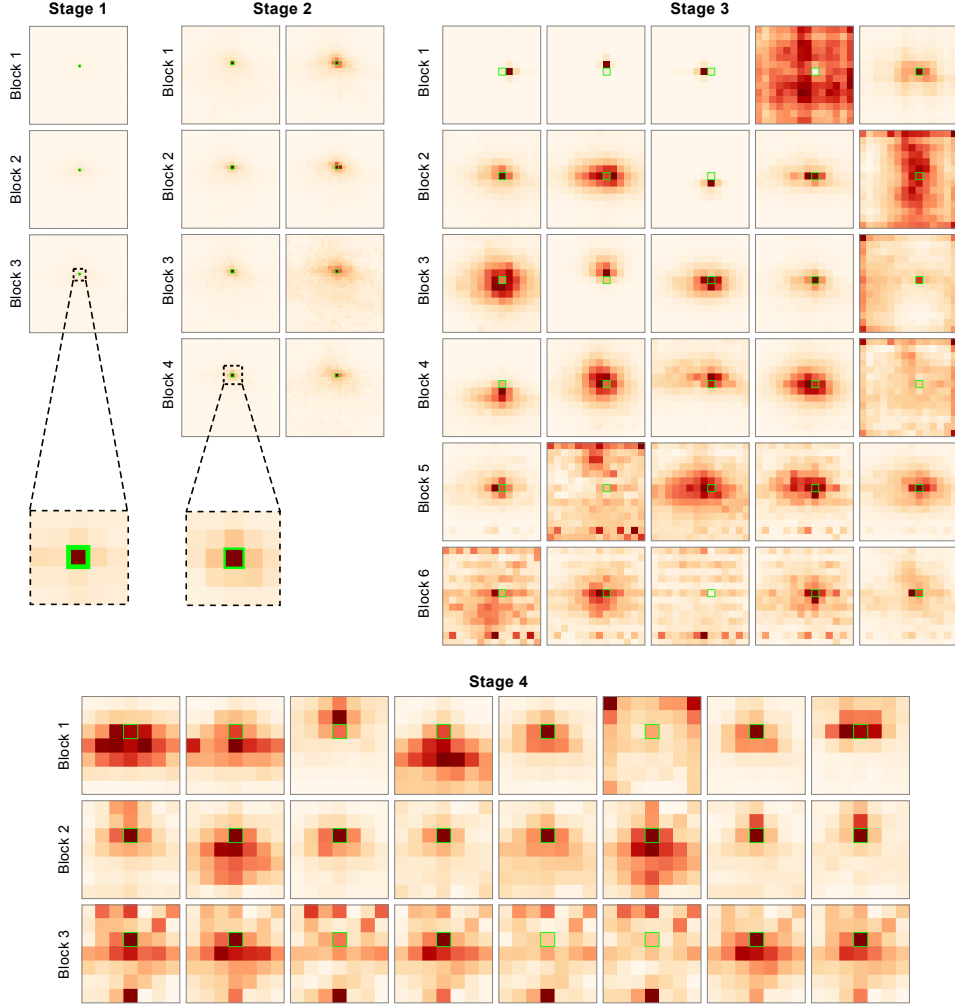


Figure 3: Attention probabilities of PVT-S with standard MSA layers. For each stage, we visualize the attention map of each head (*columns*) at each block (*rows*). All attention maps are averaged over 100 validation images. Each map shows the attention probabilities of a query pixel (*green rectangle*) to other pixels. Darker color indicates higher attention probability and vice versa. Best viewed in color.

### 4.3 Object Detection and Instance Segmentation on COCO

In this section, we conduct experiments on COCO 2017 [21] dataset to show the performance of LIT on object detection and instance segmentation. COCO is a large-scale dataset which contains  $\sim 118K$  images for the training set and  $\sim 5K$  images for the validation set. For a fair comparison, we evaluate LIT-Ti and LIT-S on two base detectors: RetinaNet [20] and Mask R-CNN [10]. For the experiments with both detectors, we consider CNN-based ResNet [11] and Transformer-based models including



Table 6: Object detection performance on the COCO val2017 split using the RetinaNet framework.

Backbone	RetinaNet						
	Params (M)	AP	AP <sub>50</sub>	AP <sub>75</sub>	AP <sub>S</sub>	AP <sub>M</sub>	AP <sub>L</sub>
ResNet-50 [11]	38	36.3	55.3	38.6	19.3	40.0	48.8
PVT-S [38]	34	40.4	61.3	43.0	25.0	42.9	55.7
LIT-Ti	<b>30</b>	<b>41.6</b>	<b>62.8</b>	<b>44.7</b>	<b>25.7</b>	<b>44.4</b>	<b>56.4</b>
ResNet-101 [11]	57	38.5	57.8	41.2	21.4	42.6	51.1
Swin-Ti [22]	39	41.5	62.1	<b>44.2</b>	25.1	<b>44.9</b>	55.5
LIT-S	39	<b>41.6</b>	<b>62.7</b>	44.1	<b>25.6</b>	44.7	<b>56.5</b>

Table 7: Object detection and instance segmentation performance on the COCO val2017 split using the Mask R-CNN framework. AP<sup>b</sup> and AP<sup>m</sup> denote the bounding box AP and mask AP, respectively.

Backbone	Mask R-CNN						
	Params (M)	AP <sup>b</sup>	AP <sup>b</sup> <sub>50</sub>	AP <sup>b</sup> <sub>75</sub>	AP <sup>m</sup>	AP <sup>m</sup> <sub>50</sub>	AP <sup>m</sup> <sub>75</sub>
ResNet-50 [11]	44	38.0	58.6	41.4	34.4	55.1	36.7
PVT-S [38]	44	40.4	62.9	43.8	37.8	60.1	40.3
LIT-Ti	<b>40</b>	<b>42.0</b>	<b>64.9</b>	<b>45.6</b>	<b>39.1</b>	<b>61.9</b>	<b>41.9</b>
ResNet-101 [11]	63	40.4	61.1	44.2	36.4	57.7	38.8
Swin-Ti [22]	48	42.2	64.6	46.2	39.1	61.6	42.0
LIT-S	48	<b>42.9</b>	<b>65.6</b>	<b>46.9</b>	<b>39.6</b>	<b>62.3</b>	<b>42.4</b>

PVT-S [38] and Swin-Ti [22]. Following common practice [2, 38], we measure the performance of all models by Average Precision (AP) in COCO.

**Implementation details.** All models are trained on 8 V100 GPUs, with  $1 \times$  schedule (12 epochs) and a total batch size of 16. We use AdamW [23] optimizer with a step decay learning rate scheduler. Following PVT [38], the initial learning rates are set to  $1 \times 10^{-4}$  and  $2 \times 10^{-4}$  for RetinaNet and Mask R-CNN, respectively. The weight decay is set to  $1 \times 10^{-4}$  for all models. Results of Swin-Ti based detectors are adopted from Chu *et al.* [4]. At the training stage, we initialize the backbone with the pretrained weights on ImageNet. The training images are resized to the shorter size of 800 pixels, and the longer size is at most 1333 pixels. During inference, we fix the shorter side of an image to 800 pixels.

**Results on COCO.** Table 6 shows the comparisons of different backbones on object detection based on RetinaNet. By comparing LIT with PVT and ResNet counterparts, we find that our model outperforms both backbones on object detection in almost all metrics. Similar results can be found in Table 7, where LIT again surpasses compared methods on object detection and instance segmentation using the Mask R-CNN framework. The consistent improvement demonstrates the effectiveness of LIT for dense prediction tasks.

#### 4.4 Semantic Segmentation on ADE20K

We conduct experiments on ADE20K [45] to show the performance of LIT models on semantic segmentation. ADE20K is a widely adopted dataset for semantic segmentation, which has  $\sim 20K$  training images,  $\sim 2K$  validation images and  $\sim 3K$  test images. For a fair comparison, we evaluate LIT models with Semantic FPN [17]. Following the common practice in [38, 22], we measure the model performance by mIoU.

**Implementation details.** All models are trained on 8 V100 GPUs, with 8K steps and a total batch size of 16. The AdamW [23] optimizer is adopted with an initial learning rate of  $1 \times 10^{-4}$ . Learning rate is decayed by the polynomial decay schedule with the power of 0.9. We set the weight decay to  $1 \times 10^{-4}$ . All backbones are initialized with the pretrained weights on ImageNet. At the training stage, we randomly resize and crop the images to  $512 \times 512$ . During inference, images are scaled to the short size of 512.

**Results on ADE20K.** We compare different backbones on the ADE20K validation set in Table 8. From the results, we observe that LIT-Ti outperforms ResNet-50 and PVT-S by 4.6% and 1.5% mIoU, respectively. For LIT-S, our model again surpasses ResNet-101 and Swin-Ti, with 2.9% and 0.2%

improvement on mIoU, respectively. The overall performance demonstrates the effectiveness of the proposed LIT models on semantic segmentation.

Table 8: Semantic segmentation performance with different backbones on the ADE20K validation set.

Backbone	Semantic FPN	
	Params (M)	mIoU (%)
ResNet-50 [11]	29	36.7
PVT-S [38]	28	39.8
LIT-Ti	24	<b>41.3</b>
ResNet-101 [11]	48	38.8
Swin-Ti [22]	32	41.5
LIT-S	32	<b>41.7</b>

## 5 Conclusion and Future Work

In this paper, we have introduced LIT, a hierarchical vision transformer which pays less attention in the early stages to ease the huge computational cost of self-attention modules over high-resolution representations. Specifically, LIT applies MLP blocks in the first two stages to focus on local patterns while employing standard Transformer blocks with sufficient heads in the later stages to handle long range dependencies. Moreover, we have proposed a deformable token merging module, which is learned to adaptively merge informative patches to an output unit, with enhanced geometric transformations. Extensive experiments on ImageNet and COCO have demonstrated that LIT achieves better performance compared with existing state-of-the-art HVT methods. Future works may include finding better architectural configurations of LIT with neural architecture search (NAS) and improving MLP blocks in the early stages to enhance the capability of LIT to encode local patterns. Besides, one may also consider applying efficient self-attention mechanisms in the later stages to achieve better efficiency, such as kernelization [27], low-rank decomposition [37], memory [28], and sparsity [3] schemes, etc. Also note that LIT potentially brings some negative societal impacts, such as the unintended use for surveillance and the huge energy consumption and carbon emissions of training LIT models on ImageNet.

## References

- [1] Jimmy Ba, J. Kiros, and Geoffrey E. Hinton. Layer normalization. *ArXiv*, abs/1607.06450, 2016.
- [2] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, pages 213–229, 2020.
- [3] Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. Generating long sequences with sparse transformers. *arXiv preprint arXiv:1904.10509*, 2019.
- [4] Xiangxiang Chu, Zhi Tian, Yuqing Wang, Bo Zhang, Haibing Ren, Xiaolin Wei, Huaxia Xia, and Chunhua Shen. Twins: Revisiting the design of spatial attention in vision transformers. *Arxiv preprint 2104.13840*, 2021.
- [5] Jean-Baptiste Cordonnier, Andreas Loukas, and Martin Jaggi. On the relationship between self-attention and convolutional layers. In *ICLR*, 2020.
- [6] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *ICCV*, pages 764–773, 2017.
- [7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *ACL*, pages 4171–4186, 2019.
- [8] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *ICLR*, 2021.

- [9] Benjamin Graham, Alaaeldin El-Nouby, Hugo Touvron, Pierre Stock, Armand Joulin, Hervé Jégou, and Matthijs Douze. Levit: a vision transformer in convnet’s clothing for faster inference. *arXiv preprint arXiv:22104.01136*, 2021.
- [10] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross B. Girshick. Mask R-CNN. In *ICCV*, pages 2980–2988, 2017.
- [11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016.
- [12] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv: Learning*, 2016.
- [13] Byeongho Heo, Sangdoo Yun, Dongyoon Han, Sanghyuk Chun, Junsuk Choe, and Seong Joon Oh. Rethinking spatial dimensions of vision transformers. *arXiv preprint arXiv:2103.16302*, 2021.
- [14] Qibin Hou, Ming-Ming Cheng, Xiaowei Hu, Ali Borji, Zhuowen Tu, and Philip H. S. Torr. Deeply supervised salient object detection with short connections. *TPAMI*, 41(4):815–828, 2019.
- [15] Han Hu, Zheng Zhang, Zhenda Xie, and Stephen Lin. Local relation networks for image recognition. In *ICCV*, pages 3463–3472, 2019.
- [16] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, pages 448–456, 2015.
- [17] Alexander Kirillov, Ross B. Girshick, Kaiming He, and Piotr Dollár. Panoptic feature pyramid networks. In *CVPR*, pages 6399–6408, 2019.
- [18] Yawei Li, Kai Zhang, Jie Zhang Cao, Radu Timofte, and Luc Van Gool. Localvit: Bringing locality to vision transformers. *arXiv preprint arXiv:2104.05707*, 2021.
- [19] Tsung-Yi Lin, Piotr Dollár, Ross B. Girshick, Kaiming He, Bharath Hariharan, and Serge J. Belongie. Feature pyramid networks for object detection. In *CVPR*, pages 936–944, 2017.
- [20] Tsung-Yi Lin, Priya Goyal, Ross B. Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *ICCV*, pages 2999–3007, 2017.
- [21] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. In David J. Fleet, Tomás Pajdla, Bernt Schiele, and Tinne Tuytelaars, editors, *ECCV*, pages 740–755, 2014.
- [22] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. *arXiv preprint arXiv:2103.14030*, 2021.
- [23] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019.
- [24] Wenjie Luo, Yujia Li, Raquel Urtasun, and Richard S. Zemel. Understanding the effective receptive field in deep convolutional neural networks. In *NeurIPS*, pages 4898–4906, 2016.
- [25] Zizheng Pan, Bohan Zhuang, Jing Liu, Haoyu He, and Jianfei Cai. Scalable visual transformers with hierarchical pooling. *arXiv preprint arXiv:2103.10619*, 2021.
- [26] Niki Parmar, Prajit Ramachandran, Ashish Vaswani, Irwan Bello, Anselm Levskaya, and Jon Shlens. Stand-alone self-attention in vision models. In *NeurIPS*, pages 68–80, 2019.
- [27] Hao Peng, Nikolaos Pappas, Dani Yogatama, Roy Schwartz, Noah A Smith, and Lingpeng Kong. Random feature attention. In *ICLR*, 2021.
- [28] Jack W Rae, Anna Potapenko, Siddhant M Jayakumar, and Timothy P Lillicrap. Compressive transformers for long-range sequence modelling. In *ICLR*, 2020.
- [29] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *IJCV*, pages 211–252, 2015.
- [30] Gyumin Shim, Jinsun Park, and In So Kweon. Robust reference-based super-resolution with similarity-aware deformable convolution. In *CVPR*, pages 8422–8431, 2020.
- [31] Aravind Srinivas, Tsung-Yi Lin, Niki Parmar, Jonathon Shlens, Pieter Abbeel, and Ashish Vaswani. Bottleneck transformers for visual recognition. *arXiv preprint arXiv:2101.11605*, 2021.
- [32] Hugues Thomas, Charles R. Qi, Jean-Emmanuel Deschaud, Beatriz Marcotegui, François Goulette, and Leonidas J. Guibas. Kpconv: Flexible and deformable convolution for point clouds. In *ICCV*, pages 6410–6419, 2019.
- [33] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. *arXiv preprint arXiv:2012.12877*, 2020.

- [34] Hugo Touvron, Matthieu Cord, Alexandre Sablayrolles, Gabriel Synnaeve, and Hervé Jégou. Going deeper with image transformers. *arXiv preprint arXiv:2103.17239*, 2021.
- [35] Ashish Vaswani, Prajit Ramachandran, Aravind Srinivas, Niki Parmar, Blake Hechtman, and Jonathon Shlens. Scaling local self-attention for parameter efficient visual backbones. In *CVPR*, 2021.
- [36] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, pages 5998–6008, 2017.
- [37] Sinong Wang, Belinda Li, Madian Khabza, Han Fang, and Hao Ma. Linformer: Self-attention with linear complexity. *arXiv preprint arXiv:2006.04768*, 2020.
- [38] Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. *arXiv preprint arXiv:2102.12122*, 2021.
- [39] Haiping Wu, Bin Xiao, Noel Codella, Mengchen Liu, Xiyang Dai, Lu Yuan, and Lei Zhang. Cvt: Introducing convolutions to vision transformers. *arXiv preprint arXiv:2103.15808*, 2021.
- [40] Zhe Wu, Li Su, and Qingming Huang. Cascaded partial decoder for fast and accurate salient object detection. In *CVPR*, pages 3907–3916, 2019.
- [41] Haotian Yan, Zhe Li, Weijian Li, Changhu Wang, Ming Wu, and Chuang Zhang. Contnet: Why not use convolution and transformer at the same time? *arXiv preprint arXiv:2104.13497*, 2021.
- [42] Kun Yuan, Shaopeng Guo, Ziwei Liu, Aojun Zhou, Fengwei Yu, and Wei Wu. Incorporating convolution designs into visual transformers. *arXiv preprint arXiv:2103.11816*, 2021.
- [43] Li Yuan, Yunpeng Chen, Tao Wang, Weihao Yu, Yujun Shi, Francis EH Tay, Jiashi Feng, and Shuicheng Yan. Tokens-to-token vit: Training vision transformers from scratch on imagenet. *arXiv preprint arXiv:2101.11986*, 2021.
- [44] Matthew D. Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *ECCV*, pages 818–833, 2014.
- [45] Bolei Zhou, Hang Zhao, Xavier Puig, Tete Xiao, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Semantic understanding of scenes through the ADE20K dataset. *IJCV*, pages 302–321, 2019.
- [46] Daquan Zhou, Bingyi Kang, Xiaojie Jin, Linjie Yang, Xiaochen Lian, Qibin Hou, and Jiashi Feng. Deepvit: Towards deeper vision transformer. *arXiv preprint arXiv:2103.11886*, 2021.
- [47] Xizhou Zhu, Han Hu, Stephen Lin, and Jifeng Dai. Deformable convnets V2: more deformable, better results. In *CVPR*, pages 9308–9316, 2019.
- [48] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. In *ICLR*, 2021.