Vision Transformer with Progressive Sampling

Xiaoyu Yue*[1] Shuyang Sun*[2] Zhanghui Kuang[3] Meng Wei[4]
Philip Torr[2] Wayne Zhang[3,6] Dahua Lin[1,5]
[1]Centre for Perceptual and Interactive Intelligence [2]University of Oxford
[3]SenseTime Research [4]Tsinghua University [5]The Chinese University of Hong Kong
[6]Qing Yuan Research Institute, Shanghai Jiao Tong University
xyyue@cpii.hk {kevinsun, phst}@robots.ox.ac.uk weim18@mails.tsinghua.edu.cn
{kuangzhanghui, wayne.zhang}@sensetime.com dhlin@ie.cuhk.edu.hk

Abstract

Transformers with powerful global relation modeling abilities have been introduced to fundamental computer vision tasks recently. As a typical example, the Vision Transformer (ViT) directly applies a pure transformer architecture on image classification, by simply splitting images into tokens with a fixed length, and employing transformers to learn relations between these tokens. However, such naive tokenization could destruct object structures, assign grids to uninterested regions such as background, and introduce interference signals. To mitigate the above issues, in this paper, we propose an iterative and progressive sampling strategy to locate discriminative regions. At each iteration, embeddings of the current sampling step are fed into a transformer encoder layer, and a group of sampling offsets is predicted to update the sampling locations for the next step. The progressive sampling is differentiable. When combined with the Vision Transformer, the obtained PS-ViT network can adaptively learn where to look. The proposed PS-ViT is both effective and efficient. When trained from scratch on ImageNet, PS-ViT performs 3.8% higher than the vanilla ViT in terms of top-1 accuracy with about $4\times$ *fewer parameters and* $10\times$ *fewer FLOPs. Code is available at https://github.com/yuexy/PS-ViT.*

1. Introduction

Transformers [39, 11] have become the de-facto standard architecture for natural language processing tasks. Thanks to their powerful global relation modeling abilities, researchers attempt to introduce them to fundamental computer vision tasks such as image classification [6, 38, 12, 44, 32], object detection [56, 4, 53, 10, 37] and image segmentation [40] recently. However, transformers are initially tailored for processing mid-size sequences, and of quadratic computational complexity w.r.t. the sequence length. Thus, they cannot directly be used to process images with massive pixels.

*Equal Contribution.

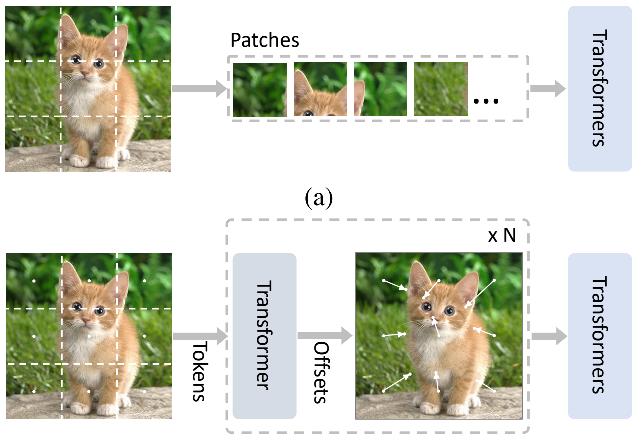

Figure 1. Comparison between the naive tokenization scheme in ViT [12] and the progressive sampling in our proposed PS-ViT. (a) The naive tokenization scheme generates a sequence of image patches which are embedded and then fed into a stack of transformers. (b) Our PS-ViT iteratively samples discriminative locations. $\times N$ indicates N sampling iterations.

To overcome the computational complexity issue, the pioneer Vision Transformer (ViT) [12] adopts a naive tokenization scheme that partitions one image into a sequence of regularly spaced patches, which are linearly projected into tokens. In this way, the image is converted into hundreds of visual tokens, which are fed into a stack of transformer encoder layers for classification. ViT attains excellent results, especially when pre-trained on large-scale datasets, which proves that full-transformer architecture is

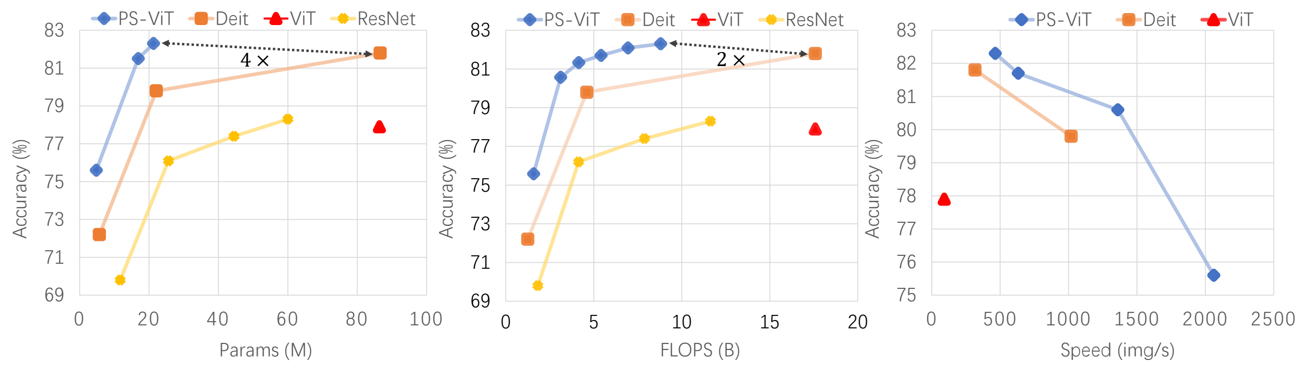

Figure 2. Comparisons between PS-ViT with state-of-the-art networks in terms of top-1 accuracy on ImageNet, parameter number, FLOPs, and speed. The chart on the left, middle and right show top-1 accuracy *vs*. parameter numbers, FLOPs and speed respectively. The speed is tested on the same V100 with a batch size of 128 for fair comparison.

a promising alternative for vision tasks. However, the limitations of such a naive tokenization scheme are obvious. First, the hard splitting might separate some highly semantically correlated regions that should be modeled with the same group of parameters, which destructs inherent object structures and makes the input patches to be less informative. Figure 1 (a) shows that the cat head is divided into several parts, resulting in recognition challenges based on one part only. Second, tokens are placed on regular grids irrespective of the underlying image content. Figure 1 (a) shows that most grids focus on the uninterested background, which might lead to the interesting foreground object is submerged in interference signals.

The human vision system organizes visual information in a completely different way than indiscriminately processing a whole scene at once. Instead, it progressively and selectively focuses attention on interesting parts of the visual space when and where it is needed while ignoring uninterested parts, combining information from different fixations over time to understand the scene [33].

Inspired by the procedure above, we propose a novel transformer-based progressive sampling module, which is able to learn where to look in images, to mitigate the issues caused by the naive tokenization scheme in ViT [12]. Instead of sampling from fixed locations, our proposed module updates the sampling locations in an iterative manner. As shown in Figure 1 (b), at each iteration, tokens of the current sampling step are fed into a transformer encoder layer, and a group of sampling offsets is predicted to update the sampling locations for the next step. This mechanism utilizes the capabilities of the transformer to capture global information to estimate offsets towards regions of interest, by combining with the local contexts and the positions of current tokens. In this way, attention progressively converges to discriminative regions of images step by step as what human vision does. Our proposed progressive sampling is differentiable, and readily plugged into ViT instead of the hard splitting, to construct end-to-end Vision Transforms with Progressive Sampling Networks dubbed as PS-ViT. Thanks to task-driven training, PS-ViT tends to sample object regions correlated with semantic structures. Moreover, it pays more attention to foreground objects while less to ambiguous background compared with simple tokenization.

The proposed PS-ViT outperforms the current state-of-the-art transformer-based approaches when trained from scratch on ImageNet. Concretely, it achieves 82.3% top-1 accuracy on ImageNet which is higher than that of the recent ViT's variant DeiT [38] with only about $4\times$ fewer parameters and $2\times$ fewer FLOPs. As shown in Figure 2, we observe that PS-ViT is remarkably better, faster, and more parameter-efficient compared to state-of-the-art transformer-based networks ViT and DeiT.

2. Related Work

Transformers are first proposed for sequence models such as machine translation [39]. Benefiting from their powerful global relation modeling abilities, and highly efficient training, transformers achieve significant improvements and become the de-facto standard in many Natural Language Processing (NLP) tasks [11, 3, 30, 29, 48].

Transformers in Computer Vision. Inspired by the success of transformers in NLP tasks, many researchers attempt to apply transformers, or attention mechanism in computer vision tasks, such as image classification [6, 38, 12, 44, 32, 2, 18, 36], object detection [56, 4, 53, 10, 50, 37], image segmentation [40], low-level image processing [5, 47, 27], video understanding [42] generation [43], multi-modality understanding [7, 35, 22], and Optical Character Recognition (OCR) [41, 49, 34]. Transformers' powerful modelling capacity comes at the cost of computational complexity. Their consumed memory and computation grow quadratically w.r.t. the token length, which prevents them to being directly applied to images with massive pixels as tokens.

Axial attention [17] applied attention along a single axis of the tensor without flattening to reduce the computational resource requirement. iGPT [6] simply down-sampled images to one low resolution, trained a sequence of transformers to auto-regressively predict pixels and achieved promising performance with a linear probe. ViT [12] regularly partitioned one high-resolution image into 16 $\times$ 16 patches, which were fed into one pure transformer architecture for classification, and attained excellent results even compared to state-of-the-art convolutional networks for the first time. However, ViT needs pretraining on large-scale datasets, thereby limiting their adoption. DeiT [38] proposed a data-efficient training strategy and a teacher-student distillation mechanism [16], and improved ViT's performance greatly. Moreover, it is trained on ImageNet only, and thus considerably simplifies the overall pipeline of ViT. Our proposed PS-ViT also starts from ViT. Instead of splitting pixels into a small number of visual tokens, we propose a novel progressive sampling strategy to avoid structure destruction and focus more attention on interesting regions.

Hard Visual Attention. PS-ViT as a series of glimpses akin to hard visual attention [25, 1, 46, 13], makes decisions based on a subset of locations only in the input image. However, PS-ViT is differentiable and can be easily trained in an end-to-end fashion while previous hard visual attention approaches are non-differentiable and trained with Reinforcement Learning (RL) methods. These RL-based methods have proven to be less effective when scaled onto more complex datasets [13]. Moreover, our PS-ViT targets at progressively sampling discriminative tokens for Vision Transformers while previous approaches locate interested regions for convolutional neural networks [25, 1, 13] or sequence decoders [46]. Our work is also related to the deformable convolution [9, 54] and deformable attention [55] mechanism, however, the motivation and the way of pixel sampling in this work are different from what proposed in the deformable convolution and attention mechanism.

3. Methodology

In this section, we first introduce our progressive sampling strategy and then describe the overall architecture of our proposed PS-ViT network. Finally, we will elaborate on the details of PS-ViT. Symbols and notations of our method are presented in Table 1.

3.1. Progressive Sampling

ViT [12] regularly partitions one image into 16 $\times$ 16 patches, which are linearly projected into a set of tokens, regardless of the content importance of image regions and the integral structure of objects. To pay more attention to interesting regions of images and mitigate the problem of structure destruction, we propose one novel progressive sampling module. As it is differentiable, it is adaptively driven via the following vision transformer based image classification task.

Name	Description
$\mathbf{F} \in \mathbb{R}^{C\times H\times W}$	the feature map extracted by the feature extractor module
$\mathbf{p}_t \in \mathbb{R}^{2\times(n\times n)}$	the sampling points at the iteration t
$\mathbf{P}_t \in \mathbb{R}^{C\times(n\times n)}$	the position embeddings at the iteration t
$\mathbf{o}_t \in \mathbb{R}^{2\times(n\times n)}$	the sampling offsets at the iteration t
$\mathbf{T}_t^{'} \in \mathbb{R}^{C\times(n\times n)}$	tokens sampled from $\mathbf{F}$ at the iteration t
$\mathbf{T}_t \in \mathbb{R}^{C\times(n\times n)}$	tokens predicted by the progressive sampling module at the iteration t

Table 1. The list of symbols and notations used in this paper.

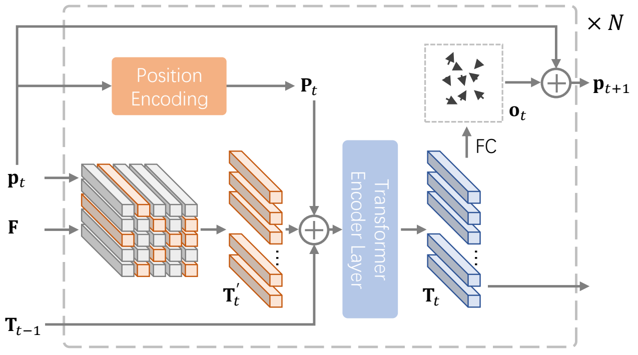

Figure 3. The architecture of the progressive sampling module. At each iteration, given the sampling location $\mathbf{p}_t$ and the feature map $\mathbf{F}$, we sample the initial tokens $\mathbf{T}_t^{'}$ at $\mathbf{p}_t$ over $\mathbf{F}$, which are element-wisely added with the positional encodings $\mathbf{P}_t$ generated based on $\mathbf{p}_t$, and the output tokens $\mathbf{T}_{t-1}$ of the last iteration, and then fed into one transformation encoder layer to predict the tokens $\mathbf{T}_t$ of the current iteration. The offset matrix $\mathbf{o}_t$ is predicted via one fully-connected layer based on $\mathbf{T}_t$, which is added with $\mathbf{p}_t$ to obtain the sampling positions $\mathbf{p}_{t+1}$ for the next iteration. The above procedure is iterated N times.

Our progressive sampling module is an iterative framework. Given the input feature map $\mathbf{F} \in \mathbb{R}^{C\times H\times W}$ with C, H, and W being the feature channel dimension, height and width respectively, it finally outputs a sequence of tokens $\mathbf{T}_N \in \mathbb{R}^{C\times(n\times n)}$, where $(n \times n)$ indicates the number of samples over one image and N is the total iterative number in the progressive sampling module.

As shown in Figure 3, at each iteration, the sampling locations are updated via adding them with the offset vectors of the last iteration. Formally,

$$\mathbf{p}_{t+1} = \mathbf{p}_t + \mathbf{o}_t,\ t \in \{1, \ldots, N-1\}, \tag{1}$$

where $\mathbf{p}_t \in \mathbb{R}^{2\times(n\times n)}$, and $\mathbf{o}_t \in \mathbb{R}^{2\times(n\times n)}$ indicate the sampling location matrix and the offset matrix predicted at the iteration t. For the first iteration, we initialize the $\mathbf{p}_1$ to be the regularly-spaced locations as done in ViT [12]. Concretely, the i-th location $\mathbf{p}_1^i$ is given by

$$\begin{aligned} \mathbf{p}_1^i &= [\pi_i^y s_h + s_h/2, \pi_i^x s_w + s_w/2] \\ \pi_i^y &= \lfloor i/n \rfloor \\ \pi_i^x &= i - \pi_i^y * n \\ s_h &= H/n \\ s_w &= W/n, \end{aligned} \tag{2}$$

where π_i^y and π_i^x map the location index i to the row index and the column one respectively. $\lfloor\cdot\rfloor$ indicates the floor operation. s_h and s_w are the step size in the y and x axial direction respectively. Initial tokens are then sampled over the input feature map at the sampled locations as follows:

$$\mathbf{T}_t^{'} = \mathbf{F}(\mathbf{p}_t),\ t \in \{1, \ldots, N\}, \tag{3}$$

where $\mathbf{T}_t^{'} \in \mathbb{R}^{C\times(n\times n)}$ is the initial sampled tokens at the iteration t. As elements of $\mathbf{p}_t$ are fractional, the sampling is implemented via the bilinear interpolation operation, which is differentiable w.r.t. both the input feature map $\mathbf{F}$ and the sampling locations $\mathbf{p}_t$. The initial sampled tokens, the output tokens of the last iteration, and the positional encodings of the current sampling locations are further element-wisely added before being fed into one transformer encoder layer to get the output tokens of the current iteration. Formally, we have

$$\begin{aligned} \mathbf{P}_t &= \mathbf{W}_t \mathbf{p}_t \\ \mathbf{X}_t &= \mathbf{T}_t^{'} \oplus \mathbf{P}_t \oplus \mathbf{T}_{t-1} \\ \mathbf{T}_t &= \text{Transformer}(\mathbf{X}_t),\ t \in \{1, \ldots, N\}, \end{aligned} \tag{4}$$

where $\mathbf{W}_t \in \mathbb{R}^{C\times 2}$ is the linear transformation that projects the sampled locations $\mathbf{p}_t$ to the positional encoding matrix $\mathbf{P}_t$ of size $C \times (n \times n)$, all iterations share the same $\mathbf{W}_t$. $\oplus$ indicates the element-wise addition while Transformer$(\cdot)$ is the mulit-head self-attention based transformer encoder layer, which will be elaborated in Section 3.3. Note that $\mathbf{T}_0$ is a zero matrix in Equation (4). ViT [12] uses the 2-D sinusoidal positional embeddings of patch indices. Since their patches are regularly spaced, patch indices can exactly encode the relative coordinates of patch centers in one image. However, it does not hold true in our case as our sampled locations are non-isometric as shown in Figure 1. We project the normalized absolute coordinates of sampled locations to one embedding space as the positional embeddings instead. Finally, the sampling location offsets are predicted for the next iteration except at the last iteration as follows:

$$\mathbf{o}_t = \mathbf{M}_t \mathbf{T}_t,\ t \in \{1, \ldots, N-1\}, \tag{5}$$

where $\mathbf{M}_t \in \mathbb{R}^{2\times C}$ is the learnable linear transformation for predicting the sampling offset matrix.

With the progressive sampling strategy, the sampled locations progressively converge to interesting regions of images. Therefore, we name it by progressive sampling.

3.2. Overall Architecture

As shown in Figure 4, the architecture of the PS-ViT consists of four main components: 1) a feature extractor module to predict dense tokens; 2) a progressive sampling module to sample discriminative locations; 3) a vision transformer module that follows the similar configuration of ViT [12] and DeiT [38]; 4) a classification module.

The feature extractor module aims at extracting the dense feature map $\mathbf{F}$, where the progressive sampling module can simple tokens $\mathbf{T}_t$. Each pixel of the dense feature map $\mathbf{F}$ can be treated as a token associated with a patch of the image. We employ the convolutional stem and the first two residual blocks in the first stage of the ResNet50 [14] as our feature extractor module because the convolution operator is especially effective at modeling spatially local contexts.

The vision transformer module follows the architecture adopted in ViT [12] and DeiT [38]. We pad an extra token named by the classification token $\mathbf{T}_{cls} \in \mathbb{R}^{C\times 1}$ to the output tokens $\mathbf{T}_N$ of the last iteration in the progressive sampling module, and fed them into the vision transformer module. Formally,

$$\overline{\mathbf{T}} = \text{VTM}([\mathbf{T}_{cls}, \mathbf{T}_N]), \tag{6}$$

where VTM indicates the vision transformer module function which is a stack of transformer encoder layers, and $\overline{\mathbf{T}} \in \mathbb{R}^{C\times(n\times n+1)}$ is the output. Note that, the positional information has been fused into $\mathbf{T}_N$ in the progressive sampling module, so we do not need to add positional embedding here. The classification token refined through the vision transformer module is finally used to predict the image classes. We use the cross entropy loss to end-to-end train the proposed PS-ViT network.

3.3. Implementation

Transformer Encoder Layer. The transformer encoder layer serves as the basic building block for the progressive sampling module and the vision transformer module. Each transformer encoder layer has a *multi-head self-attention* and a *feed-forward unit*.

Given queries $\mathbf{Q} \in \mathbb{R}^{D\times L}$, keys $\mathbf{K} \in \mathbb{R}^{D\times L}$ and values $\mathbf{V} \in \mathbb{R}^{D\times L}$ with D being the dimension and L being the sequence length, the scaled dot-product self attention can be calculated as:

$$\text{Attn}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}(\mathbf{Q}^T\mathbf{K}/\sqrt{D})\mathbf{V}^T, \tag{7}$$

where $\mathbf{Q}^T$ indicates the transpose of $\mathbf{Q}$, and softmax$(\cdot)$ is the softmax operation applied over each row of the input

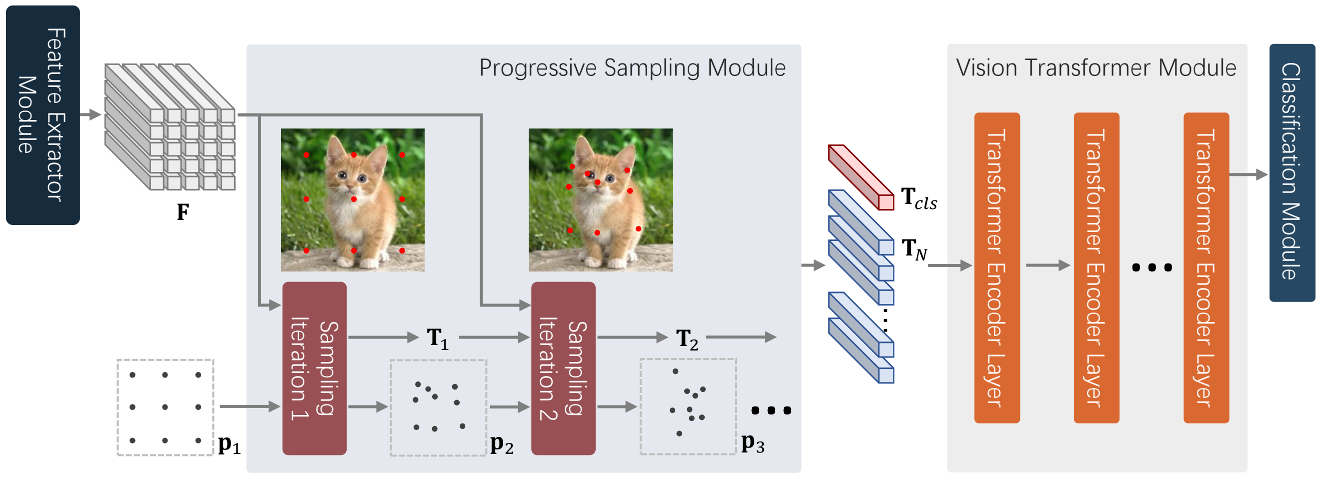

Figure 4. Overall architecture of the proposed Progressive Sampling Vision Transformer (PS-ViT). Given an input image, its feature map $\mathbf{F}$ is first extracted by the feature extractor module. Tokens $\mathbf{T}_i$ are then sampled progressively and iteratively at adaptive locations $\mathbf{p}_i$ over $\mathbf{F}$ in the progressive sampling module. The final output tokens $\mathbf{T}_N$ of the progressive sampling module are padded with the classification token $\mathbf{T}_{cls}$ and further fed into the vision transformer module to refine $\mathbf{T}_{cls}$, which is finally classified in the classification module.

Networks	N	N_v	C	M	#params	#FLOPs
PS-ViT-Ti	4	8	192	3	4.7 M	1.6 B
PS-ViT-Ti†	4	8	192	3	3.6 M	1.6 B
PS-ViT-B	4	10	384	6	21.3 M	5.4 B
PS-ViT-B†	4	10	384	6	16.9 M	5.4 B

Table 2. PS-ViT configurations. † indicates weight sharing between different iterations in the Progressive Sampling Module (PSM). N, N_v, C and M are the iteration number in PSM, the transformer encoder layer number in the vision transformer module, the dimension of tokens, and the head number in each transformer respectively.

matrix. For Multi-Head self-Attention (MHA), the queries, keys and values are generated via linear transformations on the inputs for M times with one individual learned weight for each head. Then attention function is applied in parallel on queries, keys and values of each head. Formally,

$$\begin{aligned} \text{MHA}(\mathbf{Z}) &= \mathbf{W}^o[\mathbf{H}_1, \dots, \mathbf{H}_M]^T, \\ \mathbf{H}_i &= \text{Attn}(\mathbf{W}_i^Q\mathbf{Z}, \mathbf{W}_i^K\mathbf{Z}, \mathbf{W}_i^V\mathbf{Z}), \end{aligned} \tag{8}$$

where $\mathbf{W}^o \in \mathbb{R}^{\frac{C}{M}\times C}$ is a learnable linear projection. $\mathbf{W}_i^Q \in \mathbb{R}^{\frac{C}{M}\times C}$, $\mathbf{W}_i^K \in \mathbb{R}^{\frac{C}{M}\times C}$ and $\mathbf{W}_i^V \in \mathbb{R}^{\frac{C}{M}\times C}$ are the linear projections for the queries, keys and values of the i-th head respectively.

The feed-forward unit of the transformer encoder layer consists of two fully connected layers with one GELU non-linear activation [15] between them and the latent variable dimension being $3C$. For simplicity, the transformer encoder layers in both the progressive sampling module and the vision transformer module keep the same settings.

Progressive Sampling Back-propagation. The back-propagation of the progressive sampling is straightforward. According to Equation (1) and Equation (3), for each sampling location i, the gradient w.r.t. the sampling offsets $\mathbf{o}_t^i$ at the iteration t is computed as:

$$\begin{aligned} \frac{\partial \mathbf{T}'^i_t}{\partial \mathbf{o}^i_{t-1}} &= \frac{\partial \mathbf{F}(\mathbf{p}^i_{t-1} + \mathbf{o}^i_{t-1})}{\partial \mathbf{o}^i_{t-1}} \\ &= \sum_{\mathbf{q}} \frac{\partial K(\mathbf{q}, \mathbf{p}^i_{t-1} + \mathbf{o}^i_{t-1})}{\partial \mathbf{o}^i_{t-1}} \mathbf{F}(\mathbf{q}), \end{aligned} \tag{9}$$

where $K(\cdot, \cdot)$ is the kernel for bilinear interpolation to calculate weights for each integral spatial location $\mathbf{q}$.

Network Configuration. The feature dimension C, the iteration number N in the progressive sampling module, the vision transformer layer number N_v in the vision transformer module, and the head number M in each transformer layer affect the model size, FLOPs, and performances. In this paper, we configure them with different speed-performance tradeoffs in Table 2 so that the proposed PS-ViT can be used in different application scenarios. The number of sampling points along each spatial dimension n is set as 14 by default.

Considering the sampling in each iteration is conducted over the same feature map $\mathbf{F}$ in the progressive sampling module, we try to share weights between those iterations to further reduce the number of trainable parameters. As shown in Table 2, about 25% parameters can be saved in this setting.

Model	Image size	Params (M)	FLOPs (B)	Top-1 (%)	Top-5 (%)
CNN-based					
R-18 [14]	224^2	11.7	1.8	69.8	89.1
R-50 [14]	224^2	25.6	4.1	76.1	92.9
R-101 [14]	224^2	44.5	7.9	77.4	93.5
X-50-32×4d [45]	224^2	25.0	4.3	79.3	94.5
X-101-32×4d [45]	224^2	44.2	8.0	80.3	95.1
RegNetY-4GF [31]	224^2	20.6	4.0	79.4	-
RegNetY-6.4GF [31]	224^2	30.6	6.4	79.9	-
RegNetY-16GF [31]	224^2	83.6	15.9	80.4	-
Transformer-based					
ViT-B/16 [12]	384^2	86.4	55.5	77.9	-
DeiT-Ti [38]	224^2	5.7	1.3	72.2	-
DeiT-S [38]	224^2	22.1	4.6	79.8	-
DeiT-B [38]	224^2	86.4	17.6	81.8	-
PS-ViT-Ti/14	224^2	4.8	1.6	75.6	92.9
PS-ViT-B/10	224^2	21.3	3.1	80.6	95.2
PS-ViT-B/14	224^2	21.3	5.4	81.7	95.8
PS-ViT-B/18	224^2	21.3	8.8	82.3	96.1

Table 3. Comparison with state-of-the-art networks on ImageNet with single center crop testing. The number after "/" is the sampling number in each axial direction. *e.g.*, PS-ViT-Ti/14 indicates PS-ViT-Ti with 14×14 sampling locations.

4. Experiments

Epochs	300
Optimizer	AdamW
Batch size	512
Learning rate	0.0005
Learning rate decay	cosine
Weight decay	0.05
Warmup epochs	5
Label smooth	0.1
Dropout	0.1
Rand Augment	(9, 0.5)
Mixup probability	0.8
CutMix probability	1.0

Table 4. Training strategy and hyper-parameter settings.

4.1. Experimental Details on ImageNet

All the experiments for image classification are conducted on the ImageNet 2012 dataset [21] that includes 1k classes, 1.2 million images for training, and 50 thousand images for validation. We train our proposed PS-ViT on ImageNet without pretraining on large-scale datasets. We train all the models of PS-ViT using PyTorch [28] with 8 GPUs. Inspired by the data-efficient training as done in [38], we use the AdamW [24] as the optimizer. The total training epoch number and the batch size are set to 300 and 512 respectively. The learning rate is initialized with 0.0005, and decays with the cosine annealing schedule [23]. We regularize the loss via the smoothing label with $\epsilon = 0.1$. We use random crop, Rand-Augment [8], Mixup[52], and CutMix [51] to augment images during training. Images are resized to 256×256, and cropped at the center with 224×224 size when testing. Training strategy and its hyper-parameter settings are summarized in Table 4.

4.2. Results on ImageNet

We compare our proposed PS-ViT with state-of-the-art networks on the standard image classification benchmark ImageNet in terms of parameter numbers, FLOPS, and top-1 and top-5 accuracies in Table 3.

Comparison with CNN based networks. Our PS-ViTs considerably outperform ResNets [14] while with much fewer parameters and FLOPs. Specifically, Compared with ResNet-18, PS-ViT-Ti/14 absolutely improves the top-1 accuracy by 5.8% while reducing 6.9 M parameters and 0.2 B FLOPs. We can observe a similar trend when comparing PS-ViT-B/10 (PS-ViT-B/14) and ResNet-50 (ResNet-101). Our proposed PS-ViT achieves superior performance and computational efficiency when compared with the state-of-the-art CNN based network RegNet [31]. Particularly, when compared with RegNetY-16GF, PS-ViT-B/18 improves the top-1 accuracy by 1.9% with about a quarter of parameters and a half of FLOPS.

Comparison with transformer based networks. Table 3 shows that our proposed PS-ViT outperforms ViT [12] and its recent variant DeiT [38]. In particular, PS-ViT-B/18 achieves 82.3% top-1 accuracy which is 0.5% higher than the baseline model DeiT-B while with 21 M parameters and 8.8 B FLOPs only. Our performance gain attributes to two parts. First, PS-ViT samples CNN-based tokens which is more efficient than raw image patches used in ViT [12] and DeiT [38]. Second, our progressive sampling module can adaptively focus on regions of interest and produce more semantically correlated tokens than the naive tokenization used in [12, 38].

4.3. Ablation Studies

The PS-ViT models predict on the class token in all the ablation studies.

A larger sampling number n leads to better performance. We first evaluate how the sampling number parameter n affects the PS-ViT performance. The sequence length of sampled tokens which is fed into the vision transformer module is n^2. The more the sampled tokens, the more information PS-ViT can extract. However, sampling more tokens would increase the computation and memory usage. Table 5 reports the FLOPs, and top-1 and top-5 ac-

n	Params (M)	FLOPs (B)	Top-1 (%)	Top-5 (%)
10	21.3	3.1	80.6	95.2
12		4.2	81.3	95.5
14		5.4	81.7	95.8
16		7.0	82.1	95.8
18		8.8	82.3	96.1

Table 5. Effect of the sampling number n in each axial direction.

Model	N	Top-1 (%)	Top-5 (%)
PS-ViT-B/14	1	80.6	95.3
	2	81.5	95.6
	4	81.7	95.8
	6	81.8	95.7
	8	81.9	95.7
	9	81.7	95.8
	10	81.6	95.8

Table 6. Effect of the iteration number N in the progressive sampling module.

	Top-1 (%)	Top-5 (%)
ViT$*$	78.4	94.1
PS-ViT-B/14	81.7	95.8

Table 7. Comparison between our PS-ViT with ViT. $*$ means the model with the same model configuration and training strategy.

Model	Params (M)	Top-1 (%)	Top-5 (%)
PS-ViT-Ti/14	4.8	75.6	92.9
PS-ViT-Ti†/14	3.7	74.1	92.3
PS-ViT-B/10	21.3	80.6	95.2
PS-ViT-B†/10	16.9	80.0	94.8
PS-ViT-B/12	21.3	81.3	95.5
PS-ViT-B†/12	16.9	80.9	95.3
PS-ViT-B/14	21.3	81.7	95.8
PS-ViT-B†/14	16.9	81.5	95.6

Table 8. Comparison PS-ViT with and without weight sharing in the progressive sampling module. $\dagger$ indicates weight sharing.

curacies with different n. It has been shown that the FlOPs increases as n becomes larger, and the accuracy increases when $n \leq 16$ and plateaus when $n > 16$. Considering the speed-accuracy trade-off, we set $n = 14$ by default except as otherwise noted.

The performance can be further improved with more iterations of progressive sampling. We then evaluate the effect of the iteration number N of the progressive sampling module in Table 6. To keep the computational complexity unchanged, all models in Table 6 have $14 - N$ transformer layers in the vision transformer module, and totally 14 transformer layers in the entire network. $N = 1$ indicates the sampling points will not be updated. It has been shown that PS-ViT performs the best when $N = 8$ and the accuracy begins to decline when $N > 8$. As we keep the total number of transformer layers unchanged, increasing N will result in the decrease of transformer layers in lateral modeling, which might damage the performance. Considering the accuracy improvement is negligible from $N = 4$ to $N = 8$, we set $N = 4$ by default except as otherwise noted.

Fair comparison with ViT. The network hyper-parameters in the transformer encoder of PS-ViT are different from the original setting of ViT. For a fair comparison, we further study how ViT performs when the network hyper-parameters are set to be the same as ours. We set the number of layers, channels, heads, and the number of tokens to be the same as what was proposed in PS-ViT-B/14, and train the network under the same training regime. As shown in Table 7, ViT achieves 78.4% top-1 accuracy, which is greatly inferior to its PS-ViT counterpart. We thereby conclude that the progressive sampling module can fairly boost the performance of ViT.

Sharing weights between sampling iterations. Model size (parameter number) is one of the key factors when deploying deep models on terminal devices. Our proposed PS-ViT is very terminal device friendly as it can share weights in the progressive sampling module with a negligible performance drop. Table 8 compares PS-ViT with and without weight sharing in the progressive sampling module. It has been shown that weight sharing can reduce the parameter number by about 21%$\sim$23% while with a slight performance drop, especially for PS-ViT-B/12 and PS-ViT-B/14.

4.4. Speed Comparison

Our proposed PS-ViT is efficient not only in theory but also in practice. Table 9 compare the efficiency of state-of-the-arts networks in terms of FLOPs and speed (images per second). For fair comparison, we measure the speed of all of the models on a server with one 32GB V100 GPU. The batch size is fixed to 128 and the number of images that can be inferred per second is reported averaged over 50 runs. It has been shown that PS-ViT is much more efficient than ViT and DeiT when their top-1 accuracies are comparable. Specifically, PS-ViT-B/14 and DeiT-B have similar accuracy around 81.7%. However, PS-ViT-B/14 achieves about 2.4 times and 3.3 times as fast as DeiT-B in terms of speed and FLOPs respectively. PS-ViT-B/10 speeds up ViT-B/16 by about 14.6 times and 17.9 times in terms of speed and FLOPs while improving 2.7% top-1 accuracy.

4.5. Visualization

In order to explore the mechanism of the learnable sampling locations in our method, we visualize the predicted

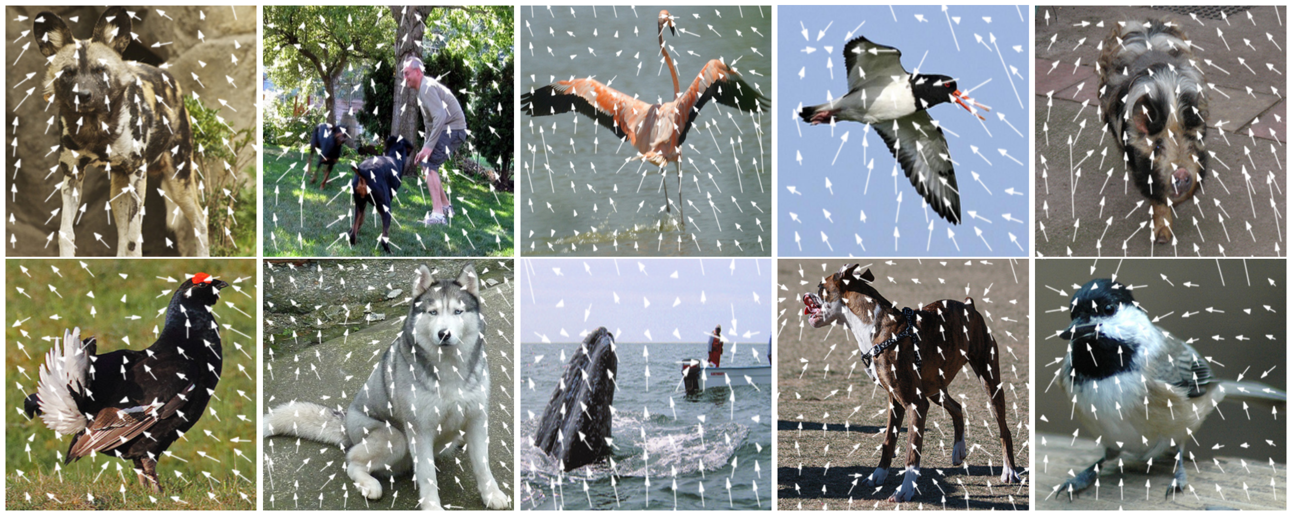
Figure 5. Visualization of sampled locations in the proposed progressive sampling module. The start points of arrows are initial sampled locations ($\mathbf{p}_1$) while the end points of arrows are the final sampled locations ($\mathbf{p}_4$).

Model	FLOPs (B)	Speed (img/s)	Top-1
RegNetY-4.0GF	4.0	1097.6	79.4
RegNetY-6.4GF	6.4	487.0	79.9
RegNetY-16GF	15.9	351.0	80.4
ViT-B/16	55.5	92.4	77.9
DeiT-S	4.6	1018.2	79.8
DeiT-B	17.6	316.1	81.8
PS-ViT-Ti/14	1.6	1955.3	75.6
PS-ViT-B/10	3.1	1348.0	80.6
PS-ViT-B/14	5.4	765.6	81.7
PS-ViT-B/18	8.8	463.8	82.3

Table 9. Comparison the efficiency of PS-ViT, and that of state-of-the-art networks in terms of FLOPs and speed.

offsets of our proposed progressive sampling module in Figure 5. We can observe that the sampling locations are adaptively adjusted according to the content of the images. Sampling points around objects tend to move to the foreground area and converge to the key parts of objects. With this mechanism, discriminative regions such as the chicken head are sampled densely, retaining the intrinsic structure information of highly semantically correlated regions.

4.6. Transfer Learning

In addition to ImageNet, we also transfer PS-ViT to downstream tasks to demonstrate its generalization ability. We follow the practice done in DeiT [38] for fair comparison. Table 10 shows results for models that have been pretrained on ImageNet and finetuned for other datasets including CIFAR-10 [20], CIFAR-100 [20], Flowers-102 [26] and Stanford Cars [19]. PS-ViT-B/14 can perform on-par-with or even better than DeiT-B with about $4\times$ fewer FLOPS and parameters on all these datasets.

Model	IM	C10	C100	Flowers	Cars
ViT-B/16	77.9	98.1	87.1	89.5	-
ViT-L/16	76.5	97.9	86.4	89.7	-
DeiT-B	81.8	99.1	90.8	98.4	92.1
PS-ViT-B/14	81.7	99.0	90.8	98.8	92.9

Table 10. Top-1 accuracy on other datasets. ImageNet and CIFAR are abbreviated to "IM" and "C".

5. Conclusions

In this paper, we propose an efficient Vision Transformers with Progressive Sampling (PS-ViT). PS-ViT first extracts feature maps via a feature extractor, and then progressively selects discriminative tokens with one progressive sampling module. The sampled tokens are fed into a vision transformer module and the classification module for image classification. PS-ViT mitigates the structure destruction issue in the ViT and adaptively focuses on interesting regions of objects. It achieves considerable improvement on ImageNet compared with ViT and its recent variant DeiT. We also provide a deeper analysis of the experimental results to investigate the effectiveness of each component. Moreover, PS-ViT is more efficient than its transformer based competitors both in theory and in practice.

Acknowledgement. This work was supported by the Centre for Perceptual and Interactive Intelligence (CPII) and the Innovation and Technology Commission of the Hong Kong Special Administrative Region, China (Enterprise Support Scheme under the Innovation and Technology Fund B/E030/18).

References

[1] Jimmy Ba, Volodymyr Mnih, and Koray Kavukcuoglu. Multiple object recognition with visual attention. *ICLR*, 2015. 3

[2] Irwan Bello, Barret Zoph, Ashish Vaswani, Jonathon Shlens, and Quoc V Le. Attention augmented convolutional networks. In *ICCV*, 2019. 2

[3] Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *arXiv:2005.14165*, 2020. 2

[4] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, 2020. 1, 2

[5] Hanting Chen, Yunhe Wang, Tianyu Guo, Chang Xu, Yiping Deng, Zhenhua Liu, Siwei Ma, Chunjing Xu, Chao Xu, and Wen Gao. Pre-trained image processing transformer. In *CVPR*, 2021. 2

[6] Mark Chen, Alec Radford, Rewon Child, Jeffrey Wu, Heewoo Jun, David Luan, and Ilya Sutskever. Generative pretraining from pixels. In *ICML*, 2020. 1, 2, 3

[7] Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. Uniter: Universal image-text representation learning. In *ECCV*, 2020. 2

[8] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *CVPRW*, 2020. 6

[9] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 764–773, 2017. 3

[10] Zhigang Dai, Bolun Cai, Yugeng Lin, and Junying Chen. Up-detr: Unsupervised pre-training for object detection with transformers. In *CVPR*, 2021. 1, 2

[11] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of Deep Bidirectional Transformers for Language Understanding. *NAACL*, 2019. 1, 2

[12] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021. 1, 2, 3, 4, 6

[13] Gamaleldin F Elsayed, Simon Kornblith, and Quoc V Le. Saccader: improving accuracy of hard attention models for vision. *arXiv:1908.07644*, 2019. 3

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 4, 6

[15] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv:1606.08415*, 2016. 5

[16] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv:1503.02531*, 2015. 3

[17] Jonathan Ho, Nal Kalchbrenner, Dirk Weissenborn, and Tim Salimans. Axial attention in multidimensional transformers. *arXiv:1912.12180*, 2019. 3

[18] Han Hu, Zheng Zhang, Zhenda Xie, and Stephen Lin. Local relation networks for image recognition. In *ICCV*, 2019. 2

[19] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *Proceedings of the IEEE international conference on computer vision workshops*, pages 554–561, 2013. 8

[20] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 8

[21] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *NeurIPS*, 2012. 6

[22] Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. Visualbert: A simple and performant baseline for vision and language. *arXiv:1908.03557*, 2019. 2

[23] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv:1608.03983*, 2016. 6

[24] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv:1711.05101*, 2017. 6

[25] Volodymyr Mnih, Nicolas Heess, Alex Graves, and Koray Kavukcuoglu. Recurrent models of visual attention. *arXiv:1406.6247*, 2014. 3

[26] Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *2008 Sixth Indian Conference on Computer Vision, Graphics & Image Processing*, pages 722–729. IEEE, 2008. 8

[27] Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Lukasz Kaiser, Noam Shazeer, Alexander Ku, and Dustin Tran. Image transformer. In *ICML*, 2018. 2

[28] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017. 6

[29] Matthew E Peters, Mark Neumann, Robert L Logan IV, Roy Schwartz, Vidur Joshi, Sameer Singh, and Noah A Smith. Knowledge enhanced contextual word representations. *arXiv preprint arXiv:1909.04164*, 2019. 2

[30] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. 2018. 2

[31] Ilija Radosavovic, Raj Prateek Kosaraju, Ross Girshick, Kaiming He, and Piotr Dollár. Designing network design spaces. In *CVPR*, 2020. 6

[32] Prajit Ramachandran, Niki Parmar, Ashish Vaswani, Irwan Bello, Anselm Levskaya, and Jonathon Shlens. Stand-alone self-attention in vision models. In *NeurIPS*, 2019. 1, 2

[33] Ronald A Rensink. The dynamic representation of scenes. *Visual cognition*, 2000. 2

[34] Fenfen Sheng, Zhineng Chen, and Bo Xu. Nrtr: A no-recurrence sequence-to-sequence model for scene text recognition. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*, 2019. 2

[35] Chen Sun, Austin Myers, Carl Vondrick, Kevin Murphy, and Cordelia Schmid. Videobert: A joint model for video and language representation learning. In *ICCV*, 2019. 2

[36] Shuyang Sun, Jiangmiao Pang, Jianping Shi, Shuai Yi, and Wanli Ouyang. Fishnet: A versatile backbone for image, region, and pixel level prediction. *arXiv preprint arXiv:1901.03495*, 2019. 2

[37] Zhiqing Sun, Shengcao Cao, Yiming Yang, and Kris Kitani. Rethinking transformer-based set prediction for object detection. *arXiv:2011.10881*, 2020. 1, 2

[38] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. *arXiv:2012.12877*, 2020. 1, 2, 3, 4, 6, 8

[39] 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*, 2017. 1, 2

[40] Huiyu Wang, Yukun Zhu, Bradley Green, Hartwig Adam, Alan Yuille, and Liang-Chieh Chen. Axial-deeplab: Stand-alone axial-attention for panoptic segmentation. In *ECCV*, 2020. 1, 2

[41] Peng Wang, Lu Yang, Hui Li, Yuyan Deng, Chunhua Shen, and Yanning Zhang. A simple and robust convolutional-attention network for irregular text recognition. *arXiv preprint arXiv:1904.01375*, 2019. 2

[42] Yuqing Wang, Zhaoliang Xu, Xinlong Wang, Chunhua Shen, Baoshan Cheng, Hao Shen, and Huaxia Xia. End-to-end video instance segmentation with transformers. *arXiv:2011.14503*, 2020. 2

[43] Dirk Weissenborn, Oscar Täckström, and Jakob Uszkoreit. Scaling Autoregressive Video Models. In *ICLR*, 2020. 2

[44] Bichen Wu, Chenfeng Xu, Xiaoliang Dai, Alvin Wan, Peizhao Zhang, Masayoshi Tomizuka, Kurt Keutzer, and Peter Vajda. Visual transformers: Token-based image representation and processing for computer vision. *arXiv:2006.03677*, 2020. 1, 2

[45] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *CVPR*, 2017. 6

[46] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *ICML*, 2015. 3

[47] Fuzhi Yang, Huan Yang, Jianlong Fu, Hongtao Lu, and Baining Guo. Learning texture transformer network for image super-resolution. In *CVPR*, 2020. 2

[48] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V Le. Xlnet: Generalized autoregressive pretraining for language understanding. *arXiv:1906.08237*, 2019. 2

[49] Xiaoyu Yue, Zhanghui Kuang, Chenhao Lin, Hongbin Sun, and Wayne Zhang. Robustscanner: Dynamically enhancing positional clues for robust text recognition. In *European Conference on Computer Vision*, 2020. 2

[50] Xiaoyu Yue, Zhanghui Kuang, Zhaoyang Zhang, Zhenfang Chen, Pan He, Yu Qiao, and Wei Zhang. Boosting up scene text detectors with guided cnn. *arXiv preprint arXiv:1805.04132*, 2018. 2

[51] 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*, 2019. 6

[52] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *ICLR*, 2018. 6

[53] Minghang Zheng, Peng Gao, Xiaogang Wang, Hongsheng Li, and Hao Dong. End-to-end object detection with adaptive clustering transformer. *arXiv:2011.09315*, 2020. 1, 2

[54] Xizhou Zhu, Han Hu, Stephen Lin, and Jifeng Dai. Deformable convnets v2: More deformable, better results. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9308–9316, 2019. 3

[55] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. *arXiv preprint arXiv:2010.04159*, 2020. 3

[56] 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. 1, 2