Pale Transformer: A General Vision Transformer Backbone with Pale-Shaped Attention

Sitong Wu[1, 2] Tianyi Wu[1, 2], Haoru Tan[3], Guodong Guo[1, 2] *

[1]Institute of Deep Learning, Baidu Research, Beijing, China
[2]National Engineering Laboratory for Deep Learning Technology and Application, Beijing, China
[3]School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing, China
{wusitong, wutianyi01, guoguodong01}@baidu.com, tanhaoru2018@ia.ac.cn

Abstract

Recently, Transformers have shown promising performance in various vision tasks. To reduce the quadratic computation complexity caused by the global self-attention, various methods constrain the range of attention within a local region to improve its efficiency. Consequently, their receptive fields in a single attention layer are not large enough, resulting in insufficient context modeling. To address this issue, we propose a Pale-Shaped self-Attention (PS-Attention), which performs self-attention within a pale-shaped region. Compared to the global self-attention, PS-Attention can reduce the computation and memory costs significantly. Meanwhile, it can capture richer contextual information under the similar computation complexity with previous local self-attention mechanisms. Based on the PS-Attention, we develop a general Vision Transformer backbone with a hierarchical architecture, named Pale Transformer, which achieves 83.4%, 84.3%, and 84.9% Top-1 accuracy with the model size of 22M, 48M, and 85M respectively for 224 × 224 ImageNet-1K classification, outperforming the previous Vision Transformer backbones. For downstream tasks, our Pale Transformer backbone performs better than the recent state-of-the-art CSWin Transformer by a large margin on ADE20K semantic segmentation and COCO object detection & instance segmentation. The code will be released on https://github.com/BR-IDL/PaddleViT.

Introduction

Inspired by the success of Transformer (Vaswani et al. 2017) on a wide range of tasks in natural language processing (NLP) (McCann et al. 2017; Howard and Ruder 2018), Vision Transformer (ViT) (Dosovitskiy et al. 2021) first employed a pure Transformer architecture for image classification, which shows the promising performance of Transformer architecture for vision tasks. However, the quadratic complexity of the global self-attention results in expensive computation costs and memory usage especially for high-resolution scenarios, making it unaffordable for applications in various vision tasks.

A typical way to improve the efficiency is to replace the global self-attention with local ones. A crucial and challenging issue is how to enhance the modeling capability under the local settings. For example, Swin (Liu et al. 2021) and Shuffle Transformer (Huang et al. 2021) proposed shifted window and shuffled window, respectively (Figure 1(b)), and alternately used two different window partitions (i.e., regular window and the proposed window) in consecutive blocks to build cross-window connections. MSG Transformer (Fang et al. 2021) manipulated the messenger tokens to exchange information across windows. Axial self-attention (Wang et al. 2020) treated the local attention region as a single row or column of the feature map (Figure 1(c)). CSWin (Dong et al. 2021) proposed cross-shaped window self-attention (Figure 1(d)), which can be regarded as a multiple row and column expansion of axial self-attention. Although these methods achieve excellent performance and are even superior to the CNN counterparts, the dependencies in each self-attention layer are not rich enough for capturing sufficient contextual information.

In this work, we propose a Pale-Shaped self-Attention (PS-Attention) to capture richer contextual dependencies efficiently. Specifically, the input feature maps are first split into multiple pale-shaped regions spatially. Each pale-shaped region (abbreviating as pale) is composed of the same number of interlaced rows and columns of the feature map. The intervals between adjacent rows or columns are equal for all the pales. For example, the pink shadow in Figure 1(e) indicates one of the pales. Then, self-attention is performed within each pale. For any token, it can directly interact with other tokens within the same pale, which endows our method with the capacity of capturing richer contextual information in a single PS-Attention layer. To further improve the efficiency, we develop a more efficient parallel implementation of the PS-Attention. Benefit from the larger receptive fields and stronger context modeling capability, our PS-Attention shows superiority to the existing local self-attention mechanisms illustrated in Figure 1.

Based on the proposed PS-Attention, we design a general vision transformer backbone with a hierarchical architecture, named Pale Transformer. We scale our approach up to get a series of models, including Pale-T (22M), Pale-S (48M), and Pale-B (85M), reaching significantly better performance than previous approaches. Our Pale-T achieves 83.4% Top-1 classification accuracy on ImageNet-1k, 50.4% single-scale mIoU on ADE20K (semantic segmentation), 47.4 box mAP (object detection) and 42.7 mask

*Corresponding author.

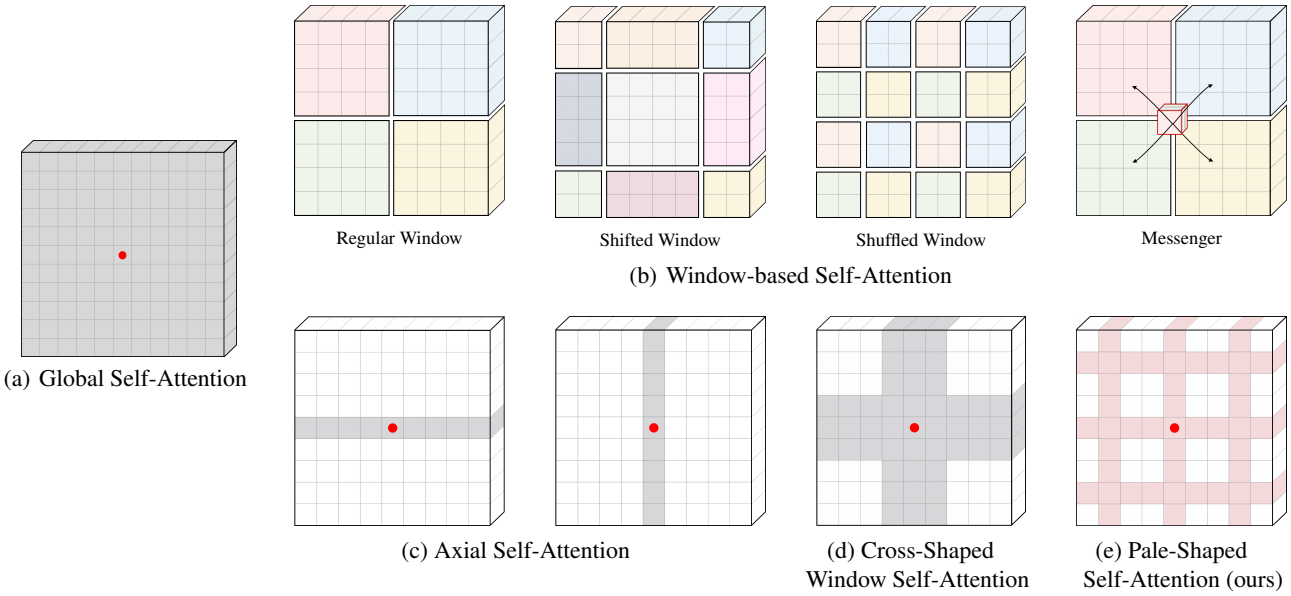

Figure 1: Illustration of different self-attention mechanisms in Transformer backbones. (a) is the standard global self-attention. (b) Window-based self-attention mechanisms perform attention inside each window, and introduce various strategies to build cross-window connections. Different colors in (b) represent different windows. In (c), (d), and (e), the input features are first split into multiple groups, one of which is illustrated by the shadow area, and the self-attention is conducted within each group. Thus, for a reference token denoted by the red dot, it can interact directly with the tokens covered by the shadow area.

mAP (instance segmentation) on COCO, outperforming the state-of-the-art backbones by +0.7%, +1.1%, +0.7, and +0.5, respectively. Furthermore, our largest variant Pale-B is also superior to the previous methods, achieving 84.9% Top-1 accuracy on ImageNet-1K, 52.2% single-scale mIoU on ADE20K, 49.3 box mAP and 44.2 mask mAP on COCO.

Related Work

ViT (Dosovitskiy et al. 2021), which takes the input image as a sequence of patches, has paved a new way and shown promising performance for many vision tasks dominated by CNNs over the years. A line of previous Vision Transformer backbones mainly focused on the following two aspects to better adapt to vision tasks: (1) Enhancing the locality of Vision Transformers. (2) Seeking a better trade-off between performance and efficiency.

Locally-Enhanced Vision Transformers

Different from CNNs, the inductive bias for local connections is not involved in the original Transformer, which may lead to insufficient extraction of local structures, such as lines, edges, and color conjunctions. Many works are devoted to strengthening the local feature extraction of Vision Transformers. The earliest approach is to replace the single-scale architecture of ViT with a hierarchical one to obtain multi-scale features (Wang et al. 2021b). Such design is followed by many works afterward (Liu et al. 2021; Huang et al. 2021; Yang et al. 2021; Dong et al. 2021). Another way is to combine CNNs and Transformers. Mobile-Former (Chen et al. 2021b), Conformer (Peng et al. 2021) and DS-Net (Mao et al. 2021) integrated the CNN and Transformer features by the well-designed dual-branch structures. In contrast, Local ViT (Li et al. 2021b), CvT (Wu et al. 2021a) and Shuffle Transformer (Huang et al. 2021) only inserted several convolutions into some components of Transformer. Besides, some works obtain richer features by fusing the multi-branch with different scales (Chen, Fan, and Panda 2021) or cooperating with local attention (Han et al. 2021; Zhang et al. 2021; Chu et al. 2021a; Li et al. 2021a; Yuan et al. 2021b).

Efficient Vision Transformers

The mainstream research on improving the efficiency for Vision Transformer backbones has two folds: reducing the redundant calculations via pruning strategies and designing more efficient self-attention mechanisms.

Pruning Strategies for Vision Transformers. For pruning, the existing methods can be divided into three categories: (1) Token Pruning. DVT (Wang et al. 2021d) proposed a cascade Transformer architecture to adaptively adjust the number of tokens according to the hardness for classification of the input image. Considering that tokens with irrelevant or even confusing information may be detrimental to image classification, some works proposed to locate discriminative regions and progressively drop less informative tokens by learnable sampling (Rao et al. 2021; Yue et al. 2021) and reinforcement learning (Pan et al. 2021) strategies. However, such unstructured sparsity results in incompatibility with dense prediction tasks. Some structure-preserving token selection strategies were implemented via token pooling (Chen et al. 2021a) and a slow-fast updat-

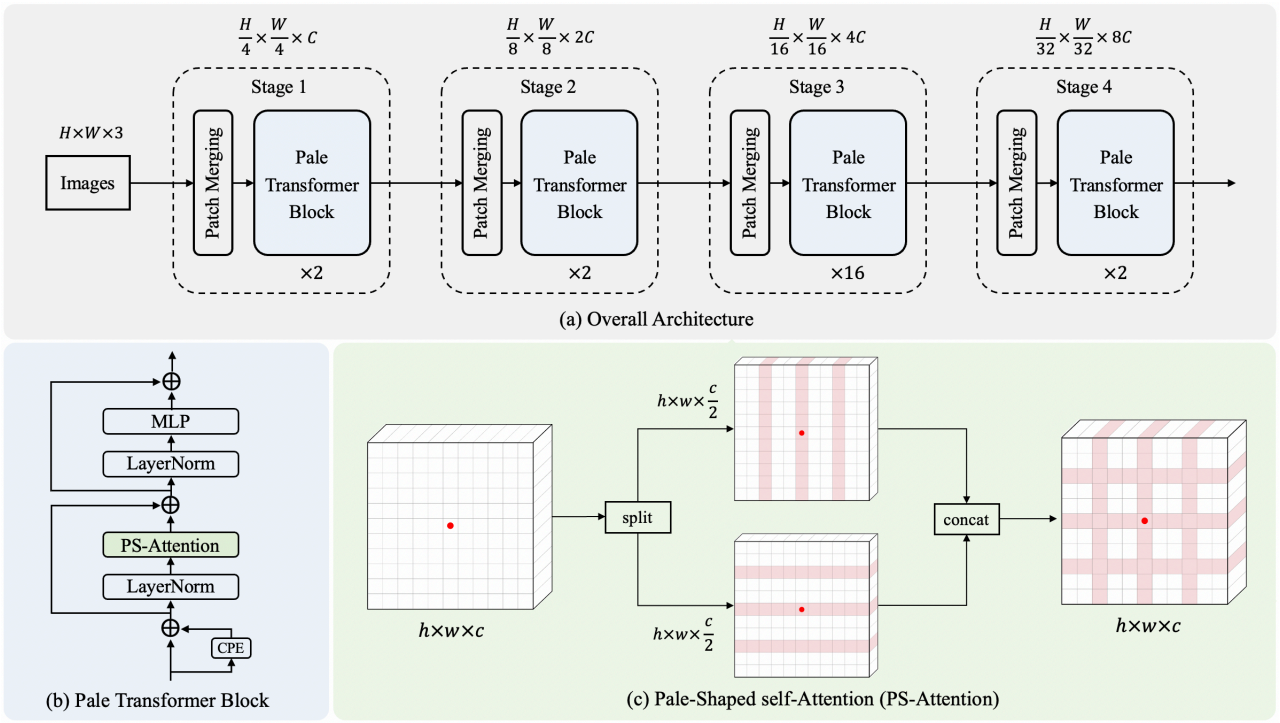

Figure 2: (a) The overall architecture of our Pale Transformer. (b) The composition of each block. (c) Illustration of parallel implementation of PS-Attention. For a reference token (red dot), it can directly interact with the tokens within the shadow area.

ing (Xu et al. 2021). (2) Channel Pruning. VTP (Zhu et al. 2021a) presented a simple but effective framework to remove the reductant channels. (3) Attention Sharing. Based on the observation that attention maps from continuous blocks are highly correlated, PSViT (Chen et al. 2021a) was proposed to reuse the attention calculation process between adjacent layers.

Efficient Self-Attention Mechanisms. Considering that the quadratic computation complexity is caused by self-attention, many methods are committed to improving its efficiency while avoiding performance decay (Wang et al. 2021b; Zhu et al. 2021b; Liu et al. 2021; Huang et al. 2021). One way is to reduce the sequence length of key and value. PVT (Wang et al. 2021b) proposed a spatial reduction attention to downsample the scale of key and value before computing attention. Deformable attention (Zhu et al. 2021b) used a linear layer to select several keys from the full set, which can be regarded as a sparse version of global self-attention. However, excessive downsampling will lead to information confusion, and deformable attention relies heavily on a high-level feature map learned by CNN and may not be directly used on the original input image. Another way is to replace the global self-attention with local self-attention, which limits the range of each self-attention layer into a local region. As shown in Figure 1(b), the feature maps are first divided into several non-overlapping square regular windows (indicated with diverse colors), and the self-attention is performed within each window individually. The key challenge for the design of local self-attention mechanisms is to bridge the gap between local and global receptive fields. A typical manner is to build connections across regular square windows. For example, alternately using regular window and another newly designed window partition manner (shifted window (Liu et al. 2021) or shuffled window (Huang et al. 2021) in Figure 1(b)) in consecutive blocks, and manipulating messenger tokens to exchange information across windows (Fang et al. 2021). Besides, axial self-attention (Wang et al. 2020) achieves longer-range dependencies in horizontal and vertical directions respectively by performing self-attention in each single row or column of the feature map. CSWin (Dong et al. 2021) proposed a cross-shaped window self-attention region including multiple rows and columns. Although these existing local attention mechanisms can provide opportunities for breaking through the local receptive fields to some extent, their dependencies are not rich enough to capture sufficient contextual information in a single self-attention layer, which limits the modeling capacity of the whole network.

The most related to our work is CSWin (Dong et al. 2021), which developed a cross-shaped window self-attention mechanism for computing self-attention in the horizontal and vertical stripes, while our proposed PS-Attention computes self-attention in the pale-shaped regions. Moreover, the receptive fields of each token in our method are much wider than CSWin, which also endows our approach with stronger context modeling capacity.

Methodology

In this section, we first present our Pale-Shaped self-Attention (PS-Attention) and its efficient parallel implementation. Then, the composition of the Pale Transformer block is given. Finally, we describe the overall architecture and variants configurations of our Pale Transformer backbone.

Pale-Shaped self-Attention

For capturing dependencies varied from short-range to long-range, we propose Pale-Shaped self-Attention (PS-Attention), which computes self-attention within a pale-shaped region (abbreviating as pale). As shown in the pink shadow of Figure 1(e), one pale contains s_r interlaced rows and s_c interlaced columns, which covers a region containing $(s_r w + s_c h - s_r s_c)$ tokens. We define (s_r, s_c) as the pale size. Given an input feature map $X \in \mathcal{R}^{h \times w \times c}$, we first split it into multiple pales $\{P_1, ..., P_N\}$ with the same size (s_r, s_c), where $P_i \in \mathcal{R}^{(s_r w + s_c h - s_r s_c) \times c}, i \in \{1, 2, ..., N\}$. The number of pales is equal to $N = \frac{h}{s_r} = \frac{w}{s_c}$, which can be ensured by padding or interpolation operation. For all pales, intervals between adjacent rows or columns are the same. The self-attention is then performed within each pale individually. As illustrated in Figure 1, the receptive field of PS-Attention is significantly wider and richer than all the previous local self-attention mechanisms, enabling more powerful context modeling capacity.

Efficient Parallel Implementation. To further improve the efficiency, we decompose the vanilla PS-Attention mentioned above into row-wise and column-wise attention, which perform self-attention within row-wise and column-wise token groups, respectively. Specifically, as shown in Figure 2(c), we first divide the input feature $X \in \mathcal{R}^{h \times w \times c}$ into two independent parts $X_r \in \mathcal{R}^{h \times w \times \frac{c}{2}}$ and $X_c \in \mathcal{R}^{h \times w \times \frac{c}{2}}$ in the channel dimension, which are then split into multiple groups for row-wise and column-wise attention respectively.

$$X_r = [X_r^1, ..., X_r^{N_r}], X_c = [X_c^1, ..., X_c^{N_c}], \tag{1}$$

where $N_r = h/s_r$, $N_c = w/s_c$, $X_r^i \in \mathcal{R}^{s_r \times w \times c}$ contains s_r interlaced rows, and $X_c^j \in \mathcal{R}^{h \times s_c \times c}$ contains s_c interlaced columns.

Then, the self-attention is conducted within each row-wise and column-wise token group, respectively. Similar to (Wu et al. 2021a), we use three separable convolution layers ϕ_Q, ϕ_K, and ϕ_V to generate the query, key, and value.

$$\begin{aligned} Y_r^i &= \text{MSA}(\phi_Q(X_r^i), \phi_K(X_r^i), \phi_V(X_r^i)), \\ Y_c^i &= \text{MSA}(\phi_Q(X_c^i), \phi_K(X_c^i), \phi_V(X_c^i)), \end{aligned} \tag{2}$$

where $i \in \{1, 2, ..., N\}$, and MSA indicates the Multi-head Self-Attention (Dosovitskiy et al. 2021).

Finally, the outputs of row-wise and column-wise attention are concatenated along channel dimension, resulting in the final output $Y \in \mathcal{R}^{h \times w \times c}$,

$$Y = \text{Concat}(Y_r, Y_c), \tag{3}$$

where $Y_r = [Y_r^1, ..., Y_r^{N_r}]$ and $Y_c = [Y_c^1, ..., Y_c^{N_c}]$.

Compared to the vanilla implementation of PS-Attention within the whole pale, such a parallel mechanism has a lower computation complexity. Furthermore, the padding operation only needs to ensure h can be divisible by s_r and w can be divisible by s_c, rather than $\frac{h}{s_r} = \frac{w}{s_c}$. Therefore, it is also conducive to avoiding excessive padding.

Complexity Analysis. Given the input feature of size $h \times w \times c$ and pale size (s_r, s_c), the standard global self-attention has a computational complexity of

$$\mathcal{O}_{\text{Global}} = 4hwc^2 + 2c(hw)^2, \tag{4}$$

however, our proposed PS-Attention under the parallel implementation has a computational complexity of

$$\mathcal{O}_{\text{Pale}} = 4hwc^2 + hwc(s_c h + s_r w + 27) << \mathcal{O}_{\text{Global}}, \tag{5}$$

which can obviously alleviate the computation and memory burden compared with the global one, since $2hw >> (s_c h + s_r w + 27)$ always holds. The detailed derivations of Eq. (4) and Eq. (5) are provided in the supplementary material.

Pale Transformer Block

As shown in Figure 2(b), our Pale Transformer block consists of three sequential parts, the conditional position encoding (CPE) for dynamically generating the positional embedding, the proposed PS-Attention module for capturing contextual information, and the MLP module for feature projection. The forward pass of the l-th block can be formulated as follows:

$$\tilde{X}^l = X^{l-1} + \text{CPE}(X^{l-1}), \tag{6}$$

$$\hat{X}^l = \tilde{X}^l + \text{PS-Attention}\Big(\text{LN}(\tilde{X}^l)\Big), \tag{7}$$

$$X^l = \hat{X}^l + \text{MLP}\Big(\text{LN}(\hat{X}^l)\Big), \tag{8}$$

where $\text{LN}(\cdot)$ refers to layer normalization (Ba, Kiros, and Hinton 2016). The CPE (Chu et al. 2021b) is implemented as a simple depth-wise convolution, which is widely used in previous works (Wu et al. 2021b; Chu et al. 2021a) for its compatibility with an arbitrary size of input. The PS-Attention module defined in Eq. (7) is constructed by sequentially performing Eq. (1) to Eq. (3). The MLP module defined in Eq. (8) consists of two linear projection layers to expand and contract the embedding dimension sequentially, which is the same as (Dosovitskiy et al. 2021) for fair comparisons.

Overall Architecture and Variants

As illustrated in Figure 2(a), the Pale Transformer consists of four hierarchical stages for capturing multi-scale features by following the popular design in CNNs (He et al. 2016) and Transformers (Liu et al. 2021; Dong et al. 2021). Each stage contains a patch merging layer and multiple Pale Transformer blocks. The patch merging layer aims to spatially downsample the input features by a certain ratio and expand the channel dimension by twice for a better representation capacity. For fair comparisons, we use the overlapping convolution for patch merging, the same as (Wu et al. 2021a;

Stage	Output Stride	Layer	Pale-T	Pale-S	Pale-B
1	4	Patch Merging	$P_1 = 4$ $C_1 = 64$	$P_1 = 4$ $C_1 = 96$	$P_1 = 4$ $C_1 = 128$
		Pale Transformer Block	$\begin{bmatrix} S_1 = 7 \\ H_1 = 2 \\ R_1 = 4 \end{bmatrix} \times 2$	$\begin{bmatrix} S_1 = 7 \\ H_1 = 2 \\ R_1 = 4 \end{bmatrix} \times 2$	$\begin{bmatrix} S_1 = 7 \\ H_1 = 4 \\ R_1 = 4 \end{bmatrix} \times 2$
2	8	Patch Merging	$P_2 = 2$ $C_2 = 128$	$P_2 = 2$ $C_2 = 192$	$P_2 = 2$ $C_2 = 256$
		Pale Transformer Block	$\begin{bmatrix} S_2 = 7 \\ H_2 = 4 \\ R_2 = 4 \end{bmatrix} \times 2$	$\begin{bmatrix} S_2 = 7 \\ H_2 = 4 \\ R_2 = 4 \end{bmatrix} \times 2$	$\begin{bmatrix} S_2 = 7 \\ H_2 = 8 \\ R_2 = 4 \end{bmatrix} \times 2$
3	16	Patch Merging	$P_3 = 2$ $C_3 = 256$	$P_3 = 2$ $C_3 = 384$	$P_3 = 2$ $C_3 = 512$
		Pale Transformer Block	$\begin{bmatrix} S_3 = 7 \\ H_3 = 8 \\ R_3 = 4 \end{bmatrix} \times 16$	$\begin{bmatrix} S_3 = 7 \\ H_3 = 8 \\ R_3 = 4 \end{bmatrix} \times 16$	$\begin{bmatrix} S_3 = 7 \\ H_3 = 16 \\ R_3 = 4 \end{bmatrix} \times 16$
4	32	Patch Merging	$P_4 = 2$ $C_4 = 512$	$P_4 = 2$ $C_4 = 768$	$P_4 = 2$ $C_4 = 1024$
		Pale Transformer Block	$\begin{bmatrix} S_4 = 7 \\ H_4 = 16 \\ R_4 = 4 \end{bmatrix} \times 2$	$\begin{bmatrix} S_4 = 7 \\ H_4 = 16 \\ R_4 = 4 \end{bmatrix} \times 2$	$\begin{bmatrix} S_4 = 7 \\ H_4 = 32 \\ R_4 = 4 \end{bmatrix} \times 2$

Table 1: Detailed configurations of Pale Transformer Variants.

Dong et al. 2021). Specifically, the spatial downsampling ratio is set to 4 for the first stage and 2 for the last three stages, implementing by 7×7 convolution with stride 4 and 3×3 convolution with stride 2, respectively. The outputs of the patch merging layer are fed into the subsequent Pale Transformer blocks, with the number of tokens kept constant. Following (Liu et al. 2021; Dong et al. 2021), we simply apply an average pooling operation on the top of the last block to obtain a representative token for the final classification head, which is composed of a single linear projection layer.

Variants. The definitions of model hyper-parameters for the i-th stage are listed below:

- P_i: the spatial reduction factor for patch merging layer,
- C_i: the embedding dimension of tokens,
- S_i: the pale size for the PS-Attention,
- H_i: the head number for the PS-Attention,
- R_i: the expansion ratio for the MLP module.

By varying the hyper-parameters H_i and C_i in each stage, we design three variants of our Pale Transformer, named Pale-T (Tiny), Pale-S (Small), and Pale-B (Base), respectively. Table 1 shows the detailed configurations of all variants. Note that all variants have the same depth with $[2, 2, 16, 2]$ in four stages. In each stage of these variants, we set the pale size $s_r = s_c = S_i = 7$, and use the same MLP expansion ratio of $R_i = 4$. Thus, the main differences among Pale-T, Pale-S, and Pale-B lie in the embedding dimension of tokens and the head number for the PS-Attention in four stages, *i.e.*, variants vary from narrow to wide.

Backbone	Params	FLOPs	Top-1 (%)
RegNetY-4G (Radosavovic et al. 2020)	21M	4.0G	80.0
DeiT-S (Touvron et al. 2021)	22M	4.6G	79.8
PVT-S (Wang et al. 2021b)	25M	3.8G	79.8
T2T-14 (Yuan et al. 2021a)	22M	6.1G	80.7
DPT-S (Chen et al. 2021c)	26M	4.0G	81.0
TNT-S (Han et al. 2021)	24M	5.2G	81.3
Swin-T (Liu et al. 2021)	29M	4.5G	81.3
Twins-SVT-S (Chu et al. 2021a)	24M	2.8G	81.3
CvT-13 (Wu et al. 2021a)	20M	4.5G	81.6
ViL-S (Zhang et al. 2021)	25M	4.9G	82.0
PVTv2-B2 (Wang et al. 2021a)	25M	4.0G	82.0
Focal-T (Yang et al. 2021)	29M	4.9G	82.2
Shuffle-T (Huang et al. 2021)	29M	4.6G	82.5
CSWin-T (Dong et al. 2021)	23M	4.3G	82.7
LV-ViT-S* (Jiang et al. 2021)	26M	6.6G	83.3
Pale-T (ours)	22M	4.2G	**83.4**
Pale-T* (ours)	22M	4.2G	**84.2**
RegNetY-8G (Radosavovic et al. 2020)	39M	8.0G	81.7
PVT-M (Wang et al. 2021b)	44M	6.7G	81.2
T2T-19 (Yuan et al. 2021a)	39M	9.8G	81.4
DPT-M (Chen et al. 2021c)	46M	6.9G	81.9
CvT-21 (Wu et al. 2021a)	32M	7.1G	82.5
Swin-S (Liu et al. 2021)	50M	8.7G	83.0
MViT-B-24 (Fan et al. 2021)	54M	10.9G	83.1
Twins-SVT-B (Chu et al. 2021a)	56M	8.3G	83.1
PVTv2-B3 (Wang et al. 2021a)	45M	6.9G	83.2
ViL-M (Zhang et al. 2021)	40M	8.7G	83.3
Focal-S (Yang et al. 2021)	51M	9.1G	83.5
Shuffle-S (Huang et al. 2021)	50M	8.9G	83.5
CSWin-S (Dong et al. 2021)	35M	6.9G	83.6
Refined-ViT-S (Zhou et al. 2021)	25M	7.2G	83.6
VOLO-D1* (Yuan et al. 2021b)	27M	6.8G	84.2
Pale-S (ours)	48M	9.0G	**84.3**
Pale-S* (ours)	48M	9.0G	**85.0**
RegNetY-16G (Radosavovic et al. 2020)	84M	16.0G	82.9
ViT-B/16‡	86M	55.4G	77.9
PVT-L (Wang et al. 2021b)	61M	9.8G	81.7
DeiT-B (Touvron et al. 2021)	86M	17.5G	81.8
T2T-24 (Yuan et al. 2021a)	64M	15.0G	82.2
TNT-B (Han et al. 2021)	66M	14.1G	82.8
ViL-B (Zhang et al. 2021)	56M	13.4G	83.2
Swin-B (Liu et al. 2021)	88M	15.4G	83.3
Twins-SVT-L (Chu et al. 2021a)	99M	14.8G	83.3
PVTv2-B5 (Wang et al. 2021a)	82M	11.8G	83.8
Focal-B (Yang et al. 2021)	90M	16.0G	83.8
Shuffle-B (Huang et al. 2021)	88M	15.6G	84.0
LV-ViT-M* (Jiang et al. 2021)	56M	16.0G	84.1
CSWin-B (Dong et al. 2021)	78M	15.0G	84.2
Refined-ViT-M (Zhou et al. 2021)	55M	13.5G	84.6
VOLO-D2* (Yuan et al. 2021b)	59M	14.1G	85.2
Pale-B (ours)	85M	15.6G	**84.9**
Pale-B* (ours)	85M	15.6G	**85.8**

Table 2: Comparisons of different backbones on ImageNet-1K validation set. All the approaches are trained and evaluated with the size of 224×224, except for the ViT-B‡ with size 384×384. The superscript "$\star$" indicates employing MixToken and token labeling loss (Jiang et al. 2021) during training.

Experiments

We first compare our Pale Transformer with the state-of-the-art Transformer backbones on ImageNet-1K (Russakovsky et al. 2015) for image classification. To further demonstrate the effectiveness and generalization of our backbone, we conduct experiments on ADE20K (Zhou et al. 2019) for semantic segmentation (Wu et al. 2021b, 2020; Zhang et al. 2019; Wu et al. 2021c), and COCO (Lin et al. 2014) for object detection & instance segmentation. Finally, we dig into the design of key components of our Pale Transformer to

Backbone	Params	FLOPs	Mask R-CNN (1x)					
			AP^{box}	AP^{box}_{50}	AP^{box}_{75}	AP^{mask}	AP^{mask}_{50}	AP^{mask}_{75}
ResNet-50 (He et al. 2016)	44M	260G	38.0	58.6	41.4	34.4	55.1	36.7
PVT-S (Wang et al. 2021b)	44M	245G	40.4	62.9	43.8	37.8	60.1	40.3
ViL-S (Zhang et al. 2021)	45M	174G	41.8	64.1	45.1	38.5	61.1	41.4
Twins-S (Chu et al. 2021a)	44M	228G	42.7	65.6	46.7	39.6	62.5	42.6
DPT-S (Chen et al. 2021c)	46M	-	43.1	65.7	47.2	39.9	62.9	43.0
Swin-T (Liu et al. 2021)	48M	264G	43.7	66.6	47.6	39.8	63.3	42.7
RegionViT-S+ (Chen, Panda, and Fan 2021)	51M	183G	44.2	67.3	48.2	40.8	64.1	44.0
Focal-T (Yang et al. 2021)	49M	291G	44.8	67.7	49.2	41.0	64.7	44.2
PVTv2-B2 (Wang et al. 2021a)	45M	-	45.3	67.1	49.6	41.2	64.2	44.4
CSWin-T (Dong et al. 2021)	42M	279G	46.7	68.6	51.3	42.2	65.6	45.4
Pale-T (ours)	41M	306G	**47.4**	**69.2**	**52.3**	**42.7**	**66.3**	**46.2**
ResNeXt-101-32 (He et al. 2016)	63M	340G	41.9	62.5	45.9	37.5	59.4	40.2
PVT-M (Wang et al. 2021b)	64M	302G	42.0	64.4	45.6	39.0	61.6	42.1
ViL-M (Zhang et al. 2021)	60M	261G	43.4	65.9	47.0	39.7	62.8	42.1
DPT-M (Chen et al. 2021c)	66M	-	43.8	66.2	48.3	40.3	63.1	43.4
Twins-B (Chu et al. 2021a)	76M	340G	45.1	67.0	49.4	41.1	64.1	44.4
RegionViT-B+ (Chen, Panda, and Fan 2021)	93M	307G	45.4	68.4	49.6	41.6	65.2	44.8
PVTv2-B3 (Wang et al. 2021a)	65M	-	47.0	68.1	51.7	42.5	65.7	45.7
Focal-S (Yang et al. 2021)	71M	401G	47.4	69.8	51.9	42.8	66.6	46.1
CSWin-S (Dong et al. 2021)	54M	342G	47.9	70.1	52.6	43.2	67.1	46.2
Pale-S (ours)	68M	432G	**48.4**	**70.4**	**53.2**	**43.7**	**67.7**	**47.1**
ResNeXt-101-64 (He et al. 2016)	101M	493G	42.8	63.8	47.3	38.4	60.6	41.3
PVT-L (Wang et al. 2021b)	81M	364G	42.9	65.0	46.6	39.5	61.9	42.5
ViL-B (Zhang et al. 2021)	76M	365G	45.1	67.2	49.3	41.0	64.3	44.2
Twins-L (Chu et al. 2021a)	120M	474G	45.2	67.5	49.4	41.2	64.5	44.5
PVTv2-B4 (Wang et al. 2021a)	82M	-	47.5	68.7	52.0	42.7	66.1	46.1
Focal-B (Yang et al. 2021)	110M	533G	47.8	70.2	52.5	43.2	67.3	46.5
CSWin-B (Dong et al. 2021)	97M	526G	48.7	70.4	53.9	43.9	67.8	47.3
Pale-B (ours)	105M	595G	**49.3**	**71.2**	**54.1**	**44.2**	**68.1**	**47.8**

Table 3: Comparisons on COCO val2017 with Mask R-CNN framework and 1x training schedule for object detection and instance segmentation.

better understand the method.

Image Classification on ImageNet-1K

Settings. All the variants are trained from scratch for 300 epochs on 8 V100 GPUs with a total batch size of 1024. Both the training and evaluation are conducted with the input size of 224×224 on ImageNet-1K dataset. Detailed configurations are provided in the supplementary material.

Results. Table 2 compares the performance of our Pale Transformer with the state-of-the-art CNNs and Vision Transformer backbones on ImageNet-1K validation set. Compared to the advanced CNNs, our Pale variants are +3.4%, +2.6%, and +2.0% better than the well-known RegNet (Radosavovic et al. 2020) models, respectively, under the similar computation complexity. Meanwhile, our Pale Transformer outperforms the state-of-the-art Transformer-based backbones, and is +0.7% higher than the most related CSWin Transformer for all variants under the similar model size and FLOPs. Note that LV-ViT (Jiang et al. 2021) and VOLO (Yuan et al. 2021b), using additional MixToken augmentation and token labeling loss (Jiang et al. 2021) for training, seem to be on par with our approach. For fair comparisons, we use these two tricks on our Pale models, labeled by * as the superscript. Pale-T* achieves +0.9% gain than LV-ViT-S* with fewer computation costs. Pale-S* and Pale-B* achieve 85.0% and 85.8%, outperforming VOLO by +0.8% and +0.6%, respectively.

Semantic Segmentation on ADE20K

Settings. To demonstrate the superiority of our Pale Transformer for dense prediction tasks, we conduct experiments on ADE20K with the widely-used UperNet (Xiao et al. 2018) as decoder for fair comparisons to other backbones. Detailed settings are described in the supplementary material.

Results. Table 4 shows the comparisons of UperNet with various excellent Transformer backbones on ADE20K validation set. We report both the single-scale (SS) and multi-scale (MS) mIoU for better comparison. Our Pale variants are consistently superior to the state-of-the-art method by a large margin. Specifically, our Pale-T and Pale-S outperform the state-of-the-art CSWin by +1.1% and +1.2% SS mIoU, respectively. Besides, our Pale-B achieves 52.5%/53.0% SS/MS mIoU, surpassing the previous best by +1.3% and +1.2%, respectively. These results demonstrate the stronger context modeling capacity of our Pale Transformer for dense prediction tasks.

Object Detection and Instance Segmentation on COCO

Settings. We evaluate the performance of our Pale Transformer backbone on COCO benchmark for object detection and instance segmentation, utilizing Mask R-CNN (He et al. 2017) framework under 1x schedule (12 training epochs). Details can be found in the supplementary material.

Backbone	Params	FLOPs	SS mIoU	MS mIoU
DeiT-S (Touvron et al. 2021)	52M	1099G	-	44.0
Swin-T (Liu et al. 2021)	60M	945G	44.5	45.8
Focal-T (Yang et al. 2021)	62M	998G	45.8	47.0
Shuffle-T (Huang et al. 2021)	60M	949G	46.6	47.6
CrossFormer-S (Wang et al. 2021c)	62M	980G	47.6	48.4
LV-ViT-S (Jiang et al. 2021)	44M	-	47.9	48.6
CSWin-T (Dong et al. 2021)	60M	959G	49.3	50.4
Pale-T (ours)	52M	996G	**50.4**	**51.2**
Swin-S (Liu et al. 2021)	81M	1038G	47.6	49.5
Focal-S (Yang et al. 2021)	85M	1130G	48.0	50.0
Shuffle-S (Huang et al. 2021)	81M	1044G	48.4	49.6
VOLO-D1 (Yuan et al. 2021b)	-	-	-	50.5
LV-ViT-M (Jiang et al. 2021)	77M	-	49.4	50.6
CrossFormer-B (Wang et al. 2021c)	84M	1090G	49.7	50.6
CSWin-S (Dong et al. 2021)	65M	1027G	50.0	50.8
Pale-S (ours)	80M	1135G	**51.2**	**52.2**
Swin-B (Liu et al. 2021)	121M	1188G	48.1	49.7
Shuffle-B (Huang et al. 2021)	121M	1196G	49.0	50.5
Focal-B (Yang et al. 2021)	126M	1354G	49.0	50.5
CrossFormer-L (Wang et al. 2021c)	126M	1258M	50.4	51.4
CSWin-B (Dong et al. 2021)	109M	1222G	50.8	51.7
LV-ViT-L (Jiang et al. 2021)	209M	-	50.9	51.8
Pale-B (ours)	119M	1311G	**52.2**	**53.0**

Table 4: Comparisons of different backbones with UperNet as decoder on ADE20K for semantic segmentation. All backbones are pretrained on ImageNet-1K with the size of 224×224. FLOPs are calculated with a resolution of 512×2048.

Results. As shown in Table 3, for object detection, our Pale-T, Pale-S, and Pale-B achieve 47.4, 48.4, and 49.2 box mAP for object detection, surpassing the previous best CSWin Transformer by +0.7, +0.5, and +0.6, respectively. Besides, our variants also have consistent improvement on instance segmentation, which are +0.5, +0.5, and +0.3 mask mAP higher than the previous best backbone.

Ablation Study

We conduct ablation studies for the key designs of our Pale Transformer on image classification and downstream tasks. All the experiments are performed with the Tiny variant under the same training settings as mentioned above. We also analyze the influence of position encoding in the supplementary material.

Effect of Pale Size. The pale sizes of four stages $\{S_1, S_2, S_3, S_4\}$ control the trade-off between the richness of contextual information and computation costs. As shown in Table 7, increasing the pale size (from 1 to 7) can continuously improve performance across all tasks, while further up to 9 does not bring obvious and consistent improvements but more FLOPs. Therefore, we use $S_i = 7, i \in \{1, 2, 3, 4\}$ for all the tasks by default.

Comparisons with Different Implementations of PS-Attention. We compare three implementations of our PS-Attention. The vanilla PS-Attention directly conducts self-attention within the whole pale region, which can be approximated as two more efficient implementations, sequential and parallel. The sequential one computes self-attention in row and column directions alternately in consecutive blocks,

Pale size in four stages	ImageNet-1K Top-1 (%)	ADE20K SS mIoU (%)	COCO AP^{box}	COCO AP^{mask}
1 1 1 1	82.4	47.9	46.1	41.5
3 3 3 3	82.9	49.4	46.7	42.3
5 5 5 5	83.1	49.7	46.8	42.4
7 7 7 7	**83.4**	**50.4**	**47.4**	**42.7**
9 9 9 9	83.3	50.6	47.4	42.6

Table 5: Ablation study for different choices of pale size. The complete table with parameters and FLOPs can be found in the supplementary material.

Attention mode	ImageNet-1K Top-1 (%)	ADE20K SS mIoU (%)	COCO AP^{box}	COCO AP^{mask}
Axial	82.4	47.9	46.1	41.5
Cross-Shaped	82.8	49.0	46.6	42.2
Pale (vanilla)	83.4	50.3	47.1	42.3
Pale (sequential)	82.9	49.5	46.9	42.2
Pale (parallel)	**83.4**	**50.4**	**47.4**	**42.7**

Table 6: Ablation study for different attention modes. Params and FLOPs of all the experiments are provided in the supplementary material.

while the parallel one performs row-wise and column-wise attention in parallel within each block. As shown in Table 8, the parallel PS-Attention achieves the best results on all tasks, even slightly better than the vanilla one by +0.3/0.4 box/mask mAP on COCO. We attribute this to that the excessive padding for the non-square input size in vanilla PS-Attention will result in slight performance degradation.

Comparisons with other Axial-based Attentions. In order to compare our PS-Attention with the most related axial-based self-attention mechanisms directly, we replace the PS-Attention of our Pale-T with the axial self-attention (Wang et al. 2020) and cross-shaped window self-attention (Dong et al. 2021), respectively. As shown in Table 8, our PS-Attention outperforms these two mechanisms obviously.

Conclusion

This work presented a new effective and efficient self-attention mechanism, named Pale-Shaped self-Attention (PS-Attention), which performs self-attention in a pale-shaped region. PS-Attention can model richer contextual dependencies than the previous local self-attention mechanisms. In order to further improve its efficiency, we designed a parallel implementation for PS-Attention, which decomposes the self-attention within the whole pale into row-wise and column-wise attention. It is also conducive to avoiding excessive padding operations. Based on the proposed PS-Attention, we developed a general Vision Transformer backbone, called Pale Transformer, which can achieve state-of-the-art performance on ImageNet-1K for image classification. Furthermore, our Pale Transformer is superior to the previous Vision Transformer backbones on ADE20K for semantic segmentation, and COCO for object detection & instance segmentation.

References

Ba, J. L.; Kiros, J. R.; and Hinton, G. E. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*.

Chen, B.; Li, P.; Li, B.; Li, C.; Bai, L.; Lin, C.; Sun, M.; Yan, J.; and Ouyang, W. 2021a. PSViT: Better Vision Transformer via Token Pooling and Attention Sharing. *arXiv preprint arXiv:2108.03428*.

Chen, C.-F.; Fan, Q.; and Panda, R. 2021. Crossvit: Cross-attention multi-scale vision transformer for image classification. *arXiv preprint arXiv:2103.14899*.

Chen, C.-F.; Panda, R.; and Fan, Q. 2021. RegionViT: Regional-to-Local Attention for Vision Transformers. *arXiv preprint arXiv:2106.02689*.

Chen, Y.; Dai, X.; Chen, D.; Liu, M.; Dong, X.; Yuan, L.; and Liu, Z. 2021b. Mobile-Former: Bridging MobileNet and Transformer. *arXiv preprint arXiv:2108.05895*.

Chen, Z.; Zhu, Y.; Zhao, C.; Hu, G.; Zeng, W.; Wang, J.; and Tang, M. 2021c. *DPT: Deformable Patch-Based Transformer for Visual Recognition*, 2899–2907. New York, NY, USA: Association for Computing Machinery. ISBN 9781450386517.

Chu, X.; Tian, Z.; Wang, Y.; Zhang, B.; Ren, H.; Wei, X.; Xia, H.; and Shen, C. 2021a. Twins: Revisiting the design of spatial attention in vision transformers. *arXiv preprint arXiv:2104.13840*, 1(2): 3.

Chu, X.; Tian, Z.; Zhang, B.; Wang, X.; Wei, X.; Xia, H.; and Shen, C. 2021b. Conditional Positional Encodings for Vision Transformers. *arXiv preprint arXiv:2102.10882*.

Cubuk, E. D.; Zoph, B.; Mane, D.; Vasudevan, V.; and Le, Q. V. 2019. Autoaugment: Learning augmentation strategies from data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 113–123.

Dong, X.; Bao, J.; Chen, D.; Zhang, W.; Yu, N.; Yuan, L.; Chen, D.; and Guo, B. 2021. CSWin Transformer: A General Vision Transformer Backbone with Cross-Shaped Windows. *arXiv preprint arXiv:2107.00652*.

Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; Uszkoreit, J.; and Houlsby, N. 2021. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In *International Conference on Learning Representations*.

Fan, H.; Xiong, B.; Mangalam, K.; Li, Y.; Yan, Z.; Malik, J.; and Feichtenhofer, C. 2021. Multiscale Vision Transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 6824–6835.

Fang, J.; Xie, L.; Wang, X.; Zhang, X.; Liu, W.; and Tian, Q. 2021. MSG-Transformer: Exchanging Local Spatial Information by Manipulating Messenger Tokens. *arXiv preprint arXiv:2105.15168*.

Han, K.; Xiao, A.; Wu, E.; Guo, J.; Xu, C.; and Wang, Y. 2021. Transformer in transformer. *arXiv preprint arXiv:2103.00112*.

He, K.; Gkioxari, G.; Dollár, P.; and Girshick, R. 2017. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, 2961–2969.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–778.

Howard, J.; and Ruder, S. 2018. Universal Language Model Fine-tuning for Text Classification. In *ACL*.

Huang, G.; Sun, Y.; Liu, Z.; Sedra, D.; and Weinberger, K. Q. 2016. Deep networks with stochastic depth. In *European conference on computer vision*, 646–661. Springer.

Huang, Z.; Ben, Y.; Luo, G.; Cheng, P.; Yu, G.; and Fu, B. 2021. Shuffle Transformer: Rethinking Spatial Shuffle for Vision Transformer. *arXiv preprint arXiv:2106.03650*.

Jiang, Z.; Hou, Q.; Yuan, L.; Zhou, D.; Shi, Y.; Jin, X.; Wang, A.; and Feng, J. 2021. All Tokens Matter: Token Labeling for Training Better Vision Transformers. *arXiv preprint arXiv:2104.10858*.

Li, J.; Yan, Y.; Liao, S.; Yang, X.; and Shao, L. 2021a. Local-to-Global Self-Attention in Vision Transformers. *arXiv preprint arXiv:2107.04735*.

Li, Y.; Zhang, K.; Cao, J.; Timofte, R.; and Van Gool, L. 2021b. LocalViT: Bringing Locality to Vision Transformers. *arXiv preprint arXiv:2104.05707*.

Lin, T.-Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; and Zitnick, C. L. 2014. Microsoft coco: Common objects in context. In *European conference on computer vision*, 740–755. Springer.

Liu, Z.; Lin, Y.; Cao, Y.; Hu, H.; Wei, Y.; Zhang, Z.; Lin, S.; and Guo, B. 2021. Swin Transformer: Hierarchical Vision Transformer Using Shifted Windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 10012–10022.

Loshchilov, I.; and Hutter, F. 2019. Decoupled Weight Decay Regularization. In *ICLR*.

Mao, M.; Zhang, R.; Zheng, H.; Gao, P.; Ma, T.; Peng, Y.; Ding, E.; and Han, S. 2021. Dual-stream Network for Visual Recognition. *arXiv preprint arXiv:2105.14734*.

McCann, B.; Bradbury, J.; Xiong, C.; and Socher, R. 2017. Learned in Translation: Contextualized Word Vectors. In *NIPS*.

Pan, B.; Jiang, Y.; Panda, R.; Wang, Z.; Feris, R.; and Oliva, A. 2021. IA-RED^2: Interpretability-Aware Redundancy Reduction for Vision Transformers. *arXiv preprint arXiv:2106.12620*.

Peng, Z.; Huang, W.; Gu, S.; Xie, L.; Wang, Y.; Jiao, J.; and Ye, Q. 2021. Conformer: Local Features Coupling Global Representations for Visual Recognition. *arXiv preprint arXiv:2105.03889*.

Radosavovic, I.; Kosaraju, R. P.; Girshick, R.; He, K.; and Dollár, P. 2020. Designing network design spaces. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 10428–10436.

Rao, Y.; Zhao, W.; Liu, B.; Lu, J.; Zhou, J.; and Hsieh, C.-J. 2021. DynamicViT: Efficient Vision Transformers with Dynamic Token Sparsification. *arXiv preprint arXiv:2106.02034*.

Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M.; et al. 2015. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3): 211–252.

Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; and Rabinovich, A. 2015. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 1–9.

Szegedy, C.; Vanhoucke, V.; Ioffe, S.; Shlens, J.; and Wojna, Z. 2016. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2818–2826.

Touvron, H.; Cord, M.; Douze, M.; Massa, F.; Sablayrolles, A.; and Jégou, H. 2021. Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning*, 10347–10357. PMLR.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. In *Advances in neural information processing systems*, 5998–6008.

Wang, H.; Zhu, Y.; Green, B.; Adam, H.; Yuille, A.; and Chen, L.-C. 2020. Axial-deeplab: Stand-alone axial-attention for panoptic segmentation. In *European Conference on Computer Vision*, 108–126. Springer.

Wang, W.; Xie, E.; Li, X.; Fan, D.-P.; Song, K.; Liang, D.; Lu, T.; Luo, P.; and Shao, L. 2021a. Pvtv2: Improved baselines with pyramid vision transformer. *arXiv preprint arXiv:2106.13797*.

Wang, W.; Xie, E.; Li, X.; Fan, D.-P.; Song, K.; Liang, D.; Lu, T.; Luo, P.; and Shao, L. 2021b. Pyramid Vision Transformer: A Versatile Backbone for Dense Prediction Without Convolutions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 568–578.

Wang, W.; Yao, L.; Chen, L.; Cai, D.; He, X.; and Liu, W. 2021c. CrossFormer: A Versatile Vision Transformer Based on Cross-scale Attention. *arXiv preprint arXiv:2108.00154*.

Wang, Y.; Huang, R.; Song, S.; Huang, Z.; and Huang, G. 2021d. Not All Images are Worth 16x16 Words: Dynamic Vision Transformers with Adaptive Sequence Length. *arXiv preprint arXiv:2105.15075*.

Wu, H.; Xiao, B.; Codella, N.; Liu, M.; Dai, X.; Yuan, L.; and Zhang, L. 2021a. CvT: Introducing Convolutions to Vision Transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 22–31.

Wu, S.; Wu, T.; Lin, F.; Tian, S.; and Guo, G. 2021b. Fully Transformer Networks for Semantic Image Segmentation. *arXiv preprint arXiv:2106.04108*.

Wu, T.; Tang, S.; Zhang, R.; Cao, J.; and Zhang, Y. 2020. Cgnet: A light-weight context guided network for semantic segmentation. *IEEE Transactions on Image Processing*, 30: 1169–1179.

Wu, T.; Tang, S.; Zhang, R.; and Guo, G. 2021c. Consensus feature network for scene parsing. *IEEE Transactions on Multimedia*.

Xiao, T.; Liu, Y.; Zhou, B.; Jiang, Y.; and Sun, J. 2018. Unified perceptual parsing for scene understanding. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 418–434.

Xu, Y.; Zhang, Z.; Zhang, M.; Sheng, K.; Li, K.; Dong, W.; Zhang, L.; Xu, C.; and Sun, X. 2021. Evo-ViT: Slow-Fast Token Evolution for Dynamic Vision Transformer. *arXiv preprint arXiv:2108.01390*.

Yang, J.; Li, C.; Zhang, P.; Dai, X.; Xiao, B.; Yuan, L.; and Gao, J. 2021. Focal Self-attention for Local-Global Interactions in Vision Transformers. *arXiv preprint arXiv:2107.00641*.

Yuan, L.; Chen, Y.; Wang, T.; Yu, W.; Shi, Y.; Jiang, Z.-H.; Tay, F. E.; Feng, J.; and Yan, S. 2021a. Tokens-to-Token ViT: Training Vision Transformers From Scratch on ImageNet. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 558–567.

Yuan, L.; Hou, Q.; Jiang, Z.; Feng, J.; and Yan, S. 2021b. Volo: Vision outlooker for visual recognition. *arXiv preprint arXiv:2106.13112*.

Yue, X.; Sun, S.; Kuang, Z.; Wei, M.; Torr, P. H.; Zhang, W.; and Lin, D. 2021. Vision Transformer With Progressive Sampling. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 387–396.

Yun, S.; Han, D.; Oh, S. J.; Chun, S.; Choe, J.; and Yoo, Y. 2019. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 6023–6032.

Zhang, H.; Cisse, M.; Dauphin, Y. N.; and Lopez-Paz, D. 2018. mixup: Beyond Empirical Risk Minimization. In *International Conference on Learning Representations*.

Zhang, P.; Dai, X.; Yang, J.; Xiao, B.; Yuan, L.; Zhang, L.; and Gao, J. 2021. Multi-Scale Vision Longformer: A New Vision Transformer for High-Resolution Image Encoding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2998–3008.

Zhang, R.; Tang, S.; Zhang, Y.; Li, J.; and Yan, S. 2019. Perspective-adaptive convolutions for scene parsing. *IEEE transactions on pattern analysis and machine intelligence*, 42(4): 909–924.

Zhou, B.; Zhao, H.; Puig, X.; Xiao, T.; Fidler, S.; Barriuso, A.; and Torralba, A. 2019. Semantic understanding of scenes through the ade20k dataset. *International Journal of Computer Vision*, 127(3): 302–321.

Zhou, D.; Shi, Y.; Kang, B.; Yu, W.; Jiang, Z.; Li, Y.; Jin, X.; Hou, Q.; and Feng, J. 2021. Refiner: Refining Self-attention for Vision Transformers. *arXiv preprint arXiv:2106.03714*.

Zhu, M.; Han, K.; Tang, Y.; and Wang, Y. 2021a. Visual Transformer Pruning. *arXiv preprint arXiv:2104.08500*.

Zhu, X.; Su, W.; Lu, L.; Li, B.; Wang, X.; and Dai, J. 2021b. Deformable DETR: Deformable Transformers for End-to-End Object Detection. In *International Conference on Learning Representations*.

Appendix

In this appendix, we first provide the detailed experimental settings for classification, semantic segmentation, object detection, and instance segmentation, respectively. Then, we study the effect of the position encoding method of our Pale Transformer, and provide more detailed comparisons of the ablation studies in the body of our paper in terms of the model size and computation costs. Finally, the detailed derivations of computation complexity for the global self-attention and our PS-Attention are given.

Detailed Experimental Settings

Image Classification on ImageNet-1K

We follow most of the settings in DeiT (Touvron et al. 2021), Swin (Liu et al. 2021) and CSWin (Dong et al. 2021) for fair comparisons. In detail, we use AdamW (Loshchilov and Hutter 2019) optimizer with a weight decay of 0.05. The initial learning rate is set to 1e-3 and progressively decays after each iteration by a cosine schedule. The linear warmup takes up 20 epochs. We use the random horizontal flipping (Szegedy et al. 2015), color jitter, Mixup (Zhang et al. 2018), CutMix (Yun et al. 2019) and AutoAugment (Cubuk et al. 2019) as data augmentation. We also adopt some common regularizations, such as Label-Smoothing (Szegedy et al. 2016) and stochastic depth (Huang et al. 2016). The maximal stochastic depth rate is set to 0.1, 0.3, and 0.5 for Pale-T, Pale-S, and Pale-B, respectively. All the variants are trained from scratch for 300 epochs on 8 V100 GPUs with the input size of 224×224 and a total batch size of 1024. During the evaluation, the images are first resized to 256×256 and then center-cropped to 224×224.

Semantic Segmentation on ADE20K

We conduct experiments on the widely-used and challenging ADE20K (Zhou et al. 2019) scene parsing dataset, which contains 20210, 2000, and 3352 images for training, validation, and testing, respectively, with 150 fine-grained object categories. For fair comparisons, we use our ImageNet-1k pretrained Pale Transformer as backbone and UperNet as the decoder, and follow the same training settings as (Liu et al. 2021). Specifically, all the models are trained for total 160k iterations with a batch size of 16. The AdamW (Loshchilov and Hutter 2019) optimizer with weight decay 0.01 is used. The initial learning rate is set to 6e-5 and decay with a polynomial scheduler after the 1500-iterations warmup. The stochastic depth rate is set to 0.3, 0.3, and 0.5 for Pale-T, Pale-S, and Pale-B, respectively. Both single-scale and multi-scale inference are reported for performance comparison. For multi-scale inference, factors vary from 0.75 to 1.75 with 0.25 as the interval. Auxiliary losses are added to the output of stage 3 of the backbone with factor 0.4, which is the same as the previous works (Liu et al. 2021; Dong et al. 2021) for fair comparisons. For data augmentation during training, we follow the default configurations of mmsegmentation, such as random crop, random flipping, random rescaling (with ratio range from 0.5 to 2.0), and random photometric distortion.

Object Detection & Instance Segmentation on COCO

We compare the performance of our Pale Transformer backbone on COCO benchmark for object detection and instance segmentation, with the typical Mask R-CNN (He et al. 2017) framework. We follow the same training strategies as (Dong et al. 2021). In detail, we train and evaluate our Pale Transformer under the normal 1x schedule, and all the models are trained for 12 epochs on 8 GPUs with the total batch size of 16 and single-scale input (shorter size is resized to 800 and longer size is no more than 1333). We use AdamW (Loshchilov and Hutter 2019) as the optimizer with a weight decay of 0.001 for Pale-T and Pale-S and 0.05 for Pale-B. For all models, the learning rate is set to 0.0001 initially and decay at epoch 8 and 11 with a ratio of 0.1. We set the stochastic depth rate to 0.2, 0.3, and 0.5 for Pale-T, Pale-S, and Pale-B, respectively. FLOPs are compared under the input size of 1280×800.

Further Ablation Study

In this section, we first provide the complete version of Table 5 and Table 6 in the body of our paper, including the parameters and FLOPs comparisons, shown in Table 7 and Table 8. Then, we analyze the effect of different position encoding methods for our Pale Transformer backbone.

Effect of Position Encoding. The position encoding plays an important role in Transformers, as it can introduce the spatial location awareness for feature aggregation of self-attention. Here, we compare several widely-used position encoding methods, e.g., no position encoding (no pos.), absolute position encoding (APE) (Dosovitskiy et al. 2021) and conditional position encoding (CPE) (Chu et al. 2021b). As shown in Table 9, CPE performs best. Not using any position encoding will cause serious performance degradation, which demonstrates the effectiveness of the position encoding in Vision Transformer models.

Derivations of the Computational Complexity

In this section, we derive the computation complexity of the global self-attention and our PS-Attention in detail.

Computational Complexity of Global Self-Attention

Supposing that the size of input feature map is denoted as $h \times w \times c$. The global self-attention (Dosovitskiy et al. 2021) has three parts. Firstly, the input feature $X \in \mathcal{R}^{h \times w \times c}$ is first sent into three independent linear layers to generate query $Q \in \mathcal{R}^{h \times w \times c}$, key $K \in \mathcal{R}^{h \times w \times c}$, and value $V \in \mathcal{R}^{h \times w \times c}$, respectively. Thus, the computational complexity of the generation of Q, K, and V is

$$\mathcal{O}^{\text{qkv}}_{\text{Global}} = 3hwc^2. \tag{9}$$

Secondly, the attention map A is computed by $\text{softmax}(QK^T/\sqrt{d})$. Then, the aggregated feature is obtained by the matrix multiplication between the normalized attention map A and the value V. The computational

Pale size in four stages	ImageNet-1K			ADE20K			COCO			
	Params	FLOPs	Top-1 (%)	Params	FLOPs	SS mIoU (%)	Params	FLOPs	AP^{box}	AP^{mask}
1 1 1 1	22M	3.8G	82.4	52M	929G	47.9	41M	253G	46.1	41.5
3 3 3 3	22M	4.1G	82.9	52M	950G	49.4	41M	269G	46.7	42.3
5 5 5 5	22M	4.4G	83.1	52M	972G	49.7	41M	283G	46.8	42.4
7 7 7 7	22M	4.2G	**83.4**	52M	996G	50.4	41M	306G	**47.4**	**42.7**
9 9 9 9	22M	5.4G	83.3	52M	1021G	**50.6**	41M	322G	**47.4**	42.6

Table 7: Ablation study for different choices of pale size.

Attention mode	ImageNet-1K			ADE20K			COCO			
	Params	FLOPs	Top-1 (%)	Params	FLOPs	SS mIoU (%)	Params	FLOPs	AP^{box}	AP^{mask}
Axial	22M	3.8G	82.4	52M	929G	47.9	41M	253G	46.1	41.5
Cross-Shaped	22M	4.2G	82.8	52M	996G	49.0	41M	306G	46.6	42.2
Pale (vanilla)	22M	5.4G	83.4	52M	2677G	50.2	41M	668G	47.1	42.3
Pale (sequential)	22M	4.2G	82.9	52M	996G	49.5	41M	306G	46.9	42.2
Pale (parallel)	22M	4.2G	**83.4**	52M	996G	**50.4**	41M	306G	**47.4**	**42.7**

Table 8: Ablation study for different attention modes.

Position Encoding	ImageNet-1K			ADE20K			COCO			
	Params	FLOPs	Top-1 (%)	Params	FLOPs	SS mIoU (%)	Params	FLOPs	AP^{box}	AP^{mask}
no pos.	22M	4.2G	82.5	51M	996G	48.3	41M	306G	46.2	41.5
APE	22M	4.2G	82.9	59M	996G	49.6	49M	306G	46.8	42.2
CPE	22M	4.2G	**83.4**	52M	996G	**50.4**	41M	306G	**47.4**	**42.7**

Table 9: Ablation study for different position encoding methods.

complexity of these two processes is

$$\mathcal{O}^{\text{attn}}_{\text{Global}} = 2c(hw)^2. \tag{10}$$

Finally, the aggregated feature also needs to pass through a linear projection layer generally with the complexity of

$$\mathcal{O}^{\text{proj}}_{\text{Global}} = hwc^2. \tag{11}$$

Thus, the overall computational complexity of the global self-attention is

$$\begin{aligned}\mathcal{O}_{\text{Global}} &= \mathcal{O}^{\text{qkv}}_{\text{Global}} + \mathcal{O}^{\text{attn}}_{\text{Global}} + \mathcal{O}^{\text{proj}}_{\text{Global}} \\ &= 4hwc^2 + 2c(hw)^2.\end{aligned} \tag{12}$$

Computational Complexity of Our PS-Attention

Similarly, given the input feature of size $h \times w \times c$ and pale size (s_r, s_c), our PS-Attention(parallel) also contains three processes. Firstly, the three individual 3×3 separable convolutions are used to generate the query $Q \in \mathcal{R}^{h\times w\times c}$, key $K \in \mathcal{R}^{h\times w\times c}$, and value $V \in \mathcal{R}^{h\times w\times c}$, respectively, with the complexity of

$$\mathcal{O}^{\text{qkv}}_{\text{Pale}} = 3(9hwc + hwc^2) = 27hwc + 3hwc^2. \tag{13}$$

Second, we decompose the self-attention within the whole pale region into row-wise and column-wise self-attention. The computational complexity of these two parallel branches are as follows

$$\begin{aligned}\mathcal{O}^{\text{row}}_{\text{Pale}} &= hw^2cs_r, \\ \mathcal{O}^{\text{column}}_{\text{Pale}} &= h^2wcs_c.\end{aligned} \tag{14}$$

Finally, the linear projection layer has the complexity of

$$\mathcal{O}^{\text{proj}}_{\text{Pale}} = hwc^2. \tag{15}$$

Therefore, the overall complexity of our parallel PS-Attention is

$$\begin{aligned}\mathcal{O}_{\text{Pale}} &= \mathcal{O}^{\text{qkv}}_{\text{Pale}} + \mathcal{O}^{\text{row}}_{\text{Pale}} + \mathcal{O}^{\text{column}}_{\text{Pale}} + \mathcal{O}^{\text{proj}}_{\text{Pale}} \\ &= 4hwc^2 + hwc(s_ch + s_rw + 27).\end{aligned} \tag{16}$$

Compared with the global self-attention, our parallel PS-Attention has lower complexity, since $2hw >> (s_ch + s_rw + 27)$ always holds.