# PSLT: A Light-weight Vision Transformer with Ladder Self-Attention and Progressive Shift

Gaojie Wu, Wei-Shi Zheng*, Yutong Lu and Qi Tian

**Abstract**—Vision Transformer (ViT) has shown great potential for various visual tasks due to its ability to model long-range dependency. However, ViT requires a large amount of computing resource to compute the global self-attention. In this work, we propose a ladder self-attention block with multiple branches and a progressive shift mechanism to develop a light-weight transformer backbone that requires less computing resources (e.g. a relatively small number of parameters and FLOPs), termed Progressive Shift Ladder Transformer (PSLT). First, the ladder self-attention block reduces the computational cost by modelling local self-attention in each branch. In the meanwhile, the progressive shift mechanism is proposed to enlarge the receptive field in the ladder self-attention block by modelling diverse local self-attention for each branch and interacting among these branches. Second, the input feature of the ladder self-attention block is split equally along the channel dimension for each branch, which considerably reduces the computational cost in the ladder self-attention block (with nearly $\frac{1}{3}$ the amount of parameters and FLOPs), and the outputs of these branches are then collaborated by a pixel-adaptive fusion. Therefore, the ladder self-attention block with a relatively small number of parameters and FLOPs is capable of modelling long-range interactions. Based on the ladder self-attention block, PSLT performs well on several vision tasks, including image classification, objection detection and person re-identification. On the ImageNet-1k dataset, PSLT achieves a top-1 accuracy of 79.9% with 9.2M parameters and 1.9G FLOPs, which is comparable to several existing models with more than 20M parameters and 4G FLOPs.

**Index Terms**—Multimedia Information Retrieval, Light-weight Vision Transformer, Ladder Self-Attention.

## 1 Introduction

Convolutional neural networks (CNNs) have shown considerable promise as general-purpose backbones for computer vision tasks. Since AlexNet [1] was proposed for the ImageNet image classification challenge [2], CNN architectures have become increasingly powerful, with more careful designs [3], [4], [5], deeper connections [6], [7] and wider dimensions [8]. Thus, CNNs appear to be indispensable for various computer vision tasks. CNNs are successful due to the inductive bias implied in convolutional computations, and this inductive bias ensures that CNNs are generalizable as investigated in [9], [10].

Another prevalent architecture that has shown extraordinary achievements in computer vision tasks is the vision transformer. In the transformer, which was first proposed for sequence modelling and transduction tasks in natural language processing [11], [12], features can interact globally by modelling attention of long-range dependency. Vision Transformer (ViT) models global interaction by computing attention among the feature map according to the projected query, key and value of each pixel. ViT has shown these achievements due to the considerable amount of computing resource in the model. However, models with large number of parameters are difficult to deploy in edge-computing devices with limited memory storage and computing resources, such as FPGAs. And models with large number of FLOPs always require much time for inference, because FLOPs measures the number of float-point operations for inference.

In this work, we aim to develop a light-weight vision transformer backbone with a relatively small number of parameters and FLOPs. Existing methods [13], [14], [15], [16] have focused on reducing the number of float-point operations by evolving the form of computing self-attention; however, the receptive field in the window-based self-attention block is restricted in most of these methods [15], [16], and only pixels divided in the same windows can interact with each other in the window-based self-attention block. Therefore, the interactions between pixels in different windows cannot be modelled in one block.

Thus, we propose a light-weight ladder self-attention block with multiple branches and a progressive shift mechanism is introduced to enlarge the receptive field explicitly. The expansion of the receptive field for the ladder self-attention block is accomplished according to the following strategy. First, diverse local self-attentions are modelled by steering pixels at the same spatial position to model interactions with pixels in diverse windows for different branches. Second, the progressive shift mechanism transmits the output features of the current branch to the subsequent branch. The self-attention in the subsequent branch is computed with the participation of the output features in the current branch, allowing interactions among features in different windows in the two branches. As a result, the ladder self-attention block with the progressive shift mechanism can model long-range interactions among pixels

* *Corresponding author*

- *Gaojie Wu and Yutong Lu are with the School of Computer Science and Engineering, Sun Yat-sen University, Guangzhou 510275, China. E-mail: wugj7@mail2.sysu.edu.cn, yutong.lu@nscc-gz.cn.*
- *Wei-Shi Zheng is with the School of Data and Computer Science, Sun Yat-sen University, Guangzhou 510275, China, with Peng Cheng Laboratory, Shenzhen 518005, China, and also with the Key Laboratory of Machine Intelligence and Advanced Computing (Sun Yat-sen University), Ministry of Education, China. E-mail: wszheng@ieee.org /zhwshi@mail.sysu.edu.cn.*
- *Qi Tian is with the Cloud & AI BU, Huawei, China (tian.qi1@huawei.com).*

divided in different windows. In addition, in the ladder self-attention block, each branch only takes an equal proportion of input channels of the block, which considerably reduces the number of parameters and FLOPs. And all the channels in these branches are aggregated in the proposed pixel-adaptive fusion module to generate the output of the ladder self-attention block. Based on the above designs, we developed a light-weight general-purpose backbone with a relatively small number of parameters.

The overall framework of our model is shown in Figure 1. According to the conclusion of [9], [10] that the convolution in the early stages helps the model learn better and faster. PSLT adopts light-weight convolutional blocks in the early stages and the ladder self-attention blocks in the latter stages. Our PSLT has several important characteristics:

1) PSLT uses light-weight ladder self-attention blocks, which greatly reduce the number of trainable parameters and FLOPs. The ladder self-attention block first divides the input feature map into several equal proportions along the channel axis. Then, each part of the feature map is sent to an individual branch to compute the self-attention similarity.
2) The ladder-self attention block is designed to acquire large receptive field, requiring relatively small number of computing resources. PSLT models local attention on each branch for computing efficiency; and more importantly, without introducing extra parameters, PSLT adopts the progressive shift mechanism to model interactions among pixels in different windows to enlarge the receptive field of each ladder self-attention block. The progressive shift mechanism forms a block shaped like a ladder.

Our proposed PSLT achieves excellent performance on visual tasks such as image classification, object detection and person re-identification with a relatively small number of trainable parameters. With less than 10 million parameters and 2G FLOPs, PSLT achieves a top-1 accuracy of 79.9% on ImageNet-1k image classification at input resolution $224 \times 224$ without pretraining on extra large dataset.

We notice that recently there exist light-weight transformers [17], [18], [19] that combine local interaction (convolution) and global self-attention in one block. In this work, we provide another perspective, and our PSLT is capable of modelling long-range interaction by effectively incorporating diverse local self-attentions and evolving the self-attention of the latter branches in the ladder self-attention block with a relatively small number of parameters and FLOPs.

In summary, our contributions are as follows:

- We propose a light-weight ladder self-attention block with multiple branches and considerably less parameters. A pixel-adaptive fusion module is developed to aggregate features from multiple branches with adaptive weights along both the spatial and channel dimensions.
- We propose a progressive shift mechanism in the ladder self-attention block to enlarge the receptive field. The progressive mechanism not only allows modelling interaction among pixels in different windows for long-range dependencies, but also reduces the number of parameters necessary to obtain the value for computing self-attention in the following branch.
- We develop a general-purpose backbone (PSLT) with a relatively small number of parameters. PSLT is constructed with light-weight convolutional blocks and the ladder self-attention blocks, improving its generalization ability.

## 2 Related Work

A comparison of various models is shown in Table 1. Our PSLT was manually developed and has a relatively small number of parameters, and PSLT does not utilize a search phase before training. In the following section, we detail some related works.

### 2.1 Convolutional Neural Networks

Since the development of AlexNet [1], CNNs have become the most popular general-purpose backbones for various computer vision tasks. More effective and deeper convolutional neural networks have been proposed to further improve the model capacity for feature representation, such as VGG [6], ResNet [3], DenseNet [7] and ResNext [5]. However, neural networks with large scales are difficult to deploy in edge devices, which have limited memory resources. Thus, various works have focused on designing light-weight neural networks with less trainable parameters.

**- Light-weight CNN.** To decrease the number of trainable parameters, MobileNets [20], [21], [22] substitute the standard convolution operation with a more efficient combination of depthwise and pointwise convolution. ShuffleNet [23] uses group convolution and channel shuffle to further simplify the model. The manual design of neural networks is time consuming, and automatically designing convolutional neural architectures has shown great potential for developing high-performance neural networks [24], [25], [26]. EfficientNet [27] investigates the model width, depth and resolution with a neural architecture search [24], [25], [28].

Although CNNs can effectively model local interactions, adaptation to input data is absent in standard convolution. Usually, the adaptive methods are capable of improving model capacity by the dynamic weights or receptive fields with respective to different input images. RedNet [29] produces dynamic weights for distinct input data. Deformable convolution [30] produces an adaptive location for the convolution kernels. Our proposed backbone (PSLT) inherits the virtue of convolution that implies inductive bias when modelling local interactions, and PSLT takes advantage of self-attention [11] to model long-range dependency and adapt to input data with dynamic weights.

### 2.2 Vision Transformer

Transformers [11], [12] are widely used in natural language processing, and the transformer architecture for computer vision tasks has evolved since ViT [31] was proposed for image classification. ViT has one-stage structure that produces features with only one scale. PiT [39] adopts depthwise convolution [20] to decrease the spatial dimension. CrossViT

TABLE 1
Comparison of various general-purpose backbones. "T" indicates that the model adopts the transformer architecture. "Adaptive" indicates that the model can adapt to the input image. "Light" denotes that the model has a relatively small number of parameters and FLOPs. "NAS" denotes that the model needs another search process before training from scratch.

| Model | Architecture | Adaptive | Light | NAS |
|---|---|---|---|---|
| VGG [6] | CNN | ✗ | ✗ | ✗ |
| ResNet [3] | CNN | ✗ | ✗ | ✗ |
| DenseNet [7] | CNN | ✗ | ✗ | ✗ |
| MobileNet [20] | CNN | ✗ | ✓ | ✗ |
| ShuffleNet [23] | CNN | ✗ | ✓ | ✗ |
| EfficientNet | CNN | ✓ | ✗ | ✓ |
| RedNet [29] | CNN | ✓ | ✗ | ✗ |
| ViT [31] | T | ✓ | ✗ | ✗ |
| PS-ViT [32] | T | ✓ | ✗ | ✗ |
| Swin [15] | T | ✓ | ✗ | ✗ |
| VOLO [33] | T | ✓ | ✗ | ✗ |
| PVT [14] | T | ✓ | ✗ | ✗ |
| MobileViT [19] | T | ✓ | ✓ | ✗ |
| EfficientFormer [34] | T | ✓ | ✓ | ✓ |
| CoATNet | CNN+T | ✓ | ✗ | ✗ |
| CvT [35] | CNN+T | ✓ | ✗ | ✗ |
| CMT [13] | CNN+T | ✓ | ✗ | ✗ |
| Conformer [36] | CNN+T | ✓ | ✗ | ✗ |
| BossNet [26] | CNN+T | ✓ | ✗ | ✓ |
| LeViT [37] | CNN+T | ✓ | ✓ | ✗ |
| Mobile-Former [38] | CNN+T | ✓ | ✓ | ✗ |
| **PSLT (Ours)** | CNN+T | ✓ | ✓ | ✗ |

[40] models multi-scale features via small and large patch embedding sizes. Recent transformer architectures [13], [14], [15], [37], [41] have adopted hierarchical structures like popular CNNs to yield multi-scale features through patch merging to progressively reduce the number of patches. To mitigate the issue that the tokenization in ViT [42] may destruct the object structure, PS-ViT [32] locates discriminative regions by iteratively sampling tokens.

Since self-attention requires a large amount of computing resources to model long-range interactions, recent works [15], [16], [43] have approximated global interaction by modelling local and long-range dependency. SwinTransformer [15] divides the feature map into multiple windows, self-attention is only conducted for interactions among pixels in the same window, and the shift operation is proposed for long-range interactions. VOLO [33] generates the outlook attention matrix for a local window. DAT [44] generates deformable attention similar to deformable convolution [30]. Twins [45] first reduces the size of the feature map and then conducts self-attention and upsamples the feature map to its original size. PVT [14], ResT [46] and CMT [13] maintain the size of the query while reducing the size of the key and value to process the self-attention operation. CSWin [47] adopts the cross-shaped window self-attention mechanism for computing efficiency.

**- CNN + Transformer** The combination of convolution and self-attention has shown great potential in computer vision tasks. CoATNet [9] investigates the best method for allocating the convolutional blocks and transformer blocks in different stages to achieve the optimal model generalization ability and capacity. BossNet [26] uses a neural architecture search method [24] to automatically determine the best method to combine the convolutional block and transformer block. LeViT [37] proposes a hybrid neural network for fast inference image classification. LeViT utilizes convolution with a kernel size of 3 to halve the feature size four times and downsamples during the self-attention operation to reduce the number of float-point operations. CvT [35] introduces depthwise and pointwise convolution to replace the fully connected layer, yielding the query, key and value in the multi-head self-attention mechanism. Conformer [36] uses a dual network structure with convolution and self-attention for enhanced representation learning. CMT [13] also introduces depthwise and pointwise convolution in multi-head self-attention mechanism, and an inverted residual FFN is used in place of a standard FFN for improvement. MPViT [48] combines the convolution and multi-scale self-attention for multi-scale representation.

TABLE 2
Comparison of light-weight vision transformers. "T" indicates that the model adopts the transformer architecture. "Local" indicates the modelling of local interaction. "Global" indicates the modelling of long-range interaction. "GSA" denotes that the model adopts global self-attention. "LSA" denotes that the model adopts local self-attention.

| Model | Architecture | Local | Global |
|---|---|---|---|
| Mobile-Former [38] | CNN+T | CNN | GSA |
| MobileViT [19] | CNN+T | CNN | GSA |
| EdgeViT [17] | CNN+T | CNN | GSA |
| EdgeNext [18] | CNN+T | CNN | GSA |
| **PSLT (Ours)** | CNN+T | LSA | LSA |

**- Light-weight vision transformers.** The achievements of vision transformers rely on the large number of computing resource (parameters and FLOPs). Mobile-Former [38] leverages the advantages of MobileNet for local processing and the advantages of the transformer for global interaction. EdgeViT [17] and EdgeNeXt [18] also combine local interaction (convolution) and global self-attention. MobileViT [19] is also a light-weight general-purpose vision transformer for mobile devices. MobileViT leverages a multi-scale sampler for efficient training. LVT [49] also develops enhanced self-attention for computing efficiency.

In this work, we propose a light-weight ladder self-attention block requiring a relatively small number of parameters and FLOPs. Instead of combining convolution and global self-attention for local and global interactions in a block, our ladder self-attention block enlarges the receptive field by modelling and interacting the diverse local self-attentions, as shown in Table 2. And modelling local self-attention is more computationally efficient than modelling global self-attention. Then we develop a light-weight transformer backbone with the ladder self-attention block in the last two stages and light-weight convolutional blocks in the first two stages, and the allocation of self-attention blocks and convolutional blocks is proved with better performance in CoAtNet [9].

# 3 Progressive Shift Ladder Transformer (PSLT)

## 3.1 Overall Architecture

In this work, we aim to develop a light-weight transformer backbone and expand the receptive field of the basic window-based self-attention block. We propose a ladder self-attention block with multiple branches. Each branch takes an equal proportion of input features and adopts the window-based self-attention, considerably reducing the number of parameters and float-point operations in the ladder self-attention block.

A progressive shift mechanism is formed to enlarge the receptive field of the ladder self-attention block with the following strategy. Each branch shifts the obtained features in different directions and divides the shifted features into multiple windows. In this manner, the output features of each branch can be aggregated from diverse windows. Furthermore, the latter branch takes the output feature of the previous branch as input for computing self-attention, allowing pixels in different windows of various branches to interact with one another. With the progressive shift mechanism, the light-weight ladder self-attention block is capable of modelling long-range interactions. Because the output feature of each branch is integrated with pixels in different spatial windows, a pixel-adaptive fusion module is developed to effectively integrate the output features of all the branches with adaptive weights along both the spatial and channel dimensions.

Based on the above considerations, we finally develop a light-weight general-purpose backbone with a relatively small number of trainable parameters based on the proposed ladder self-attention block, termed Progressive Shift Ladder Transformer (PSLT). An overview of PSLT is shown in Figure 1. PSLT leverages the advantages of convolution for local interaction and self-attention for long-range interaction. An input image is passed through a stem convolution layer and four stages consisting of convolutional blocks or the proposed ladder self-attention blocks. The output feature of the final stage is sent to the global average pooling layer and classifier.

To ensure that PSLT is applicable to various computer vision tasks, we adopt a four-stage architecture following the Swin Transformer [15] to yield the hierarchical features. To improve the model generalization ability and capacity, PSLT adopts light-weight convolutional blocks in MobileNetV2 [21] with a squeeze-and-excitation (SE) block [50] in the first two stages and ladder self-attention in the final two stages. In the final two stages, the proposed light-weight ladder self-attention blocks are applied to model long-range interactions. Note that the input feature is downsampled at the beginning of each stage, and these stages jointly yield multi-scale features similar to prevalent convolutional architectures. With these hierarchical representations, PSLT can be easily applied as the backbone model in existing frameworks for various computer vision tasks. For image classification, the output feature of the final stage, with abstract semantics representing the global information, is transmitted to the global average pooling layer and classifier.

In addition, in contrast to splitting the input RGB image into non-overlapping patches (Patch Partition) for the initial representation, as is performed in most commonly used vision transformer architectures, PSLT applies the popular stem with three convolution layers for feature extraction, as splitting the image into non overlapping patches may divide the same part of an object into different patches. The $3 \times 3$ convolution in the first process is capable of modelling local interactions with the implied inductive bias.

Different from existing light-weight transformer [38], which greatly reduces the number of parameters in backbones and introduces multiple parameters in the classification head, PSLT has less parameters in the classification head. Our experiments show that PSLT uses only 6% (0.57 M of 9.2 M) of the parameters in the classification head, which is considerably lower than Mobile-Former-294M [38] using 40% (4.6 M of 11.4 M) of the parameters in the classification head of the two fully connected layers.

## 3.2 Ladder Self-Attention Block

To model interaction among pixels in different windows, we propose a ladder self-attention block with multiple branches and model long-range interaction among pixels in different branches through the progressive shift mechanism. Since the feature maps are integrated in diverse spatial windows in each branch of the ladder self-attention block, a pixel-adaptive fusion module is developed for effective feature fusion among these branches. The difference of processing feature maps in these branches allows the proposed block to obtain diverse information, and the proposed pixel-adaptive fusion module is capable of efficiently aggregating the diverse information. In the following section, we describe the details of the ladder self-attention block.

### *3.2.1 Progressive Shift for the Ladder Self-Attention Block*

The ladder self-attention block divides the input feature map into several equal proportions along the channel dimension and sends these features to multiple branches. Take the ladder self-attention block in Figure 1 for example; the input feature map with $d$ channels is divided into three parts, each with $\frac{d}{3}$ channels for each branch. PSLT adopts window-based self-attention to only compute similarity among pixels in the same windows, thus ensuring computing efficiency. Different from SwinTransformer [15] that stacks even number of window-based self-attention blocks to enlarge the receptive field with shift operation, our PSLT adopts the progressive shift mechanism to model diverse local interactions in each branch and to interact information among these branches, which explicitly enlarges the receptive field of each ladder self-attention block. In this manner, PSLT acquires large receptive field with a relatively small number of parameters and FLOPs.

For long-range interaction, PSLT proposes a progressive shift mechanism for the ladder self-attention block. First, the input features in the first branch are processed by the PSA with the W-MHSA without the shift operation, as shown in Figure 3 (a). And the features of the other two branches are shifted differently to model diverse local interactions, namely pixels from two windows in one branch may be divided in the same window in the other branches. The

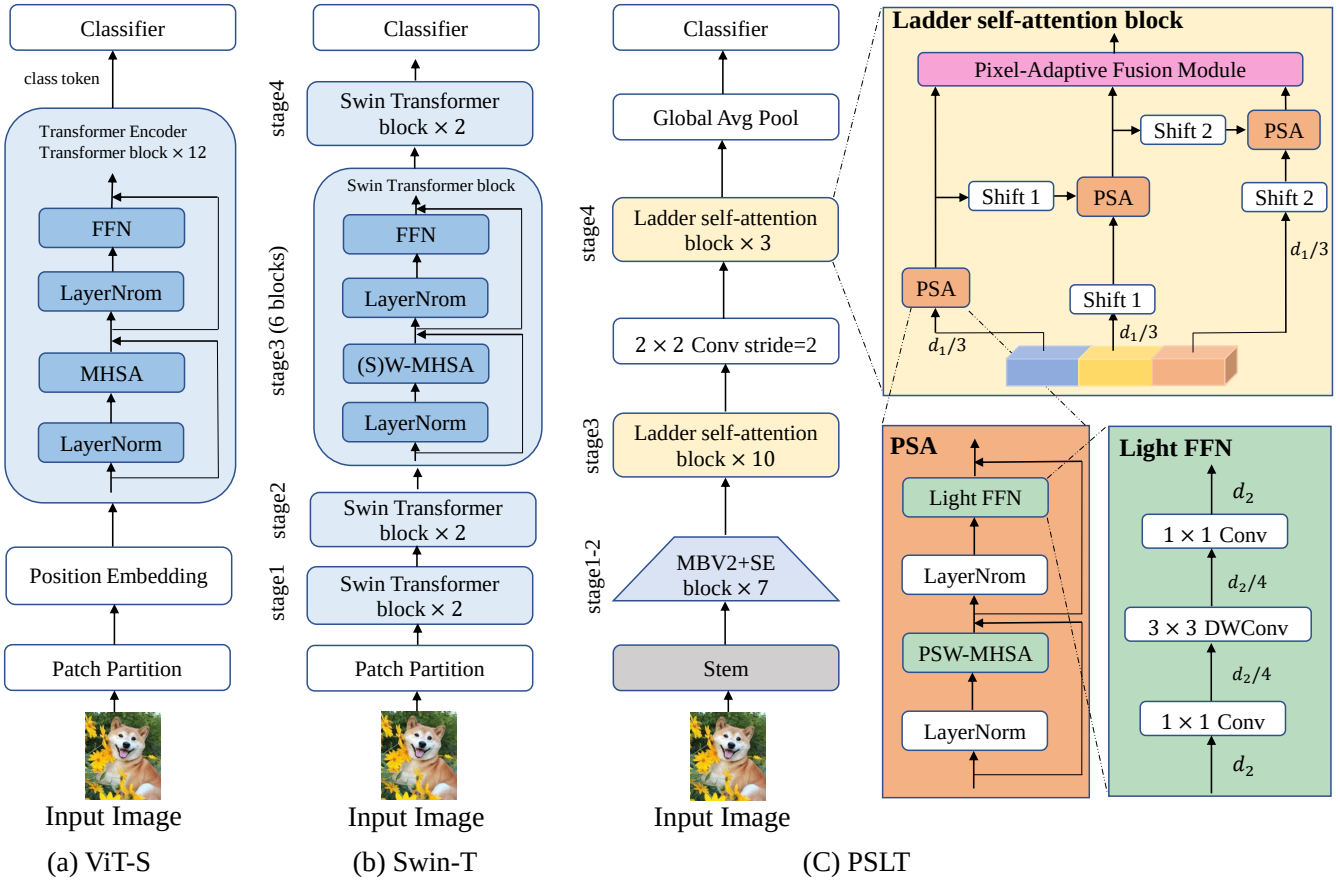



Fig. 1. Illustration of prevalent transformer architectures for image classification. (a) ViT [31] with only one stage. (b) Swin Transformer [15], which uses window-based multi-head self-attention and multi-stage blocks. (c) Our proposed PSLT adopts light-weight ladder self-attention (SA) blocks in the final two stages to model long-range dependency of pixels divided in different windows in the same block, and the light-weight convolutional blocks (MobileNetV2 [21] block with a squeeze-and-excitation (SE) block [50]) are adopted in the first two stages. The details of PSLT are described in Section 3, and the structures of the PSW-MHSA and pixel-adaptive fusion module are shown in Figure 3. "Shift 1" and "Shift 2" denotes shift operations with different directions. "$d_1$" and "$d_2$" denote the number of channels, and $d_1 = 3d_2$.

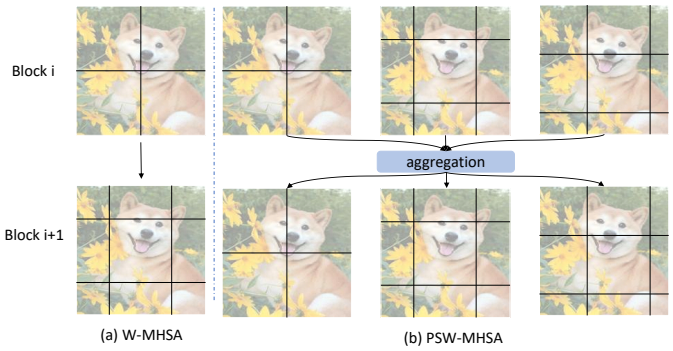
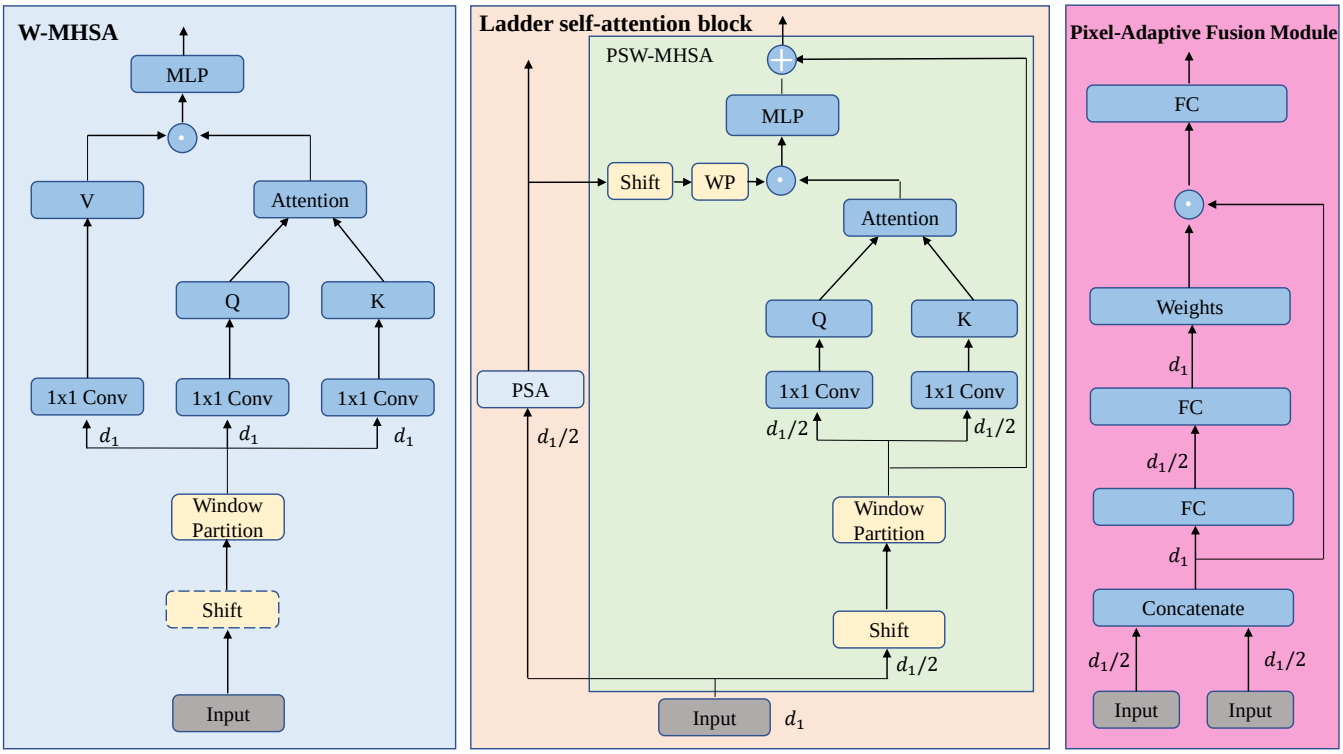



Fig. 2. Illustration of the window partition for pixel interactions in each block. (a) The original W-MHSA generates windows according to only one strategy to model local attention in each block. (b) The PSW-MHSA produces different windows with multiple strategies and fuses these features to aggregate information from diverse spatial windows.

shift direction or stride of the two branches is different from each other to model long-range interactions. An illustration of modelling interaction in different windows is shown in Figure 2. Second, the progressive shift manner transmits the output feature of the current branch to the subsequent branch. As shown in Figure 3 (b), the PSW-MHSA takes both the output feature of the previous branch and the current input feature as input for self-attention computation to model interactions among the branches.

More specifically, the PSW-MHSA applies the same shift and window partition operations to both the input feature of the current branch and the output feature of the previous branch. A $1 \times 1$ convolution is applied to the input feature to yield the query and key to compute the similarity among the feature points. Instead of yielding the value in the same manner, the PSW-MHSA takes the output of the previous branch as the value directly for the self-attention computation. The PSW-MHSA progressively delivers the output features of the branches, allowing it to model long-range dependency because after the local interactions are modelled, the pixels constrained in the divided windows of the current branch can interact with pixels outside the window in the following branches.

Instead of consecutive connection of local self-attention and shifted local self-attention for long-range interaction [15], our PSW-MHSA enables building backbone with arbitrary number of ladder self-attention blocks instead of even number of blocks. In this way, the ladder self-attention block can model long-range interactions among pixels in different windows with the help of multiple branches, and the train-




(a) W-MHSA (b) PSW-MHSA (c) Pixel-Adaptive Fusion Module

Fig. 3. Example of detailed blocks in PSLT. (a) Window-based multi-head self-attention in the Swin Transformer [15]. (b) The ladder self-attention block with two branches for example. The progressive shift mechanism transmits output feature of the previous branch to the subsequent branch, which further enlarges the receptive field. Only the PSW-MHSA is illustrated in detail; LayerNrom and Light FFN are shown in Figure 1 and omitted here for simplicity. "WP" indicates the window partition. (c) The pixel-adaptive fusion module in the ladder self-attention block. "Weights" denotes the adaptive weights along the channel and spatial dimensions, which is the output of the second fully connected layer. The details of the PSW-MHSA are described in Section 3.2.1.

ing cost (number of trainable parameters) is decreased .

Formally, the ladder self-attention blocks are computed as

$$\begin{aligned} \hat{O}_t &= \text{PSW-MHSA}(I_t, O_{t-1}) \\ \text{PSW-MHSA}(I_t, O_{t-1}) &= \text{softmax}(\frac{Q_{I_t} K_{I_t}}{\sqrt{d}}) O_{t-1} + I_t \\ O_t &= \text{LFFN}(\text{LN}(\hat{O}_t)) \\ \bar{O} &= \text{PAFM}(O_t), t = 0, 1, 2, .... \end{aligned} \tag{1}$$

The output ($\hat{O}_t$) of the PSW-MHSA in the $t$-th branch is computed according to the input feature of the $t$-th branch ($I_t$) and the output feature of the $(t-1)$-th branch ($O_{t-1}$). In the PSW-MHSA, the query ($Q_{I_t}$) and the key ($K_{I_t}$) are projected from $I_t$, and $O_{t-1}$ is directly applied as the value. The first branch is computed as $O_0 = \text{MHSA}(I_0)$, as shown in Figure 3 (a). Then, the Light FFN (LFFN) and layer norm (LN) are applied to produce the output of the $t$-th branch ($O_t$). Finally, a pixel-adaptive fusion module (PAFM) is developed to generate the output of the ladder self-attention block ($\bar{O}$) according to the outputs of all branches.

**- Complexity analysis.** Our intention is to build a light-weight backbone, and the PSW-MHSA is proposed for modelling long-range dependency. Here, we roughly compare the number of trainable parameters and computation complexity of the W-MHSA and PSW-MHSA. Taking the input of $h \times w \times d_1$ as an example, each window has $M \times M$ non-overlapping patches, and $B$ branches are contained in the PSW-MHSA. Then, the number of trainable parameters in the W-MHSA and PSW-MHSA are:

$$\begin{aligned} \phi(\text{W-MHSA}) &= 4d_1^2 \\ \phi(\text{PSW-MHSA}) &= \frac{3d_1^2}{B} + \frac{d_1^2}{B^2}. \end{aligned} \tag{2}$$

The computational complexity of the W-MHSA and PSW-MHSA are:

$$\begin{aligned} \Omega(\text{W-MHSA}) &= 4hwd_1^2 + 2M^2hwd_1 \\ \Omega(\text{PSW-MHSA}) &= \frac{3hwd_1^2}{B} + \frac{hwd_1^2}{B^2} + 2M^2hwd_1. \end{aligned} \tag{3}$$

As can be seen, when $B = 1$, the PSW-MHSA has the same number of trainable parameters and complexity as the W-MHSA. When $B > 1$, the number of trainable parameters and the computation complexity are both considerably reduced ($B$ is set to 3 by default).

### 3.2.2 *Light FFN*

As presented in Figure 1 (b), the output of the MHSA is processed by the FFN. To further decrease the number of trainable parameters and float-point operations in PSLT, a Light FFN is proposed to replace the original FFN, as shown in Figure 1 (c). In contrast to the original FFN, which uses

a fully connected layer with $d$ channels for both the input and output features, the Light FNN first projects the input with $d_2$ channels to a narrower feature with $\frac{d_2}{4}$ channels. Then, a depthwise convolution is applied to model local interactions, and a pointwise convolution is adopted to restore the channels.

**- Complexity analysis.** Taking the input of an FFN layer of size $h \times w \times d_2$ as an example, the number of trainable parameters of the FFN and LFFN are:

$$\begin{aligned} \phi(\text{FFN}) &= d^2 \\ \phi(\text{LFFN}) &= \frac{d_2^2}{2} + \frac{9d_2}{4}. \end{aligned} \tag{4}$$

The computational complexity of the FFN and LFFN are:

$$\begin{aligned} \Omega(\text{FFN}) &= hwd_2^2 \\ \Omega(\text{LFFN}) &= \frac{hwd_2^2}{2} + \frac{9hwd_2}{16}. \end{aligned} \tag{5}$$

In general, the number of input feature channels $d_2$ is substantially larger than $\frac{9}{2}$; thus, the LFFN is more computationally economical than the FFN, as it has less trainable parameters and float-point operations.

#### 3.2.3 Pixel-Adaptive Fusion Module

PSLT divides the input feature map equally along the channel dimension for the multiple branches to decrease the number of parameters and computation complexity. To model long-range dependency, the progressive shift mechanism in the ladder self-attention block divides the same pixel into different windows in each branch. Because the output features of the branches are produced by integrating feature information in different windows, PSLT proposes a pixel-adaptive fusion module to efficiently aggregate multi-branch features with adaptive weights along the spatial and channel dimensions.

More specifically, the output features of all the branches are concatenated and sent to two fully connected layers to yield the weights for each pixel. The weights indicate the importance of the features from all the branches for each pixel in the feature map. To ensure that the number of trainable parameters is not increased, the first fully connected layer halves the number of channels. The features are then multiplied by the weights and then sent to a fully connected layer for feature fusion along the channel dimension.

Different from the conventional squeeze-and-excitation block [50] measuring only the importance of the channels, we introduce the pixel-adaptive fusion module to integrate the output features of all branches with adaptive weights in both the spatial and channel dimensions for two reasons. First, because the pixels in each branch are divided into different windows, the pixels at various spatial locations contain distinct information. Second, along the channel dimension, the input feature map is split equally for each branch, so the pixel information at the same spatial location is different among the branches. The experimental results show that our proposed pixel-adaptive fusion module is more suitable for feature fusion than the squeeze-and-excitation block [50].

TABLE 3
Specification for PSLT. "conv2d" and "MBV2Block" indicate the standard convolutional operation and the light-weight block in the MobileNetV2 [21] with the squeeze-and-excitation layer [50]. "conv2d↓" and "MBV2Block+SE↓" denote the operation or block for downsampling with stride 2. "FC" denotes the fully connected layer. PSLT has a total of 9.223M parameters according to the table.

| Stage | Input | Block | #Out | Stride | #Params |
|---|---|---|---|---|---|
| Stem | $224^2 \times 3$ | $3 \times 3$ conv2d↓ | 36 | 2 | 0.018M |
| | $112^2 \times 36$ | $3 \times 3$ conv2d $\times 2$ | 36 | 1 | |
| 1 | $112^2 \times 36$ | MBV2Block+SE↓ | 72 | 2 | 0.125M |
| | $56^2 \times 72$ | MBV2Block+SE $\times 2$ | 72 | 1 | |
| 2 | $56^2 \times 72$ | MBV2Block+SE↓ | 144 | 2 | 0.817M |
| | $28^2 \times 144$ | MBV2Block+SE $\times 2$ | 144 | 1 | |
| | $28^2 \times 144$ | MBV2Block+SE↓ | 288 | 2 | |
| 3 | $14^2 \times 288$ | Ladder SA block $\times 10$ | 288 | 1 | 3.879M |
| 4 | $14^2 \times 288$ | $2 \times 2$ conv2d↓ | 576 | 2 | 3.808M |
| | $7^2 \times 576$ | Ladder SA block $\times 3$ | 576 | 1 | |
| head | $7^2 \times 576$ | $7 \times 7$, pool | 576 | 1 | 0.576M |
| | $1^2 \times 576$ | FC | 1000 | 1 | |

### 3.3 Network Specification

Table 3 shows the architecture of PSLT, which contains 9.2M trainable parameters for image classification on the ImageNet dataset [2]. PSLT stacks the light-weight convolutional blocks (MBV2Block+SE) in the first two stages and the proposed ladder self-attention blocks in the final two stages. PSLT starts with three $3 \times 3$ convolution as stem to produce the input feature of the first stage. In both stage 1 and stage 2, the first convolutional block has a stride of 2 to downsample the input feature map. The final block in stage 2 is utilized to downsample the input feature map of stage 3. A $2 \times 2$ convolution is applied to downsample the feature map of stage 4. Each time the size of the feature map is halved, the number of feature channels doubles.

Finally, the classifier head uses global average pooling on the output of the last block. The globally pooled feature passes through a fully connected layer to yield an image classification score. Without the classifier head, PSLT with multi-scale features can easily be applied as a backbone in existing methods for various visual tasks.

**- Detailed configuration.** Following SwinTransformer [15], the window size is set to $7 \times 7$, and the window partition and position embedding is produced with the mechanism similar to [15]. The number of heads in PSW-MHSA is set to 4 and 8 in the last two stages respectively. The stride in the shift operation (shown in Figure 1 (c)) in PSW-MHSA is set to 3. The shift operation is similar to that in [15], but the shift direction is different from each other in the last two branches.

## 4 EXPERIMENT

In this section, we evaluate the effectiveness of the proposed PSLT by conducting experiments on several tasks, including ImageNet-1k image classification [2], COCO objection detection [51] and Market-1501 person re-identification [52].

### 4.1 Image Classification

#### 4.1.1 ImageNet Results

**- Experimental setting.** We evaluate our proposed PSLT on ImageNet [2] classification. The ImageNet dataset has 1,000 classes, including 1.2 million images for training and 50 thousand images for validation. We train our PSLT on the training set, and the top-1 accuracy on the validation set and the ImageNet-V2 [53] is reported. For a fair comparison, we follow Mobile-Former [38] when performing the ImageNet classification experiments. The images were randomly cropped to $224 \times 224$. The data augmentation methods include horizontal flipping [54], mix-up [55], auto-augmentation [56] and random erasing [57]. Our PSLT was trained from scratch on 8 A6000 GPUs with label smoothing [58] using the AdamW [59] optimizer for 450 epochs with a cosine learning rate decay. The distillation is not adopted for training. The initial learning rate was set to $6 \times 10^{-4}$, and the weight decay was set to 0.025 by default. For images in the validation set, we adopted the center crop, with images cropped to $224 \times 224$ for evaluation.

**- Experimental results.** Table 4 shows a comparison of the performance of our proposed PSLT and the performance of several convolution-based models and transformer-based models. The first block shows the performance of our proposed PSLT. The second block in Table 4 presents the performance of models with less than 10M parameters, including light-weight convolutional neural networks and recently proposed transformer-based models. Compared to an efficient convolution-based model with a similar number of parameters (ShuffleNetV2 + WeightNet [61]), PSLT achieves a significantly higher top-1 accuracy (79.9% vs. 75.0%) with slightly fewer parameters (9.2M vs. 9.6M) but more FLOPs (1.9G vs. 307M). With a similar number of parameters (9.2M vs. 9.5M) and FLOPs (1.9G vs. 2.0G), PSLT achieves a top-1 accuracy that is 2% higher than the pure transformer-based DeiT-2G (79.9% vs. 77.6%). Several recent works have focused on leveraging the advantages of CNNs and transformer-based architectures to improve model generalization ability and capacity, including LeViT [37] and CMT [13]. It should be noted that our proposed PSLT (trained for 450 epochs without distillation) achieves a higher top-1 accuracy (79.9% vs. 78.6%) than LeViT (trained for 1,000 epochs with distillation) with a similar number of parameters. With a similar number of parameters (4.3M vs. 4.2M) and FLOPs (876M vs. 0.6G), PSLT-Tiny achieves a slightly higher top-1 accuracy (74.9% vs. 74.4%) than EdgeViT-XXS [17]. And PSLT achieves comparable performance to MPViT-T [41] with a similar number of FLOPs.

Although our PSLT shows slightly inferior performance compared to the EfficientNet (79.9% vs. 80.1%) with similar number of parameters, the EfficientNet is obtained by the carefully designed neural architecture search algorithm, which is time-consuming. And our PSLT achieves higher performance than the EfficientNet on the experiments of COCO for object detection in Table 6 and instance segmentation in Table 7. Furthermore, our PSLT still achieves higher performance than EfficientFormer-L1 which automatically finds efficient transformers with similar neural architecture search algorithm to EfficientNet.

Compared to those recently developed lightweight networks [17], [18], [19], [37], [48], our PSLT provides a different perspective for acting as an effective backbone with a relatively small number of parameters and FLOPs. The above experimental results demonstrate that PSLT with diverse local self-attentions for long-range dependency is capable of achieving comparable performance, and modelling the local self-attention requires less computing resources than modelling the global self-attention.

The third block in Table 4 shows the performance of models with 10M $\sim$ 20M parameters. PSLT outperforms pure transformer-based architectures with a similar number of parameters and FLOPs (9.2M/1.9G), including PvT-Tiny (13.2M/1.9G) and DeiT-2G (6.5M/2.0G), and PSLT achieves a slightly higher top-1 accuracy than Swin-2G (79.9% vs. 79.2%) with slightly fewer parameters. PSLT slightly outperforms CNN-based models and transformer-based models, where PSLT achieves a higher top-1 accuracy than Mobile-Former (79.9% vs. 79.3%) with significantly fewer parameters (9.2M vs. 14.0M) and more FLOPs (1.9G vs. 508M), and PSLT achieves a top-1 accuracy that is 3% higher than ConT-S (79.9% vs. 76.5%) with a similar number of parameters and FLOPs. PSLT also achieves comparable performance to the searched light-weight model EfficientFormer-L1 [34] (79.9% vs. 79.2%), where PSLT requires a relatively smaller number of parameters (9.2M vs. 12.2M) and FLOPs (1.9G vs. 2.4G).

The last block in Table 4 shows the performance of models with more than 20M parameters for reference, including the commonly used ResNet and its evolved architectures. Compared to the pure transformer models, PSLT achieves a comparable top-1 accuracy to PVT-S (79.9% vs. 79.8%) with nearly 50% fewer parameters (9.2M vs. 24.5M) and FLOPs (1.9G vs. 3.8G). PSLT-Large with smaller amount of parameters and FLOPs (16.0M/3.4G) also outperforms recent transformer structures, including DeiT-S (22.0M/4.6G), PS-ViT-B/10 (21.3M/3.1G) and BossNet-T0 (3.4G). And the PSLT-Large also achieves a slightly higher top-1 accuracy than Swin [15] (81.5% vs. 81.3%) with fewer parameters (16.0M vs. 29.0M) and FLOPs (3.4G vs. 4.5G).

The ImageNet classification experiments show that our proposed PSLT outperforms both convolution-based and transformer-based models with similar number of parameters. These results validate the effectiveness of the proposed ladder self-attention block.

#### 4.1.2 CIFAR Results

**- Experimental setting.** The CIFAR-10/100 [70] datasets contain 10/100 classes, including 50,000 images for training and 10,000 images for testing, with an image resolution of $32 \times 32$. We trained PSLT from scratch on the training set, and the top-1 accuracy on the test set is reported. PSLT was trained for 300 epochs using the AdamW [59] optimizer with cosine learning rate decay using an initial learning rate of $5 \times 10^{-4}$ and a weight decay of 0.025.

**- Experimental results.** Table 5 shows the image classification performance of the vision transformer models on

TABLE 4
Image classification performance on the ImageNet without pretraining. Models with similar number of parameters are presented for comparison. "Input" indicates the scale of the input images. "Top-1" ("V2 Top-1") denotes the top-1 accuracy on ImageNet (ImageNet-V2). "#Params" refers to the number of trainable parameters. "#FLOPs" is calculated according to the corresponding input size. "+" indicates that the performance is cited from Mobile-Former [38]. "*" indicates that the model was trained with distilling from an external teacher. "‡" denotes extra techniques are applied, such as multi-scale sampler [19] and EMA [2]. "PAcc" ("FAcc") is the ratio of Top-1 accuracy to the number of parameters (FLOPs).

| Model | NAS | Input | #Params | #FLOPs | Top-1 | PAcc (%/M) | FAcc (%/G) | V2 top-1 |
|---|---|---|---|---|---|---|---|---|
| **PSLT-Tiny(Ours)** | ✗ | $224^2$ | 4.3M | 876M | 74.9% | 17.42 | 85.5 | 63.0% |
| **PSLT(Ours)** | ✗ | $224^2$ | 9.2M | 1.9G | 79.9% | 8.68 | 42.05 | 68.6% |
| **PSLT-Large(Ours)** | ✗ | $224^2$ | 16.0M | 3.4G | 81.5% | 5.1 | 24.0 | 70.3% |
| MobileNetV3 [22] | ✓ | $224^2$ | 4.0M | 155M | 73.3% | 18.33 | 472.9 | – |
| EdgeViT-XXS [17] | ✗ | $224^2$ | 4.1M | 0.6G | 74.4% | 18.14 | 124 | – |
| Mobile-Former [38] | ✗ | $224^2$ | 4.6M | 96M | 72.8% | 15.83 | 758.3 | – |
| MobileViTv2-S [19]‡ | ✗ | $256^2$ | 4.9M | 1.8G | 78.1% | 15.94 | 43.39 | – |
| LVT [49] | ✗ | $224^2$ | 5.5M | 0.9G | 74.8% | 13.6 | 83.11 | – |
| MobileViT-S [19]‡ | ✗ | $256^2$ | 5.6M | 2.0G | 78.4% | 14 | 39.2 | – |
| EdgeNeXt-S [18]‡ | ✗ | $224^2$ | 5.6M | 965M | 78.8% | 14.07 | 81.66 | – |
| MPViT-T [48] | ✗ | $224^2$ | 5.8M | 1.6G | 78.2% | 13.48 | 48.88 | – |
| HRFormer-T [60] | ✗ | $224^2$ | 8.0M | 1.8G | 78.5% | 9.81 | 43.61 | – |
| LeViT [37]* | ✗ | $224^2$ | 9.2M | 406M | 78.6% | 8.54 | 193.6 | 66.6% |
| EfficientNet [27] | ✓ | $224^2$ | 9.2M | 1.0G | 80.1% | 8.71 | 80.1 | 68.8% |
| RedNet-26 [29] | ✗ | $224^2$ | 9.2M | 1.7G | 75.9% | 8.25 | 44.65 | – |
| CMT-Ti [13] | ✗ | $192^2$ | 9.5M | 0.6G | 79.2% | 8.34 | 132 | – |
| ShuffleV2+Weight [61] | ✗ | $224^2$ | 9.6M | 307M | 75.0% | 7.81 | 244.3 | – |
| ConT-S [62] | ✗ | $224^2$ | 10.1M | 1.5G | 76.5% | 7.57 | 51 | – |
| Coat-Lite mini [63] | ✗ | $224^2$ | 11.0M | 2.0G | 79.1% | 7.19 | 39.55 | – |
| Shunted-T [64] | ✗ | $224^2$ | 11.5M | 2.1G | 79.8% | 6.94 | 38 | – |
| GC ViT-XXT [65] | ✗ | $224^2$ | 12.0M | 2.1G | 79.6% | 6.63 | 37.9 | – |
| PoolFormer-S12 [66] | ✗ | $224^2$ | 12.0M | $\sim$ 4.2G | 77.2% | 6.43 | 18.38 | – |
| EfficientFormer-L1 [34] | ✓ | $224^2$ | 12.2M | 2.4G | 79.2% | 6.49 | 33 | – |
| Swin-2G [15]+ | ✗ | $224^2$ | 12.8M | 2.0G | 79.2% | 6.19 | 39.6 | – |
| PvT-Tiny [14] | ✗ | $224^2$ | 13.2M | 1.9G | 75.1% | 5.69 | 39.53 | – |
| ResT-Small [46] | ✗ | $224^2$ | 13.7M | 1.9G | 79.6% | 5.81 | 41.89 | – |
| Mobile-Former [38] | ✗ | $224^2$ | 14.0M | 508M | 79.3% | 5.66 | 156.1 | – |
| HRNet-W18 [67] | ✗ | $224^2$ | 21.3M | 4.0G | 76.8% | 3.56 | 19.2 | – |
| PS-ViT-B/10 [32] | ✗ | $224^2$ | 21.3M | 3.1G | 80.6% | 3.78 | 26 | – |
| A-ViT-S [68] | ✗ | $224^2$ | 22.0M | 3.6G | 78.6% | 3.57 | 21.83 | – |
| DeiT-S [69] | ✗ | $224^2$ | 22.0M | 4.6G | 79.8% | 3.63 | 17.35 | 68.5% |
| BossNet-T0 [26] | ✓ | $224^2$ | - - | 3.4G | 80.8% | – | 23.76 | – |
| PVT-S [14] | ✗ | $224^2$ | 24.5M | 3.8G | 79.8% | 3.24 | 21 | – |
| Res2Net-50 [4] | ✗ | $224^2$ | 25.0M | 4.2G | 78.0% | 3.12 | 18.57 | – |
| ResNet-50 [3] | ✗ | $224^2$ | 25.6M | 4.1G | 76.2% | 2.98 | 18.59 | – |
| RedNet-101 [29] | ✗ | $224^2$ | 25.6M | 4.7G | 79.1% | 3.09 | 16.83 | – |
| VOLO [33] | ✗ | $224^2$ | 27.0M | 6.8G | 84.2% | 3.12 | 12.38 | – |
| CrossFormer-T [16] | ✗ | $224^2$ | 27.8M | 2.9G | 81.5% | 2.93 | 28.1 | – |
| Swin [15] | ✗ | $224^2$ | 29.0M | 4.5G | 81.3% | 2.80 | 18.07 | – |
| WRN [8] | ✗ | $224^2$ | 68.9M | – | 78.1% | 1.13 | – | – |
| ViT-B/16 [31] | ✗ | $384^2$ | 86M | 55.5G | 77.9% | 0.91 | 1.4 | 67.5% |

the CIFAR-10 and CIFAR-100 datasets. As shown in the table, our PSLT achieves comparable performance compared to the commonly-used convolutional neural networks with a small amount of parameters. Compared with WRN [8], PSLT achieves comparable performance on the CIFAR10/100 with similar amount of parameters but higher top-1 accuracy on ImageNet with much less parameters in Table 4. Compared to the vision transformers, our PSLT shows higher model generalization ability with fewer training images, where PSLT achieves relatively higher top-1 accuracy on both CIFAR-10 (95.43% vs. 95.0%) and CIFAR-100 (79.1% vs. 78.63%) than MOA-T with less trainable parameters (8.7M vs. 30M).

### 4.2 Objection Detection

**- Experimental setting.** We conduct objection detection experiments on COCO 2017 [51], which contains 118k images in the training set and 5k images in the validation set of 80 classes. Following PVT [14], which uses standard settings to generate multi-scale feature maps, we evaluate

TABLE 5
Image classification performance of the models on the CIFAR-10/100 without pretraining. The top-1 accuracy rate on CIFAR-10/100 is reported. "+" denotes that results were taken from [71]. "FLOPs" is not reported for the compared methods in literatures.

| Model | Input | #Params | CIFAR-10 | CIFAR-100 |
|---|---|---|---|---|
| ResNet [3] | $32^2$ | 19.4M | 92.07% | – |
| WRN-40-4 [8] | $32^2$ | 8.9M | 95.47% | 78.82% |
| ResNeXt-29 [5] | $32^2$ | 68.1M | 96.43% | 82.69% |
| DenseNet [7] | $32^2$ | 27.2M | 94.17% | 76.58% |
| PVT-T [14]$^+$ | $32^2$ | 13M | 91.0% | 72.80% |
| Swin [15]$^+$ | $32^2$ | 27.5M | 94.41% | 78.07% |
| MOA-T [71] | $32^2$ | 30M | 95.0% | 78.63% |
| **PSLT (Ours)** | $32^2$ | 8.7M | 95.43% | 79.1% |

our proposed PSLT as a backbone with two standard detectors: RetinaNet [73] (one-stage) and Mask R-CNN [74] (two-stage). During training, our backbone (PSLT) was first initialized with the pretrained weights on ImageNet [2], and the newly added layers were initialized with Xavier [75]. Our PSLT was trained with a batch size of 16 on 8 GeForce 1080Ti GPUs. We adopt the AdamW [59] optimizer with an initial learning rate of $1 \times 10^{-4}$. Following the common PVT settings, we adopted a $1\times$ or $3\times$ training schedule (12 or 36 epochs) to train the detection models. The shorter side of the training images was resized to 800 pixels, while the longer side of the images is set to a maximum of 1,333 pixels. The shorter side of the validation images was fixed to 800 pixels. With the $3\times$ training schedule, the shorter side of the input image was randomly resized in the range of [640, 800].

**- Experimental results.** Table 6 shows the performance of the backbones on COCO val2017; the backbones were initialized with pretrained weights on ImageNet using RetinaNet for object detection. As shown in the table, with a similar number of trainable parameters, PSLT achieves comparable performance. With the $1\times$ training scheduler, PSLT outperforms Mobile-Former, where PSLT achieving a higher AP (41.2 vs. 38.0) with slightly more parameters (19.1 M vs. 17.9 M) and FLOPs (192 G vs. 181G). The last block of Table 6 presents models with more than 30M parameters; PSLT achieves higher AP (41.2 vs. 40.4) than PVT-S with considerably fewer parameters (19.1 M vs. 34.2M) and FLOPs (192 G vs. 226 G). Compared with the commonly used backbone ResNet-50, PSLT achieves a considerably higher AP (41.2 vs. 35.3) with nearly 50% less trainable parameters (19.1 M vs. 37.7 M) and fewer FLOPs (192 G vs. 239 G). With the $3\times$ training scheduler, PSLT shows higher performance with a similar number of parameters, where PSLT achieving a 1.7-point higher AP than PVT-S (43.9 vs. 42.2) with considerably fewer parameters.

Similar results are shown in Table 7, including segmentation experiments on Mask R-CNN. With the $1\times$ training scheduler, PSLT outperforms ResNet-50, where PSLT achieving 2.8 points higher box AP (40.8 vs. 38.0) and 2.9 points higher mask AP (37.3 vs. 34.4) with fewer parameters (28.9M vs. 44.2M) and FLOPs (211 G vs. 253 G). PSLT shows comparable performance to PVT-S (40.8 vs. 40.4 box AP and 37.3 vs. 37.8 mask AP) with considerably fewer parameters (28.9M vs. 44.1M) and FLOPs (211G vs. 305G). Similar results are observed with the $3\times$ training scheduler; PSLT outperforms PVT-T, where PSLT achieving 2.9 points higher $AP^b$ and 1.5 points higher $AP^m$ with a smaller number of parameters (28.9M vs. 32.9M) and FLOPs (211 G vs. 240 G). Moreover, PSLT achieves comparable performance to ResNet-50, which has more than 40M parameters.

For a fair comparison with Swin [15], we implement our PSLT-Large with the same experimental setting as Swin [15] with Sparse R-CNN [72] framework (in Table 6) and with Mask R-CNN [74] framework (in Table 7). Our PSLT-Large achieves comparable performance with less parameters and FLOPs.

### 4.3 Semantic Segmentation

**- Experimental setting.** Semantic segmentation experiments are conducted with a challenging scene parsing dataset, ADE20K [76]. ADE20K contains 150 fine-grained semantic categories with 20210, 2000 and 3352 images in the training, validation and test sets, respectively. Following PVT [14], we evaluate our proposed PSLT as a backbone on the basis of the Semantic FPN [77], a simple segmentation method without dilated convolution [78]. In the training phase, the backbone was first initialized with the pretrained weights on ImageNet [2], and the newly added layers were initialized with Xavier [75]. Our PSLT was trained for 80k iterations with a batch size of 16 on 4 GeForce 1080Ti GPUs. We adopted the AdamW [59] optimizer with an initial learning rate of $2 \times 10^{-4}$. The learning rate follows the polynomial decay rate with a power of 0.9. The training images were randomly resized and cropped to 512 $\times$ 512, and the images in the validation set were rescaled, with the shorter side set to 512 pixels during testing.
**- Experimental results.** Table 8 shows the semantic segmentation performance on the ADE20K validation set for backbones initialized with pretrained weights on ImageNet with Sematic FPN [77]. As shown in the table, PSLT outperforms the ResNet-based models [3], where PSLT achieving a 2.3% points higher than ResNet-50 (39.0% vs. 36.7%) with nearly half the number of parameters (13.0M vs. 28.5M) and 20% less FLOPs than ResNet-50 (32.0G vs. 45.6G). With almost the same number parameters and FLOPs, PSLT achieves a 2.3% points higher than PVT-T (39.0% vs 36.7%).

For a fair comparison with Swin [15], we implement our PSLT-Large with the same experimental setting as Swin [15] with UperNet [79] framework. Our PSLT-Large achieves comparable performance with less parameters and FLOPs.

### 4.4 Person Re-identification

**- Experimental setting.** Person re-identification (ReID) experiments are conducted with Market-1501 [52], a dataset that contains 12,936 images of 751 people in the training set and 19,732 images of 750 different people in the validation set. All the images in the Market-1501 dataset were captured with six cameras. We trained our PSLT on the training set to classify each person, similar to an image classifier with 751 training classes. The output feature of the final stage in PSLT was utilized to compute the similarity between the query image and the gallery images for validation, and the rank-1 accuracy and mean average precision (mAP) are

TABLE 6
Object detection performance on COCO. The $AP$ value on the validation set is reported. "MS" denotes that multi-scale training [14] was used. "+" indicates that the result is taken from [14]. FLOPs is calculated at input resolution $1280 \times 800$. "*" denotes that the model is trained based on Sparse R-CNN [72] following [15].

| Backbone | #Params | #FLOPs | RetinaNet 1x | | | | | | RetinaNet 3x + MS | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | $AP$ | $AP_{50}$ | $AP_{75}$ | $AP_S$ | $AP_M$ | $AP_L$ | $AP$ | $AP_{50}$ | $AP_{75}$ | $AP_S$ | $AP_M$ | $AP_L$ |
| **PSLT(Ours)** | 19.1M | 192G | 41.2 | 61.3 | 44.0 | 25.5 | 45.0 | 54.8 | 43.9 | 64.3 | 46.9 | 27.6 | 47.6 | 56.9 |
| Mobile-Former [38] | 17.9M | 181G | 38.0 | 58.3 | 40.3 | 22.9 | 41.2 | 49.7 | – | – | – | – | – | – |
| EfficientNet [27] | 20.0M | 151.2G | 40.5 | 60.5 | 43.1 | 22.2 | 44.8 | 56.6 | 42.4 | 62.0 | 45.6 | 24.1 | 46.1 | 58.3 |
| ResNet-18 [3]$^+$ | 21.3M | 168G | 31.8 | 49.6 | 33.6 | 16.3 | 34.2 | 43.2 | 35.4 | 53.9 | 37.6 | 19.5 | 38.2 | 46.8 |
| PVT-T [14] | 23.0M | 221G | 36.7 | 56.9 | 38.9 | 22.6 | 38.8 | 50.0 | 39.4 | 59.8 | 42.0 | 25.5 | 42.0 | 52.1 |
| RedNet-50 [29] | 27.8M | 210G | 38.3 | 58.2 | 40.5 | 21.1 | 41.8 | 50.9 | – | – | – | – | – | – |
| ConT-M [62] | 27.0M | 217G | 37.9 | 58.1 | 40.2 | 23.0 | 40.6 | 50.4 | – | – | – | – | – | – |
| PVT-S [14] | 34.2M | 226G | 40.4 | 61.3 | 43.0 | 25.0 | 42.9 | 55.7 | 42.2 | 62.7 | 45.0 | 26.2 | 45.2 | 57.2 |
| ResNet-50 [3]$^+$ | 37.7M | 239G | 36.3 | 55.3 | 38.6 | 19.3 | 40.0 | 48.8 | 39.0 | 58.4 | 41.8 | 22.4 | 42.8 | 51.6 |
| **PSLT-Large (Ours)*** | 97.6M | 147.7G | – | – | – | – | – | – | 48.2 | 67.2 | 52.8 | 32.3 | 51.1 | 62.4 |
| Swin [15]* | 110.0M | 172.0G | – | – | – | – | – | – | 47.9 | 67.3 | 52.3 | – | – | – |

TABLE 7
Object detection and instance segmentation performance on COCO. $AP^b$ and $AP^m$ denote the bounding box AP and mask AP on the validation set, respectively. "MS" denotes that multi-scale training [14] was used. "+" indicates that the result was taken from [14]. FLOPs is calculated at input resolution $1280 \times 800$.

| Backbone | #Params | #FLOPs | Mask R-CNN 1x | | | | | | Mask R-CNN 3x + MS | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | $AP^b$ | $AP^b_{50}$ | $AP^b_{75}$ | $AP^m$ | $AP^m_{50}$ | $AP^m_{75}$ | $AP^b$ | $AP^b_{50}$ | $AP^b_{75}$ | $AP^m$ | $AP^m_{50}$ | $AP^m_{75}$ |
| **PSLT (Ours)** | 28.9M | 211G | 40.8 | 61.8 | 44.9 | 37.3 | 59.0 | 39.9 | 42.7 | 63.1 | 47.1 | 38.9 | 60.5 | 41.9 |
| EfficientNet [27] | 30.2M | 169.3G | 40.1 | 61.2 | 43.5 | 36.4 | 58.0 | 38.9 | 42.1 | 63.3 | 45.5 | 38.0 | 60.2 | 40.6 |
| RedNet-50 [29] | 34.2M | 224G | 40.2 | 61.4 | 43.7 | 36.1 | 58.1 | 38.2 | – | – | – | – | – | – |
| ResNet-18 [3]$^+$ | 31.2M | – – | 34.0 | 54.0 | 36.7 | 31.2 | 51.0 | 32.7 | 36.9 | 57.1 | 40.0 | 33.6 | 53.9 | 35.7 |
| PVT-T [14] | 32.9M | 240G | 36.7 | 59.2 | 39.3 | 35.1 | 56.7 | 37.3 | 39.8 | 62.2 | 43.0 | 37.4 | 59.8 | 39.9 |
| PVT-S [14] | 44.1M | 305G | 40.4 | 62.9 | 43.8 | 37.8 | 60.1 | 40.3 | 43.0 | 65.3 | 46.9 | 39.9 | 62.5 | 42.8 |
| ResNet-50 [3]$^+$ | 44.2M | 253G | 38.0 | 58.6 | 41.4 | 34.4 | 55.1 | 36.7 | 41.0 | 61.7 | 44.9 | 37.1 | 58.4 | 40.1 |
| **PSLT-Large (Ours)** | 35.6M | 242.3G | 44.1 | 65.9 | 48.3 | 39.6 | 62.6 | 42.4 | 46.7 | 68.0 | 51.5 | 41.8 | 65.0 | 45.2 |
| Swin [15] | 48.0M | 267.0G | 43.7 | 66.6 | 47.7 | 39.8 | 63.3 | 42.7 | 46.0 | 68.1 | 50.3 | 41.6 | 65.1 | 44.9 |

TABLE 8
Semantic segmentation performance of different backbones on the ADE20K. The mean intersection over union (MIoU) is reported. "+" indicates that the result is taken from [14]. "*" denotes that the model is trained based on UperNet [79] following [15].

| Backbone | Semantic FPN | | | |
|---|---|---|---|---|
| | #Params | Input | FLOPs | mIoU |
| **PSLT (Ours)** | 13.0M | $512^2$ | 32.0G | 39.0% |
| ResNet-18 [3]$^+$ | 15.5M | $512^2$ | 32.2G | 32.9% |
| PVT-T [14] | 17.0M | $512^2$ | 33.2G | 35.7% |
| ResNet-50 [3]$^+$ | 28.5M | $512^2$ | 45.6G | 36.7% |
| PVT-S [14] | 28.2M | $512^2$ | 44.5G | 39.8% |
| **PSLT-Large(Ours)*** | 47.8M | $512^2$ | 229.9G | 46.1% |
| Swin [15]* | 60M | $512^2$ | 945G | 46.1% |

TABLE 9
Performance of the models on the Market-1501 when the models are first pretrained on ImageNet. "Rank-1" indicates the rank-1 accuracy rate. "mAP" denotes the mean Average Precision. "FLOPs" is not reported for the compared methods in literatures.

| Model | Input | #Params | Rank-1 | mAP |
|---|---|---|---|---|
| OSNet [81] | $256 \times 128$ | 2.2M | 94.8% | 84.9 |
| CDNet [82] | $256 \times 128$ | 1.8M | 95.1% | 86.0 |
| Auto-ReID [83] | $256 \times 128$ | 11.4M | 94.5% | 85.1 |
| TransReID [84] | $256 \times 128$ | ∼50M | 94.7% | 86.8 |
| PCB (+RPP) [85] | $256 \times 128$ | ∼26M | 93.8% | 81.6 |
| VPM [86] | $256 \times 128$ | ∼26M | 93.0% | 80.8 |
| BagofTrick [80] | $256 \times 128$ | ∼26M | 94.5% | 85.9 |
| **PSLT (Ours)** | $256 \times 128$ | 9.0M | 94.3% | 86.2 |

reported. For a fair comparison, we follow Bag of Tricks [80] to perform the ReID experiments. The data augmentation techniques that we apply to the training images include horizontal flipping, central padding, randomly cropping to $256 \times 128$, normalizing by subtracting the channel mean and dividing by the channel standard deviation, and random erasing [57]. The data augmentation techniques applied to the test images include resizing to $256 \times 128$ and normalizing by subtracting the channel mean and dividing by the channel standard deviation.

The proposed PSLT was trained for 360 epochs under the guidance of the cross-entropy loss and triplet loss [87] using the Adam optimizer. Similarly, PSLT adopts the BNNeck [80] structure to compute the triplet loss. The output feature after the global average pooling layer first passed through a BatchNorm layer before being sent to the classifier. The feature before the BatchNorm layer was used to compute the triplet loss, and the output score of the classifier was

TABLE 10
Comparison of model inference. "Mem" denotes the peak memory for evaluation. "FPS" is the number of images processed for one second.

| Model | #Params | #FLOPs | Mem | FPS | Top-1 |
|---|---|---|---|---|---|
| **PSLT-Tiny** | 4.3M | 876M | 1.9G | 669.2 | 74.9% |
| **PSLT** | 9.2M | 1.9G | 2.1G | 638.2 | 79.9% |
| **PSLT-Large** | 16.0M | 3.4G | 2.3G | 557.7 | 81.5% |
| DeiT-S [69] | 22.0M | 4.6G | 1.8G | 1456.9 | 79.8% |
| Swin [15] | 29.0M | 4.5G | 2.5G | 755.2 | 81.3% |
| Swin$^{\times 3}$ [15] | 247.2M | 38.1G | 4.8G | 272.4 | – |

used to compute the cross-entropy loss for training. For validation, the feature after the BatchNorm layer was adopted to compute the similarity. A batch size of 64 was applied during training. The initial learning rate is set to $1.5 \times 10^{-4}$, and the learning rate was decayed by a factor of 10 at epochs 150, 225 and 300. The weight decay was set to $5 \times 10^{-4}$.

**- Experimental results.** Table 9 shows the performance on Market-1501 of models that were pretrained on ImageNet. The models in the first block are designed specifically for person ReID. The second block presents the performance of general-purpose backbones with improvement methodologies on Market-1501. PSLT outperforms some methods with general-purpose backbones. PSLT achieves a higher rank-1 accuracy (94.3% vs. 93.8%) and mAP (86.2 vs. 81.6) than PCB (+RPP) with fewer parameters (9.0M vs. 26M). In addition, PSLT achieves comparable performance to Auto-ReID, which is automatically designed for person ReID. Compared to Auto-ReID, PSLT achieves a higher mAP (86.2 vs. 85.1) and the same rank-1 accuracy with fewer parameters (9.0M vs. 11.4M). PSLT also shows comparable performance to OSNet, with a slightly higher mAP (86.2 vs. 84.9) and slightly lower rank-1 accuracy (94.3% vs. 94.8%). However, PSLT performs worse than specifically designed state-of-the-art models (CDNet and TransReID).

As can be seen, our PSLT achieves comparable performance to general-purpose models in the second block and slightly inferior performance to the models specifically designed for ReID in the first block. The experimental results demonstrate that our PSLT is generalizable to the person re-identification tasks.

### 4.5 Discussion

We have shown our model performance in Table 4 and our PSLT achieves comparable top-1 accuracy with relative less parameters and FLOPs. Usually, the FLOPs is adopted to measure the total float operations for processing one image. Intuitively, the model with lower FLOPs can achieve higher FPS which measures the amount of processed images in one second. However, according to Table 10, the FPS for the listed three kinds of models seems counter-intuitive. For example, the Swin contains relatively less FLOPs than DeiT-S but processes nearly a half amount of images as the DeiT-S does in one second. The same phenomenon occurs when comparing our PSLT-Large and Swin. The main reason is that the computing optimization is not implemented for Swin and our PSLT-Large directly on the accelerators such as GPUs, including the window partition in Swin and PSLT-Large and the computation of multiple branches in our PSLT-Large. Furthermore, we examine the FPS of Swin$^{\times 3}$ with 3 times of the original channel dimension, and find that the FLOPs is nearly 9 times of the Swin but the FPS is only 1/3 of the Swin. Thus, more direct optimization on the computing hardware (GPU) for each transformer model could essential for the FPS. However, the optimization of the computing hardware for fast inference can be investigated in the future but is not the main scope of our work.

TABLE 11
Ablation study on the ImageNet without pretraining. "#Branches" indicates the number of branches in the ladder self-attention block.

| Model | #Branches | #Params | #FLOPs | Top-1 |
|---|---|---|---|---|
| PSLT | 2 | 11.0M | 2.1G | 80.2% |
| PSLT | 3 | 9.2M | 1.9G | 79.9% |
| PSLT | 4 | 8.4M | 1.8G | 79.0% |
| PSLT | 6 | 7.5M | 1.7G | 77.9% |

TABLE 12
Ablation study on the ImageNet without pretraining. "Shift" denotes that the shift operation was used in the ladder self-attention block to model diverse local self-attentions. "FD" indicates whether progressively transmitting features is adopted. "PAFM" indicates the pixel-adaptive fusion module.

| Model | Shift | FD | PAFM | #Params | #FLOPs | Top-1 |
|---|---|---|---|---|---|---|
| PSLT | ✓ | ✓ | ✓ | 9.2M | 1.9G | 79.9% |
| – – | ✗ | ✓ | ✓ | 9.2M | 1.9G | 79.3% |
| – – | ✓ | ✗ | ✓ | 9.7M | 2.0G | 79.2% |

## 5 Ablation Study

In this section, we conduct ImageNet classification experiments to demonstrate that the ladder self-attention block and progressive shift mechanism in the proposed PSLT are effective. Here, all models were trained with an input image resolution of $224^2$ following the experimental settings described in Section 4.1.1.

**- Number of branches in the ladder self-attention block.** As shown in Section 3.2, the ladder self-attention block contains multiple branches, and more branches denotes that the developed backbone has less trainable parameters. In the ladder self-attention block with 6 branches, the shift stride in each branch is set to 3 (with two directions), 1 (with two directions), 5. The blocks with less branches are constructed similarly to the above. Table 11 shows the performance of PSLT with various number of branches in the ladder self-attention blocks. Although the model with more branches requires less computing resources, the performance of the model gets worse. For balance, we set the number of branches as 3 by default.

**- Shift operation in the ladder self-attention block.** As described in Section 3.2.1, PSLT adopts the shift operation in each branch of the ladder self-attention block to model long-range interactions. The third line in Table 12 shows the performance of PSLT without the shift operation, *i.e.*, all shift operations in Figures 1 and 3 are removed. The experimental results show that with the shift operation, PSLT achieves a higher top-1 accuracy (79.9% vs. 79.3%) without increasing the number of training parameters or FLOPs.

TABLE 13
Ablation study of FFN layer on ImageNet without pre-training. "LFFN" denotes our Light FNN layer.

| Model | FFN | #Params | #FLOPs | Top-1 |
|---|---|---|---|---|
| PSLT | LFFN | 9.2M | 1.9G | 79.9% |
| – – | FFN | 12.8M | 2.3G | 79.7% |

TABLE 14
Ablation study of the fusion module for the multiple branches on the ImageNet without pretraining. "Module" indicates the method for feature fusion. "Weight" denotes the existence of the adaptive weights for feature fusion. "Pointwise" indicates the existence of a fully connected layer for fusing the features along the channel dimension. "AW" denotes that only the first two fully connected layers are preserved in the pixel-adaptive fusion module. "Concat" indicates that the output features of the multiple branches are concatenated. "SE" denotes that the SE [50] block is adopted.

| Module | Weight | Pointwise | #Params | #FLOPs | Top-1 |
|---|---|---|---|---|---|
| PAFM | ✓ | ✓ | 9.2M | 1.9G | 79.9% |
| FC | ✗ | ✓ | 7.4M | 1.7G | 78.8% |
| AW | ✓ | ✗ | 7.4M | 1.7G | 78.6% |
| Concat | ✗ | ✗ | 5.6M | 1.5G | 77.3% |
| SE [50] | ✓ | ✓ | 9.2M | 1.7G | 79.0% |

**- Feature delivery in the ladder self-attention block.** For long-range interactions, PSLT progressively delivers features of previous branch, where the latter branch takes the output features of the previous branch as its value when computing self-attention. Here, we investigate whether delivering the output features is effective. A multi-branch block is adopted, and the shift operation is still used for each branch, *i.e.*, only the horizontal connections in Figure 1 are removed, and the PSW-MHSA in each branch is replaced with the W-MHSA in Figure 3. The last line in Table 12 presents the performance of the model without horizontal delivery. Since the value of self-attention in each branch needs to be computed from the input feature, the number parameters in the model without horizontal delivery is larger than the PSLT. With the horizontal connections in the ladder self-attention block, PSLT achieves a higher top-1 accuracy (79.9% vs. 79.2%) with slightly fewer parameters (9.2M vs. 9.7M).

By modelling diverse local interactions and interacting among the branches, the progressive shift mechanism is capable of enlarging the receptive field of the ladder self-attention block. The last two lines in Table 12 show performance of models without the two strategies respectively. Obviously, PSLT with the progressive shift mechanism achieves higher performance. Besides, with similar amount of computing resources, PSLT also outperforms the Swin-2G as shown in Table 4, where Swin-2G is implemented in the same way as SwinTransformer [15]. The above experimental results show that the progressive shift mechanism improves model capacity with large receptive field.

**- Light FFN layer.** For further cutting down the computing budgets, PSLT proposes the Light FNN (LFFN) layer to replace the original FFN layer in self-attention block. Table 13 shows the comparison between PSLT and the model with FFN (only Light FNN layer in PSLT is replaced with the original FFN layer), and PSLT achieves comparable performance, where PSLT with LFFN achieves comparable top-1 accuracay (79.9% vs. 79.7%) with relatively smaller amount of parameters (9.2M vs. 12.8M) and FLOPs (1.9G vs. 2.3G).

TABLE 15
Ablation study of the proposed modules for PSLT on the ImageNet without pretraining. "V1" and "V2" denote the top-1 accuracy on ImageNet and ImageNet-V2 validation sets respectively.

| Module | LSA | LFFN | PAFM | #Params | #FLOPs | V1 | V2 |
|---|---|---|---|---|---|---|---|
| ViT-B [31] | ✗ | ✗ | ✗ | 86M | 55.5G | 77.9% | 67.5% |
| LSA | ✓ | ✗ | ✗ | 9.2M | 2.0G | 77.2% | 64.8% |
| +LFFN | ✓ | ✓ | ✗ | 5.6M | 1.5G | 77.3% | 64.6% |
| +PAFM | ✓ | ✓ | ✓ | 9.2M | 1.9G | 79.9% | 68.6% |

**- Pixel-Adaptive Fusion Module.** As described in Section 3.2.3, to effectively integrate the features of each branch in the ladder self-attention block, PSLT adopts a pixel-adaptive fusion module. In this module, the weight of each pixel in the feature map is first multiplied by the feature; then, the weighted features are integrated in a fully connected layer to fuse the features along the channel dimension. "AW" and "FC" denote the module without one of the two steps, respectively. The experimental results in Table 14 demonstrate the effectiveness of our proposed pixel-adaptive fusion module.

Furthermore, we implemented another module ("SE" in Table 14) for comparison, which only computes the weights along the channel dimension (a pointwise following a squeeze-and-excitation layer). The model with only channel adaptive weights achieves top-1 accuracy of 79.0% with 9.2M parameters and 1.7G FLOPs. The experimental results demonstrate that adaptive weights in both the spatial and channel dimensions significantly improves performance.

**- Stacking the proposed modules.** As described in Section 3.1, our proposed PSLT is composed of the proposed ladder self-attention block, Light FFN and the pixel-adaptive fusion module. Table 15 shows the performance of models by stacking the proposed modules one by one, which demonstrates the effectiveness of the modules. Compared to the ViT, although applying our ladder self-attetion (LSA) and Light FFN (LFFN) does not improve the performance, the number of parameters and FLOPs are largely reduced, where the parameters have been reduced by 9 times and 15 times respectively, and the FLOPs have been reduced by 27 times and 37 times respectively. The pixel-adaptive fusion module (PAFM) improves the model performance by integrating information of all branches in the block. *The top-1 accuracy of our PSLT and DeiT-S on both the ImageNet and ImageNet-V2 validation set conforms to the linear fit in [53]. Our PSLT (including DeiT-S) achieves smaller improvement over ViT-B on ImageNet-V2 validation (2% and 1% top-1 accuracy improvement on ImageNet and ImageNet-V2 validation set, respectively). Such a phenomenon has been similarly observed in [37], [69]. As described in [88], this shows the model generalization ability is challenged on different validation set.*

## 6 Conclusion

In this work, we propose a ladder self-attention block with multiple branches requiring a relatively small number of

parameters and FLOPs. To improve the computation efficiency and enlarge the receptive field of the ladder self-attention block, the input feature map is split into several parts along the channel dimension, and a progressive shift mechanism is proposed for long-range dependency by modelling diverse local self-attention on each branch and interacting among these branches. A pixel-adaptive fusion module is finally designed to integrate features from multiple branches with adaptive weights along both the spatial and channel dimensions. Furthermore, the ladder self-attention block adopts a Light FFN layer to decrease the number of trainable parameters and float-point operations. Based on the above designs, we develop a general-purpose backbone (PSLT) with a relatively small number of parameters and FLOPs. Overall, PSLT applies convolutional blocks in the early stages and ladder self-attention blocks in the latter stages. PSLT achieves comparable performance with existing methods on several computer vision tasks, including image classification, object detection and person re-identification.

Our work explores a new perspective to model long-range interaction by effectively utilizing diverse local self-attentions, while there are recent works [17], [18], [38], [48] implement effective methods to combine the convolution and global self-attention. We find that modelling local self-attention is more environmental-friendly than modelling global self-attention.

For future work, we will investigate the direct computing optimization of our PSLT for fast inference. And we will investigate a new computation manner for self-attention because the similarity computation is time consuming. For example, we will investigate whether simple addition is powerful enough to model interactions in the self-attention computation. PSLT shows comparable performance without elaborately selecting the hyperparameters, such as the number of blocks, the channel dimension or the number of heads in the ladder self-attention blocks. Furthermore, we can automatically select parameters that significantly improve model performance with neural architecture search techniques.

## ACKNOWLEDGMENT

This work was supported partially by the NSFC (U21A20471,U1911401,U1811461), Guangdong NSF Project (No. 2020B1515120085, 2018B030312002), Guangzhou Research Project (201902010037), and the Key-Area Research and Development Program of Guangzhou (202007030004).

## REFERENCES

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, 2012.

[2] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, and M. Bernstein, "Imagenet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.

[3] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.

[4] S.-H. Gao, M.-M. Cheng, K. Zhao, X.-Y. Zhang, M.-H. Yang, and P. Torr, "Res2net: A new multi-scale backbone architecture," *IEEE transactions on pattern analysis and machine intelligence*, vol. 43, no. 2, pp. 652–662, 2019.

[5] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1492–1500.

[6] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[7] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700–4708.

[8] S. Zagoruyko and N. Komodakis, "Wide residual networks," *arXiv preprint arXiv:1605.07146*, 2016.

[9] Z. Dai, H. Liu, Q. Le, and M. Tan, "Coatnet: Marrying convolution and attention for all data sizes," *Advances in Neural Information Processing Systems*, vol. 34, 2021.

[10] T. Xiao, M. Singh, E. Mintun, T. Darrell, P. Dollár, and R. Girshick, "Early convolutions help transformers see better," *Advances in Neural Information Processing Systems*, vol. 34, pp. 30 392–30 400, 2021.

[11] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[12] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[13] J. Guo, K. Han, H. Wu, C. Xu, Y. Tang, C. Xu, and Y. Wang, "Cmt: Convolutional neural networks meet vision transformers," *arXiv preprint arXiv:2107.06263*, 2021.

[14] W. Wang, E. Xie, X. Li, D.-P. Fan, K. Song, D. Liang, T. Lu, P. Luo, and L. Shao, "Pyramid vision transformer: A versatile backbone for dense prediction without convolutions," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 568–578.

[15] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 10 012–10 022.

[16] W. Wang, L. Yao, L. Chen, D. Cai, X. He, and W. Liu, "Crossformer: A versatile vision transformer based on cross-scale attention," *arXiv e-prints*, pp. arXiv–2108, 2021.

[17] J. Pan, A. Bulat, F. Tan, X. Zhu, L. Dudziak, H. Li, G. Tzimiropoulos, and B. Martinez, "Edgevits: Competing light-weight cnns on mobile devices with vision transformers," *arXiv preprint arXiv:2205.03436*, 2022.

[18] M. Maaz, A. Shaker, H. Cholakkal, S. Khan, S. W. Zamir, R. M. Anwer, and F. S. Khan, "Edgenext: Efficiently amalgamated cnn-transformer architecture for mobile vision applications," *arXiv preprint arXiv:2206.10589*, 2022.

[19] S. Mehta and M. Rastegari, "Mobilevit: light-weight, general-purpose, and mobile-friendly vision transformer," *arXiv preprint arXiv:2110.02178*, 2021.

[20] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.

[21] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4510–4520.

[22] A. Howard, M. Sandler, G. Chu, L.-C. Chen, B. Chen, M. Tan, W. Wang, Y. Zhu, R. Pang, V. Vasudevan *et al.*, "Searching for mobilenetv3," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 1314–1324.

[23] X. Zhang, X. Zhou, M. Lin, and J. Sun, "Shufflenet: An extremely efficient convolutional neural network for mobile devices," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 6848–6856.

[24] B. Zoph and Q. Le, "Neural architecture search with reinforcement learning," in *ICLR 2017 : International Conference on Learning Representations 2017*, 2017.

[25] Z. Guo, X. Zhang, H. Mu, W. Heng, Z. Liu, Y. Wei, and J. Sun, "Single path one-shot neural architecture search with uniform sampling," in *European Conference on Computer Vision*. Springer, 2020, pp. 544–560.

[26] C. Li, T. Tang, G. Wang, J. Peng, B. Wang, X. Liang, and X. Chang, "Bossnas: Exploring hybrid cnn-transformers with block-wisely

self-supervised neural architecture search," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 12 281–12 291.

[27] M. Tan and Q. V. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *International Conference on Machine Learning*, 2019, pp. 6105–6114.

[28] H. Liu, K. Simonyan, and Y. Yang, "Darts: Differentiable architecture search," in *ICLR 2019 : 7th International Conference on Learning Representations*, 2019.

[29] D. Li, J. Hu, C. Wang, X. Li, Q. She, L. Zhu, T. Zhang, and Q. Chen, "Involution: Inverting the inherence of convolution for visual recognition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 12 321–12 330.

[30] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei, "Deformable convolutional networks," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 764–773.

[31] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[32] X. Yue, S. Sun, Z. Kuang, M. Wei, P. H. Torr, W. Zhang, and D. Lin, "Vision transformer with progressive sampling," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 387–396.

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

[34] Y. Li, G. Yuan, Y. Wen, E. Hu, G. Evangelidis, S. Tulyakov, Y. Wang, and J. Ren, "Efficientformer: Vision transformers at mobilenet speed," *arXiv preprint arXiv:2206.01191*, 2022.

[35] H. Wu, B. Xiao, N. Codella, M. Liu, X. Dai, L. Yuan, and L. Zhang, "Cvt: Introducing convolutions to vision transformers," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 22–31.

[36] Z. Peng, W. Huang, S. Gu, L. Xie, Y. Wang, J. Jiao, and Q. Ye, "Conformer: Local features coupling global representations for visual recognition," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 367–376.

[37] B. Graham, A. El-Nouby, H. Touvron, P. Stock, A. Joulin, H. Jégou, and M. Douze, "Levit: a vision transformer in convnet's clothing for faster inference," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 12 259–12 269.

[38] Y. Chen, X. Dai, D. Chen, M. Liu, X. Dong, L. Yuan, and Z. Liu, "Mobile-former: Bridging mobilenet and transformer," *arXiv preprint arXiv:2108.05895*, 2021.

[39] B. Heo, S. Yun, D. Han, S. Chun, J. Choe, and S. J. Oh, "Rethinking spatial dimensions of vision transformers," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 11 936–11 945.

[40] C.-F. R. Chen, Q. Fan, and R. Panda, "Crossvit: Cross-attention multi-scale vision transformer for image classification," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 357–366.

[41] H. Fan, B. Xiong, K. Mangalam, Y. Li, Z. Yan, J. Malik, and C. Feichtenhofer, "Multiscale vision transformers," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 6824–6835.

[42] L. Yuan, Y. Chen, T. Wang, W. Yu, Y. Shi, Z.-H. Jiang, F. E. Tay, J. Feng, and S. Yan, "Tokens-to-token vit: Training vision transformers from scratch on imagenet," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 558–567.

[43] Q. Yu, Y. Xia, Y. Bai, Y. Lu, A. L. Yuille, and W. Shen, "Glance-and-gaze vision transformer," *Advances in Neural Information Processing Systems*, vol. 34, 2021.

[44] Z. Xia, X. Pan, S. Song, L. E. Li, and G. Huang, "Vision transformer with deformable attention," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 4794–4803.

[45] X. Chu, Z. Tian, Y. Wang, B. Zhang, H. Ren, X. Wei, H. Xia, and C. Shen, "Twins: Revisiting the design of spatial attention in vision transformers," *Advances in Neural Information Processing Systems*, vol. 34, 2021.

[46] Q. Zhang and Y.-B. Yang, "Rest: An efficient transformer for visual recognition," *Advances in Neural Information Processing Systems*, vol. 34, pp. 15 475–15 485, 2021.

[47] X. Dong, J. Bao, D. Chen, W. Zhang, N. Yu, L. Yuan, D. Chen, and B. Guo, "Cswin transformer: A general vision transformer backbone with cross-shaped windows," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 12 124–12 134.

[48] Y. Lee, J. Kim, J. Willette, and S. J. Hwang, "Mpvit: Multi-path vision transformer for dense prediction," *arXiv preprint arXiv:2112.11010*, 2021.

[49] C. Yang, Y. Wang, J. Zhang, H. Zhang, Z. Wei, Z. Lin, and A. Yuille, "Lite vision transformer with enhanced self-attention," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 11 998–12 008.

[50] J. Hu, L. Shen, S. Albanie, G. Sun, and E. Wu, "Squeeze-and-excitation networks," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, vol. 42, no. 8, 2018, pp. 2011–2023.

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

[52] L. Zheng, L. Shen, L. Tian, S. Wang, J. Wang, and Q. Tian, "Scalable person re-identification: A benchmark," in *2015 IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 1116–1124.

[53] B. Recht, R. Roelofs, L. Schmidt, and V. Shankar, "Do imagenet classifiers generalize to imagenet?" in *International Conference on Machine Learning*. PMLR, 2019, pp. 5389–5400.

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

[55] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," *arXiv preprint arXiv:1710.09412*, 2017.

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

[57] Z. Zhong, L. Zheng, G. Kang, S. Li, and Y. Yang, "Random erasing data augmentation," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 34, no. 07, 2020, pp. 13 001–13 008.

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

[59] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," *arXiv preprint arXiv:1711.05101*, 2017.

[60] Y. Yuan, R. Fu, L. Huang, W. Lin, C. Zhang, X. Chen, and J. Wang, "Hrformer: High-resolution transformer for dense prediction," *arXiv preprint arXiv:2110.09408*, 2021.

[61] N. Ma, X. Zhang, J. Huang, and J. Sun, "Weightnet: Revisiting the design space of weight networks," in *European Conference on Computer Vision*. Springer, 2020, pp. 776–792.

[62] H. Yan, Z. Li, W. Li, C. Wang, M. Wu, and C. Zhang, "Contnet: Why not use convolution and transformer at the same time?" *arXiv preprint arXiv:2104.13497*, 2021.

[63] W. Xu, Y. Xu, T. Chang, and Z. Tu, "Co-scale conv-attentional image transformers," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 9981–9990.

[64] S. Ren, D. Zhou, S. He, J. Feng, and X. Wang, "Shunted self-attention via multi-scale token aggregation," *arXiv preprint arXiv:2111.15193*, 2021.

[65] A. Hatamizadeh, H. Yin, J. Kautz, and P. Molchanov, "Global context vision transformers," *arXiv e-prints*, pp. arXiv–2206, 2022.

[66] W. Yu, M. Luo, P. Zhou, C. Si, Y. Zhou, X. Wang, J. Feng, and S. Yan, "Metaformer is actually what you need for vision," *arXiv preprint arXiv:2111.11418*, 2021.

[67] K. Sun, B. Xiao, D. Liu, and J. Wang, "Deep high-resolution representation learning for human pose estimation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 5693–5703.

[68] H. Yin, A. Vahdat, J. M. Alvarez, A. Mallya, J. Kautz, and P. Molchanov, "A-vit: Adaptive tokens for efficient vision transformer," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 10 809–10 818.

[69] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jégou, "Training data-efficient image transformers & distillation through attention," in *International Conference on Machine Learning*. PMLR, 2021, pp. 10 347–10 357.

[70] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," *Handbook of Systemic Autoimmune Diseases*, 2009.
[71] K. Patel, A. M. Bur, F. Li, and G. Wang, "Aggregating global features into local vision transformer," *arXiv preprint arXiv:2201.12903*, 2022.
[72] P. Sun, R. Zhang, Y. Jiang, T. Kong, C. Xu, W. Zhan, M. Tomizuka, L. Li, Z. Yuan, C. Wang *et al.*, "Sparse r-cnn: End-to-end object detection with learnable proposals," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 14 454–14 463.
[73] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2980–2988.
[74] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2961–2969.
[75] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings, 2010, pp. 249–256.
[76] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba, "Scene parsing through ade20k dataset," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 633–641.
[77] A. Kirillov, R. Girshick, K. He, and P. Dollár, "Panoptic feature pyramid networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 6399–6408.
[78] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," *arXiv preprint arXiv:1511.07122*, 2015.
[79] T. Xiao, Y. Liu, B. Zhou, Y. Jiang, and J. Sun, "Unified perceptual parsing for scene understanding," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 418–434.
[80] H. Luo, Y. Gu, X. Liao, S. Lai, and W. Jiang, "Bag of tricks and a strong baseline for deep person re-identification," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2019, pp. 0–0.
[81] K. Zhou, Y. Yang, A. Cavallaro, and T. Xiang, "Omni-scale feature learning for person re-identification," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019, pp. 3702–3712.
[82] H. Li, G. Wu, and W.-S. Zheng, "Combined depth space based architecture search for person re-identification," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 6729–6738.
[83] R. Quan, X. Dong, Y. Wu, L. Zhu, and Y. Yang, "Auto-reid: Searching for a part-aware convnet for person re-identification," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019, pp. 3749–3758.
[84] S. He, H. Luo, P. Wang, F. Wang, H. Li, and W. Jiang, "Transreid: Transformer-based object re-identification," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 15 013–15 022.
[85] Y. Sun, L. Zheng, Y. Yang, Q. Tian, and S. Wang, "Beyond part models: Person retrieval with refined part pooling (and a strong convolutional baseline)," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 480–496.
[86] Y. Sun, Q. Xu, Y. Li, C. Zhang, Y. Li, S. Wang, and J. Sun, "Perceive where to focus: Learning visibility-aware part-level features for partial person re-identification," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 393–402.
[87] A. Hermans, L. Beyer, and B. Leibe, "In defense of the triplet loss for person re-identification." *arXiv preprint arXiv:1703.07737*, 2017.
[88] V. Shankar, R. Roelofs, H. Mania, A. Fang, B. Recht, and L. Schmidt, "Evaluating machine accuracy on imagenet," in *International Conference on Machine Learning*. PMLR, 2020, pp. 8634–8644.

**Gaojie Wu** is currently a Ph.D candidate from school of computer science and engineering, Sun Yat-sen University. His research interests mainly focus on neural architecture search and computer vision.

**Wei-Shi Zheng** is now a full Professor with Sun Yat-sen University. Dr. Zheng received his Ph.D. degree in Applied Mathematics from Sun Yat-sen University in 2008. His research interests include person/object association and activity understanding in visual surveillance, and the related large-scale machine learning algorithm. Especially, Dr. Zheng has active research on person re-identification in the last five years. He has ever joined Microsoft Research Asia Young Faculty Visiting Programme. He has ever served as area chairs of CVPR, ICCV, BMVC and IJCAI. He is an IEEE MSA TC member. He is an associate editor of the Pattern Recognition Journal. He is a recipient of the Excellent Young Scientists Fund of the National Natural Science Foundation of China, and a recipient of the Royal Society-Newton Advanced Fellowship of the United Kingdom.

**Yutong Lu** is Professor in Sun Yat-sen University (SYSU), Director of National supercomputing center in Guangzhou. Her extensive research and development experience has spanned several generations of domestic supercomputers in China. Her continuing research interests include HPC, Cloud computing, storage and file system, and advanced programming environment. At present, she is devoted to the research and implementation of system and application for the convergence of HPC, Bigdata and AI on supercomputer.

**Qi Tian** (Fellow, IEEE) received the B.E. degree in electronic engineering from Tsinghua University and the Ph.D. degree in electrical and computer engineering (ECE) from the University of Illinois at Urbana–Champaign (UIUC), Champaign, IL, USA. He was a Visiting Chaired Professor with the Center for Neural and Cognitive Computation, Tsinghua University, and a Lead Researcher with the Media Computing Group, Microsoft Research Asia (MSRA). He is currently the Chief Scientist of artificial intelligence with Huawei Cloud & AI, a Changjiang Chaired Professor of the Ministry of Education, an Overseas Outstanding Youth, and an Overseas Expert by the Chinese Academy of Sciences.