Computational Visual Media
https://doi.org/10.1007/s41095-022-0274-8 Vol. 8, No. 3, September 2022, 415–424

Research Article

PVT v2: Improved baselines with Pyramid Vision Transformer

Wenhai Wang[1,2] (✉), Enze Xie[3], Xiang Li[4], Deng-Ping Fan[5], Kaitao Song[4], Ding Liang[6], Tong Lu[2], Ping Luo[3], and Ling Shao[7]

Abstract Transformers have recently lead to encouraging progress in computer vision. In this work, we present new baselines by improving the original Pyramid Vision Transformer (PVT v1) by adding three designs: (i) a linear complexity attention layer, (ii) an overlapping patch embedding, and (iii) a convolutional feed-forward network. With these modifications, PVT v2 reduces the computational complexity of PVT v1 to linearity and provides significant improvements on fundamental vision tasks such as classification, detection, and segmentation. In particular, PVT v2 achieves comparable or better performance than recent work such as the Swin transformer. We hope this work will facilitate state-of-the-art transformer research in computer vision. Code is available at `https://github.com/whai362/PVT`.

Keywords transformers; dense prediction; image classification; object detection; semantic segmentation

1 Introduction

Recent studies on transformers for computer vision are converging on the backbone network [1–8] for downstream vision tasks, such as image classification, object detection, and instance and semantic segmentation. To date, there have been promising results. For example, Vision Transformer (ViT) [1] first showed that a pure transformer can archive state-of-the-art performance in image classification. The Pyramid Vision Transformer (PVT v1) [3] showed that a pure transformer backbone can also surpass CNN counterparts for dense prediction tasks such as detection and segmentation [9–11]. Later, Swin transformer [5], CoaT [6], LeViT [7], and Twins [8] further improved classification, detection, and segmentation performance with transformer backbones.

This work aims to establish stronger and more feasible baselines built on the PVT v1 framework. We report three design improvements: (i) a linear complexity attention layer, (ii) an overlapping patch embedding, and (iii) a convolutional feed-forward network, which are orthogonal to the PVT v1 framework, and when used with it, can bring better image classification, object detection, and instance and semantic segmentation results. We call the improved framework PVT v2; it has 6 different size variants, from B0 to B5 according to the number of parameters. In particular, PVT v2-B5 yields an 83.8% top-1 error on ImageNet, better than Swin-B [5] and Twins-SVT-L [8], while having fewer parameters and using fewer GFLOPs. Moreover, GFL [12] with PVT-B2 archives 50.2 AP on COCO 2017 val, 2.6 AP higher when using Swin-T [5], and 5.7 AP higher when using ResNet50 [13]. We hope these improved baselines will provide a reference for future research on vision transformers.

2 Related work

We now discuss transformer backbones related to

1 Shanghai AI Laboratory, Shanghai 200232, China. E-mail: wangwenhai362@gmail.com (✉).
2 Department of Computer Science and Technology, Nanjing University, Nanjing 210023, China. E-mail: lutong@nju.edu.cn.
3 Department of Computer Science, the University of Hong Kong, Hong Kong 999077, China. E-mail: E. Xie, xieenze@hku.hk; P. Luo, pluo@cs.hku.hk.
4 School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing 210014, China. E-mail: X. Li, xiang.li.implus@njust.edu.cn; K. Song, kt.song@njust.edu.cn.
5 Computer Vision Lab, ETH Zurich, Zurich 8092, Switzerland. E-mail: dengpfan@gmail.com.
6 SenseTime, Beijing 100080, China. E-mail: liangding@sensetime.com.
7 Inception Institute of Artificial Intelligence, Abu Dhabi, United Arab Emirates. E-mail: ling.shao@inceptioniai.org.
Manuscript received: 2021-12-22; accepted: 2022-02-08

this work. ViT [1] treats each image as a sequence of tokens (patches) with a fixed length, and then feeds them to multiple transformer layers to perform classification. It was the first work to demonstrate that a pure transformer can archive state-of-the-art image classification results given sufficient training data (e.g., ImageNet-22k [14], JFT-300M). DeiT [2] further explores a data-efficient training strategy and a distillation approach for ViT.

To improve image classification results, recent methods make tailored changes to ViT. T2T ViT [15] progressively concatenates tokens within an overlapping sliding window into a single token. TNT [16] utilizes inner and outer transformer blocks to generate pixel and patch embeddings. CPVT [17] replaces the fixed size position embedding in ViT with conditional position encodings, making it easier to process images of arbitrary resolution. CrossViT [18] processes image patches of different sizes via a dual-branch transformer. LocalViT [19] incorporates depth-wise convolution into vision transformers to improve the local continuity of features.

To adapt to dense prediction tasks such as object detection, instance and semantic segmentation, certain methods [3–8] introduce the pyramid structure in CNNs to the design of transformer backbones. PVT v1 was the first pyramid structure transformer, presenting a hierarchical transformer with four stages, and showing that a pure transformer backbone can be as versatile as CNN counterparts and provide better results for detection and segmentation tasks. Later, various improvements [4–8] were made to enhance local continuity of features and to remove the fixed size position embedding. For example, the Swin transformer [5] replaces the fixed size position embedding with relative position biases, and restricts self-attention within shifted windows. CvT [4], CoaT [6], and LeViT [7] introduce convolution-like operations into vision transformers. Twins [8] combines local and global attention mechanisms to obtain a stronger feature representation.

3 Methodology

3.1 Limitations of PVT v1

PVT v1 [3] has three main limitations: (i) like ViT [1], when processing high-resolution input (with the shorter side being 800+ pixels), the computational requirements of PVT v1 are relatively large, (ii) PVT v1 treats an image as a sequence of non-overlapping patches, which loses local continuity in the image to a certain extent, and (iii) the position encoding in PVT v1 is fixed-size, which is inflexible when images of arbitrary size must be processed. These problems limit the utility of PVT v1 for vision tasks.

To address these issues, we propose PVT v2, which improves PVT v1 through three designs, given in Sections 3.2–3.4.

3.2 Linear spatial reduction attention

First, to reduce the high computational cost caused by attention operations, we propose a linear spatial reduction attention (SRA) layer, illustrated in Fig. 1. Unlike SRA [3] which uses convolutions for spatial reduction, linear SRA uses average pooling to reduce the spatial dimension ($h \times w$) to a fixed size ($P \times P$) before the attention operation. In this way, linear SRA enjoys linear computational and memory costs like a convolutional layer. Specifically, given an input of size $h \times w \times c$, the complexity of SRA and linear SRA are

$$\Omega(\text{SRA}) = \frac{2h^2w^2c}{R^2} + hwc^2R^2 \tag{1}$$

$$\Omega(\text{linear SRA}) = 2hwP^2c \tag{2}$$

where R is the spatial reduction ratio of SRA [3], and P is the pooling size of linear SRA, which is set to 7.

3.3 Overlapping patch embedding

Secondly, to model the local continuity information, we utilize an overlapping patch embedding to tokenize images. As shown in Fig. 2(a), we enlarge the patch window, making adjacent windows overlap by half of their area, and pad the feature map with zeros to keep the resolution. In this work, we use convolution with zero padding to implement overlapping patch embedding. Specifically, given input of size $h \times w \times c$, we feed it to a convolution with stride S, kernel size

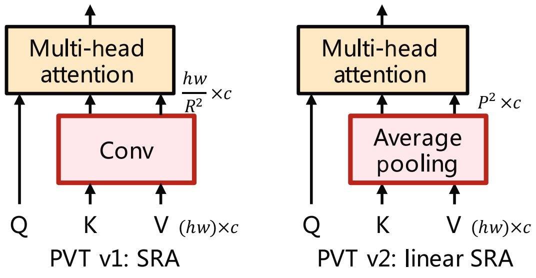
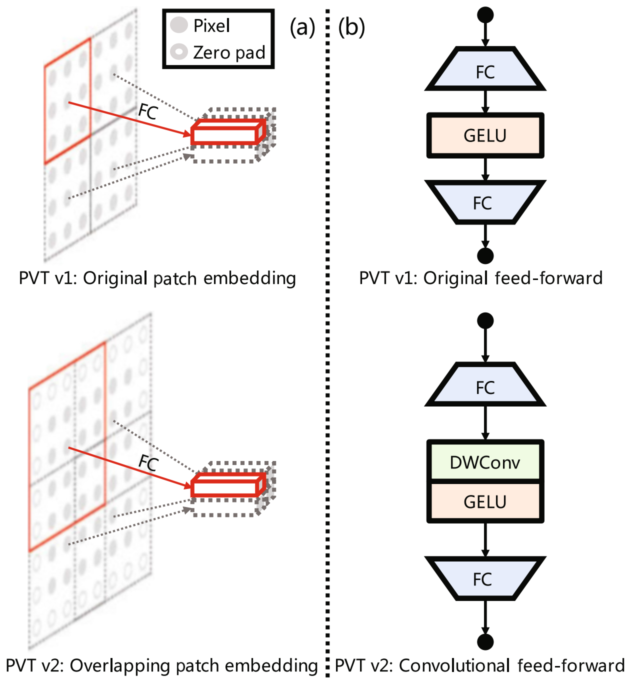


Fig. 1 SRA in PVT v1 and linear SRA in PVT v2.



PVT v1: Original patch embedding

PVT v1: Original feed-forward

PVT v2: Overlapping patch embedding

PVT v2: Convolutional feed-forward

Fig. 2 Two improvements in PVT v2: (a) overlapping patch embedding, (b) convolutional feed-forward network.

$2S-1$, padding size $S-1$, and c' kernels. The output size is $(h/S)(w/S)C'$.

3.4 Convolutional feed-forward

Thirdly, inspired by Refs. [17, 19, 20], we remove the fixed-size position encoding [1], and introduce zero padding position encoding into PVT. As shown in Fig. 2(b), we add a 3×3 depth-wise convolution [21] with padding size of 1 between the first fully-connected (FC) layer and GELU [22] in feed-forward networks.

3.5 Details of PVT v2 series

We scale up PVT v2 from B0 to B5 By changing the hyper-parameters, which are as follows for Stage i:

S_i: stride of the overlapping patch embedding
C_i: number of channels of output
L_i: number of encoder layers
R_i: reduction ratio of the SRA
P_i: adaptive average pooling size of the linear SRA
N_i: number of heads of the efficient self-attention
E_i: expansion ratio of the feed-forward layer [23]

Table 1 gives detailed information for the PVT v2 series. Our design follows the principles of ResNet [24]: (i) the channel dimension increases and the spatial resolution shrinks as the layers get deeper, and (ii) Stage 3 has the greatest computational cost.

Table 1 Detailed settings for PVT v2 series. "-Li" denotes PVT v2 with linear SRA

	Output size	Layer name	Pyramid Vision Transformer v2						
			B0	B1	B2	B2-Li	B3	B4	B5
Stage 1	$\frac{H}{4} \times \frac{W}{4}$	Overlapping patch embedding	$S_1 = 4$						
			$C_1 = 32$	$C_1 = 64$					
		Transformer encoder	$R_1 = 8$ $N_1 = 1$ $E_1 = 8$ $L_1 = 2$	$R_1 = 8$ $N_1 = 1$ $E_1 = 8$ $L_1 = 2$	$R_1 = 8$ $N_1 = 1$ $E_1 = 8$ $L_1 = 3$	$P_1 = 7$ $N_1 = 1$ $E_1 = 8$ $L_1 = 3$	$R_1 = 8$ $N_1 = 1$ $E_1 = 8$ $L_1 = 3$	$R_1 = 8$ $N_1 = 1$ $E_1 = 8$ $L_1 = 3$	$R_1 = 8$ $N_1 = 1$ $E_1 = 4$ $L_1 = 3$
Stage 2	$\frac{H}{8} \times \frac{W}{8}$	Overlapping patch embedding	$S_2 = 2$						
			$C_2 = 64$	$C_2 = 128$					
		Transformer encoder	$R_2 = 4$ $N_2 = 2$ $E_2 = 8$ $L_2 = 2$	$R_2 = 4$ $N_2 = 2$ $E_2 = 8$ $L_2 = 2$	$R_2 = 4$ $N_2 = 2$ $E_2 = 8$ $L_2 = 3$	$P_2 = 7$ $N_2 = 2$ $E_2 = 8$ $L_2 = 3$	$R_2 = 4$ $N_2 = 2$ $E_2 = 8$ $L_2 = 3$	$R_2 = 4$ $N_2 = 2$ $E_2 = 8$ $L_2 = 8$	$R_2 = 4$ $N_2 = 2$ $E_2 = 4$ $L_2 = 6$
Stage 3	$\frac{H}{16} \times \frac{W}{16}$	Overlapping patch embedding	$S_3 = 2$						
			$C_3 = 160$	$C_3 = 320$					
		Transformer encoder	$R_3 = 2$ $N_3 = 5$ $E_3 = 4$ $L_3 = 2$	$R_3 = 2$ $N_3 = 5$ $E_3 = 4$ $L_3 = 2$	$R_3 = 2$ $N_3 = 5$ $E_3 = 4$ $L_3 = 6$	$P_3 = 7$ $N_3 = 5$ $E_3 = 4$ $L_3 = 6$	$R_3 = 2$ $N_3 = 5$ $E_3 = 4$ $L_3 = 18$	$R_3 = 2$ $N_3 = 5$ $E_3 = 4$ $L_3 = 27$	$R_3 = 2$ $N_3 = 5$ $E_3 = 4$ $L_3 = 40$
Stage 4	$\frac{H}{32} \times \frac{W}{32}$	Overlapping patch embedding	$S_4 = 2$						
			$C_4 = 256$	$C_4 = 512$					
		Transformer encoder	$R_4 = 1$ $N_4 = 8$ $E_4 = 4$ $L_4 = 2$	$R_4 = 1$ $N_4 = 8$ $E_4 = 4$ $L_4 = 2$	$R_4 = 1$ $N_4 = 8$ $E_4 = 4$ $L_4 = 3$	$P_4 = 7$ $N_4 = 8$ $E_4 = 4$ $L_4 = 3$	$R_4 = 1$ $N_4 = 8$ $E_4 = 4$ $L_4 = 3$	$R_4 = 1$ $N_4 = 8$ $E_4 = 4$ $L_4 = 3$	$R_4 = 1$ $N_4 = 8$ $E_4 = 4$ $L_4 = 3$

3.6 Advantages of PVT v2

Combining these improvements, PVT v2 can (i) achieve more local continuity of images and feature maps, (ii) process variable-resolution input more readily, and (iii) enjoy the same linear complexity as a CNN.

4 Experiments

4.1 Image classification

4.1.1 Setting

Image classification experiments were performed on the ImageNet-1K dataset [27], which comprises 1.28 million training images and 50k validation images in 1000 categories. All models were trained on the training set for fair comparison and we report the top-1 error on the validation set. We followed DeiT [2] and applied random cropping, random horizontal flipping [28], label-smoothing regularization [29], mixup [30], and random erasing [31] for data augmentation. During training, we employed AdamW [32] with a momentum of 0.9, a mini-batch size of 128, and a weight decay of 5×10^{-2} to optimize models. The initial learning rate was set to 10^{-3} and decreased following a cosine schedule [33]. All models were trained for 300 epochs from scratch on 8 V100 GPUs. We applied a 224 × 224 center crop on the validation set for benchmarking to evaluate the classification accuracy.

4.1.2 Results

In Table 2, we see that PVT v2 provides best results for ImageNet-1K classification. Compared to PVT v1, PVT v2 uses similar flops and number of parameters, but the image classification accuracy is improved. For example, PVT v2-B1 is 3.6% higher than PVT v1-Tiny, and PVT v2-B4 is 1.9% higher than PVT-Large.

Compared to other recent counterparts, PVT v2 series also have large advantages in terms of accuracy and model size. For example, PVT v2-B5 achieves 83.8% ImageNet top-1 accuracy, which is 0.5% higher than Swin transformer [5] and Twins [8], while using fewer parameters and GFLOPS.

Table 2 Image classification performance on the ImageNet validation set. #Param = millions of parameters. GFLOPs is calculated for input of size 224 × 224. * = performance of the method trained under the strategy of its original paper. Acc = top-1 accuracy. -Li = PVT v2 with linear SRA

Method	#Param	GFLOPs	Acc (%)
PVT v2-B0 (ours)	**3.4**	**0.6**	**70.5**
ResNet18* [24]	11.7	1.8	69.8
DeiT-Tiny/16 [2]	**5.7**	**1.3**	72.2
PVT v1-Tiny [3]	13.2	1.9	75.1
PVT v2-B1 (ours)	13.1	2.1	**78.7**
ResNet50* [24]	25.6	4.1	76.1
ResNeXt50-32x4d* [25]	25.0	4.3	77.6
RegNetY-4G [26]	21.0	4.0	80.0
DeiT-Small/16 [2]	22.1	4.6	79.9
T2T-ViT$_t$-14 [15]	22.0	6.1	80.7
PVT v1-Small [3]	24.5	3.8	79.8
TNT-S [16]	23.8	5.2	81.3
Swin-T [5]	29.0	4.5	81.3
CvT-13 [4]	**20.0**	4.5	81.6
CoaT-Lite Small [6]	**20.0**	4.0	81.9
Twins-SVT-S [8]	24.0	**2.8**	81.7
PVT v2-B2-Li (ours)	22.6	3.9	**82.1**
PVT v2-B2 (ours)	25.4	4.0	82.0
ResNet101* [24]	44.7	7.9	77.4
ResNeXt101-32x4d* [25]	44.2	8.0	78.8
RegNetY-8G [26]	39.0	8.0	81.7
T2T-ViT$_t$-19 [15]	39.0	9.8	81.4
PVT v1-Medium [3]	44.2	6.7	81.2
CvT-21 [4]	**32.0**	7.1	82.5
PVT v2-B3 (ours)	45.2	**6.9**	**83.2**
ResNet152* [24]	60.2	11.6	78.3
T2T-ViT$_t$-24 [15]	64.0	15.0	82.2
PVT v1-Large [3]	61.4	9.8	81.7
TNT-B [16]	66.0	14.1	82.8
Swin-S [5]	**50.0**	8.7	83.0
Twins-SVT-B [8]	56.0	**8.3**	83.2
PVT v2-B4 (ours)	62.6	10.1	**83.6**
ResNeXt101-64x4d* [25]	83.5	15.6	79.6
RegNetY-16G [26]	84.0	16.0	82.9
ViT-Base/16 [1]	86.6	17.6	81.8
DeiT-Base/16 [2]	86.6	17.6	81.8
Swin-B [5]	88.0	15.4	83.3
Twins-SVT-L [8]	99.2	14.8	83.7
PVT v2-B5 (ours)	**82.0**	**11.8**	**83.8**

4.2 Object detection

4.2.1 Setting

Object detection experiments were conducted on the challenging COCO benchmark [9]. All models were trained on COCO 2017 train (118k images) and evaluated on COCO 2017 val (5k images). We verified the effectiveness of PVT v2 backbones with mainstream detectors, including RetinaNet [34], Mask R-CNN [35], Cascade Mask R-CNN [36], ATSS [37], GFL [12], and Sparse R-CNN [38]. Before training, we used weights pre-trained on ImageNet to initialize the backbone and Xavier [39] to initialize the newly added layers. We trained all models with batch size 16 on 8 V100 GPUs, and adopted AdamW [32] with an initial learning rate of 10^{-4} as optimizer. Following common practice [34, 35, 40], we adopted a 1× or 3× training schedule (12 or 36 epochs) to train all detection models. Training images were resized to have a shorter side of 800 pixels, while the longer side did not exceed 1333 pixels. When using the 3× training schedule, we randomly resized the shorter side of the input image to lie within the range $[640, 800]$. In the testing phase, the shorter side of the input image was fixed to 800 pixels.

4.2.2 Results

As Table 3 reports, PVT v2 significantly outperforms PVT v1 on both one-stage and two-stage object detectors with similar model size. For example, PVT v2-B4 achieves 46.1 AP with RetinaNet [34], and 47.5 AP^{b} with Mask R-CNN [35], surpassing models with PVT v1 by 3.5 AP and 4.6 AP^{b}, respectively. We present some qualitative object detection and instance segmentation results on COCO 2017 val [9] in Fig. 3, which also shows the good results from our models.

For a fair comparison between PVT v2 and Swin transformer [5], we kept all settings the same, including ImageNet-1K pre-training and COCO fine-tuning strategies. We evaluated Swin transformer and PVT v2 on four state-of-the-art detectors, including Cascade R-CNN [36], ATSS [37], GFL [12], and Sparse R-CNN [38] (Table 4). We see that PVT v2 obtains much better AP than Swin transformer for all detectors, showing its better feature representation ability. For example, on ATSS, PVT v2 uses a similar number of parameters and flops to Swin-T, but PVT v2 achieves 49.9 AP, 2.7 higher than Swin-T. Our PVT v2-Li reduces the computation from 258 to 194 GFLOPs, while only sacrificing a little performance.

4.3 Semantic segmentation

4.3.1 Settings

Following PVT v1 [3], we chose ADE20K [10] to benchmark semantic segmentation. For a fair comparison, we tested the PVT v2 backbones by using them with Semantic FPN [41]. In the training phase, the backbone was initialized with weights pre-trained on ImageNet [14], and the newly added layers were initialized with Xavier [39]. We optimized our models using AdamW [32] with an initial learning rate of 10^{-4}. Following common practices [41, 42], we trained our models for 40k iterations with a batch size of 16 on 4 V100 GPUs. The learning rate decayed following a polynomial decay schedule with a power of 0.9. We randomly resized and cropped images to 512×512 for training, and rescaled the shorter side to 512 pixels during testing.

4.3.2 Results

As Table 5 shows, when using Semantic FPN [41] for semantic segmentation, PVT v2 consistently outperforms PVT v1 [3] and other counterparts. For example, using almost the same number of parameters and GFLOPs, PVT v2-B1/B2/B3/B4

Table 3 Object detection and instance segmentation on COCO 2017 val. #P = millions of parameters. AP^{b} = bounding box AP. AP^{m} = mask AP. -Li = PVT v2 with linear SRA

Backbone	RetinaNet 1×							Mask R-CNN 1×						
	#P	AP	AP_{50}	AP_{75}	AP_{S}	AP_{M}	AP_{L}	#P	AP^{b}	$\mathrm{AP}^{\mathrm{b}}_{50}$	$\mathrm{AP}^{\mathrm{b}}_{75}$	AP^{m}	$\mathrm{AP}^{\mathrm{m}}_{50}$	$\mathrm{AP}^{\mathrm{m}}_{75}$
PVT v2-B0	**13.0**	**37.2**	**57.2**	**39.5**	**23.1**	**40.4**	**49.7**	**23.5**	**38.2**	**60.5**	**40.7**	**36.2**	**57.8**	**38.6**
ResNet18 [24]	**21.3**	31.8	49.6	33.6	16.3	34.3	43.2	**31.2**	34.0	54.0	36.7	31.2	51.0	32.7
PVT v1-Tiny [3]	23.0	36.7	56.9	38.9	22.6	38.8	50.0	32.9	36.7	59.2	39.3	35.1	56.7	37.3
PVT v2-B1 (ours)	23.8	**41.2**	**61.9**	**43.9**	**25.4**	**44.5**	**54.3**	33.7	**41.8**	**64.3**	**45.9**	**38.8**	**61.2**	**41.6**
ResNet50 [24]	37.7	36.3	55.3	38.6	19.3	40.0	48.8	44.2	38.0	58.6	41.4	34.4	55.1	36.7
PVT v1-Small [3]	34.2	40.4	61.3	43.0	25.0	42.9	55.7	44.1	40.4	62.9	43.8	37.8	60.1	40.3
PVT v2-B2-Li (ours)	**32.3**	43.6	64.7	46.8	28.3	47.6	57.4	**42.2**	44.1	66.3	48.4	40.5	63.2	43.6
PVT v2-B2 (ours)	35.1	**44.6**	**65.6**	**47.6**	**27.4**	**48.8**	**58.6**	45.0	**45.3**	**67.1**	**49.6**	**41.2**	**64.2**	**44.4**
ResNet101 [24]	56.7	38.5	57.8	41.2	21.4	42.6	51.1	63.2	40.4	61.1	44.2	36.4	57.7	38.8
ResNeXt101-32x4d [25]	56.4	39.9	59.6	42.7	22.3	44.2	52.5	**62.8**	41.9	62.5	45.9	37.5	59.4	40.2
PVT v1-Medium [3]	**53.9**	41.9	63.1	44.3	25.0	44.9	57.6	63.9	42.0	64.4	45.6	39.0	61.6	42.1
PVT v2-B3 (ours)	55.0	**45.9**	**66.8**	**49.3**	**28.6**	**49.8**	**61.4**	64.9	**47.0**	**68.1**	**51.7**	**42.5**	**65.7**	**45.7**
PVT v1-Large [3]	**71.1**	42.6	63.7	45.4	25.8	46.0	58.4	**81.0**	42.9	65.0	46.6	39.5	61.9	42.5
PVT v2-B4 (ours)	72.3	**46.1**	**66.9**	**49.2**	**28.4**	**50.0**	**62.2**	82.2	**47.5**	**68.7**	**52.0**	**42.7**	**66.1**	**46.1**
ResNeXt101-64x4d [25]	95.5	41.0	60.9	44.0	23.9	45.2	54.0	101.9	42.8	63.8	47.3	38.4	60.6	41.3
PVT v2-B5 (ours)	**91.7**	**46.2**	**67.1**	**49.5**	**28.5**	**50.0**	**62.5**	**101.6**	**47.4**	**68.6**	**51.9**	**42.5**	**65.7**	**46.0**

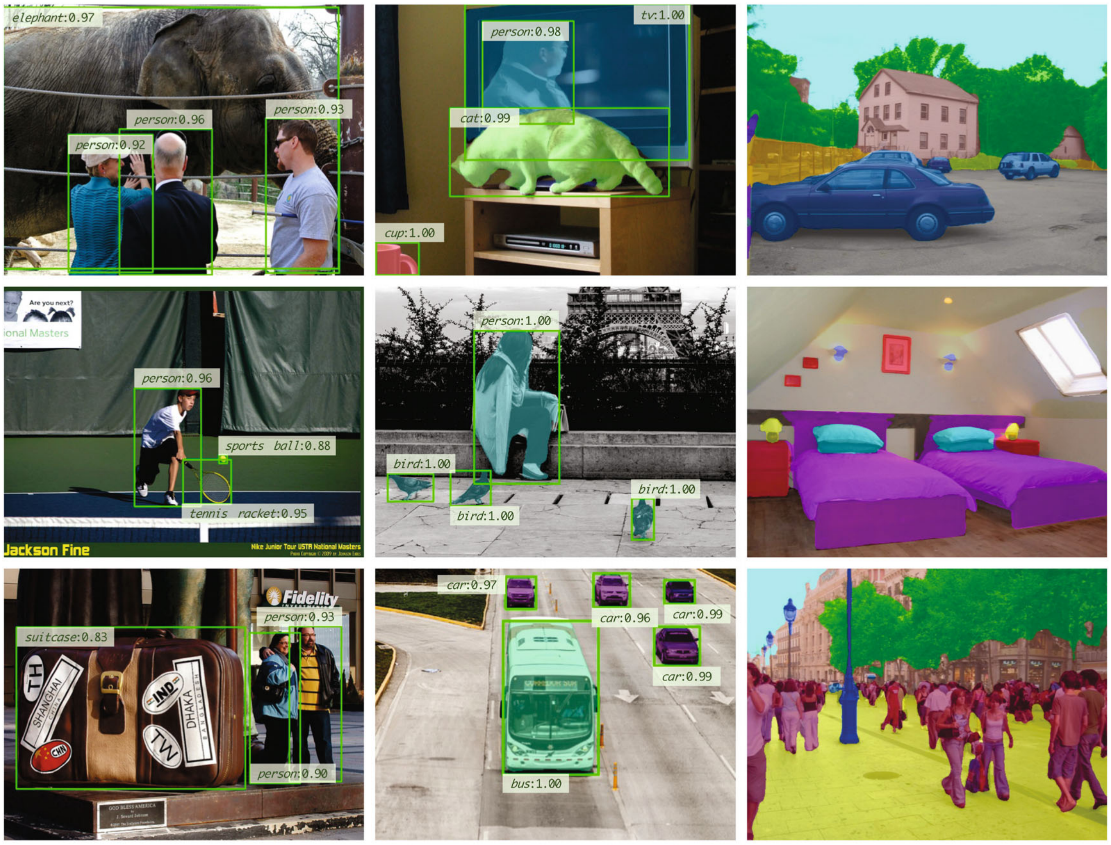
Table 4 Comparison with Swin transformer on object detection. AP^{b} = bounding box AP. #P = millions of parameters. #G = GFLOPs calculated for an input size 1280 × 800. -Li = PVT v2 with linear SRA

Backbone	Method	AP^{b}	$\mathrm{AP}^{\mathrm{b}}_{50}$	$\mathrm{AP}^{\mathrm{b}}_{75}$	#P	#G
ResNet50 [24]	Cascade Mask R-CNN	46.3	64.3	50.5	82	739
Swin-T [5]		50.5	69.3	54.9	86	745
PVT v2-B2-Li (ours)		50.9	69.5	55.2	**80**	**725**
PVT v2-B2 (ours)		**51.1**	**69.8**	**55.3**	83	788
ResNet50 [24]	ATSS	43.5	61.9	47.0	32	205
Swin-T [5]		47.2	66.5	51.3	36	215
PVT v2-B2-Li (ours)		48.9	68.1	53.4	**30**	**194**
PVT v2-B2 (ours)		**49.9**	**69.1**	**54.1**	33	258
ResNet50 [24]	GFL	44.5	63.0	48.3	32	208
Swin-T [5]		47.6	66.8	51.7	36	215
PVT v2-B2-Li (ours)		49.2	68.2	53.7	**30**	**197**
PVT v2-B2 (ours)		**50.2**	**69.4**	**54.7**	33	261
ResNet50 [24]	Sparse R-CNN	44.5	63.4	48.2	106	166
Swin-T [5]		47.9	67.3	52.3	110	172
PVT v2-B2-Li (ours)		48.9	68.3	53.4	**104**	**151**
PVT v2-B2 (ours)		**50.1**	**69.5**	**54.9**	107	215

Table 5 Semantic segmentation results for different backbones using the ADE20K validation set. #P = millions of parameters. #G = GFLOPs with input size 512 × 512. -Li = PVT v2 with linear SRA

Backbone	Semantic FPN		
	#P	#G	mIoU (%)
PVT v2-B0 (ours)	**7.6**	**25.0**	**37.2**
ResNet18 [24]	**15.5**	**32.2**	32.9
PVT v1-Tiny [3]	17.0	33.2	35.7
PVT v2-B1 (ours)	17.8	34.2	**42.5**
ResNet50 [24]	28.5	45.6	36.7
PVT v1-Small [3]	28.2	44.5	39.8
PVT v2-B2-Li (ours)	**26.3**	**41.0**	45.1
PVT v2-B2 (ours)	29.1	45.8	**45.2**
ResNet101 [24]	47.5	65.1	38.8
ResNeXt101-32x4d [25]	**47.1**	64.7	39.7
PVT v1-Medium [3]	48.0	**61.0**	41.6
PVT v2-B3 (ours)	49.0	62.4	**47.3**
PVT v1-Large [3]	**65.1**	**79.6**	42.1
PVT v2-B4 (ours)	66.3	81.3	**47.9**
ResNeXt101-64x4d [25]	86.4	103.9	40.2
PVT v2-B5 (ours)	**85.7**	**91.1**	**48.7**

provide at least 5.3% higher mIoU than PVT v1-Tiny/Small/Medium/Large. Moreover, although PVT-Large uses 12% less GFLOPs than ResNeXt101-64x4d, its mIoU is still 8.5% higher. Figure 3 shows some qualitative semantic segmentation results on ADE20K [10]. These results demonstrate that PVT v2



Object Detection on COCO Instance Segmentation on COCO Semantic Segmentation on ADE20K

Fig. 3 Results for object detection and instance segmentation on COCO 2017 val [9], and semantic segmentation on ADE20K [10]. Left to right: results generated by PVT v2-B2-based RetinaNet [34], Mask R-CNN [35], and Semantic FPN [41].

backbones can extract powerful features for semantic segmentation, benefiting from the improved designs.

4.4 Ablation study

4.4.1 Model analysis

Ablation experiments on PVT v2 are reported in Table 6. We see that all three designs improve the model in terms of result quality, number of parameters, or computational requirements.

4.4.2 Overlapping patch embedding

Overlapping patch embedding (OPE) is important. Comparing #1 and #2 in Table 6, the model with OPE obtains better top-1 accuracy (81.1% vs. 79.8%) on ImageNet and better AP (42.2 vs. 40.4) on COCO than when using the original patch embedding (PE) [1]. OPE is effective because it can model the local continuity of images and feature maps via overlapping sliding windows.

4.4.3 Convolutional feed-forward network

The convolutional feed-forward network (CFFN) matters. Compared to the original feed-forward network (FFN) [1], our CFFN contains a zero-padding convolutional layer, which can capture local continuity of the input tensor. In addition, due to the positional information introduced by zero-padding in OPE and CFFN, we can remove the fixed-size positional embeddings used in PVT v1, giving the model the flexibility to handle variable resolution input. As reported in #2 and #3 in Table 6, CFFN brings 0.9 points improvement on ImageNet (82.0% vs. 81.1%) and 2.4 points improvement on COCO, demonstrating its effectiveness.

4.4.4 Linear SRA

Linear SRA (LSRA) contributes to a better model. As reported in #3 and #4 in Table 6, compared to SRA [3], LSRA significantly reduces the computational load (in GFLOPs) of the model by 22%, while providing comparable top-1 accuracy on ImageNet (82.1% vs. 82.0%), and only 1 point lower AP on COCO (43.6 vs. 44.6). These results show the lower computational cost and good effects of LSRA.

Table 6 Ablation experiments on PVT v2. OPE, CFFN, and LSRA represent overlapping patch embedding, convolutional feed-forward network (PVT v2-B2), and linear SRA (PVT v2-B2-Li), respectively. #P = millions of parameters. #G = GFLOPs. Acc = top-1 accuracy

#	Setting	Acc	RetinaNet 1×		
		(%)	#P	#G	AP
1	PVT v1-Small [3]	79.8	34.2	285.8	40.4
2	+ OPE	81.1	34.9	288.6	42.2
3	++ CFFN	82.0	35.1	290.7	**44.6**
4	+++ LSRA	**82.1**	**32.3**	**227.4**	43.6

4.4.5 Computational complexity

As Fig. 4 shows, with increasing input scale, the growth rate of the computational requirements in GFLOPS for the proposed PVT v2-B2-Li are much lower than for PVT v1-Small [3], and are similar to those of ResNet-50 [13]. This demonstrates that PVT v2-Li successfully addresses the high computational overheads caused by the attention layer.

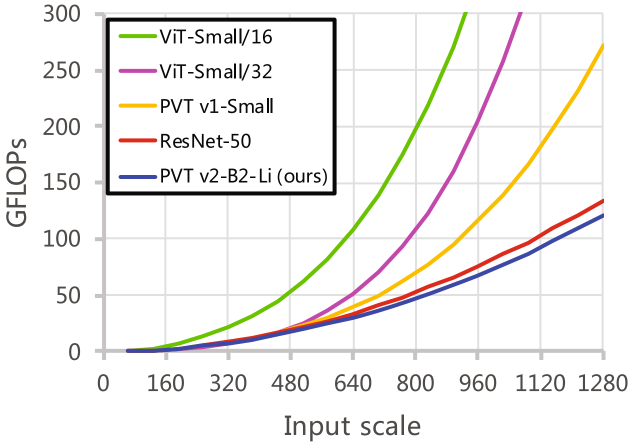


Fig. 4 GFLOPs required for different input sizes.

5 Conclusions

We studied the limitations of the Pyramid Vision Transformer (PVT v1) and improved it with three designs: an overlapping patch embedding, a convolutional feed-forward network, and a linear spatial reduction attention layer. Extensive experiments on different tasks, such as image classification, object detection, and semantic segmentation demonstrate that the proposed PVT v2 is stronger than its predecessor PVT v1 and other state-of-the-art transformer-based backbones, with comparable numbers of parameters. We hope these improved baselines will provide a reference for future research in vision transformers.

Acknowledgements

This work was supported by the National Natural Science Foundation of China under Grant Nos. 61672273 and 61832008, the Science Foundation for Distinguished Young Scholars of Jiangsu under Grant No. BK20160021, the Postdoctoral Innovative

Talent Support Program of China under Grant Nos. BX20200168 and 2020M681608, and the General Research Fund of Hong Kong under Grant No. 27208720.

Declaration of competing interest

The authors have no competing interests to declare that are relevant to the content of this article.

References

[1] Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; S. Gelly, S.; et al. An image is worth 16×16 words: Transformers for image recognition at scale. In: Proceedings of the International Conference on Learning Representations, 2021.

[2] Touvron, H.; Cord, M.; Douze, M.; Massa, F.; Jégou, H. Training data-efficient image transformers & distillation through attention. In: Proceedings of the 38th International Conference on Machine Learning, 2021.

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

[4] Wu, H.; Xiao, B.; Codella, N.; Liu, M.; Dai, X.; Yuan, L.; Zhang, L. CvT: Introducing convolutions to vision transformers. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, 22–31, 2021.

[5] Liu, Z.; Lin, Y.; Cao, Y.; Hu, H.; Wei, Y.; Zhang, Z.; Lin, S.; Guo, B. Swin transformer: Hierarchical vision transformer using shifted windows. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, 10012–10022, 2021.

[6] Xu, W.; Xu, Y.; Chang, T.; Tu, Z. Co-scale conv-attentional image transformers. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, 9981–9990, 2021.

[7] Graham, B.; El-Nouby, A.; Touvron, H.; Stock, P.; Joulin, A.; Jégou, H. LeViT: A vision transformer in ConvNet's clothing for faster inference. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, 12259–12269, 2021.

[8] Chu, X.; Tian, Z.; Wang, Y.; Zhang, B.; Ren, H.; Wei, X.; Xia, H.; Shen, C. Twins: Revisiting the design of spatial attention in vision transformers. In: Proceedings of the 35th Conference on Neural Information Processing Systems, 2021.

[9] Lin, T. Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; Zitnick, C. L. Microsoft COCO: Common objects in context. In: *Computer Vision – ECCV 2014. Lecture Notes in Computer Science, Vol. 8693.* Fleet, D.; Pajdla, T.; Schiele, B.; Tuytelaars, T. Eds. Springer Cham, 740–755, 2014.

[10] Zhou, B. L.; Zhao, H.; Puig, X.; Fidler, S.; Barriuso, A.; Torralba, A. Scene parsing through ADE20K dataset. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 5122–5130, 2017.

[11] Dong, B.; Wang, W.; Fan, D.-P.; Li, J.; Fu, H.; Shao, L. Polyp-PVT: Polyp segmentation with pyramid vision transformers. *arXiv preprint* arXiv:2108.06932, 2021.

[12] Li, X.; Wang, W.; Wu, L.; Chen, S.; Hu, X.; Li, J.; Tang, J.; Yang, J. Generalized focal loss: Learning qualified and distributed bounding boxes for dense object detection. In: Proceedings of the 34th Conference on Neural Information Processing Systems, 2020.

[13] He, K. M.; Zhang, X. Y.; Ren, S. Q.; Sun, J. Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification. In: Proceedings of the IEEE International Conference on Computer Vision, 1026–1034, 2015.

[14] Deng, J.; Dong, W.; Socher, R.; Li, L. J.; Kai, L.; Li, F. F. ImageNet: A large-scale hierarchical image database. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 248–255, 2009.

[15] Yuan, L.; Chen, Y.; Wang, T.; Yu, W.; Shi, Y.; Jiang, Z.; Tay, F. E.; Feng, J.; Yan, S. Tokens-to-token ViT: Training vision transformers from scratch on ImageNet. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, 558–567, 2021.

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

[17] Chu, X.; Tian, Z.; Zhang, B.; Wang, X.; Wei, X.; Xia, H.; Shen, C. Conditional positional encodings for vision transformers. *arXiv preprint* arXiv:2102.10882, 2021.

[18] Chen, C.-F.; Fan, Q.; Panda, R. CrossViT: Cross-attention multi-scale vision transformer for image classification. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, 357–366, 2021.

[19] Li, Y.; Zhang, K.; Cao, J.; Timofte, R.; van Gool, L. LocalViT: Bringing locality to vision transformers. *arXiv preprint* arXiv:2104.05707, 2021.

[20] Islam, M. A.; Jia, S.; Bruce, N. D. B. How much position information do convolutional neural networks encode? In: Proceedings of the International Conference on Learning Representations, 2020.

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

[22] Hendrycks, D.; Gimpel, K. Gaussian error linear units (GELUs). *arXiv preprint* arXiv:1606.08415, 2016.

[23] Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; Polosukhin, I. Attention is all you need. In: Proceedings of the 31st International Conference on Neural Information Processing Systems, 6000–6010, 2017.

[24] He, K. M.; Zhang, X. Y.; Ren, S. Q.; Sun, J. Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 770–778, 2016.

[25] Xie, S. N.; Girshick, R.; Dollar, P.; Tu, Z. W.; He, K. M. Aggregated residual transformations for deep neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 5987–5995, 2017.

[26] Radosavovic, I.; Kosaraju, R. P.; Girshick, R.; He, K. M.; Dollár, P. Designing network design spaces. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 10425–10433, 2020.

[27] Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S. A.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M.; et al. ImageNet large scale visual recognition challenge. *International Journal of Computer Vision* Vol. 115, No. 3, 211–252, 2015.

[28] Szegedy, C.; Liu, W.; Jia, Y. Q.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going deeper with convolutions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 1–9, 2015.

[29] Szegedy, C.; Vanhoucke, V.; Ioffe, S.; Shlens, J.; Wojna, Z. Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2818–2826, 2016.

[30] Zhang, H.; Cisse, M.; Dauphin, Y. N.; Lopez-Paz, D. mixup: Beyond empirical risk minimization. In: Proceedings of the International Conference on Learning Representations, 2018.

[31] Zhong, Z.; Zheng, L.; Kang, G. L.; Li, S. Z.; Yang, Y. Random erasing data augmentation. *Proceedings of the AAAI Conference on Artificial Intelligence* Vol. 34, No. 7, 13001–13008, 2020.

[32] Loshchilov, I.; Hutter, F. Decoupled weight decay regularization. In: Proceedings of the International Conference on Learning Representations, 2019.

[33] Loshchilov, I.; Hutter, F. SGDR: Stochastic gradient descent with warm restarts. In: Proceedings of the International Conference on Learning Representations, 2017.

[34] Lin, T. Y.; Goyal, P.; Girshick, R.; He, K. M.; Dollár, P. Focal loss for dense object detection. In: Proceedings of the IEEE International Conference on Computer Vision, 2999–3007, 2017.

[35] He, K. M.; Gkioxari, G.; Dollár, P.; Girshick, R. Mask R-CNN. In: Proceedings of the IEEE International Conference on Computer Vision, 2980–2988, 2017.

[36] Cai, Z. W.; Vasconcelos, N. Cascade R-CNN: Delving into high quality object detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 6154–6162, 2018.

[37] Zhang, S. F.; Chi, C.; Yao, Y. Q.; Lei, Z.; Li, S. Z. Bridging the gap between anchor-based and anchor-free detection via adaptive training sample selection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 9756–9765, 2020.

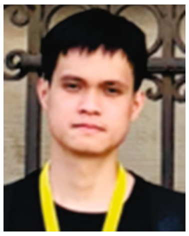
[38] Sun, P. Z.; Zhang, R. F.; Jiang, Y.; Kong, T.; Xu, C. F.; Zhan, W.; Tomizuka, M.; Li, L.; Yuan, Z.; Wang, C.; et al. Sparse R-CNN: End-to-end object detection with learnable proposals. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 14449–14458, 2021.

[39] Glorot, X.; Bengio, Y. Understanding the difficulty of training deep feedforward neural networks. In: Proceedings of the 13th International Conference on Artificial Intelligence and Statistics, 249–256, 2010.

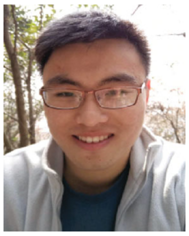
[40] Chen, K.; Wang, J. Q.; Pang, J. M.; Cao, Y. H.; Xiong, Y.; Li, X.; Sun, S.; Feng, W.; Liu, Z.; Xu, J.; et al. MMDetection: Open MMLab detection toolbox and benchmark. *arXiv preprint* arXiv:1906.07155, 2019.

[41] Kirillov, A.; Girshick, R.; He, K. M.; Dollár, P. Panoptic feature pyramid networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 6392–6401, 2019.

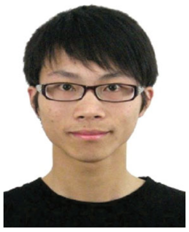
[42] Chen, L. C.; Papandreou, G.; Kokkinos, I.; Murphy, K.; Yuille, A. L. DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs. *IEEE Transactions on Pattern Analysis and Machine Intelligence* Vol. 40, No. 4, 834–848, 2018.



Wenhai Wang received his B.S. degree from Nanjing University of Science and Technology, China, in 2016. He is currently a Ph.D. student with the Department of Computer Science, Nanjing University. His main research interests include scene text detection and recognition, deep neural network exploration, object detection, and instance segmentation.



Enze Xie received his B.S. degree from Nanjing University of Aeronautics and Astronautics, China, in 2016, and his M.S. degree from Tongji University, China, in 2019. He is currently a Ph.D. student with the Department of Computer Science, the University of Hong Kong. His main research interests include object detection and instance segmentation.

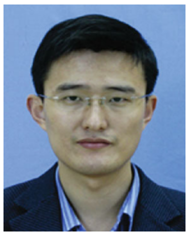


Xiang Li received his B.S. degree in computer science from Nanjing University of Science and Technology in 2013, where he is currently working towards a Ph.D. degree in pattern recognition and intelligent systems. His research interests include computer vision, pattern recognition, data mining, and deep learning.

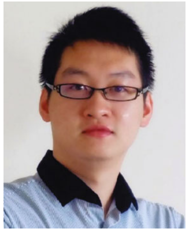
Deng-Ping Fan is a postdoctoral researcher at ETH Zurich, Switzerland. He received his Ph.D. degree from Nankai University in 2019. He joined the Inception Institute of Artificial Intelligence (IIAI) in 2019. He has published about 30+ top journal and conference papers. His research interests include computer vision, deep learning, and saliency detection.

Kaitao Song received his Ph.D. degree in computer science from Nanjing University of Science and Technology in 2021. His research interests focus on machine learning and deep learning algorithms for natural language processing and speech processing, including pre-trained language models, neural machine translation, music generation, text summarization, neural architecture search for NLP, audio speech recognition, text-to-speech synthesis, etc.

Ding Liang has been working for SenseTime Ltd., since he graduated from Tsinghua University. He is now an associate director and head of the OCR team. His main research interests include OCR, face recognition, and model compression.



Tong Lu received his Ph.D. degree in computer science from Nanjing University in 2005, where he also received his M.Sc. and B.Sc. degrees in 2002 and 1997, respectively. He served as associate professor and assistant professor in the Department of Computer Science and Technology at Nanjing University from 2007 and 2005, respectively, where he is now a full professor. He is also a member of the National Key Laboratory of Novel Software Technology in China. He has published over 130 papers and authored 2 books, and received more than 30 international and Chinese patents. His current interests are in multimedia, computer vision, and pattern recognition algorithms and systems.



Ping Luo is an assistant professor in the Department of Computer Science, The University of Hong Kong. He received his Ph.D. degree in 2014 from Information Engineering, the Chinese University of Hong Kong, and was a postdoctoral fellow there from 2014 to 2016. He joined SenseTime Research as a principal research scientist from 2017 to 2018. His research interests are machine learning and computer vision. He has published 100+ peer-reviewed articles in top-tier conferences and journals. He was named a young innovator under 35 by MIT Technology Review (TR35) Asia Pacific.

Ling Shao is the CEO and Chief Scientist of the Inception Institute of AI (IIAI), Abu Dhabi, United Arab Emirates (UAE). He was the initiator and Founding Provost and Executive Vice President of the Mohamed bin Zayed University of Artificial Intelligence (the world's first AI University), UAE. His research interests include computer vision, machine learning, and medical imaging. He is a fellow of the IEEE, the IAPR, the IET, and the BCS.