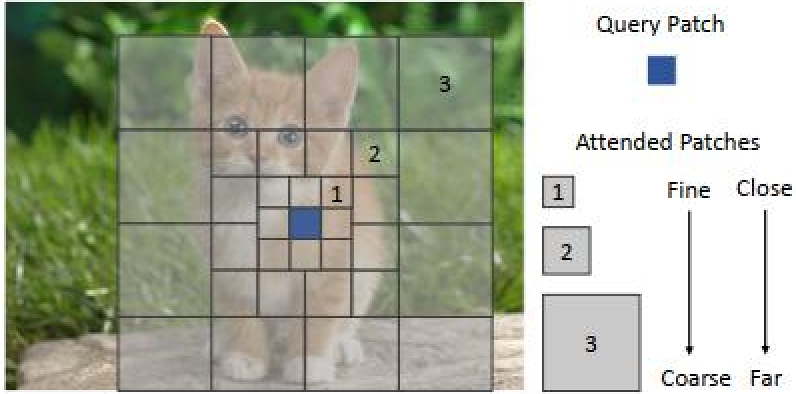
Focal Self-attention for Local-Global Interactions in Vision Transformers

Jianwei Yang[1] **Chunyuan Li**[1] **Pengchuan Zhang**[1] **Xiyang Dai**[2] **Bin Xiao**[2]
Lu Yuan[2] **Jianfeng Gao**[1]
[1]Microsoft Research at Redmond, [2]Microsoft Cloud + AI
{jianwyan,chunyl,penzhan,xidai,bixi,luyuan,jfgao}@microsoft.com

full self-attentions



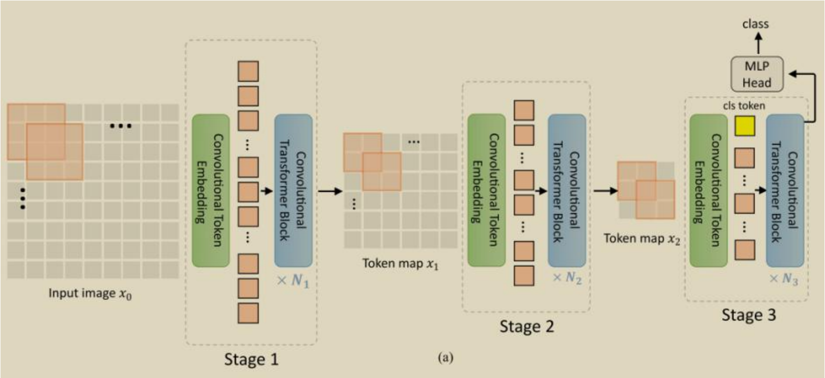
Through the visualization of full self-attentions, we indeed observe that it learns to attend local surroundings (like CNNs) and the global contexts at the same time.

Standard self-attention can capture both short- and long-range interactions at fine-grain, but it suffers from high computational cost when it performs the attention on high-resolution feature maps

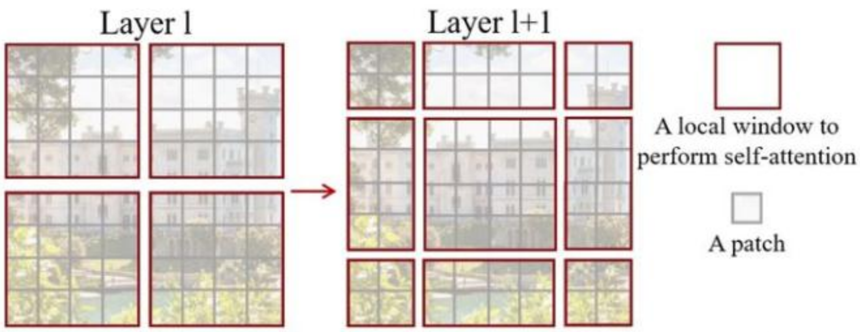
When it comes to high-resolution images for dense predictions such as object detection or segmentation, a global and fine-grained self-attention becomes non-trivial due to the quadratic computational cost with respect to the number of grids in feature maps.

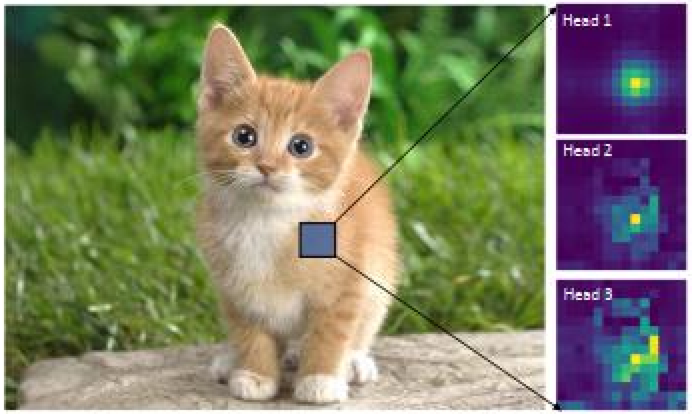
Recent works alternatively exploited either a coarse-grained global self-attention or a fine-grained local self-attention to reduce the computational burden.

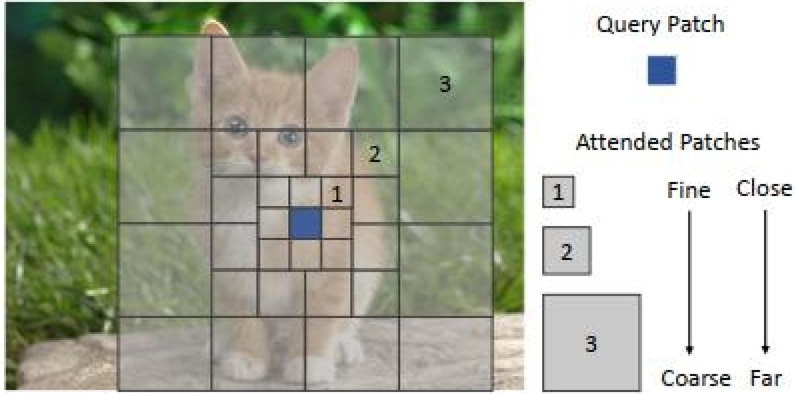
coarse-grained global self-attention



fine-grained local self-attention

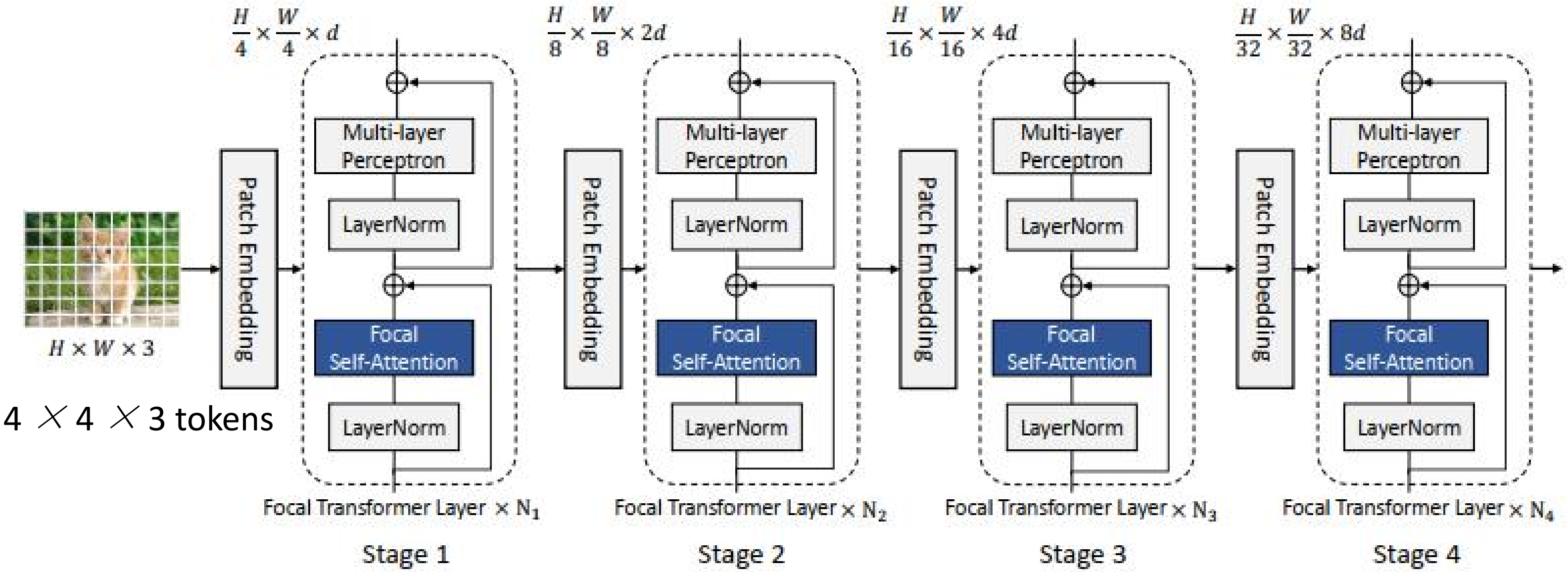






This paper presents a new self-attention mechanism to capture both local and global interactions in Transformer layers for high-resolution inputs.

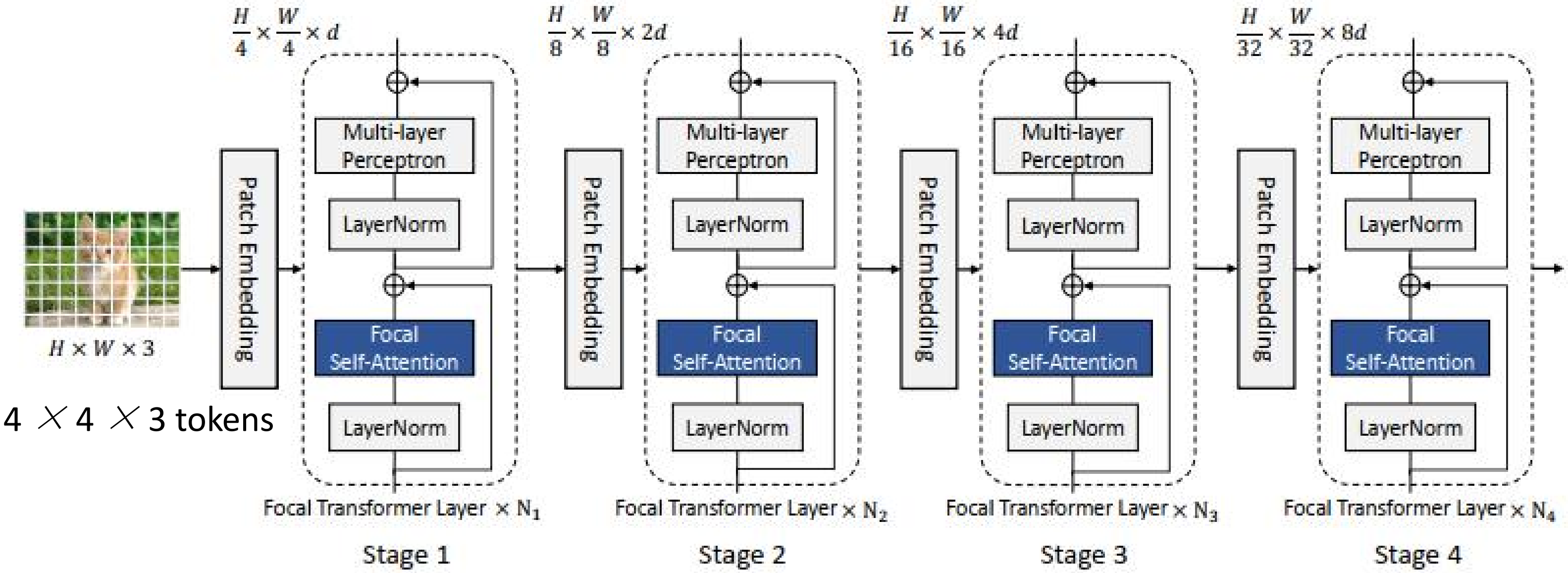
Each token attends its closest surrounding tokens at fine granularity and the tokens far away at coarse granularity, and thus can capture both short- and long-range visual



4 $\times$ 4 $\times$ 3 tokens

multi-scale design architecture (swin-Transformer), which allows us to obtain high-resolution feature maps at earlier stages.

Patch embedding layer to reduce the spatial size of feature map by factor 2, while the feature dimension is increased by 2

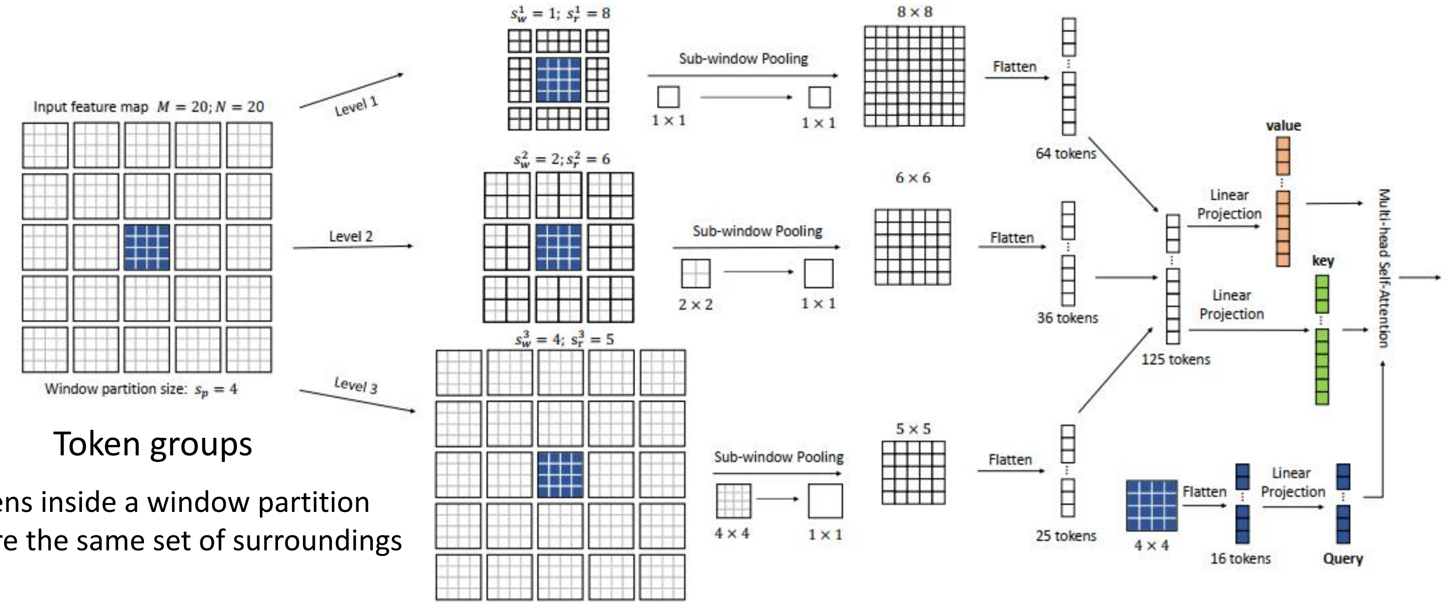


classification tasks: take the average of the output from last stage and send it to a classification layer.

object detection: the feature maps from last 3 or all 4 stages are fed to the detector head, depending on the particular detection method we use.

The model capacity : can be customized by varying the input feature dimension *d* and the number of focal Transformer layers at each stage.

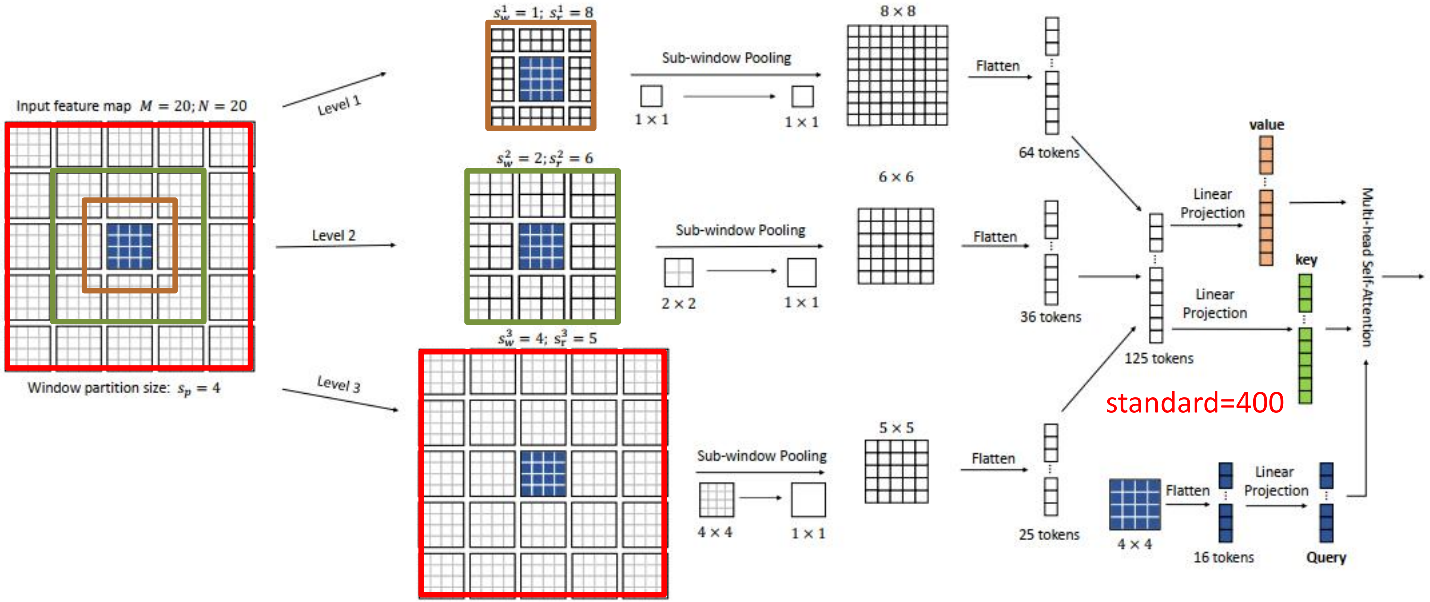
window-wise Focal self-attention



Token groups

tokens inside a window partition
share the same set of surroundings

a feature map of $x \in \mathcal{R}^{M \times N \times d}$ with spatial size $M \times N$, we first partition it into a grid of windows with size $s_p \times s_p$. Then, we find the surroundings for each window rather than individual tokens. In the following, we elaborate the window-wise focal self-attention.



- **Focal levels** L – the number of granularity levels we extract the tokens for our focal self-attention. In Fig. 1, we show 3 focal levels in total for example.
- **Focal window size** s_w^l – the size of sub-window on which we get the summarized tokens at level $l \in \{1, ..., L\}$, which are 1, 2 and 4 for the three levels in Fig. 1.
- **Focal region size** s_r^l – the number of sub-windows horizontally and vertically in attended regions at level l, and they are 3, 4 and 4 from level 1 to 3 in Fig. 1.

	Output Size	Layer Name	Focal-Tiny	Focal-Small	Focal-Base	
stage 1	56×56	Patch Embedding	$p_1 = 4; c_1 = 96$	$p_1 = 4; c_1 = 96$	$p_1 = 4; c_1 = 128$	
	56×56	Transformer Block	$\begin{bmatrix} s^0_{w,r} = \{1, 13\} \\ s^1_{w,r} = \{7, 7\} \end{bmatrix} \times 2$	$\begin{bmatrix} s^0_{w,r} = \{1, 13\} \\ s^1_{w,r} = \{7, 7\} \end{bmatrix} \times 2$	$\begin{bmatrix} s^0_{w,r} = \{1, 13\} \\ s^1_{w,r} = \{7, 7\} \end{bmatrix} \times 2$	W： 49
stage 2	28×28	Patch Embedding	$p_2 = 2; c_2 = 192$	$p_2 = 2; c_2 = 192$	$p_2 = 2; c_2 = 256$	
	28×28	Transformer Block	$\begin{bmatrix} s^0_{w,r} = \{1, 13\} \\ s^1_{w,r} = \{7, 5\} \end{bmatrix} \times 2$	$\begin{bmatrix} s^0_{w,r} = \{1, 13\} \\ s^1_{w,r} = \{7, 5\} \end{bmatrix} \times 2$	$\begin{bmatrix} s^0_{w,r} = \{1, 13\} \\ s^1_{w,r} = \{7, 5\} \end{bmatrix} \times 2$	W： 35
stage 3	14×14	Patch Embedding	$p_3 = 2; c_3 = 384$	$p_3 = 2; c_3 = 384$	$p_3 = 2; c_3 = 512$	
	14×14	Transformer Block	$\begin{bmatrix} s^0_{w,r} = \{1, 13\} \\ s^1_{w,r} = \{7, 3\} \end{bmatrix} \times 6$	$\begin{bmatrix} s^0_{w,r} = \{1, 13\} \\ s^1_{w,r} = \{7, 3\} \end{bmatrix} \times 18$	$\begin{bmatrix} s^0_{w,r} = \{1, 13\} \\ s^1_{w,r} = \{7, 3\} \end{bmatrix} \times 18$	W： 21
stage 4	7×7	Patch Embedding	$p_4 = 2; c_4 = 768$	$p_4 = 2; c_4 = 768$	$p_4 = 2; c_4 = 1024$	
	7×7	Transformer Block	$\begin{bmatrix} s^0_{w,r} = \{1, 7\} \\ s^1_{w,r} = \{7, 1\} \end{bmatrix} \times 2$	$\begin{bmatrix} s^0_{w,r} = \{1, 7\} \\ s^1_{w,r} = \{7, 1\} \end{bmatrix} \times 2$	$\begin{bmatrix} s^0_{w,r} = \{1, 7\} \\ s^1_{w,r} = \{7, 1\} \end{bmatrix} \times 2$	W： 7

Table 1: Model configurations for our focal Transformers. We introduce three configurations Focal-Tiny, Focal-Small and Focal-Base with different model capacities.

take 224 × 224 images as inputs and the window partition size is also set to 7 to make our models comparable to the Swin Transformers.

Model	#Params.	FLOPs	Top-1 (%)
ResNet-50 [34]	25.0	4.1	76.2
DeiT-Small/16 [57]	22.1	4.6	79.9
PVT-Small [63]	24.5	3.8	79.8
ViL-Small [80]	24.6	5.1	82.0
CvT-13 [67]	20.0	4.5	81.6
Swin-Tiny [44]	28.3	4.5	81.2
Focal-Tiny (Ours)	29.1	4.9	82.2
ResNet-101 [34]	45.0	7.9	77.4
PVT-Medium [63]	44.2	6.7	81.2
CvT-21 [67]	32.0	7.1	82.5
ViL-Medium [80]	39.7	9.1	83.3
Swin-Small [44]	49.6	8.7	83.1
Focal-Small (Ours)	51.1	9.1	83.5
ResNet-152 [34]	60.0	11.0	78.3
ViT-Base/16 [23]	86.6	17.6	77.9
DeiT-Base/16 [57]	86.6	17.5	81.8
PVT-Large [63]	61.4	9.8	81.7
ViL-Base [80]	55.7	13.4	83.2
Swin-Base [44]	87.8	15.4	83.4
Focal-Base (Ours)	89.8	16.0	**83.8**

Table 2: Comparison of image classification on ImageNet-1K for different models. Except for ViT-Base/16, all other models are trained and evaluated on 224×224 resolution.

Backbone	RetinaNet	Mask R-CNN	
	AP^b	AP^b	AP^m
ResNet-50 [34]	36.3	38.0	34.4
PVT-Small	40.4	40.4	37.8
ViL-Small [80]	41.6	41.8	38.5
Swin-Tiny [44]	42.0	43.7	39.8
Focal-Tiny (Ours)	**43.7** (+1.7)	**44.8** (+1.1)	**41.0** (+1.3)
ResNet-101 [34]	38.5	40.4	36.4
ResNeXt101-32x4d [70]	39.9	41.9	37.5
PVT-Medium [63]	41.9	42.0	39.0
ViL-Medium [80]	42.9	43.4	39.7
Swin-Small [44]	45.0	46.5	42.1
Focal-Small (Ours)	**45.6** (+0.6)	**47.4** (+0.9)	**42.8** (+0.7)
ResNeXt101-64x4d [70]	41.0	42.8	38.4
PVT-Large [63]	42.6	42.9	39.5
ViL-Base [80]	44.3	45.1	41.0
Swin-Base [44]	45.0	46.9	42.3
Focal-Base (Ours)	**46.3** (+1.3)	**47.8** (+0.9)	**43.2** (+0.9)

Table 3: Comparisons with CNN and Transformer baselines and SoTA methods on COCO object detection. The box mAP (AP^b) and mask mAP (AP^m) are reported for RetinaNet and Mask R-CNN trained with $1\times$ schedule. More detailed comparisons with $3\times$ schedule are in Table 4.

Method	Backbone	#Param	FLOPs	AP^b	AP^b_{50}	AP^b_{75}
C. Mask R-CNN [7]	R-50	82.0	739	46.3	64.3	50.5
	Swin-T	85.6	742	50.5	69.3	54.9
	Focal-T	86.7	770	**51.5** (+1.0)	**70.6**	**55.9**
ATSS [81]	R-50	32.1	205	43.5	61.9	47.0
	Swin-T	35.7	212	47.2	66.5	51.3
	Focal-T	36.8	239	**49.5** (+2.3)	**68.8**	**53.9**
RepPointsV2 [72]	R-50	43.4	431	46.5	64.6	50.3
	Swin-T	44.1	437	50.0	68.5	54.2
	Focal-T	45.4	491	**51.2** (+1.2)	**70.4**	**54.9**
Sparse R-CNN [55]	R-50	106.1	166	44.5	63.4	48.2
	Swin-T	109.7	172	47.9	67.3	52.3
	Focal-T	110.8	196	**49.0** (+1.1)	**69.1**	**53.2**

Table 5: Comparison with ResNet-50, Swin-Tiny across different object detection methods. We use Focal-Tiny as the backbone and train all models using $3\times$ schedule.

Backbone	#Params (M)	FLOPs (G)	RetinaNet 3x schedule + MS						Mask R-CNN 3x schedule + MS					
			AP^b	AP^b_{50}	AP^b_{75}	AP_S	AP_M	AP_L	AP^b	AP^b_{50}	AP^b_{75}	AP^m	AP^m_{50}	AP^m_{75}
ResNet50 [34]	37.7/44.2	239/260	39.0	58.4	41.8	22.4	42.8	51.6	41.0	61.7	44.9	37.1	58.4	40.1
PVT-Small[63]	34.2/44.1	226/245	42.2	62.7	45.0	26.2	45.2	57.2	43.0	65.3	46.9	39.9	62.5	42.8
ViL-Small [80]	35.7/45.0	252/174	42.9	63.8	45.6	27.8	46.4	56.3	43.4	64.9	47.0	39.6	62.1	42.4
Swin-Tiny [44]	38.5/47.8	245/264	45.0	65.9	48.4	29.7	48.9	58.1	46.0	68.1	50.3	41.6	65.1	44.9
Focal-Tiny (Ours)	39.4/48.8	265/291	**45.5**	**66.3**	**48.8**	**31.2**	**49.2**	**58.7**	**47.2**	**69.4**	**51.9**	**42.7**	**66.5**	**45.9**
ResNet101 [34]	56.7/63.2	315/336	40.9	60.1	44.0	23.7	45.0	53.8	42.8	63.2	47.1	38.5	60.1	41.3
ResNeXt101-32x4d [70]	56.4/62.8	319/340	41.4	61.0	44.3	23.9	45.5	53.7	44.0	64.4	48.0	39.2	61.4	41.9
PVT-Medium [63]	53.9/63.9	283/302	43.2	63.8	46.1	27.3	46.3	58.9	44.2	66.0	48.2	40.5	63.1	43.5
ViL-Medium [80]	50.8/60.1	339/261	43.7	64.6	46.4	27.9	47.1	56.9	44.6	66.3	48.5	40.7	63.8	43.7
Swin-Small [44]	59.8/69.1	335/354	46.4	67.0	50.1	31.0	50.1	60.3	48.5	70.2	53.5	43.3	67.3	46.6
Focal-Small (Ours)	61.7/71.2	367/401	**47.3**	**67.8**	**51.0**	**31.6**	**50.9**	**61.1**	**48.8**	**70.5**	**53.6**	**43.8**	**67.7**	**47.2**
ResNeXt101-64x4d [70]	95.5/102	473/493	41.8	61.5	44.4	25.2	45.4	54.6	44.4	64.9	48.8	39.7	61.9	42.6
PVT-Large[63]	71.1/81.0	345/364	43.4	63.6	46.1	26.1	46.0	59.5	44.5	66.0	48.3	40.7	63.4	43.7
ViL-Base [80]	66.7/76.1	443/365	44.7	65.5	47.6	29.9	48.0	58.1	45.7	67.2	49.9	41.3	64.4	44.5
Swin-Base [44]	98.4/107	477/496	45.8	66.4	49.1	29.9	49.4	60.3	48.5	69.8	53.2	43.4	66.8	46.9
Focal-Base (Ours)	100.8/110.0	514/533	**46.9**	**67.8**	**50.3**	**31.9**	**50.3**	**61.5**	**49.0**	**70.1**	**53.6**	**43.7**	**67.6**	**47.0**

Table 4: COCO object detection and segmentation results with RetinaNet [42] and Mask R-CNN [34]. All models are trained with $3\times$ schedule and multi-scale inputs (MS). The numbers before and after "/" at column 2 and 3 are the model size and complexity for RetinaNet and Mask R-CNN, respectively.

Model	W-Size	FLOPs	Top-1 (%)	AP^b	AP^m
Swin-Tiny	7	4.5	81.2	43.7	39.8
	14	4.9	82.1	44.0	40.5
Focal-Tiny	7	4.9	82.2	44.9	41.1
	14	5.2	82.3	45.5	41.5

Table 8: Impact of different window sizes (W-Size). We alter the default size 7 to 14 and observe consistent improvements for both methods.

Model	W-Shift	Top-1 (%)	AP^b	AP^m
Swin-Tiny	-	80.2	38.8	36.4
	✓	81.2	43.7	39.8
Focal-Tiny	-	82.2	44.8	41.0
	✓	81.9	44.9	41.1

Table 9: Impact of window shift (W-Shift) on Swin Transformer and Focal Transformer. Tiny models are used.

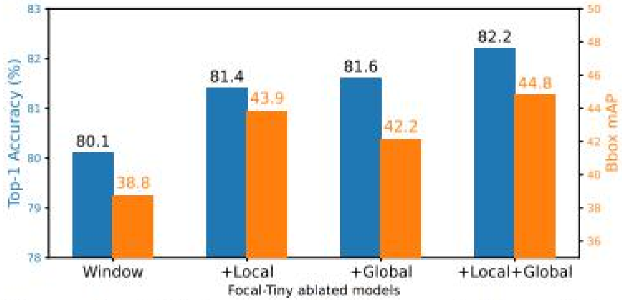


Figure 5: Ablating Focal-Tiny model by adding local, global and both interactions, respectively. Blue bars are for image classification and orange bars indicate object detection performance. Both local and global interactions are essential to obtain good performance. Better viewed in color.

Depths	Model	#Params.	FLOPs	Top-1 (%)	AP^b	AP^m
2-2-2-2	Swin	21.2	3.1	78.7	38.2	35.7
	Focal	21.7	3.4	79.9	40.5	37.6
2-2-4-2	Swin	24.7	3.8	80.2	41.2	38.1
	Focal	25.4	4.1	81.4	43.3	39.8
2-2-6-2	Swin	28.3	4.5	81.2	43.7	39.8
	Focal	29.1	4.9	82.2	44.8	41.0

Table 10: Impact of the change of model depth. We gradually reduce the number of transformer layers at the third stage from original 6 to 4 and further 2. It apparently hurts the performance but our Focal Transformers has much slower drop rate than Swin Transformer.