PyramidTNT: Improved Transformer-in-Transformer Baselines with Pyramid Architecture

Kai Han, Jianyuan Guo, Yehui Tang, Yunhe Wang

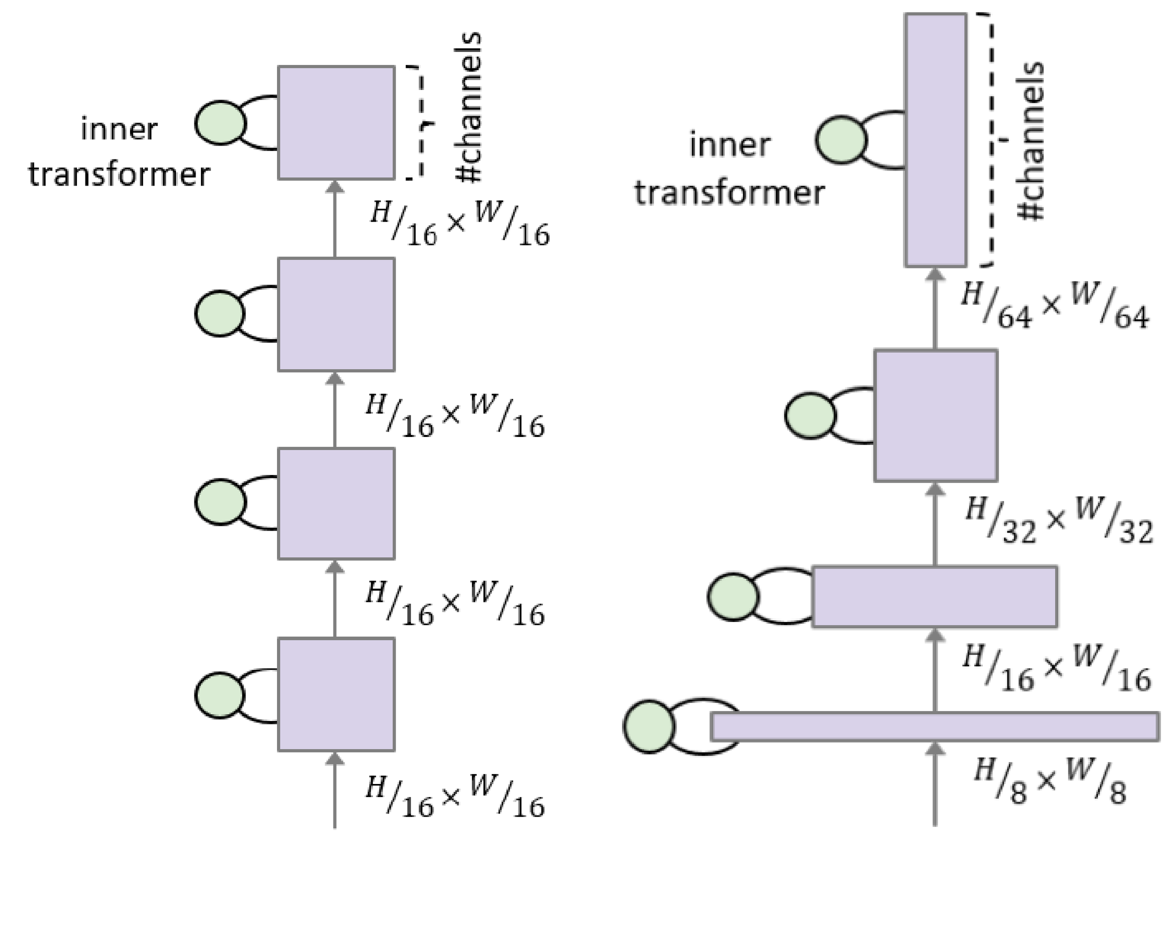
Huawei Noah's Ark Lab

NOAH'S ARK LAB

Abstract

Transformer networks have achieved great progress for computer vision tasks. Transformer-in-Transformer (TNT) architecture utilizes inner transformer and outer transformer to extract both local and global representations. In this work, we present new TNT baselines by introducing two advanced designs: 1) pyramid architecture, and 2) convolutional stem. The new ``PyramidTNT'' significantly improves the original TNT by establishing hierarchical representations. PyramidTNT achieves better performances than the previous state-of-the-art vision transformers such as Swin Transformer. We hope this new baseline will be helpful to the further research and application of vision transformer. Code is available at https://github.com/huawei-noah/CV-Backbones.

PyramidTNT Architecture



(a) TNT (b) PyramidTNT

Stage	Output size	PyramidTNT-Ti		PyramidTNT-S		PyramidTNT-M		PyramidTNT-B	
Stem	$\frac{H}{8}\times\frac{W}{8}$	Conv×5		Conv×5		Conv×5		Conv×5	
		Outer	Inner	Outer	Inner	Outer	Inner	Outer	Inner
Stage 1	$\frac{H}{8}\times\frac{W}{8}$	$\begin{bmatrix}D=80\\H_o=2\\R=4\end{bmatrix}\times2$	$\begin{bmatrix}C=5\\H_i=1\\R=1\end{bmatrix}\times1$	$\begin{bmatrix}D=128\\H_o=4\\R=4\end{bmatrix}\times2$	$\begin{bmatrix}C=8\\H_i=2\\R=1\end{bmatrix}\times1$	$\begin{bmatrix}D=192\\H_o=4\\R=4\end{bmatrix}\times2$	$\begin{bmatrix}C=12\\H_i=2\\R=1\end{bmatrix}\times1$	$\begin{bmatrix}D=256\\H_o=4\\R=4\end{bmatrix}\times2$	$\begin{bmatrix}C=16\\H_i=2\\R=1\end{bmatrix}\times1$
Downsample	$\frac{H}{16}\times\frac{W}{16}$	Patch Merging		Patch Merging		Patch Merging		Patch Merging	
		Outer	Inner	Outer	Inner	Outer	Inner	Outer	Inner
Stage 2	$\frac{H}{16}\times\frac{W}{16}$	$\begin{bmatrix}D=160\\H_o=4\\R=2\end{bmatrix}\times6$	$\begin{bmatrix}C=10\\H_i=2\\R=1\end{bmatrix}\times1$	$\begin{bmatrix}D=256\\H_o=8\\R=2\end{bmatrix}\times8$	$\begin{bmatrix}C=16\\H_i=4\\R=1\end{bmatrix}\times2$	$\begin{bmatrix}D=384\\H_o=8\\R=2\end{bmatrix}\times8$	$\begin{bmatrix}C=24\\H_i=4\\R=1\end{bmatrix}\times2$	$\begin{bmatrix}D=512\\H_o=8\\R=2\end{bmatrix}\times10$	$\begin{bmatrix}C=32\\H_i=4\\R=1\end{bmatrix}\times2$
Downsample	$\frac{H}{32}\times\frac{W}{32}$	Patch Merging		Patch Merging		Patch Merging		Patch Merging	
		Outer	Inner	Outer	Inner	Outer	Inner	Outer	Inner
Stage 3	$\frac{H}{32}\times\frac{W}{32}$	$\begin{bmatrix}D=320\\H_o=8\\R=1\end{bmatrix}\times3$	$\begin{bmatrix}C=20\\H_i=4\\R=1\end{bmatrix}\times1$	$\begin{bmatrix}D=512\\H_o=16\\R=1\end{bmatrix}\times4$	$\begin{bmatrix}C=32\\H_i=8\\R=1\end{bmatrix}\times1$	$\begin{bmatrix}D=768\\H_o=16\\R=1\end{bmatrix}\times6$	$\begin{bmatrix}C=48\\H_i=8\\R=1\end{bmatrix}\times1$	$\begin{bmatrix}D=1024\\H_o=16\\R=1\end{bmatrix}\times6$	$\begin{bmatrix}C=64\\H_i=8\\R=1\end{bmatrix}\times1$
Downsample	$\frac{H}{64}\times\frac{W}{64}$	Patch Merging		Patch Merging		Patch Merging		Patch Merging	
		Outer	Inner	Outer	Inner	Outer	Inner	Outer	Inner
Stage 4	$\frac{H}{64}\times\frac{W}{64}$	$\begin{bmatrix}D=320\\H_o=8\\R=1\end{bmatrix}\times2$	$\begin{bmatrix}C=20\\H_i=4\\R=1\end{bmatrix}\times1$	$\begin{bmatrix}D=512\\H_o=16\\R=1\end{bmatrix}\times2$	$\begin{bmatrix}C=32\\H_i=8\\R=1\end{bmatrix}\times1$	$\begin{bmatrix}D=768\\H_o=16\\R=1\end{bmatrix}\times2$	$\begin{bmatrix}C=48\\H_i=8\\R=1\end{bmatrix}\times1$	$\begin{bmatrix}D=1024\\H_o=16\\R=1\end{bmatrix}\times2$	$\begin{bmatrix}C=64\\H_i=8\\R=1\end{bmatrix}\times1$
Head	1 × 1	Pooling & FC		Pooling & FC		Pooling & FC		Pooling & FC	
Input resolution		192×192		256×256		256×256		256×256	
Parameters (M)		10.6		32.0		85.0		157.0	
FLOPs (B)		0.6		3.3		8.2		16.0	

Experiments

- ImageNet classification

Model	Params (M)	FLOPs (B)	Throughput (image/s)	Top-1 (%)
T2T-ViT-14 [41]	21.5	5.2	-	81.5
T2T-ViT-19 [41]	39.2	8.9	-	81.9
T2T-ViT-24 [41]	64.1	14.1	-	82.3
PVT-Small [36]	24.5	3.8	820	79.8
PVT-Medium [36]	44.2	6.7	526	81.2
PVT-Large [36]	61.4	9.8	367	81.7
PVTv2-B0 [35]	3.4	0.6	-	70.5
PVTv2-B2 [35]	25.4	4.0	-	82.0
PVTv2-B4 [35]	62.6	10.1	-	83.6
Swin-T [22]	29	4.5	755	81.3
Swin-S [22]	50	8.7	437	83.0
Swin-B [22]	88	15.4	278	83.3
TNT-S [11]	23.8	5.2	428	81.5
TNT-S-2 [11]	22.4	4.7	704	81.4
TNT-B [11]	65.6	14.1	246	82.9
PyramidTNT-Ti	10.6	0.6	2423	**75.2**
PyramidTNT-S	32.0	3.3	721	**82.0**
PyramidTNT-M	85.0	8.2	413	**83.5**
PyramidTNT-B	157.0	16.0	263	**84.1**

- COCO object detection

Table 4. Object detection and instance segmentation results on COCO val2017. FLOPs is calculated on 1280×800 input.

Backbone	RetinaNet 1×						Mask R-CNN 1×						
	# FLOPs	AP	AP_{50}	AP_S	AP_M	AP_L	# FLOPs	AP^b	AP^b_{50}	AP^b_{75}	AP^m	AP^m_{50}	AP^m_{75}
ResNet50 [13]	239.3G	36.3	55.3	19.3	40.0	48.8	260.1G	38.0	58.6	41.4	34.4	55.1	36.7
PVT-Small [36]	226.5G	40.4	61.3	25.0	42.9	55.7	245.1G	40.4	62.9	43.8	37.8	60.1	40.3
CycleMLP-B2 [3]	230.9G	40.6	61.4	22.9	44.4	54.5	249.5G	42.1	64.0	45.7	38.9	61.2	41.8
Swin-T [22]	244.8G	41.5	62.1	25.1	44.9	55.5	264.0G	42.2	64.6	46.2	39.1	61.6	42.0
Hire-MLP-Small [8]	237.6G	41.7	-	**25.3**	**45.4**	54.6	256.2G	42.8	65.0	46.7	39.3	62.0	42.1
PyramidTNT-S	225.9G	**42.0**	**63.1**	25.0	44.9	**57.7**	255.9G	**43.4**	**65.3**	**47.3**	**39.5**	**62.3**	**42.2**

Table 5. Instance segmentation results on COCO val2017.

Backbone	Mask R-CNN 3×							Cascade Mask R-CNN 3×						
	# FLOPs	AP^b	AP^b_{50}	AP^b_{75}	AP^m	AP^m_{50}	AP^m_{75}	# FLOPs	AP^b	AP^b_{50}	AP^b_{75}	AP^m	AP^m_{50}	AP^m_{75}
ResNet50 [13]	260.1G	41.0	61.7	44.9	37.1	58.4	40.1	738.7G	46.3	64.3	50.5	40.1	61.7	43.4
AS-MLP-T [18]	260.1G	46.0	67.5	50.7	41.5	64.6	44.5	739.0G	50.1	68.8	54.3	43.5	66.3	46.9
Swin-T [22]	264.0G	46.0	68.2	50.2	41.6	65.1	44.8	742.4G	50.5	69.3	54.9	43.7	66.6	47.1
Hire-MLP-S [8]	256.2G	46.2	68.2	50.9	42.0	65.6	45.3	734.6G	50.7	69.4	55.1	**44.2**	66.9	**48.1**
PyramidTNT-S	255.9G	**47.1**	**68.9**	**51.6**	**42.2**	**65.8**	**45.4**	794.1G	**51.0**	**69.7**	**55.3**	**44.2**	**67.0**	**48.1**