DefMamba: Deformable Visual State Space Model

Supplementary Material

A. Model Detailed Structure

In Table A1, we present the detailed architecture of our model variants, including the Tiny, Small, and Base versions, each with varying numbers of channels and blocks. During the experimental process, considering the channel redundancy mentioned in previous methods [29, 31] and the experimental findings in [23], we set the SSM_RATIO to 1. Following prior work [38], we employed two 3×3 convolutions with a stride of 2 and padding of 1 in the patch embedding layer, interspersed with LN and GELU activation. For the downsampling layer, we utilized a 3×3 convolution with a stride of 2 and padding of 1, followed by LN.

Stage	DefMamba-T	DefMamba-S	DefMamba-B
(224×224)	Patch embedding		
Stage 1 (56×56)	$N_1 = 2$, $C = 48$ K: 9	$N_1 = 2$, $C = 96$ K: 9	$N_1 = 2$, $C = 96$ K: 9
Stage 2 (28×28)	$N_2 = 2$, $C = 96$ K: 7	$N_2 = 2$, $C = 192$ K: 7	$N_2 = 3$, $C = 192$ K: 7
Stage 3 (14×14)	$N_3 = 5$, $C = 192$ K: 5	$N_3 = 6$, $C = 384$ K: 5	$N_3 = 16$, $C = 384$ K: 5
Stage 4 (7×7)	$N_4 = 2$, $C = 384$ K: 3	$N_4 = 2$, $C = 768$ K: 3	$N_4 = 2$, $C = 768$ K: 3
(1×1)	Average pool, 1000-d FC, Softmax		
#Param.	8M	32M	51M
#FLOPs	1.2G	4.8G	8.5G

Table A1. Model architecture specifications. N_i represents the number of DM blocks in the i-th stage. C represents the number of channels. K represents the kernel size of the DWConv in the Offset network. FC represents the fully connected layer.

B. More variants of object detection

To further validate the performance of our designed DefMamba variants on the object detection and instance segmentation task, we conducted experiments based on DefMamba-B. As shown in the Table B2.

Mask R-CNN 1× schedule							
Backbone	**#FLOPs**	**AP^b**	**AP^b_{50}**	**AP^b_{75}**	**AP^m**	**AP^m_{50}**	**AP^m_{75}**
ResNet-101 [15]	336G	38.2	58.8	41.4	34.7	55.7	37.2
Swin-S [24]	354G	44.8	66.6	48.9	40.9	63.4	44.2
ConvNeXt-S [25]	348G	45.4	67.9	50.0	41.8	65.2	45.1
VMamba-S [23]	400G	48.2	69.7	52.5	43.0	66.6	46.4
LocalVMamba-S [19]	414G	48.4	69.9	52.7	43.2	66.7	46.5
GrootV-S [38]	341G	48.6	70.3	53.5	43.6	67.5	47.1
DefMamba-B	349G	48.7	70.5	53.8	43.7	67.7	47.0

Table B2. Performance of Object Detection and Instance Segmentation Based on DefMamba-B on COCO. AP^b and AP^m indicate the mean Average Precision (mAP) of detection and segmentation.

As shown in the table above, our method still outperforms previous methods at the base scale.

C. More variants of semantic segmentation

To further validate the performance of our designed DefMamba variants on the semantic segmentation task, we conducted experiments based on DefMamba-B. As shown in the Table C3.

ADE20K with crop size 512				
Backbone	**mIOU (SS)**	**mIOU (MS)**	**#Param.**	**#FLOPs**
ResNet-101 [15]	42.9	44.0	85M	1030G
Swin-S [24]	47.6	49.5	81M	1039G
ConvNeXt-S [25]	46.0	46.7	60M	939G
VMamba-S [23]	49.5	50.5	76M	1081G
PlainMamba-L3 [42]	49.1	-	81M	419G
LocalVMamba-S [19]	50.0	51.0	81M	1095G
QuadMamba-B [39]	49.7	50.8	82M	1042G
GrootV-S [38]	50.7	51.7	-	1019G
DefMamba-B	50.8	51.7	84M	1024G

Table C3. Performance of Semantic Segmentation Based on DefMamba-B on ADE20K. SS and MS denote single-scale and multi-scale testing, respectively.

As shown in the table above, our method is comparable to previous methods at the base scale.