





VSA: Learning Varied-Size Window Attention in Vision Transformers

Supplementary Material

Qiming Zhang^{1*}, Yufei Xu^{1*}, Jing Zhang¹, and Dacheng Tao^{2,1}

¹ University of Sydney, Australia

² JD Explore Academy, China

{yuxu7116,qzha2506}@uni.sydney.edu.au,

jing.zhang1@sydney.edu.au, dacheng.tao@gmail.com

A Appendix

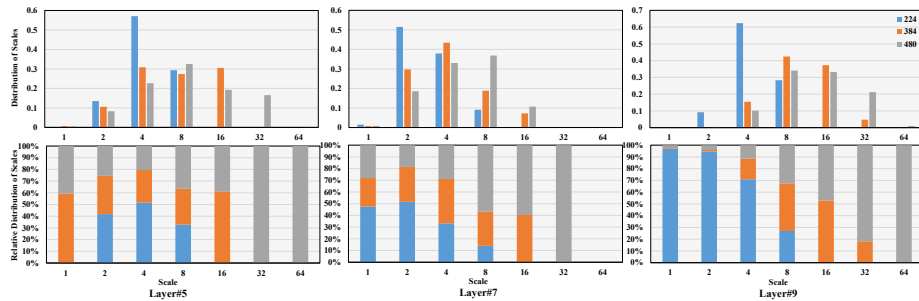


Fig. S1: The absolute and relative distribution of scales estimated by VSA with different input resolutions. The X-axis denotes the scale ratio of the varied-size window w.r.t. the default one. VSA generates the target windows at various scales to capture rich contextual information and the window size tends to become larger to adapt to large objects with the resolution going higher.

A.1 Implementation details

In this section we give details of the regression and sampling process. Different from previous work [3] that controls the regression distance by a predefined parameter, the proposed VSR module is hyper-parameter free. Denoting the coordinates of samples within one window as $\{(x_i, y_i) | i = 1 \dots n\}$ where i refers to the i_{th} token, we first disentangle the coordinates as

$$\begin{aligned} x_i &= x_i^{re} + x^{ce}, \\ y_i &= y_i^{re} + y^{ce}, \end{aligned} \tag{1}$$

* Equal contribution.

where (x^{ce}, y^{ce}) denote the center coordinates of the corresponding window and (x_i^{re}, y_i^{re}) are the relative coordinates w.r.t. the center. Then given the learned scales and offsets $s_w, o_w \in R^2$ of the corresponding window from Equation 3, they are normalized by multiplying the ratio between the window size and image size (also denoted as s_w, o_w for simplicity). This makes the VSR module learn how to expand and move the current window towards the optimal attention region by taking the window as a base. Then the tokens’ new coordinates within that window are calculated as

$$(x_i^*, y_i^*)^T = (x_i^{re}, y_i^{re})^T \cdot s_w + o_w + (x^{ce}, y^{ce})^T. \quad (2)$$

In the end, the key and value tokens are sampled according to the new coordinates (x_i^*, y_i^*) and fed into the following window-based attention layer.

Model	Params (M)	FLOPs (G)	Input Size	Training Set	ImageNet [5]		Real [1]
					Top-1	Top-5	Top-1
DeiT-S [12]	22	4.6	224	IN1k	81.2	95.4	86.8
PVT-S [14]	25	3.8	224	IN1k	79.8	-	-
ViL-S [17]	25	4.9	224	IN1k	82.4	-	-
PiT-S [8]	24	4.8	224	IN1k	80.9	-	-
TNT-S [7]	24	5.2	224	IN1k	81.3	95.6	-
MSG-T [6]	25	3.8	224	IN1k	82.4	-	-
Twins-PCPVT-S [4]	24	3.8	224	IN1k	81.2	-	-
Twins-SVT-S [4]	24	2.9	224	IN1k	81.7	-	-
T2T-ViT-14 [16]	22	5.2	224	IN1k	81.5	95.7	86.8
Swin-T [11]	29	4.5	224	IN1k	81.2	-	-
Swin-T+VSA	29	4.6	224	IN1k	82.3	96.1	87.5
ViTAEv2-S ¹ [18]	20	5.4	224	IN1k	82.2	96.1	87.5
ViTAEv2-S [18]	20	5.7	224	IN1k	82.6	96.2	87.6
ViTAEv2-S¹+VSA	20	5.6	224	IN1k	82.7	96.3	87.8
Swin-T [11]	29	14.2	384	IN1k	81.4	95.4	86.4
Swin-T+VSA	29	14.9	384	IN1k	83.2	96.5	88.0
Swin-T [11]	29	23.2	480	IN1k	81.5	95.7	86.3
Swin-T+VSA	29	24.0	480	IN1k	83.4	96.7	88.0
PiT-B [8]	74	12.5	224	IN1k	82.0	-	-
TNT-B [7]	66	14.1	224	IN1k	82.8	96.3	-
Focal-B [15]	90	16.0	224	IN1k	83.8	-	-
ViL-B [17]	56	13.4	224	IN1k	83.7	-	-
MSG-S [6]	56	8.4	224	IN1k	83.4	-	-
PVTv2-B5 [13]	82	11.8	224	IN1k	83.8	-	-
Swin-S [11]	50	8.7	224	IN1k	83.0	-	-
Swin-B [11]	88	15.4	224	IN1k	83.3	-	88.0
Shuffle-S [9]	50	8.9	224	IN1k	83.5	-	-
Swin-S+VSA	50	8.9	224	IN1k	83.6	96.6	88.4
ViTAEv2-48M	49	13.3	224	IN1k	83.8	96.6	88.4
ViTAEv2-48M¹+VSA	50	13.0	224	IN1k	83.9	-	-
Swin-B [11]	88	15.4	224	83.3	-	88.0	
Swin-B+VSA	88	16.0	224	83.9	96.7	88.6	
ViTAEv2-48M¹+VSA	50	13.0	224	IN22k+IN1k	84.9	-	-
ViTAEv2-B	90	24.3	224	IN22k+IN1k	86.1	97.9	89.9
ViTAEv2-B¹+VSA	94	23.9	224	IN22k+IN1k	86.2	97.9	90.0

¹ The full window version.

Table S1: Image classification results on ImageNet. ‘Input Size’ denotes the image size used for training and test. ‘IN1k’ and ‘IN22k’ refer to ImageNet-1k and ImageNet-22k datasets respectively.

A.2 Visual inspection

Statistics of target windows’ scales. In this part, we analyze the behavior of VSA in regressing the target windows when dealing with input images of different resolutions. Specifically, we first scale the input images to different resolutions, *i.e.*, 224×224 , 384×384 , and 480×480 and train three corresponding models respectively, whose performance has been reported in Table S1. Then, we randomly sample images from the ImageNet validation set and feed them into the three models respectively to count the scale distribution of all target windows estimated by VSR. We plot the results from the 5th, 7th, and 9th layers in Figure S1, where the X-axis denotes the scale ratio between the target windows and the default one. The Y-axis in the first row denotes the percentage of corresponding scales. For better visualization, we normalize the data independently along the Y-axis w.r.t. each scale on the X-axis, *i.e.*, obtaining the relative distribution as shown at the bottom of the figure. As can be seen, larger windows gradually dominate the VSA when the input size increases from 224×224 to 480×480 , showing the good ability of VSA in adapting to different input sizes.

A.3 Classification results

We present the classification results of various model in Table S1. As shown in the table, the proposed VSA method can consistently improve the classification results. It is noted that with the help of VSA, the full window version of ViTAEv2 outperforms its counterpart using full attention in the last two stages [18], which has much computation cost when the input images become larger as shown in [18]. Besides, when pretraining using the ImageNet-22k dataset, the full window version of ViTAEv2-B reaches 86.2% and 90.0% Top-1 accuracy on ImageNet-1k and ImageNet-1k Real datasets respectively.

	Params (M)	Cascade RCNN 1x						Cascade RCNN 3x					
		AP ^{bb}	AP ₅₀ ^{bb}	AP ₇₅ ^{bb}	AP ^{mk}	AP ₅₀ ^{mk}	AP ₇₅ ^{mk}	AP ^{bb}	AP ₅₀ ^{bb}	AP ₇₅ ^{bb}	AP ^{mk}	AP ₅₀ ^{mk}	AP ₇₅ ^{mk}
R50	82	44.3	62.7	48.4	38.3	59.7	41.2	46.3	64.3	50.5	40.1	61.7	43.4
Swin-T	86	48.1	67.1	52.2	41.7	64.4	45.0	50.2	69.2	54.7	43.7	66.6	47.3
Swin-T+VSA	86	49.8	69.0	54.1	43.0	66.2	46.4	51.3	70.3	55.8	44.6	67.6	48.1
ViTAEv2-S ¹	77	47.3	66.0	51.5	40.6	63.0	43.7	48.0	65.7	52.5	41.3	63.1	45.0
ViTAEv2-S¹+VSA	77	49.8	69.2	54.0	43.1	66.4	46.5	51.9	70.6	56.2	44.8	68.1	48.5

¹ The full window version.

Table S2: Object detection results on MS COCO with Cascade RCNN.

A.4 Downstream task results

We further present the results of Swin [11] and ViTAEv2 [18] with the proposed VSR module on MS COCO dataset [10] for detection tasks. ViTAEv2 with window attention for all stages, *i.e.*, the full window attention version, is adopted as default. We evaluate their detection performance Cascade RCNN [2] frameworks

	CascadeRCNN 3x								
	AP^{bb}	AP_s^{bb}	AP_m^{bb}	AP_l^{bb}	AP^M	AP_s^M	AP_m^M	AP_l^M	fps
Swin-T	50.4	33.8	54.1	65.2	43.7	27.3	47.5	59.0	25.2
Swin-T+VSA	51.4	35.5	54.7	66.0	44.7	28.7	48.1	59.8	22.5
Swin-S	51.9	35.2	55.7	67.7	45.0	28.8	48.7	60.6	16.0
Swin-S+VSA	52.7	36.7	56.0	68.3	45.6	29.8	49.0	61.2	14.1
Swin-B	51.9	35.4	55.2	67.4	45.0	28.9	48.3	60.4	14.4
Swin-B+VSA	52.9	36.8	56.4	68.4	45.9	30.1	49.3	61.5	12.8

Table S3: More object detection results with Cascade RCNN.

with both $1\times$ and $3\times$ settings. The detection results are available in Table S2 and Table S3 (Cascade RCNN). The results in both tables imply that VSA can successfully boost the detection performance with Cascade RCNN frameworks. Besides, the performance gain keeps when scaling to models with more parameters, *e.g.*, Swin-S and Swin-B [11] with 50M and 88 parameters, respectively, as demonstrated in Table S3. In addition, from Table S3, we can see that the improvement on small and big objects significantly exceeds the median ones of the baseline, which supports our claim that VSA can better deal with objects of different sizes.

References

1. Beyer, L., Hénaff, O.J., Kolesnikov, A., Zhai, X., Oord, A.v.d.: Are we done with imagenet? arXiv preprint arXiv:2006.07159 (2020) [2](#)
2. Cai, Z., Vasconcelos, N.: Cascade r-cnn: Delving into high quality object detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 6154–6162 (2018) [3](#)
3. Chen, Z., Zhu, Y., Zhao, C., Hu, G., Zeng, W., Wang, J., Tang, M.: Dpt: Deformable patch-based transformer for visual recognition. In: Proceedings of the 29th ACM International Conference on Multimedia. pp. 2899–2907 (2021) [1](#)
4. 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: Advances in Neural Information Processing Systems (2021) [2](#)
5. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 248–255. Ieee (2009) [2](#)
6. Fang, J., Xie, L., Wang, X., Zhang, X., Liu, W., Tian, Q.: Msg-transformer: Exchanging local spatial information by manipulating messenger tokens. arXiv preprint arXiv:2105.15168 (2021) [2](#)
7. Han, K., Xiao, A., Wu, E., Guo, J., Xu, C., Wang, Y.: Transformer in transformer. Advances in Neural Information Processing Systems **34** (2021) [2](#)
8. Heo, B., Yun, S., Han, D., Chun, S., Choe, J., Oh, S.J.: Rethinking spatial dimensions of vision transformers. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) (2021) [2](#)
9. Huang, Z., Ben, Y., Luo, G., Cheng, P., Yu, G., Fu, B.: Shuffle transformer: Rethinking spatial shuffle for vision transformer. arXiv preprint arXiv:2106.03650 (2021) [2](#)
10. 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: Proceedings of the European Conference on Computer Vision (ECCV). pp. 740–755. Springer (2014) [3](#)
11. 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 (ICCV). pp. 10012–10022 (2021) [2](#), [3](#), [4](#)
12. Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A., Jegou, H.: Training data-efficient image transformers; distillation through attention. In: International Conference on Machine Learning. PMLR (2021) [2](#)
13. Wang, W., Xie, E., Li, X., Fan, D.P., Song, K., Liang, D., Lu, T., Luo, P., Shao, L.: Pvtv2: Improved baselines with pyramid vision transformer (2021) [2](#)
14. 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 (ICCV). pp. 568–578 (2021) [2](#)
15. Yang, J., Li, C., Zhang, P., Dai, X., Xiao, B., Yuan, L., Gao, J.: Focal attention for long-range interactions in vision transformers. In: Advances in Neural Information Processing Systems (2021) [2](#)
16. Yuan, L., Chen, Y., Wang, T., Yu, W., Shi, Y., Jiang, Z.H., 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 (ICCV). pp. 558–567 (2021) [2](#)

17. Zhang, P., Dai, X., Yang, J., Xiao, B., Yuan, L., Zhang, L., Gao, J.: Multi-scale vision longformer: A new vision transformer for high-resolution image encoding. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 2998–3008 (October 2021) [2](#)
18. Zhang, Q., Xu, Y., Zhang, J., Tao, D.: Vitaev2: Vision transformer advanced by exploring inductive bias for image recognition and beyond. arXiv preprint arXiv:2202.10108 (2022) [2](#), [3](#)