BiFormer: Vision Transformer with Bi-Level Routing Attention - Supplemental

Lei Zhu[1] Xinjiang Wang[2] Zhanghan Ke[1] Wayne Zhang[2] Rynson Lau[1†]
[1] City University of Hong Kong [2] SenseTime Research
{lzhu68-c,zhanghake2-c}@my.cityu.edu.hk, {wangxinjiang,wayne.zhang}@sensetime.com
Rynson.Lau@cityu.edu.hk

A. Discussion on Regional Representations

In our proposed bi-level routing attention, we derive the regional representations ($\mathbf{Q}^r$ and $\mathbf{K}^r$) with average pooling for region-to-region routing. We justify the choice here.

In fact, as the goal of region-to-region routing is to find the most related tokens for token-to-token attention in the next step, it is reasonable to maximize *the average token-to-token affinity scores between the two regions*. However, this is equivalent to maximizing *the affinity score between the average tokens of the two regions*, because

$$\frac{1}{|\Omega| \cdot |\Omega'|} \sum_{i \in \Omega} \sum_{j \in \Omega'} \mathbf{Q}_i \mathbf{K}_j = \frac{\sum_{i \in \Omega} \mathbf{Q}_i}{|\Omega|} \cdot \frac{\sum_{j \in \Omega'} \mathbf{K}_j}{|\Omega'|}, \quad (1)$$

where we denote the set of token indices of the two regions with Ω and Ω'.

B. Throughput Comparison

To demonstrate the computation efficiency of the proposed bi-level routing attention, we compare the throughputs of models using different attention mechanisms. Specifically, we replace the shift window attention modules in Swin-T [4] with quad-tree attention [5] modules to form QuadTree-STL, and with our bi-level routing attention modules to form BiFormer-STL. We then use the widely used timm [7] script to benchmark the training and inference throughput on a 32 GB Tesla V100 GPU with a batch size of 128 and image resolution of 224×224.

As shown in Figure 1, Swin-T has the highest throughput due to its simplicity. Switching to our bi-level routing attention(BRA), the training and inference throughput of BiFormer-STL decrease by $\sim$30% and $\sim$40% respectively in comparison with Swin-T. This is caused by extra GPU kernel launch and memory transactions caused by the routing process (*i.e.* locating the regions to attend and gather key-value pairs). Nonetheless, BiFormer-STL is still $3\times \sim 6\times$ faster than QuadTree-STL. This is due to that on the one hand the recursive nature of quad-tree attention hurts the parallelism, on the other hand quad-tree attention

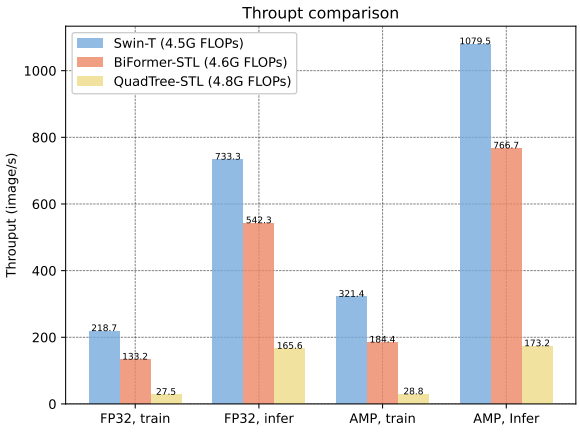

Figure 1. Throughput comparison on a 32GB Tesla V100 GPU. The suffix "STL" denotes **S**win-**T** **L**ayout, which means we use Swin-T [4] backbone with only attention module being replaced. We report results under both FP32 precision and automatic mixed precision (AMP) modes.

relies on sparse matrix multiplications which are inefficient on GPUs, while our BRA can be efficiently implemented with key-value token gathering followed by GPU-friendly dense matrix multiplications.

It is worth noting that, the overheads of both memory transactions and kernel launch incurred by the routing process can be reduced via engineering efforts such as GPU kernel fusion. We leave this optimization to our future work.

C. Choices of top-k and partition factor S

In the paper, S and k were chosen more with consideration of engineering issues. **(1)** S is chosen as a divisor of the training size to avoid padding, which slows down the training and may also degrade the performance. For example, in image classification where the resolution is $224 = 7 \times 32$, we use $S = 7$ so that it is a divisor of the size of feature maps in every stage. This is similar to SWinTransformer [4], which uses a window size of 7. **(2)** In dense

S	k	#tokens to attend	Acc	im/s (FP32)
7	1,4,16,49	64,64,64,49	**82.7**	522.3
7	1,2,8,32	64,32,32,32	82.4	563.2
7	2,8,32,49	128,128,128,49	82.6	419.9
8,4,2,1	2,2,2,1	98,98,98,49	82.3	**606.2**

Table 1. Ablation study on top-k and partition factor S.

prediction tasks, we use larger S to balance the complexity of region-level routing and token-level attention to achieve overall lower complexity. One can find hints from Eq. 9 of the paper, though we do not strictly follow the scaling rule due to the size divisor constraint. **(3)** We gradually increase k to keep a reasonable number of tokens to attend as the region size becomes smaller in later stages.

It is possible to try different combinations of S and k. We show ablation results on IN-1K in Table 1, based on BiFormer-STL (as in the paper). A key observation from these experiments is that increasing the number of tokens to attend may even hurt the accuracy. This implies the explicit sparsity constraint may serve as a regularization to avoid distractions from the background.

D. Adapting Pretrained Plain ViT with BRA

Recently, to take advantage of large-scale pretraining with masked image modeling, a new research direction emerges to adapt plain ViT [2] for dense prediction tasks [1, 3]. Here we explore adapting pre-trained plain ViT [2] for semantic segmentation with our proposed BRA.

Specifically, we replace all or part of full multi-head self-attention (MHSA) modules in DeiT-B [6] with our BRA and directly load the weights pre-trained on ImageNet before training on ADE20K dataset for semantic segmentation. In this way, the linear projection weights of BRA modules are initialized with those of the original MHSA. We compare such an adaptation with those proposed in [3], *i.e.* using local window attention (window size $w = 14$) together with several global attention or convolution propagation blocks. We set window size $w = 4$ (which is equivalent to region partition size $S = 8$ since the feature map has a resolution of 32×32) and the number of regions to attend $k = 12$, hence each query attends to $4^2 \times 12 = 192$ key-value pairs, which is comparable to the local window attention where each query attends to $14 \times 14 = 196$ key-value pairs.

Table 2 shows the results. Without propagation blocks, the architecture using BRA significantly surpasses the one with local window attention by 2.4 mAP. When further equipped with 4 global propagation blocks, the performance of both architectures is improved, while the one using BRA still has an advantage of 0.2 mAP.

attention function	mIoU(%)
local window attention ($w = 14$)	43.55
BRA($w = 4, k = 12$)	**45.92**
local window attention + 4 conv prop. blks.	44.68
local window attention + 4 global prop. blks.	46.64
BRA + 4 global prop. blks.	**46.84**

Table 2. Adapting pretrained ViT [2] with BRA for semantic segmentation on ADE20K. For decoder, we use the Simple Feature Pyramid [3] followed by with Upernet [8] head.

E. More Visualization Results

To further show how BRA works, we demonstrate more visualization results in Figure 2.

References

[1] Zhe Chen, Yuchen Duan, Wenhai Wang, Junjun He, Tong Lu, Jifeng Dai, and Yu Qiao. Vision transformer adapter for dense predictions. *arXiv preprint arXiv:2205.08534*, 2022. 2

[2] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations, ICLR 2021, 2021*, 2021. 2

[3] Yanghao Li, Hanzi Mao, Ross Girshick, and Kaiming He. Exploring plain vision transformer backbones for object detection. *arXiv preprint arXiv:2203.16527*, 2022. 2

[4] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10012–10022, 2021. 1

[5] Shitao Tang, Jiahui Zhang, Siyu Zhu, and Ping Tan. Quadtree attention for vision transformers. In *The International Conference on Learning Representations, ICLR 2022, 2022*, 2022. 1

[6] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning*, pages 10347–10357. PMLR, 2021. 2

[7] Ross Wightman. Pytorch image models. `https://github.com/rwightman/pytorch-image-models`, 2019. 1

[8] Tete Xiao, Yingcheng Liu, Bolei Zhou, Yuning Jiang, and Jian Sun. Unified perceptual parsing for scene understanding. In *Proceedings of the European conference on computer vision (ECCV)*, pages 418–434, 2018. 2

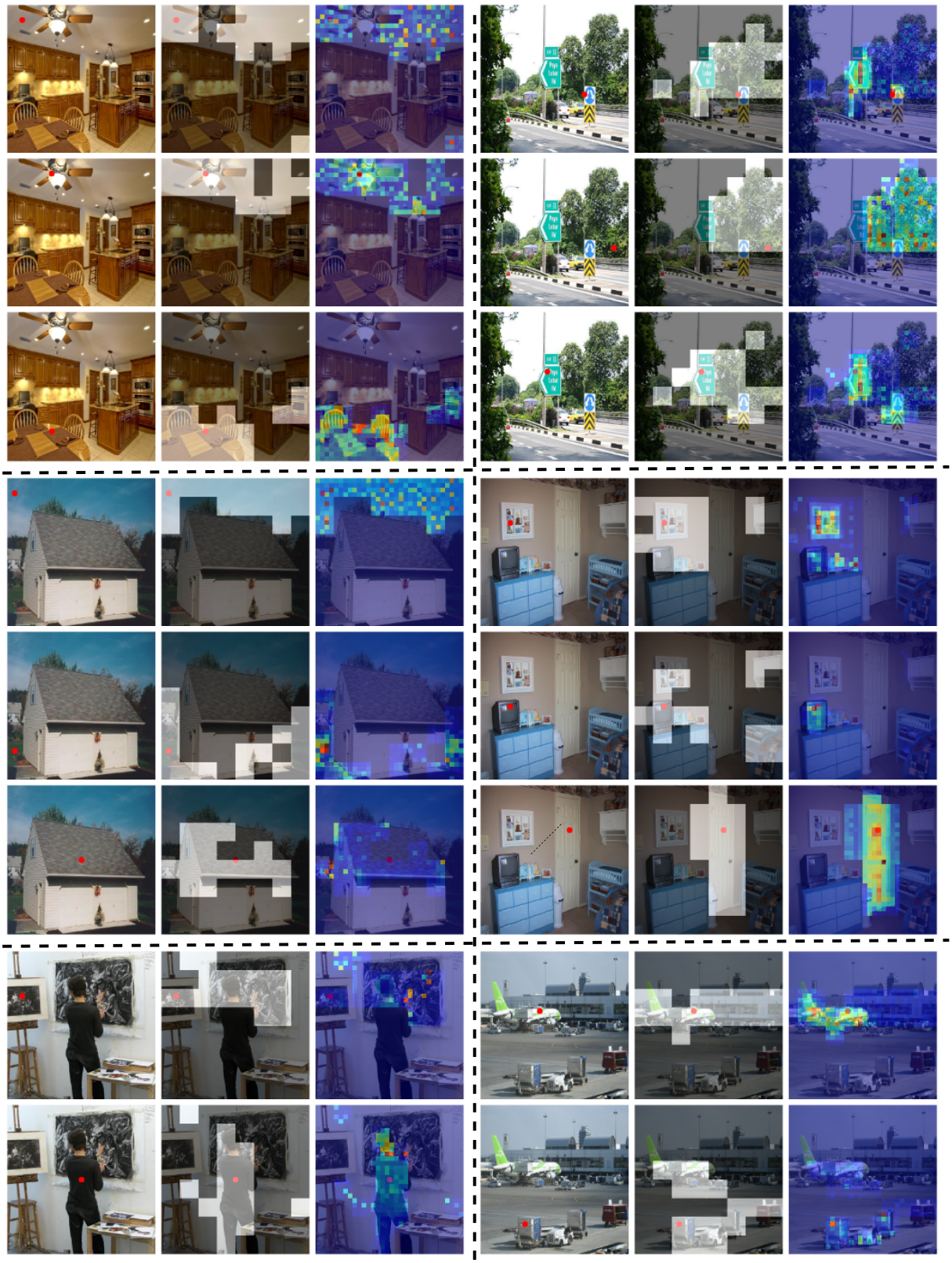

Figure 2. More attention map visualization results. For each scene, We demonstrate 2-3 query positions on the input image (left), corresponding routed regions (middle) and final attention heat map (right).