# Doubly-Fused ViT: Fuse Information from Vision Transformer Doubly with Local Representation

Li Gao, Dong Nie, Bo Li, and Xiaofeng Ren

Alibaba Group
{liangliang.gl,dong.nie,shize.lb,x.ren}@alibaba-inc.com

This supplementary material provides details, experimental results on downstream tasks and additional visual results that could not be included in the paper submission due to space limitation. In Sec. A, we first provide more details about several model architectures (i.e., DFvT-T, DFvT-S and DFvT- B). Then, we demonstrate the effectiveness and efficiency of DFvT by providing experimental results on downstream tasks (in Sec. B and C). Sec. D provides some visual results to show what we have learned in the features.

## A Model Variants

We construct models across a spectrum of different model sizes by simply changing the number of Transformer blocks at each stage, the hidden feature dimension and the MLP ratio of increasing dimension, termed DFvT-T, DFvT-S, and DFvT-B, which represents the tiny, small and base models, respectively. The details of these models are shown in Table 1.

## B Object Detection on COCO

The object detection experiments are conducted on COCO 2017 detection dataset [3], which covers 80 categories and contains 80k training images, 30k testing images, 5k validation images. We report the final results on the validation subset

| | Output Size | DFvT-T | | DFvT-S | | DFvT-B | |
|---|---|---|---|---|---|---|---|
| | | Context Module | Spatial Module | Context Module | Spatial Module | Context Module | Spatial Module |
| Patch Stem | 56 x 56 | $\begin{bmatrix} 3\times3, stride\ 2, 24 \\ 3\times3, stride\ 2, 48 \end{bmatrix}$ | | $\begin{bmatrix} 3\times3, stride\ 2, 48 \\ 3\times3, stride\ 2, 96 \end{bmatrix}$ | | $\begin{bmatrix} 3\times3, stride\ 2, 64 \\ 3\times3, stride\ 2, 128 \end{bmatrix}$ | |
| Stage 1 | 28 x 28 | $maxpool\ 3\times3, stride\ 2$ $\begin{bmatrix} 1\times1, stride\ 1, 48 \\ 3\times3, stride\ 2, 48 \\ 1\times1, stride\ 1, 48 \end{bmatrix} \begin{bmatrix} h_1=3, r=4 \end{bmatrix} \times 1$ | $[3\times3, stride\ 1, 48]$ | $maxpool\ 3\times3, stride\ 2$ $\begin{bmatrix} 1\times1, stride\ 1, 96 \\ 3\times3, stride\ 2, 96 \\ 1\times1, stride\ 1, 96 \end{bmatrix} \begin{bmatrix} h_1=3, r=2 \end{bmatrix} \times 1$ | $[3\times3, stride\ 1, 96]$ | $maxpool\ 3\times3, stride\ 2$ $\begin{bmatrix} 1\times1, stride\ 1, 128 \\ 3\times3, stride\ 2, 128 \\ 1\times1, stride\ 1, 128 \end{bmatrix} \begin{bmatrix} h_1=4, r=2 \end{bmatrix} \times 2$ | $[3\times3, stride\ 1, 128]$ |
| Patch Aggr. | 28 x 28 | $1\times1, stride\ 1, 96$ | | $1\times1, stride\ 1, 192$ | | $1\times1, stride\ 1, 256$ | |
| Stage 2 | 14 x 14 | $maxpool\ 3\times3, stride\ 2$ $\begin{bmatrix} 1\times1, stride\ 1, 96 \\ 3\times3, stride\ 2, 96 \\ 1\times1, stride\ 1, 96 \end{bmatrix} \begin{bmatrix} h_2=6, r=4 \end{bmatrix} \times 1$ | $[3\times3, stride\ 1, 96]$ | $maxpool\ 3\times3, stride\ 2$ $\begin{bmatrix} 1\times1, stride\ 1, 192 \\ 3\times3, stride\ 2, 192 \\ 1\times1, stride\ 1, 192 \end{bmatrix} \begin{bmatrix} h_2=6, r=2 \end{bmatrix} \times 1$ | $[3\times3, stride\ 1, 192]$ | $maxpool\ 3\times3, stride\ 2$ $\begin{bmatrix} 1\times1, stride\ 1, 256 \\ 3\times3, stride\ 2, 256 \\ 1\times1, stride\ 1, 256 \end{bmatrix} \begin{bmatrix} h_2=8, r=2 \end{bmatrix} \times 2$ | $[3\times3, stride\ 1, 256]$ |
| Patch Aggr. | 14 x 14 | $1\times1, stride\ 1, 192$ | | $1\times1, stride\ 1, 384$ | | $1\times1, stride\ 1, 512$ | |
| Stage 3 | 7 x 7 | $maxpool\ 3\times3, stride\ 2$ $\begin{bmatrix} 1\times1, stride\ 1, 192 \\ 3\times3, stride\ 2, 192 \\ 1\times1, stride\ 1, 192 \end{bmatrix} \begin{bmatrix} h_3=12, r=4 \end{bmatrix} \times 2$ | $[3\times3, stride\ 1, 192]$ | $maxpool\ 3\times3, stride\ 2$ $\begin{bmatrix} 1\times1, stride\ 1, 384 \\ 3\times3, stride\ 2, 384 \\ 1\times1, stride\ 1, 384 \end{bmatrix} \begin{bmatrix} h_3=12, r=2 \end{bmatrix} \times 2$ | $[3\times3, stride\ 1, 384]$ | $maxpool\ 3\times3, stride\ 2$ $\begin{bmatrix} 1\times1, stride\ 1, 512 \\ 3\times3, stride\ 2, 512 \\ 1\times1, stride\ 1, 512 \end{bmatrix} \begin{bmatrix} h_3=16, r=2 \end{bmatrix} \times 6$ | $[3\times3, stride\ 1, 512]$ |
| Patch Aggr. | 7 x 7 | $1\times1, stride\ 1, 384$ | | $1\times1, stride\ 1, 768$ | | $1\times1, stride\ 1, 1024$ | |
| Stage 4 | 7 x 7 | $\begin{bmatrix} 1\times1, stride\ 1, 384 \\ 3\times3, stride\ 1, 384 \\ 1\times1, stride\ 1, 384 \end{bmatrix} \begin{bmatrix} h_4=24, r=4 \end{bmatrix} \times 1$ | - | $\begin{bmatrix} 1\times1, stride\ 1, 768 \\ 3\times3, stride\ 1, 768 \\ 1\times1, stride\ 1, 768 \end{bmatrix} \begin{bmatrix} h_4=24, r=2 \end{bmatrix} \times 1$ | - | $\begin{bmatrix} 1\times1, stride\ 1, 1024 \\ 3\times3, stride\ 1, 1024 \\ 1\times1, stride\ 1, 1024 \end{bmatrix} \begin{bmatrix} h_4=32, r=2 \end{bmatrix} \times 2$ | - |
| Classifier | 1 x 1 | $\begin{bmatrix} avgpool \\ 1\times1, stride\ 1, 1000 \end{bmatrix}$ | | $\begin{bmatrix} avgpool \\ 1\times1, stride\ 1, 1000 \end{bmatrix}$ | | $\begin{bmatrix} avgpool \\ 1\times1, stride\ 1, 1000 \end{bmatrix}$ | |
| FLOPs | | 0.3G | | 0.8G | | 2.5G | |

**Table 1.** Detailed architecture specifications. $h_i$ denotes the number of head in block $i$, and $r$ represents the up-dimensioning ratio of mlp block.

of bounding box AP and mask AP with the representative Mask R-CNN [2] framework in Table 2.

| COCO 2017 | | | | | | | |
|---|---|---|---|---|---|---|---|
| Method | Backbone | $AP^{box}$ | $AP^{mask}$ | Params(M) | FLOPs(G) | FPS | Memory |
| SSDLite | MobileNetV2† | 22.1 | - | 4.3 | - | - | - |
| Mask RCNN | DFvT-T | 32.9 | 30.7 | 25 | 178 | 39.3 | 2.0 |
| Mask RCNN | DFvT-S | 38.0 | 35.4 | 32 | 198 | 31.5 | 2.5 |
| Mask RCNN | ResNet50† | 37.8 | 34.2 | 44 | 260 | 22.6 | 4.0 |
| Mask RCNN | ResNet101† | 40.0 | 36.1 | 63 | 336 | 16.6 | 6.4 |
| Mask RCNN | PVT-Medium† | 42.0 | 39.0 | 64 | 392 | - | - |
| Mask RCNN | Swin-Tiny† | 43.7 | 39.8 | 48 | 267 | 16.0 | 5.9 |
| Mask RCNN | DFvT-B | 43.1 | 39.2 | 58 | 242 | 22.7 | 3.9 |

**Table 2.** Object detection on COCO 2017 with Mask R-CNN framework. † denotes that the results are reported by their original papers or mmdetection library. The batch size is set to 2 when test FPS and memory on a single NVIDAI 2080 Ti GPU. When computing FLOPs, the input size is set to $1280 \times 800$.

Our DFvT-B backbone achieves 43.1 mAP on the object detection task and 39.2 mAP on the instance segmentation task, which is 3.1 higher than that of ResNet101 backbone with much fewer computational costs and faster speed. It is worth noting that our DFvT is very competitive considering inference speed and memory cost. For example, DFvT-B is as fast as ResNet50 (e.g., FPS 22.7) and occupies less memory (e.g.3.9G) with exceeding it 5.3 in box mAP. Moreover, DFvT-B achieves compelling results compared to Swin-Tiny with much less cost, that is, our FPS is 42% faster and memory occupation is 34% less.

## C Semantic Segmentation on ADE20K

ADE20K [5] is a widely-used dataset, dense pixel-level semantic annotations are provided for 20,210 training images, 2,000 validation images, and 3,000 testing images. ADE20K contains 150 semantic categories with a broad range. In the experiment setup, we utilize UperNet [4] as the semantic segmentation framework along with our DFvT-B to get final segmentation prediction maps. The results are presented in Table 3.

The mIoU of DFvT-B with UperNet on the validation set is 44.7, while that with Resnet101 and Swin-Tiny are 42.0 and 44.5, respectively. It can be seen that our proposed method is also efficient in semantic segmentation in terms of computation cost, speed and memory.

## D Visual Results

We provide visual inspection results of DFvT-B using Grad-CAM++ [1] in Figure 1. The visualization of the features shows which important regions our model

| ADE20K | | | | | | |
|---|---|---|---|---|---|---|
| Method | Backbone | val mIoU | Params(M) | FLOPs(G) | FPS | Memory |
| UperNet | MobileNetV2 | 34.8 | 31 | - | - | - |
| UperNet | DFvT-T | 36.1 | 34 | 849 | 9.0 | 7.7 |
| UperNet | DFvT-S | 40.7 | 44 | 874 | 8.6 | 8.1 |
| UperNet | ResNet50† | 40.4 | 67 | 952 | 7.6 | 8.3 |
| UperNet | ResNet101† | 42.0 | 86 | 1029 | 6.8 | OOM |
| UperNet | Swin-Tiny† | 44.5 | 60 | 945 | 6.7 | 9.5 |
| UperNet | DFvT-B | 44.7 | 72 | 925 | 7.7 | 9.4 |

**Table 3.** Semantic segmentation on ADE20K. The batch size is set to 4 when testing speed and memory cost in a single NVIDIA 2080 Ti GPU, and the image size is set to $2048 \times 512$ for FLOPs computing. † means that the mIoU results are copied from the papers or mmsegmentation library.

classifies based on. As can be seen in the figure, our proposed DFvT-B can accurately locate the targets and focus more attention on identifiable features. This can be attributed to our specially designed context and spatial modules. In particular, the context module can handle the entire concept of the object, and the spatial module can well learn fine-grained features to accurately locate the object.

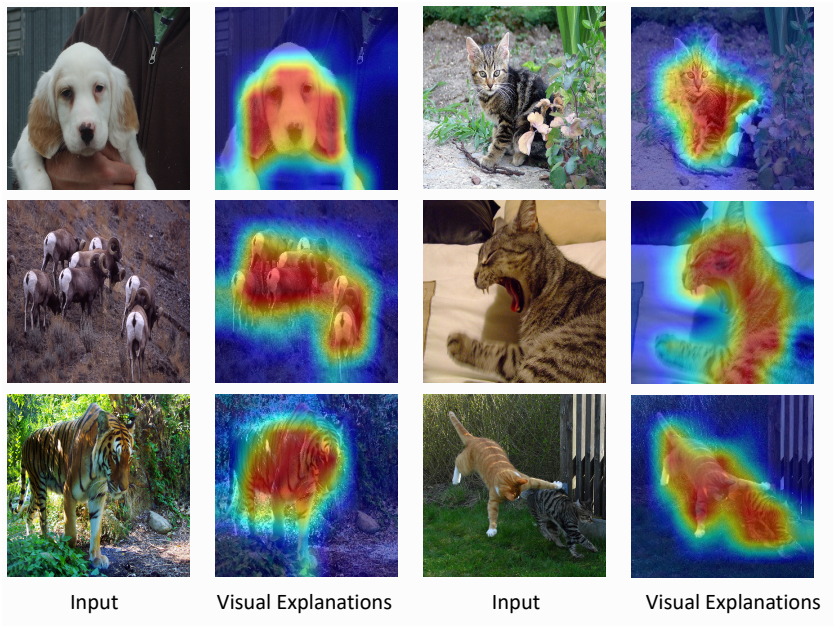


**Fig. 1.** Visualization of features of DFvT-B.

## References

1. Chattopadhay, A., Sarkar, A., Howlader, P., Balasubramanian, V.N.: Grad-cam++: Generalized gradient-based visual explanations for deep convolutional networks. In: Proceedings of the IEEE Winter Conference on Applications of Computer Vision. pp. 839–847 (2018)
2. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask r-cnn. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 2961–2969 (2017)
3. 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. pp. 740–755 (2014)
4. Xiao, T., Liu, Y., Zhou, B., Jiang, Y., Sun, J.: Unified perceptual parsing for scene understanding. In: Proceedings of the European Conference on Computer Vision. pp. 418–434 (2018)
5. Zhou, B., Zhao, H., Puig, X., Xiao, T., Fidler, S., Barriuso, A., Torralba, A.: Semantic understanding of scenes through the ade20k dataset. International Journal of Computer Vision **127**(3), 302–321 (2019)