Make nnUNets Small Again

Mattias P. Heinrich HEINRICH@IMI.UNI-LUEBECK.DE
Institute of Medical Informatics, University of Luebeck, Germany

Jannis Hagenah JANNIS.HAGENAH@ENG.OX.AC.UK
Department of Engineering Science, University of Oxford, UK

Abstract

Automatic high-quality segmentations have become ubiquitous in numerous downstream tasks of medical image analysis, i.e. shape-based pathology classification or semantically guided image registration. Public frameworks for 3D U-Nets provide numerous pre-trained models for nearly all anatomies in CT scans. Yet, the great generalisation comes at the cost of very heavy networks with millions of parameter and trillions of floating point operations for every single model in even larger ensembles. We present a novel combination of two orthogonal approaches to lower the computational (and environmental) burden of U-Nets: namely partial convolution and structural re-parameterization that tackle the intertwined challenges while keeping real world latency small.

Keywords: 3D semantic segmentation, model distillation, efficient convolutions

1. Introduction

When designing convolutional network architectures, the recent trend has moved away from the classic VGG style (Simonyan and Zisserman, 2014) that uses few plain 3×3 **spatial convolutions** with large channel sizes and carries a large parameter count. Instead, depth-separable or group convolutions are frequently seen (Sandler et al., 2018; Ma et al., 2018) in an effort to balance model size, multiply-add operations (MADs) and accuracy. In fact, the recent ConvNeXt architecture (Liu et al., 2022) could demonstrate that replacing VGG-style or ResNet-like convolutional blocks can outperform vision transformers with a very low MADs count. They achieve this by using a clever design of the convolutional block with a large kernel depth-separable convolution followed by an inverted bottleneck with 1×1 **pointwise convolutions** together with micro-design improvements.

Unfortunately, as shown in (Chen et al., 2023; Vasu et al., 2022) the number of raw operations is poorly correlated with latency (runtime) due to the much lower efficiency of modern parallel processors that are optimised for dense 3×3 convolutions with large channel sizes to reach impressive theoretical FLOPS (floating points per second)[1] of more than 10 trillion (10 TFLOPS). Our experimental validation confirms, that in an effort to lower the MADs the GPU utilisation (and hence efficiency) also massively drops leading to diminishing returns or in the worst case even longer runtimes for smaller models. A default configuration of the nnUNet (Isensee et al., 2021) typically requires >30 million parameters and >1 trillion MADs to segment a single 3D patch. Nevertheless, the training and inference algorithms are considered to be relatively efficient, due to the aforementioned high GPU utilisation of modern Nvidia GPUs.

1. Note that we purposefully use MADs and FLOPS to differentiate between computational work load and the hardware GPU capabilities at 100% utilisation.

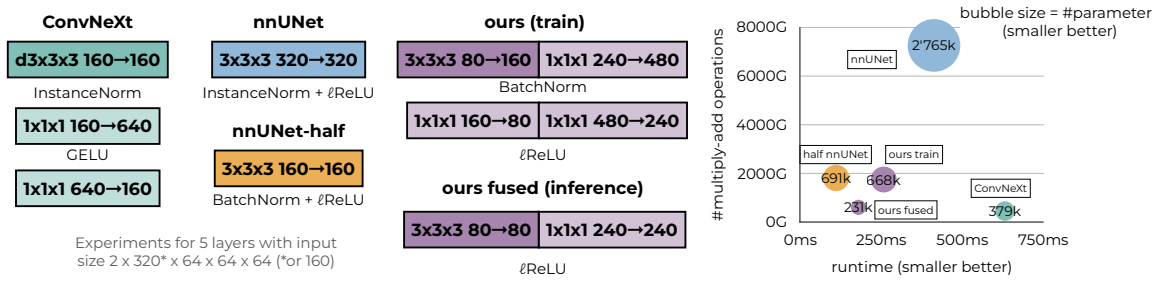


Figure 1: Concept of proposed FasterFusion block and measured inference runtimes of various choices for (grouped) spatial and pointwise convolution operators. Our concept yields a high GPU utilisation ≈54% close to $3 \times 3 \times 3$ convolutions, whereas ConvNeXt fails to convert a lower operations into inference speed.

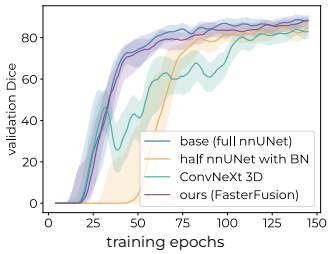
Contribution: In this work, we present a novel combination of two orthogonal approaches that tackle the computational burden and make nnUNets small again. Our model employs a new variant of *T-shaped* spatial convolutions that act only on a part of the channels individually together with full-depth pointwise convolutions. Different to (Chen et al., 2023) we also incorporate a novel version of re-parameterisation as popularised by RepVGG (Ding et al., 2021) that enables the fusion of the inverted bottleneck **FasterFusion**. Combined these contributions lead to 3-4× smaller model sizes and 2× faster inference times, while matching the accuracy and stable training convergence of full-sized models.

2. Methods

Building upon T-shaped spatial convolution (Chen et al., 2023) that perform the spatial $3 \times 3 \times 3$ kernels only on a part (in our work a quarter) of channels and use pointwise operators for the remaining one, we aim to find a good balance between reducing parameters, keeping a reasonable number of computations for training and a method that yields the fastest speed at inference. Our approach uses an inverted bottleneck with an intermediate doubled channel size to limit peak memory when using the same number of input and output channels as the blocks within the nnUNet (Isensee et al., 2021). In contrast to (Chen et al., 2023), we place a BatchNorm and no non-linearity between first and second convolution in our block. This enables us to apply re-parameterisation after training and completely fuse all three consecutive layers at inference and reduce the parameters by 2.9×. Our method can be used as drop-in replacement in any 2D or 3D convolutional network, yet in this first proof-of-concept we restrict ourselves to the popular 3D semantic segmentation architecture of the nnUNet.

Fig. 1 demonstrates the disproportionated efficiency of plain $3 \times 3 \times 3$ spatial convolutions with large channel size in comparison to depth-separable, groupwise and pointwise convolutions (when comparing nnUNet with ConvNeXt). Details on the implementation of training and fusion, which only requires us to perform a number of matrix multiplications on the respective weights once after training, are found in our open source code:

https://github.com/mattiaspaul/makennunetsmallagain. Intuitively, a larger number of trainable parameters and floating point operations will ease training whereas the fusion of blocks - *re-parameterisation* - increases efficiency for inference and substantially reduces the size of models to be stored.



	Dice Avg (Std)	Percentiles 25/50/75	Parameters
baseline nnUNet	89.7 (16.8)	92/95/97	30'600k
half nnUNet	74.9 (26.0)	68/88/94	7'656k
ConvNeXt	88.2 (14.4)	88/93/95	4'084k
ours FasterFusion	90.5 (15.6)	93/95/97	5'834k

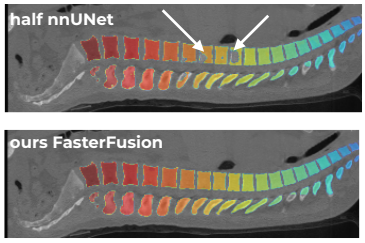


Figure 2: Left: Validation Dice over epochs shows more robust training of our method compared to other low-parameter models. Middle: Quantitative results demonstrate improved quality and 81% reduced parameters (at inference). Right: Visual segmentation examples show that using an nnUNet with halved channel sizes yields some inaccurate vertebrae.

3. Experimental results and Conclusion

We evaluate all method variants for the VerSe19 vertebrae multi-label segmentation task within the nnUNet framework, by re-orienting all patients into a prone pose with heads up and without mirror augmentation but otherwise default parameters. All models where trained on a single RTX-A4000 with 16 GB for 150 epochs on 180 training scans and evaluated on 22 test cases showing on average 10 out of 25 vertebrae. Note, that we replaced the default nnUNet layers only with our FasterFusion blocks when the channel size was 160 or above, since we found for small kernels the model size and runtime improvements were negligible. Fig. 2 highlights the fact that while using 5× fewer parameters our approach matches the quality of full-sized nnUNets. Our model excels with the highest average Dice of 90.5% and does not suffer from slow or unstable training progress as the half-sized nnUNet and ConvNeXt variants respectively.

Conclusion: Our work and its experimental findings indicate that applying T-shaped convolutions – in which $3 \times 3 \times 3$ kernels only act partially on the input channel width – together with pointwise operators within an specifically designed inverted bottleneck and re-parameterisation offers an exciting new strategy for better balancing model sizes, training effort and computational burden of the inference of deep segmentation networks in medical imaging and beyond.

Acknowledgments

I would like to thank Alex Bigalke for careful proof-reading.

References

Jierun Chen, Shiu-hong Kao, Hao He, Weipeng Zhuo, Song Wen, Chul-Ho Lee, and S-H Gary Chan. Run, don't walk: Chasing higher flops for faster neural networks. *arXiv preprint arXiv:2303.03667*, 2023.

Xiaohan Ding, Xiangyu Zhang, Ningning Ma, Jungong Han, Guiguang Ding, and Jian Sun. Repvgg: Making vgg-style convnets great again. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 13733–13742, 2021.

Fabian Isensee, Paul F Jaeger, Simon AA Kohl, Jens Petersen, and Klaus H Maier-Hein. nnu-net: a self-configuring method for deep learning-based biomedical image segmentation. *Nature methods*, 18(2):203–211, 2021.

Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11976–11986, 2022.

Ningning Ma, Xiangyu Zhang, Hai-Tao Zheng, and Jian Sun. Shufflenet v2: Practical guidelines for efficient cnn architecture design. In *Proceedings of the European conference on computer vision (ECCV)*, pages 116–131, 2018.

Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Pavan Kumar Anasosalu Vasu, James Gabriel, Jeff Zhu, Oncel Tuzel, and Anurag Ranjan. An improved one millisecond mobile backbone. *arXiv preprint arXiv:2206.04040*, 2022.