Sauron U-Net: Simple automated redundancy elimination in medical image segmentation via filter pruning (Appendices)

Anonymous Author(s)
Affiliation
Address
`email`

Appendix A. Normalization strategies

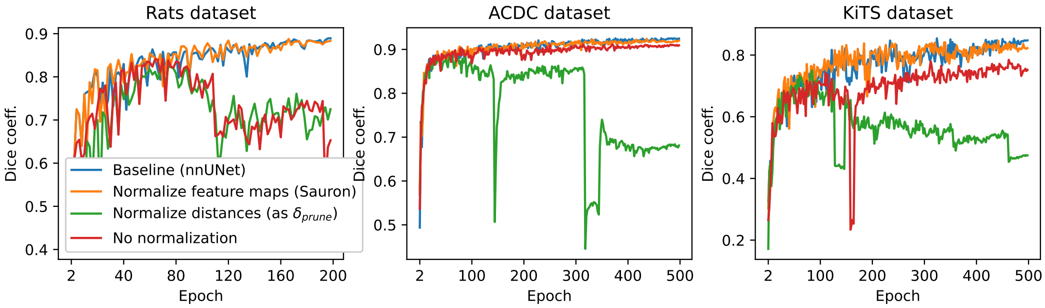

Figure S1: Validation Dice coefficients of baseline nnUNet, Sauron, and two other approaches to normalize δ_{opt}.

We studied the impact on performance of different strategies to normalize δ_{opt}. For this, we compared Sauron's normalization strategy (orange, Fig. S1), the baseline nnUNet (blue), and when not normalizing δ_{opt} (red). Additionally, we normalized δ_{opt} as in $\boldsymbol{\delta}_{prune}$ (green), i.e., instead of normalizing feature maps, the computed distances are divided by their maximum value, layer-wise (see Section 3.2).

Figure S1 shows that Sauron's normalization of the feature maps provided the closest optimization stability and performance to the baseline nnUNet.

Appendix B. nnUNet diagram

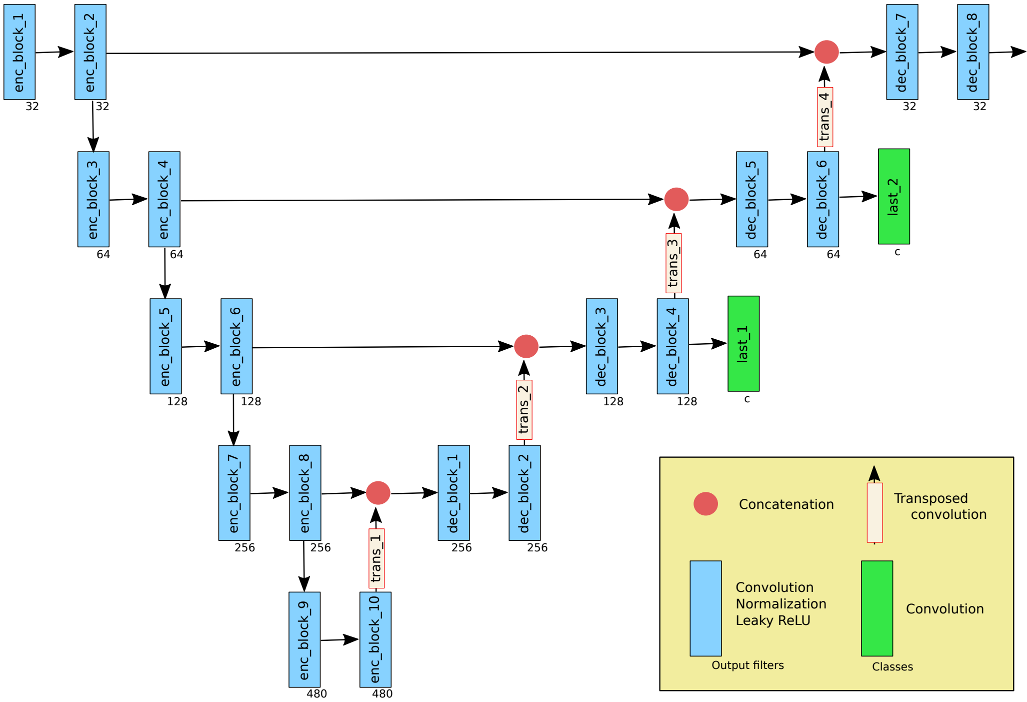

Figure S2: Diagram of an archetypal nnUNet with five levels.

nnUNet is a self-configurable U-Net optimized with extensive data augmentation, deep supervision, and polynomial learning rate decay. In our experiments, the configuration of its architecture and optimization settings depended on the dataset, as in the original publication [6]. The architectural components that depended on the dataset were the following:

- **Number of levels**: Number of block pairs in the encoder with different feature map sizes. The number of levels in Fig. S2 is five. After each even block in the decoder (except in *dec_block_2*), nnUNet computes predictions at different resolutions, enabling deep supervision (green blocks in Fig. S2).
- **Number of filters**: Number of filters of the first two blocks in the encoder. The number of filters in every level doubles with respect to the previous level, unless it exceeds 480—maximum number of filters.
- **Normalization**: We employed either Batch normalization [3] or Instance normalization [9].
- **Dimensions**: Whether we used 3D or 2D convolutions.

Appendix C. Configuration based on the dataset

nnUNet architecture, its optimization, dataset preprocessing, and data augmentation strategy varied across datasets. Such disparity in configuration aimed to tailor each model and training settings to resemble as much as possible to previous studies that reported state-of-the-art performance [4, 1, 5, 2, 10]. Tables 1, 2, and 3 list the configuration employed for each dataset. This configuration can also be seen in our publicly-available code.

C.1. Preprocessing

Rats dataset was not preprocessed. ACDC and KiTS datasets were resampled to their median voxel resolution (Tables 2 and 3 report the final voxel resolution in mm.). In KiTS dataset, images from patients 15, 23, 37, 68, 125, 133 were discarded due to their faulty ground-truth segmentation. Intensity values were clipped to $[-79, 304]$ and normalized by subtracting 101 and dividing by 76.9. Finally, images smaller than the patch size $160 \times 160 \times 80$ were padded.

C.2. Data augmentation

During training, images from Rats, ACDC, and KiTS datasets were augmented via TorchIO [8]. Images were randomly scaled and rotated with certain probability p. Their intensity values were altered via random gamma correction. Then, they were randomly flipped, and they were transformed via random elastic deformation. Particularly in ACDC dataset, 2D slices from the 3D volumes were cropped or padded to 320×320 voxels.

C.3. Architecture

The number of levels of the nnUNet models trained on Rats and KiTS datasets were five whereas in ACDC was seven. nnUNet was optimized on Rats, ACDC, and KiTS datasets with 32, 48, and 24 number of initial filters (*enc_block_1*, Appendix B.), respectively. The nnUNet models optimized on Rats and ACDC datasets were 2D whereas the model for KiTS dataset was 3D. Finally, the normalization layer utilized in Rats and KiTS datasets was Instance Normalization [9] whereas in ACDC was Batch Normalization [3].

C.4. Optimization

All models were optimized with Adam [7] with a starting learning rate of 10^{-3}, weight decay of 10^{-5}, and polynomial learning rate decay: $(1 - (e/epochs))^{0.9}$. nnUNet was optimized for 200 epochs in Rats dataset and 500 epochs in ACDC and KiTS datasets. The batch size in Rats, ACDC, and KiTS datasets was four, ten, and two, respectively.

Table 1: Rats dataset configuration summary

Preprocessing	Data augmentation	Architecture	Optimization
—	Scale $[0.9, 1.1], p = 0.5$ Rotation $[-10, 10], p = 0.5$ Gamma correction $[-0.3, 0.3], p = 0.5$ Flip axis $p = 0.5$	Five levels 32 Init. filters 2D Instance Norm.	200 Epochs Batch size: 4

Table 2: ACDC dataset configuration summary

Preprocessing	Data augmentation	Architecture	Optimization
Resample (1.25, 125, 1)	Scale $[0.85, 1.25], p = 0.2$ Rotation $[-180, 180], p = 0.2$ Elastic deformation $p = 0.3$ Gamma correction $[-0.3, 0.5], p = 0.3$ Flip axis $p = 0.5$ CropOrPad (320, 320)	Seven levels 48 Init. filters 2D Batch Norm.	500 Epochs Batch size: 10

Table 3: KiTS dataset configuration summary

Preprocessing	Data augmentation	Architecture	Optimization
Discard samples	Scale $[0.85, 1.25], p = 0.2$	Five levels	500 Epochs
Resample (1.62, 1.62, 3.22)	Rotation $[-180, 180], p = 0.2$	24 Init. filters	Batch size: 2
Clip intensities (-79, 304)	Elastic deformation $p = 0.3$	3D	
Norm. $(data - 101)/76.9$	Gamma correction $[-0.3, 0.5], p = 0.3$	Instance Norm.	
Pad $(160 \times 160 \times 80)$	Flip axis $p = 0.5$		

Appendix D. Increase/decrease in clusterability metrics

Table 4 lists the relative increase/decrease in the three clusterability measures (dip-test value, distances δ_{opt}, and average number of neighbors) for each convolutional layer. The increase/decrease is computed as the ratio between p_1 and p_2, where p_1 is the average value during the first third of the optimization, and p_2 is the average value in the last third of the training. An increase in clusterability is indicated by 1) an increase in dip-test, 2) a decrease in δ_{opt}, and 3) an increase in the average number of neighbors.

Table 4: Name of the convolutional layer (see Section Appendix B.), number of output filters, and relative increase/decrease in three clusterability measures. Gray: layers with 256 or more feature maps.

Conv. Layer	Filters	Dip-test	Distances δ_{opt}	Avg. neighbors
enc_conv_1	32	-46.0%	2.1%	-74.3%
enc_conv_2	32	-13.5%	5.4%	-91.5%
enc_conv_3	64	-15.0%	2.4%	-97.9%
enc_conv_4	64	-4.2%	-13.7%	-99.8%
enc_conv_5	128	-5.0%	-7.4%	-93.6%
enc_conv_6	128	3.3%	-9.4%	-97.5%
enc_conv_7	256	688.0%	-22.0%	75.1%
enc_conv_8	256	168.5%	-29.3%	118.5%
enc_conv_9	480	-32.4%	-53.4%	128.9%
enc_conv_10	480	118.1%	-51.1%	161.4%
dec_trans_1	256	226.7%	-29.7%	30.7%
dec_conv_1	256	212.7%	-42.4%	63.8%
dec_conv_2	256	121.9%	-8.3%	70.3%
dec_trans_2	128	18.0%	-55.0%	30.3%
dec_conv_3	128	89.0%	3.4%	3.4%
dec_conv_4	128	-15.2%	13.7%	192.5%
dec_trans_3	64	214.6%	-40.8%	1872.5%
dec_conv_5	64	-7.4%	0.0%	49.1%
dec_conv_6	64	22.9%	-3.1%	312.9%
dec_trans_4	32	130.8%	-47.1%	2769.6%
dec_conv_7	32	-9.2%	5.5%	0%
dec_conv_8	32	7.6%	9.0%	-97.4%

Appendix E. Feature maps in the second-to-last convolutional layer of the baseline (unpruned) nnUNet models

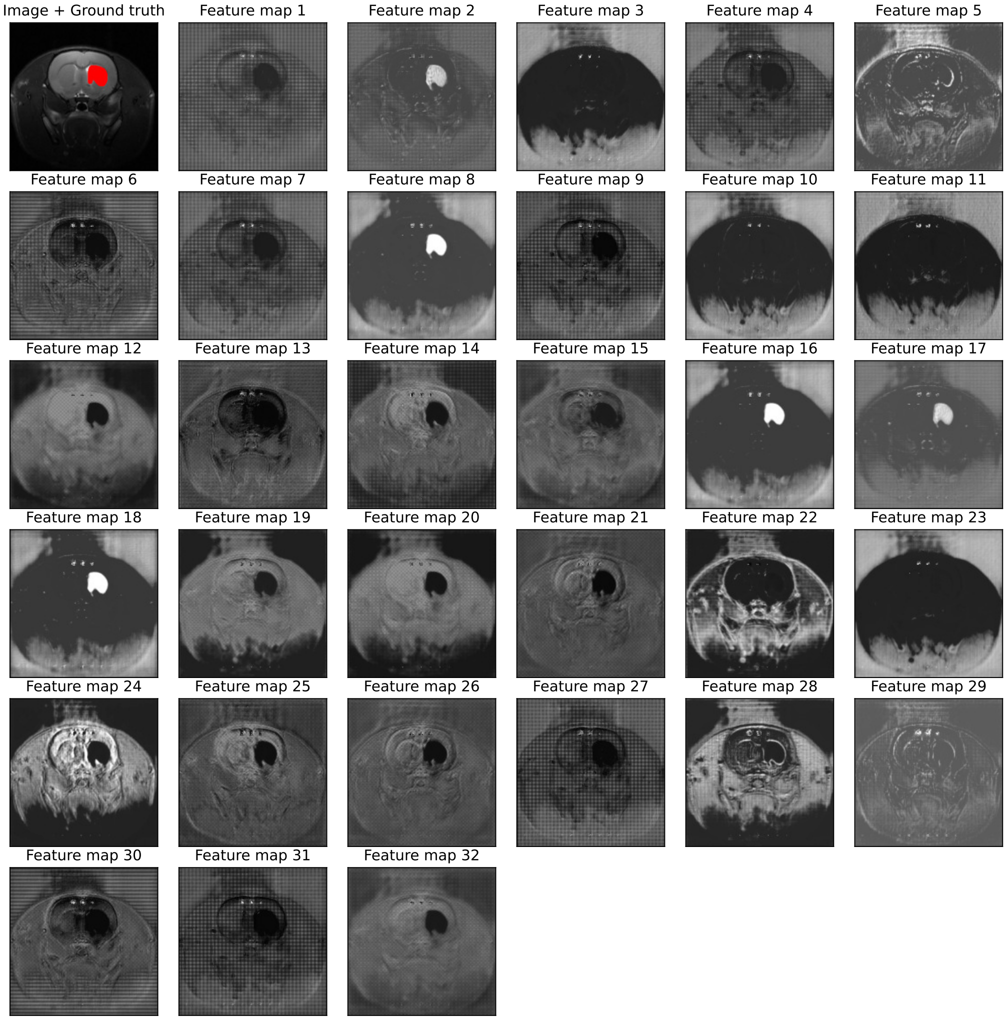

Figure S3: Rats dataset.

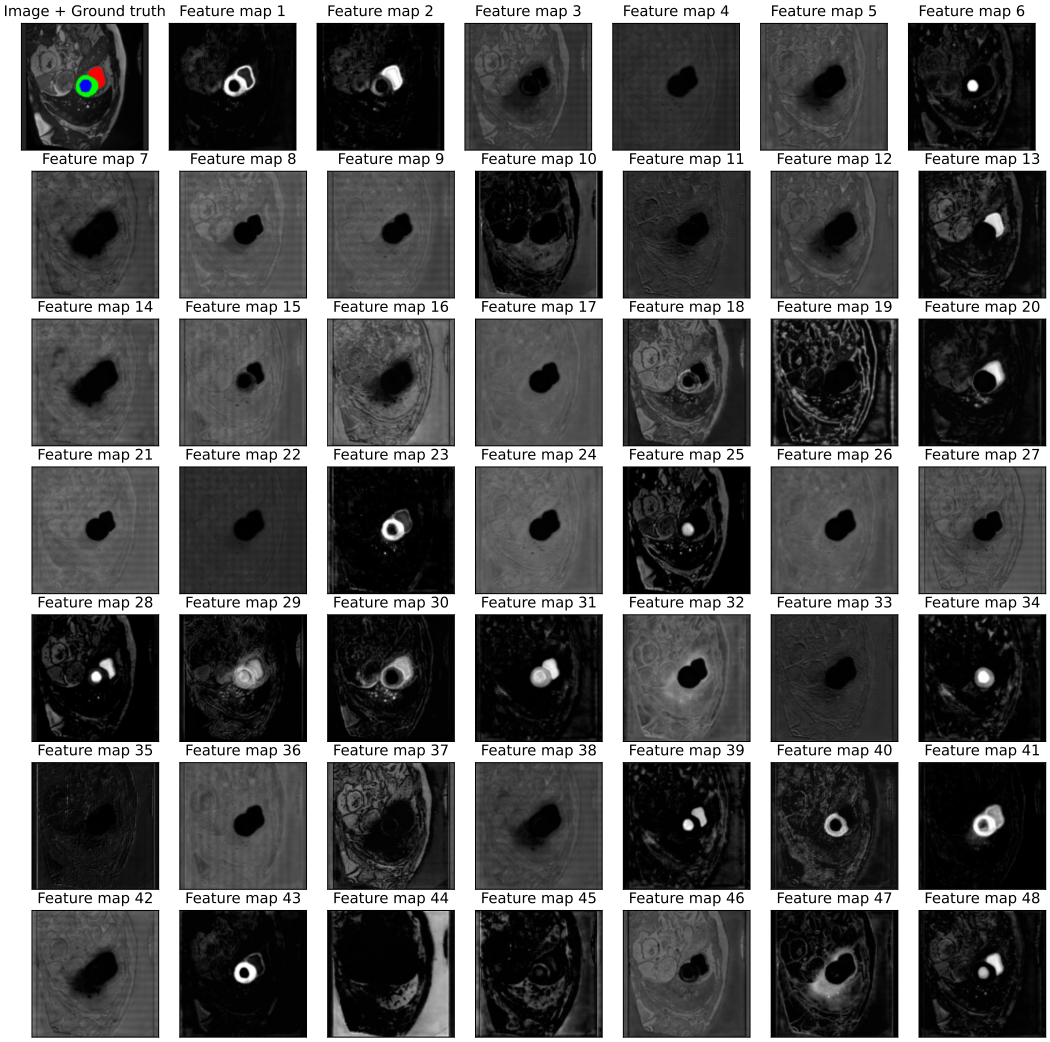

Figure S4: ACDC dataset.

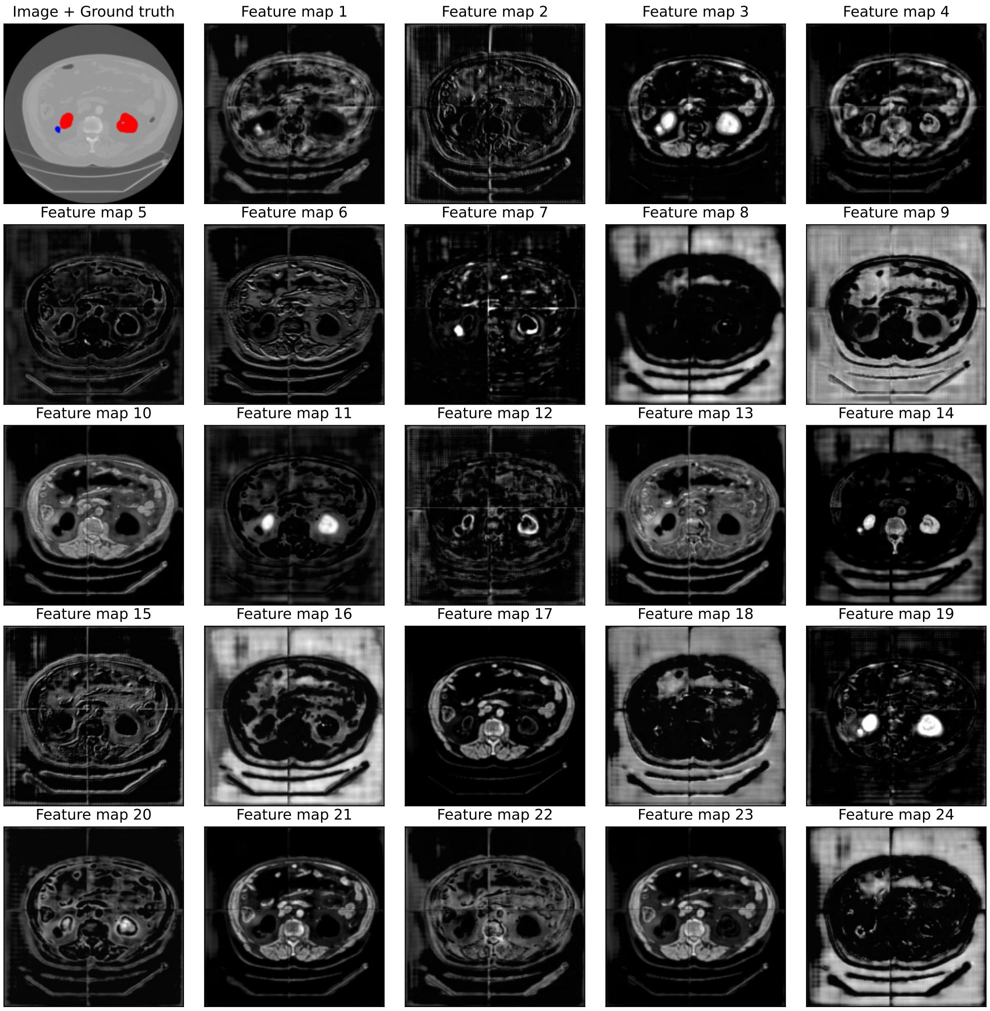

Figure S5: KiTS dataset.

References

[1] Olivier Bernard, Alain Lalande, Clement Zotti, Frederick Cervenansky, Xin Yang, Pheng-Ann Heng, Irem Cetin, Karim Lekadir, Oscar Camara, Miguel Angel Gonzalez Ballester, et al. Deep learning techniques for automatic mri cardiac multi-structures segmentation and diagnosis: Is the problem solved? *IEEE transactions on medical imaging*, 37(11):2514–2525, 2018.

[2] Nicholas Heller, Fabian Isensee, Klaus H Maier-Hein, Xiaoshuai Hou, Chunmei Xie, Fengyi Li, Yang Nan, Guangrui Mu, Zhiyong Lin, Miofei Han, et al. The state of the art in kidney and kidney tumor segmentation in contrast-enhanced ct imaging: Results of the kits19 challenge. *Medical image analysis*, 67:101821, 2021.

[3] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. PMLR, 2015.

[4] Fabian Isensee, Paul F Jaeger, Peter M Full, Ivo Wolf, Sandy Engelhardt, and Klaus H Maier-Hein. Automatic cardiac disease assessment on cine-mri via time-series segmentation and domain specific features. In *International workshop on statistical atlases and computational models of the heart*, pages 120–129. Springer, 2017.

[5] Fabian Isensee, Paul F Jäger, Simon AA Kohl, Jens Petersen, and Klaus H Maier-Hein. Automated design of deep learning methods for biomedical image segmentation. *arXiv preprint arXiv:1904.08128*, 2019.

[6] 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.

[7] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[8] Fernando Pérez-García, Rachel Sparks, and Sébastien Ourselin. Torchio: a python library for efficient loading, preprocessing, augmentation and patch-based sampling of medical images in deep learning. *Computer Methods and Programs in Biomedicine*, page 106236, 2021. ISSN 0169-2607. doi: https://doi.org/10.1016/j.cmpb.2021.106236. URL `https://www.sciencedirect.com/science/article/pii/S0169260721003102`.

[9] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Instance normalization: The missing ingredient for fast stylization. *arXiv preprint arXiv:1607.08022*, 2016.

[10] Juan Miguel Valverde, Artem Shatillo, Riccardo De Feo, Olli Gröhn, Alejandra Sierra, and Jussi Tohka. Ratlesnetv2: a fully convolutional network for rodent brain lesion segmentation. *Frontiers in neuroscience*, 14:1333, 2020.