Slim nnU-Net: Revisiting nnU-Net Self Configuration from an Optimization Angle

Yidong Zhao[1], Yi Zhang[1], and Qian Tao[1]

Department of Imaging Physics, Delft University of Technology, Delft, The Netherlands
q.tao@tudelft.nl

Abstract. nnU-Net is the de facto standard for medical image segmentation, yet its architectures remain computationally expensive. We revisit nnU-Net self-configuration from an optimization angle: instead of heuristic architecture design or post-hoc pruning, we reformulate nnU-Net training as a proximal optimization problem with non-smooth regularizations on network structures. We design the reguarlization as a mixed ℓ_1 and group ℓ_2 loss. Using proximal optimization for this mixed loss enables safe removal of entire channels without harming the segmentation loss. It unifies weight and structure learning in a single, integral training process; and "slim" nnU-Net architectures emerge as the result of optimization. We carry out experiments on both 2D cardiac MRI (ACDC) and 3D brain tumor segmentation (BraTS 2023) datasets, and benchmarked our method against a range of established baselines. We show that for both 2D and 3D nnU-Net, our method removes >99% of the weights structurally, while preserving near-baseline Dice performance. We further show that the reconfigured nnU-Nets deliver substantial run-time gains, with up to 96% FLOP reduction for 2D and 86% for 3D. Compared to baselines, slim nnU-Net achieves the best accuracy–efficiency trade-off. Our method enables more efficient inference and reduced hardware requirements for broad applications of nnU-Net[1].

Keywords: Proximal optimization · Structured pruning · Segmentation

1 Introduction

nnU-Net [10] is the de facto standard for medical image segmentation. It is a self-configuring framework that automatically adapts its architecture to data by extracting a dataset "fingerprint" (spacing, shape, modality) and applying predefined heuristics to determine the network configuration. nnU-Net has demonstrated superior performance across many segmentation challenges, and a recent large-scale benchmark [11,22] showed that its CNN-based architecture consistently outperforms transformer- or Mamba-based alternatives when experimental factors are carefully controlled.

[1] Code is available at https://github.com/Ido-zh/nnunet-sparse.

However, nnU-Net is heavily over-parameterized and computationally expensive. Its 3D full-resolution variant may process patches of 128×128×128 voxels with up to 320 feature channels in deep layers [10]. The resulting memory footprint limits usage to high-end hardware, increases inference cost, and restricts deployment on portable devices. Improving nnU-Net's computational and memory efficiency without sacrificing performance is highly relevant.

Neural network pruning [8,20] offers a tractable alternative to expensive neural architecture search (NAS) [17]. Pruning methods can be divived into two main categories. *Unstructured pruning* removes individual weights based on magnitude or gradient criteria, including Deep Compression [6], Lottery Ticket [5], and sparsification via regularization [23]. Though parameter-efficient, the irregular sparsity pattern still requires specialized hardware for real speedup [3]. In medical imaging, reductions up to 90% have been reported with minimal accuracy loss [9,15], though FLOPs are often not evaluated.

Structured pruning removes entire parameter groups (channels, layers, attention heads), producing architectures that can be accelerated on standard hardware [7,25]. Representative methods include filter-norm channel pruning [12] and ℓ_1 regularization on batch normalization scaling [14]. For medical imaging, STAMP [4] proposed simultaneous training and pruning for U-Net via targeted dropout, achieving over 85% parameter reduction without reporting FLOPs. Sauron [21] introduced feature-distance regularization to cluster feature maps and eliminate redundant filters, and reported substantial FLOP reductions.

We note that existing medical imaging pruning methods rely heavily on heuristics, such as magnitude-based importance [9,15] or feature redundancy [4,21], and typically require iterative training-pruning-retraining or additional feature analysis. This introduces computational overhead, deviates from nnU-Net's robust design principle, while providing no guarantees of convergence or optimality.

In this study, we extend nnU-Net's self-configuration principle from architecture design to architecture optimization. Without altering the initial architecture, we replace standard gradient descent with *proximal optimization*, which enables non-smooth regularization on network structure [16]. To obtain real inference speedup, we propose a mixed ℓ_1 and group-ℓ_2 regularization, which effectively promotes sparsity at both weight and filter levels. Our approach unifies architecture and weight learning in a single optimization: entire filter groups are naturally driven to zero during training and can be completely removed after training. Specifically, we make the following contributions:

- We revisit nnU-Net self-configuration by formulating its training as proximal optimization problem, with mixed ℓ_1 and group-ℓ_2 regularization. This unifies weight and structure learning in an integral optimization process.
- We follow the nnU-Net design principle, without introducing ad hoc heuristics or retraining cycles. A minimal λ setting provides interpretable control over sparsity–accuracy trade-off.
- Empirically, we reach an extreme reduction regime: over 99% of nnU-Net weights can be removed structurally while preserving near-baseline Dice performance for both 2D cardiac (ACDC) and 3D brain tumor (BraTS

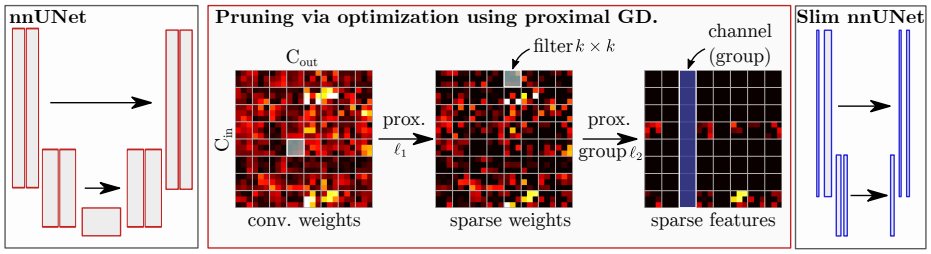

Fig. 1. Proximal gradient descent (GD) with individual and group sparsity regularization progressively induces weight and channel sparsity when training an nnUNet. After training, entire filters and channels can be removed to yield a compact Slim nnUNet.

2023) segmentation. We further demonstrate actual runtime gains: up to 96% FLOP reduction for 2D and 86% reduction for 3D. The resulting "slim nnU-Net" enables faster inference and lower hardware requirements.

2 Methods

2.1 From Standard Training to Explicit Sparsity

nnU-Net models are trained with stochastic gradient descent to minimize a segmentation loss with weight decay:

$$\min_{\theta} \quad \mathcal{L}_{seg}(\theta) + \lambda_2 \|\theta\|_2^2, \tag{1}$$

where θ denotes network parameters and λ_2 controls ℓ_2 regularization. The ℓ_2 regularization improves generalization, but cannot induce sparsity: it smoothly shrinks parameters but can never drive them to exactly zero. As a result, redundant filters remain active and computational cost at inference time is unaffected. To obtain computationally efficient models, we need to introduce explicit sparsity-inducing regularization.

2.2 Hierarchical Sparsity via Sparse Group Lasso

For nnU-Net, we introduce a sparse group lasso regularization, originally designed for regression problems [18]. It promotes sparsity in a hierarchical way:

Individual ℓ_1 sparsity. Element-wise ℓ_1 regularization [19] $\|\theta\|_1 = \sum_i |\theta_i|$, promotes exact zeros at the weight level. However, isolated zero weights cannot accelerate computation.

Grouped sparsity for structured pruning. Structured pruning demands the removal of entire filters or channels. For a convolutional kernel $W \in \mathbb{R}^{\mathrm{C_{out}} \times \mathrm{C_{in}} \times k \times k}$, we define groups $\mathcal{G}$ along output channels:

$$\mathcal{R}_{\mathrm{group}}(\theta) = \sum_{g \in \mathcal{G}} \|\theta_g\|_2 = \sum_{g \in \mathcal{G}} \sqrt{\sum_{i \in g} \theta_i^2}, \tag{2}$$

where each group θ_g contains all parameters of one output channel of shape $\mathbb{R}^{\mathrm{C_{in}} \times k \times k}$. This loss, also called Group Lasso [24], encourages weights of entire channels to collapse to zero simultaneously, leading to structured filter removal.

Mixed regularization for nnU-Net. We propose to combine sparsity at both levels:

$$\min_{\theta} \quad \mathcal{L}_{\mathrm{seg}}(\theta) + \lambda_1 \|\theta\|_1 + \lambda_g \sum_{g \in \mathcal{G}} \|\theta_g\|_2, \tag{3}$$

where λ_1 and λ_g control fine-grained and structured sparsity, respectively. In nnU-Net, deeper encoder and bottleneck layers may contain many potentially redundant filters. The ℓ_1 term suppresses weak individual weights, thereby reducing the norm of weak weight groups. This facilitates their structured removal by the second Group Lasso term. The hierarchical sparsity allows redundant filters to collapse progressively, as illustrated by Fig. 1.

2.3 Optimization via Proximal Gradient Descent

However, both ℓ_1 and grouped ℓ_2 losses are non-differentiable at zero. Standard gradient descent (GD) is not theoretically justified to optimize them: GD typically converges slowly ($\mathcal{O}\left(\frac{1}{\sqrt{t}}\right)$) for non-differentiable losses and oscillates near zero without reaching exact zeros [2]. Proximal gradient descent is the theoretically grounded solution for composite losses with smooth and non-smooth terms. It applies a gradient step on the smooth loss, followed by a proximal mapping of the non-smooth regularizer [16]. Compared to GD, it leads to more stable optimization, exact sparsity, and much faster convergence ($\mathcal{O}\left(\frac{1}{t}\right)$) [16]. Let

$$\mathcal{F}(\theta) = \mathcal{L}_{seg}(\theta) + \mathcal{R}(\theta), \tag{4}$$

where $\mathcal{R}$ denotes the sparsity regularization as in Eq. 3. The proximal operator of $\mathcal{R}$ is defined as

$$\mathrm{prox}_{\eta\mathcal{R}}(v) = \arg\min_{\theta} \left(\frac{1}{2\eta} \|\theta - v\|_2^2 + \mathcal{R}(\theta) \right), \tag{5}$$

where $\eta > 0$ is the step size. Intuitively, the proximal mapping finds a point that balances staying close to v while minimizing the regularization term $\mathcal{R}$. Each iteration consists of a gradient step (smooth part) and a proximal step (non-smooth part):

$$\theta^{(t+\frac{1}{2})} = \theta^{(t)} - \eta \nabla \mathcal{L}_{seg}(\theta^{(t)}), \quad \theta^{(t+1)} = \mathrm{prox}_{\eta\mathcal{R}}\left(\theta^{(t+\frac{1}{2})}\right). \tag{6}$$

For individual terms of ℓ_1 and group lasso, the proximal operator admits simple close-form solutions, corresponds to soft-thresholding:

$$\text{prox}_{\eta\lambda_1}(\theta_i) = \text{sign}(\theta_i)\left(|\theta_i| - \eta\lambda_1\right)_+, \tag{7}$$

$$\text{prox}_{\eta\lambda_g}(\theta_g) = \left(1 - \frac{\eta\lambda_g}{\|\theta_g\|_2}\right)_+ \theta_g, \tag{8}$$

where $(\cdot)_+ = \max(\cdot, 0)$. If $\|\theta_g\|_2 \leq \eta\lambda_g$, the entire channel becomes zero. For our regularization with mixed (convex combination of) ℓ_1 and grouped ℓ_2, the proximal operator can be decomposed sequentially [18]:

$$\theta \leftarrow \text{prox}_{\eta\lambda_g}\big(\text{prox}_{\eta\lambda_1}(\theta)\big). \tag{9}$$

This enables a principled way to optimize our total loss in Eq. 3 with good convergence. An overview of this two-step proximal map is illustrated in Fig. 1.

After training the network with proximal optimization (e.g. for the same number of iterations as nnUNet), we apply structured pruning to all convolutional and transposed convolutional filters. Channels whose group norm falls below a threshold τ are removed entirely. The corresponding input channels in subsequent layers that consume these feature maps are removed accordingly. τ is defined as the percentage relative to the maximum group norm of each layer.

3 Experiments

3.1 Datasets

We evaluate our method on both 2D and 3D medical segmentation benchmarks. For 2D experiments, we use the ACDC dataset [1] containing 150 cardiac MRI subjects, where 100 subjects are used for training and 50 subjects for testing following the standard split. For 3D experiments, we use the BraTS 2023 brain tumor segmentation dataset [13] consisting of 1251 multi-modal MRI subjects. We randomly hold out 20% (251 subjects) for testing and use the remaining 1000 subjects for training.

3.2 Methods in Comparison

We compare against the following baselines, with all methods implemented in the original nnU-Net-v2 framework [10,11]: (1) **LTH (Lottery Ticket Hypothesis)** [5]: A fraction $p = 0.2$ of weights are removed at each round based on their magnitudes and we rewind the surviving weights to an early initialization; (2) **STAMP** [4]: Structured channel pruning guided by activation-energy statistics, the lowest-scoring $p = 0.05$ fraction of channels per layer are removed; (3) **Sauron** [21]: Sparsity is promoted by training with a distance regularization of strength $\lambda = 0.5$; (4) **GD-mix**: standard gradient descent with combined individual ℓ_1 and grouped sparsity penalties, without proximal updates; (5) **prox-group**: proximal gradient descent with grouped sparsity only, i.e., $\lambda_1 = 0$ in

Eq. (3); Different values of λ_g are compared; (6) **prox-mix**: proximal gradient descent with both individual and grouped sparsity ($\lambda_1 > 0$). Different combinations of λ_g and λ_1 are compared. After training, the weights are pruned with the threshold of τ. We tested different choices of $\tau \in [0.01, 0.1]$.

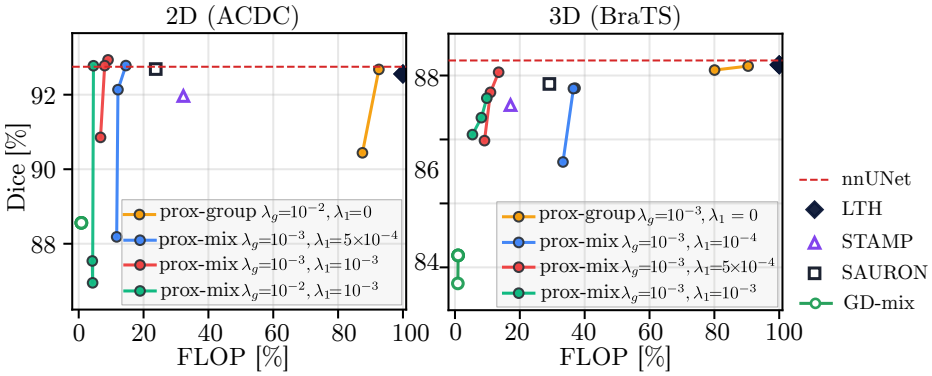

Fig. 2. The accuracy–efficiency tradeoff: Dice versus FLOPs on 2D (ACDC) and 3D (BraTS). Curves labeled with different (λ_g, λ_1) correspond to proximal variants: prox-group ($\lambda_1 = 0$) prox-mix ($\lambda_1 > 0$). For each (λ_g, λ_1) setting, a pruning threshold $\tau \in [0.01, 0.2]$ yields the efficiency–accuracy curve. Prox-mix achieves the best accuracy–efficiency trade-off (upper-left corner), especially at high sparsity levels.

4 Results and Discussions

4.1 Accuracy–Efficiency Trade-off

We report the accuracy–efficiency trade-off for all methods listed above in Fig. 2, which compares Dice versus remaining FLOPs on 2D (ACDC) and 3D (BraTS). Ideal performance would lie in the upper-left region of the plane. For both 2D and 3D, proximal gradient descent with mixed sparsity (prox-mix) consistently achieves the best accuracy–efficiency trade-off. Notably, prox-mix preserves near-baseline Dice while removing more than 95% FLOPs and over 99% of the network weights (Table 1). This demonstrates an extreme high level of parameter reduction with negligible accuracy degradation.

The hyperparameters (λ_g, λ_1) directly control the compression regime: increasing λ_g promotes more aggressive structured channel removal, while λ_1 refines intra-channel sparsity and stabilizes performance at high sparsity levels. Moderate combinations yield the most favorable Pareto frontier, whereas overly strong penalties begin to degrade Dice. In contrast, gradient descent with mixed sparsity (GD-mix) exhibits much worse trade-offs, indicating that simply feeding our mixed loss to standard optimization fails to induce structured sparsity without harming performance. To examine this curious behavior, we show the

training dynamics in Fig. 3, which suggests that with the same mixed loss, proximal GD converges much faster and reaches a much lower loss than GD. This conforms to the proximal optimization theory.

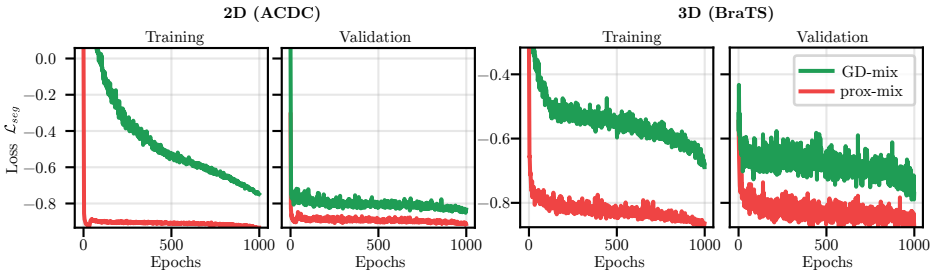

Fig. 3. Training and validation loss curves by GD (GD-mix) and proximal gradient descent (prox-mix) on 2D (ACDC) and 3D (BraTS) datasets. Proximal optimization converges faster and reaches lower loss, demonstrating more stable and effective handling of sparsity regularization.

Prox-group improves over GD-mix but remains slightly less effective than prox-mix, emprically showing the benefit of combining fine-grained and structured sparsity. Compared with LTH, STAMP, and Sauron, our method achieves comparable or higher Dice under substantially heavier reduction, particularly in the high-sparsity regime. Additionally, training a randomly initialized small nnU-Net with matched FLOPs as prox-mix led to Dice drops of 1.91% on ACDC and 1.68% on BraTS, confirming the performance degradation of directly using a small network. This suggests that directly training a small network is insufficient, highlighting the significance of pruning from an over-parameterized model.

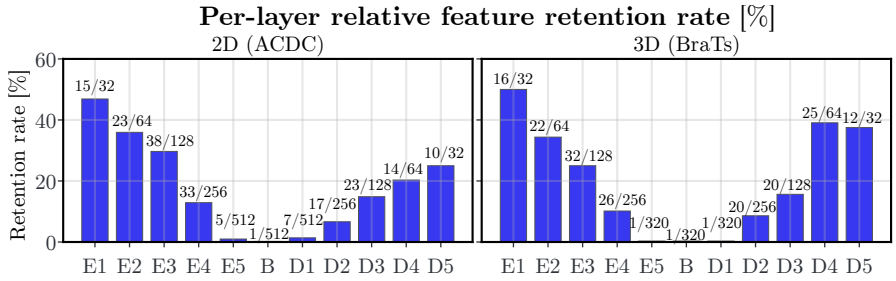

Fig. 4. Per-layer relative feature retention rate (%) across encoder (E1–E5), bottleneck (B), and decoder (D1–D5) for 2D (ACDC) and 3D (BraTS) models, showing stronger retention in early encoder layers and deeper decoder layers, with near-zero retention at the bottleneck. Above the bars we report the pruned/original channel numbers.

Table 1. Quantitative comparison of pruning methods for 2D (ACDC) and 3D (BraTS) nnU-Net. Our proposed method (prox-mix) produces ultra-sparse networks (>99% weight removal) while maintaining competitive segmentation performance. The extreme compression leads to 74.2% and 65.5% CPU speedup for 2D and 3D models, respectively, and 44.9% GPU acceleration in 3D.

	Methods	Weight [%]	FLOP [%]	Dice [%]	$\mathbf{T}_{\mathrm{GPU}}$ [ms/case]	$\mathbf{T}_{\mathrm{CPU}}$ [ms/case]
2D (ACDC)	Original	100.00	100.00	92.75±1.73	122	7298
	LTH	40.96	100.00	92.56±1.70	117	7360
	STAMP	42.75	32.28	91.96±1.94	120	3348
	Sauron	8.09	23.75	92.69±1.80	116	3375
	prox-mix	0.48	4.28	92.78±1.59	114	1880
3D (BraTs)	Original	100.00	100.00	88.47±6.53	1355	221196
	LTH	51.20	100.00	88.33±6.34	1427	221387
	STAMP	42.06	17.13	87.07±6.47	762	77780
	Sauron	4.22	29.11	87.74±6.34	954	125593
	prox-mix	0.75	13.46	88.10±6.29	747	76206

4.2 Layer-wise Sparsity Patterns

We further report nnU-Net per-layer feature retention rates across encoder, bottleneck, and decoder stages in Fig. 4. A clear structured pattern emerges: early encoder layers retain a higher proportion of features, while the bottleneck exhibits near-zero retention. Deeper decoder layers partially recover feature capacity. This consistent behavior across 2D and 3D models suggests that redundancy is concentrated in deeper latent representations, enabling aggressive compression without compromising segmentation accuracy.

4.3 Quantitative Compression and Runtime Analysis

Quantitative comparison of runtime analysis is reported in Table 1. Our prox-mix model removes up to 99.52% (2D) and 99.25% (3D) of weights while retaining Dice comparable to the original nnU-Net, placing the model in an ultra-sparse regime with negligible accuracy degradation. In the 2D setting, GPU inference time decreases from 122 ms to 114 ms per case (6.6% reduction), while CPU inference time decreases from 7298 ms to 1880 ms (74.2% reduction). For the 3D model, GPU inference time decreases from 1355 ms to 747 ms (44.9% reduction), and CPU inference time decreases from 221196 ms to 76206 ms (65.5% reduction). The substantial CPU acceleration highlights the practical benefit of structured pruning under extreme weight reduction, whereas the more moderate GPU speedup in 2D reflects hardware-level constraints such as memory bandwidth and kernel launching overhead. Notably, the larger 3D architecture gains more pronounced benefits from structured sparsity on both GPU and CPU,

suggesting that proximal mixed sparsity becomes increasingly advantageous as model scale grows.

5 Conclusion

In this study, we formulate nnU-Net training as a principled proximal optimization problem: combined with mixed individual and group sparsity regularization, it leads to a new way of nnU-Net self-configuration, directly from a single training process. On both 2D cardiac MRI (ACDC) and 3D brain tumor segmentation (BraTS 2023), we achieve over 99% of nnU-Net weight removal and significantly "slimmed" architectures, with near-baseline Dice performance. For inference, the reconfigured nnU-Net achieves up to 96% and 86% FLOP reduction in 2D and 3D, respectively. We also show that our method outperforms established baselines, in particular with respect to the accuracy-–efficiency trade-off. Our method potentially leads to more efficient inference and reduced hardware requirements for broad applications of nnU-Net.

Acknowledgments. This work was supported by the Delft AI Initiative, TU Delft, Dutch Research Council (NWO) grant no. NGF.1609.241.028, and the NVIDIA Academic Grant Program.

Disclosure of Interests. The authors have no competing interests to declare.

References

1. Bernard, O., Lalande, A., Zotti, C., 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. Boyd, S., Vandenberghe, L.: Convex Optimization. Cambridge University Press (2004)
3. Cheng, H., Zhang, M., Shi, J.Q.: A survey on deep neural network pruning: Taxonomy, comparison, analysis, and recommendations. IEEE Transactions on Pattern Analysis and Machine Intelligence **46**(12), 10558–10575 (2024)
4. Dinsdale, N.K., Jenkinson, M., Namburete, A.I.L.: STAMP: Simultaneous training and model pruning for low data regimes in medical image segmentation. Medical Image Analysis **81**, 102563 (2022)
5. Frankle, J., Carbin, M.: The lottery ticket hypothesis: Finding sparse, trainable neural networks. In: International Conference on Learning Representations (ICLR) (2019)
6. Han, S., Mao, H., Dally, W.J.: Deep compression: Compressing deep neural networks with pruning, trained quantization and Huffman coding. In: International Conference on Learning Representations (ICLR) (2016)
7. He, Y., Xiao, L.: Structured pruning for deep convolutional neural networks: A survey. IEEE Transactions on Pattern Analysis and Machine Intelligence **46**(5), 2900–2919 (2024)
8. Hoefler, T., Alistarh, D., Ben-Nun, T., Dryden, N., Peste, A.: Sparsity in deep learning: Pruning and growth for efficient inference and training in neural networks. Journal of Machine Learning Research **22**(241), 1–124 (2021)

9. Holste, G., Jiang, Z., Jaiswal, A., et al.: How does pruning impact long-tailed multi-label medical image classifiers? In: Medical Image Computing and Computer-Assisted Intervention (MICCAI). Lecture Notes in Computer Science, vol. 14226, pp. 664–674. Springer (2023)
10. Isensee, F., Jaeger, P.F., Kohl, S.A.A., Petersen, J., Maier-Hein, K.H.: nnU-Net: A self-configuring method for deep learning-based biomedical image segmentation. Nature Methods **18**(2), 203–211 (2021)
11. Isensee, F., Wald, T., Ulrich, C., Baumgartner, M., Roy, S., Maier-Hein, K., Jäger, P.F.: nnU-Net Revisited: A Call for Rigorous Validation in 3D Medical Image Segmentation. In: Medical Image Computing and Computer Assisted Intervention – MICCAI 2024. Lecture Notes in Computer Science, vol. 15009, pp. 488–498. Springer, Cham (2024). https://doi.org/10.1007/978-3-031-72114-4_47
12. Li, H., Kadav, A., Durdanovic, I., Samet, H., Graf, H.P.: Pruning filters for efficient convnets. In: International Conference on Learning Representations (ICLR) (2017)
13. Li, H.B., Conte, G.M., Hu, Q., et al.: The brain tumor segmentation (brats) challenge 2023: Brain mr image synthesis for tumor segmentation (brasyn). IEEE Transactions on Medical Imaging (2024). https://doi.org/10.1109/TMI.2023.3317837
14. Liu, Z., Li, J., Shen, Z., Huang, G., Yan, S., Zhang, C.: Learning efficient convolutional networks through network slimming. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV) (2017)
15. Mahbod, A., Entezari, R., Ellinger, I., Saukh, O.: Deep neural network pruning for nuclei instance segmentation in h&e-stained histological images. In: Applications of Medical Artificial Intelligence (AMAI) Workshop. pp. 108–117. Lecture Notes in Computer Science, Springer (2022)
16. Parikh, N., Boyd, S.: Proximal algorithms. Foundations and Trends in Optimization **1**(3), 127–239 (2014)
17. Poyser, M., Breckon, T.P.: Neural architecture search: A contemporary literature review for computer vision applications. Pattern Recognition **147**, 110052 (2024)
18. Simon, N., Friedman, J., Hastie, T., Tibshirani, R.: A sparse-group lasso. Journal of Computational and Graphical Statistics **22**(2), 231–245 (2013)
19. Tibshirani, R.: Regression shrinkage and selection via the lasso. Journal of the Royal Statistical Society: Series B **58**(1), 267–288 (1996)
20. Vadera, S., Ameen, S.: Methods for pruning deep neural networks. Ieee Access **10**, 63280–63300 (2022)
21. Valverde, J.M., Imani, V., Tohka, J.: Sauron U-Net: Simple automated redundancy elimination in medical image segmentation via filter pruning. Neurocomputing **597**, 127934 (2024)
22. Wald, T., Roy, S., Isensee, F., Ulrich, C., Ziegler, S., Trofimova, D., Stock, R., Baumgartner, M., Köhler, G., Maier-Hein, K.: Primus: Enforcing attention usage for 3d medical image segmentation. arXiv preprint arXiv:2503.01835 (2025)
23. Yang, Y., Yuan, Q.: Proxsgd: Training structured neural networks under regularization and constraints. In: International Conference on Learning Representations (ICLR) (2020)
24. Yuan, M., Lin, Y.: Model selection and estimation in regression with grouped variables. Journal of the Royal Statistical Society: Series B **68**(1), 49–67 (2006)
25. Yun, J., Lozano, A.C., Yang, E.: Adaptive proximal gradient methods for structured neural networks. Advances in neural information processing systems **34**, 24365–24378 (2021)