RT-DNAS: Real-Time Constrained Differentiable Neural Architecture Search for 3D Cardiac Cine MRI Segmentation

Qing Lu[1](✉), Xiaowei Xu[2], Shunjie Dong[3], Cong Hao[4], Lei Yang[5],
Cheng Zhuo[3], and Yiyu Shi[1]

[1] University of Notre Dame, Notre Dame, IN, USA
{qlu2,yshi4}@nd.edu

[2] Guangdong Provincial People's Hospital, Guangzhou, China
xxu8@nd.edu

[3] Zhejiang University, Hangzhou, China
{sj_dong,czhuo}@zju.edu.cn

[4] Georgia Institute of Technolog, Atlanta, GA, USA
callie.hao@gatech.edu

[5] University of New Mexico, Albuquerque, NM, USA
leiyang@unm.edu

Abstract. Accurately segmenting temporal frames of cine magnetic resonance imaging (MRI) is a crucial step in various real-time MRI guided cardiac interventions. To achieve fast and accurate visual assistance, there are strict requirements on the maximum latency and minimum throughput of the segmentation framework. State-of-the-art neural networks on this task are mostly hand-crafted to satisfy these constraints while achieving high accuracy. On the other hand, existing literature has demonstrated the power of neural architecture search (NAS) in automatically identifying the best neural architectures for various medical applications, within which differentiable NAS is a prevailing and efficient approach. However, they are mostly guided by accuracy, sometimes with computation complexity, but the importance of real-time constraints are overlooked. A major challenge is that such constraints are non-differentiable and thus are not compatible with the widely used differentiable NAS frameworks. In this paper, we present a strategy that can directly handle real-time constraints in differentiable NAS frameworks, named **RT-DNAS**. Experiments on extended 2017 MICCAI ACDC dataset show that compared with state-of-the-art manually and automatically designed architectures, RT-DNAS is able to identify neural architectures that can achieve better accuracy while satisfying the real-time constraints.

Keywords: Neural architecture search · Real-time segmentation · Cardiac cine MRI

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-16443-9_58.

L. Wang et al. (Eds.): MICCAI 2022, LNCS 13435, pp. 602–612, 2022.
https://doi.org/10.1007/978-3-031-16443-9_58

1 Introduction

The rapid advance in real-time cine Magnetic Resonance Imaging (MRI) has been deployed for visual assistance in various cardiac interventions, such as aortic valve replacement [13], cardiac electroanatomic mapping and ablation [15], electrophysiology for atrial arrhythmias [19], intracardiac catheter navigation [6], and myocardial chemoablation [17]. In these applications, the cine MRI needs to be segmented on-the-fly, for which deep neural networks (DNNs) have been a prevailing choice. The cine MRI requires a reconstruction rate of 22 frames per second (FPS) [9,18], setting the minimum throughput for segmentation using DNNs. In addition, the latency of the network should be less than 50 ms to avoid any lags that can be perceived by human eyes [1]. ICA-UNet [21] is the state-of-the-art DNN that can maintain an acceptable accuracy while meeting these timing constraints, with a latency of 39 ms and a throughput of 28.3 FPS.

However, it is observed that the latency and throughput of ICA-UNet are well above the requirements. Since the MRI frames can only be reconstructed at 22 FPS, achieving a throughput higher than that would not yield any additional benefits but only introduce computation waste. Similarly, as suggested in [1], human vision cannot differentiate any latency below 50 ms. Therefore, there is still room to further trade off throughput and latency for higher accuracy. Unfortunately, ICA-UNet is manually designed, making the exploration of such tradeoff very difficult if not impossible.

Differentiable neural architecture search (NAS) has been recently applied to automatically and efficiently identify the best neural architecture for various biomedical applications [12,14,27]. A majority of these works focus on improving the accuracy as the only metric of interest [2,7,16]. Most recently, researchers have realized that it is crucial to also consider network latency/throughput for biomedical image segmentation applications [5,8,22,23,26]. They use network complexity, in terms of either the number of parameters or the number of floating point operations (FLOPs), as a proxy for latency/throughput estimation. This is primarily because latency/throughput are non-differentiable and can hardly be incorporated into any differentiable NAS frameworks. However, as demonstrated in [10], the latency/throughput can vary significantly even for networks of the same size and FLOPs, due to the difference in computation dependencies that are decided by the network topology. As such, there is no guarantee that the network architectures identified by these frameworks can satisfy the real-time constraints. As such, they cannot be extended to address the 3D cardiac cine MRI segmentation needed for surgical visual assistance.

To address these challenges and to directly incorporate real-time constraint into neural architecture search for 3D cardiac cine MRI segmentation, in this work, we propose a novel NAS approach, termed **RT-DNAS**, where the latency constraint can be incorporated into the differentiable search paradigm. Different from existing NAS approaches where both accuracy and network complexity are considered as optimization objectives, our goal is to *achieve as high accuracy as possible while satisfying the latency and throughput constraints.* It is achieved by using a genetic algorithm (GA) to decide the final network architecture.

Experiments on extended 2017 MICCAI ACDC dataset show that RT-DNAS can achieve the highest accuracy while satisfying the real-time constraints compared with both manual designed ICA-UNet and state-of-the-art NAS frameworks.

2 Method

We build our framework based on MS-NAS [25], the state-of-the-art NAS approach for medical image segmentation with competitive performance and implementation flexibility. An overview of the framework is illustrated in Fig. 1.

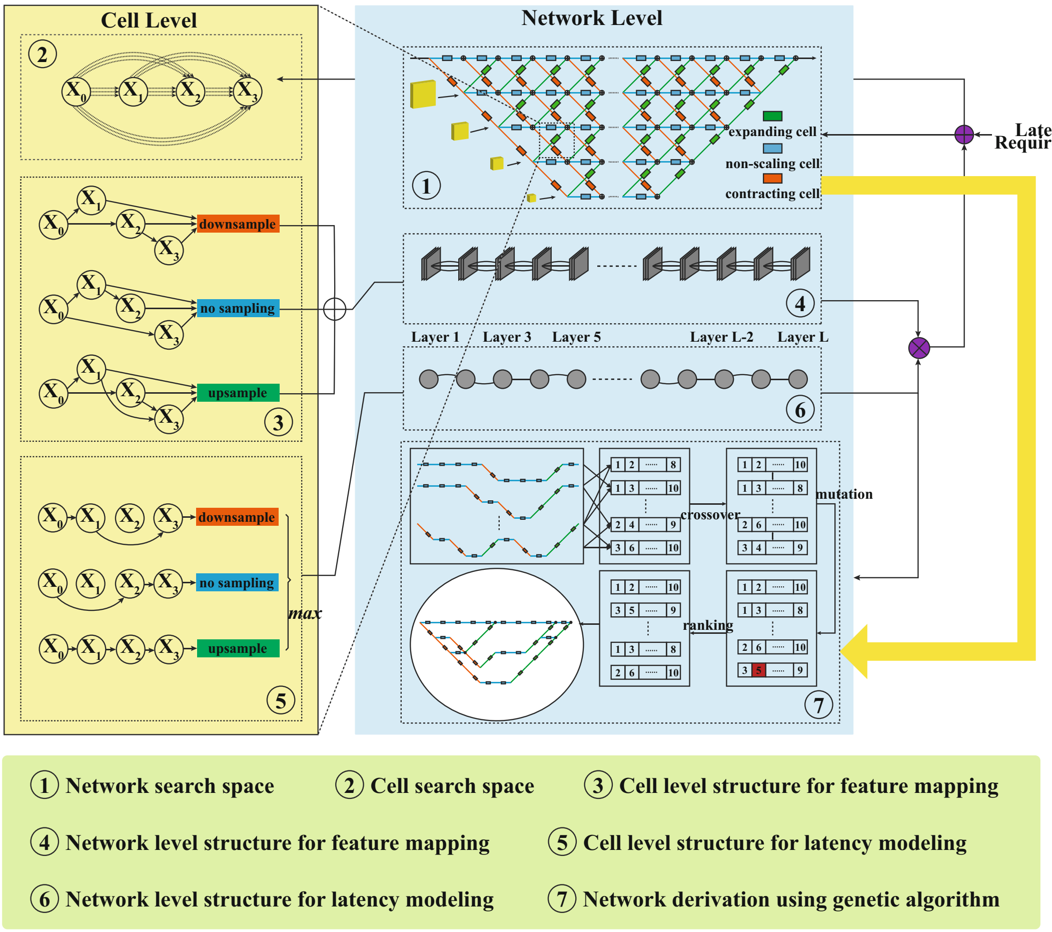


Fig. 1. Overview of the RT-DNAS workflow. We adopt the MS-NAS two-level search space. To incorporate real-time performance, the model finds favorable cell structures within latency constraint. A modified decoding method using genetic algorithm is proposed to derive the network structure.

2.1 Search Space and Method

Inspired by MS-NAS, we define a *network architecture search space* with various types of network design choices, including operation type, feature scale, feature fusion mechanism, and the connections between different operations and cells.

We define the entire NAS search space in a hierarchical manner: assume the network has L layers; each layer can be composed of multiple *cells*; each *cell* is composed of multiple *operations*.

Cell Level. A cell can be represented by a directed acyclic graph (DAG) where the vertices are tensors, and the edges represent the dependence between them. For an input tensor X_0, a cell produces N new tensors X_1, X_2, ..., X_N, and then process all of them using a scaling operation which is the only difference among cell types. The computation of each tensor depends on all its previous tensors (including X_0) and an operation o. We relax the search of the connections and operations by associating an coefficient to each combination of input tensor, output tensor, and operation. Denote the coefficient corresponding to the edge from X_j to X_i by operation o as $v^o_{i,j}$, and the output with performing o on X_j is $o(X_j)$, we can then compute X_i as

$$X_i = \sum_{j<i} \sum_{o \in \mathcal{O}} \frac{exp(v^o_{i,j})}{\sum_{j'<i} \sum_{o' \in \mathcal{O}} exp(v^{o'}_{i,j'})} o(X_j) \tag{1}$$

where $\mathcal{O}$ is the operation set including, depth-wise separable convolution, dilated convolution, etc. It is worth mentioning that relaxing the connection and operation search separately as in MS-NAS can easily cause the gradient of relaxation coefficients vanish, so we directly search input-operation pairs instead.

Network Level. The network level search builds a network consisting of multiple paths from input to output; each path is formed by L cells in series. This is completed by two phases. In the first phase, all the paths are combined to generate the feature maps at all scales. Following the original MS-NAS, we relax the selection of different types of cells, i.e. expanding cell, non-scaling cell, and contracting cell, where their output can merge in the supernet topology (Fig. 1). As a result, the performance of each path can be evaluated by the proximal accumulated weight along the cells it covers. In the second phase, final networks are derived from the best paths based on such evaluation method. To take real-time efficiency into account, we propose a second search procedure using genetic algorithm to decode the network level structure as described in Sect. 2.3.

2.2 Latency Constraint Incorporation

Searching architectures for excellent real-time performance requires considering not only accuracy but also latency and throughput. However, since throughput is naturally coupled with latency (the higher the latency, the lower the throughput), to reduce search burden, we will incorporate latency only in the search process. If the identified architectures cannot satisfy the throughput constraint, we can simply lower the maximum latency and run the search again. As such, the goal of RT-DNAS is to discover the Pareto frontier in the trade-offs between the two metrics. Inspired by an early work targeting a specific hardware accelerator [11], we define the loss function as follows:

$$\mathcal{L} = \mathcal{L}_a \cdot \mathcal{L}_t + \lambda e^{L_t - L_{ub}} \tag{2}$$

where $\mathcal{L}_a$ is the cross-entropy loss of the network evaluated using training data, $\mathcal{L}_t$ is the loss for latency to ensure real-time performance; L_t is the latency achieved by the network candidate; L_{ub} is a constant representing the latency upper bound, which is set to be 50 ms as advised by [1]; λ is a large penalty factor, indicating that if the network architecture latency excels the upper bound, the loss will increase exponentially, so that infeasible solutions can be eliminated as early as possible. Since it is difficult to measure the real latency on the target device during the search, we estimate the latency $\mathcal{L}_t$ in two steps, including cell level and network level, similar to the search space construction.

Cell Level. For each tensor-to-tensor connection, there are $|\mathcal{O}|$ operations in parallel and the latency of the connection equals to the selected operation. Therefore, a reasonable estimation of *latency expectation* is the weighted sum of all the latency values from tensor i to tensor j, as follows:

$$Lat_{i,j} = \sum_{j<i} \sum_{o \in \mathcal{O}} \frac{exp(v_{i,j}^{o})}{\sum_{j'<i} \sum_{o' \in \mathcal{O}} exp(v_{i,j'}^{o'})} Lat_{i,j}^{o}(X_j) \tag{3}$$

where $Lat_{i,j}^{o}$ is the latency of operation o with input X_j, a constant for a specific operation on a fixed-size tensor. With the latency between any pair of tensors available, the latency of the cell is computed as:

$$Lat_{cell} = max_{\mathbf{x} \in \mathcal{X}} Path(\mathbf{x}) + S_{cell} \tag{4}$$

where S_{cell} is a constant representing the latency of scaling operation; $\mathbf{x}$ is a set of the tensors that form a path from X_0 to X_N; $\mathcal{X}$ is the set of all possible paths. Note that Eq. 4 can be solved by finding the longest path within the cell.

Network Level. The overall latency of a network architecture can be estimated using the summation of the latency of each layer. Similar to cell level latency computation, we take a weighted sum of all the cell latency values to compute layer latency expectation based on the probability of each cell occurring in a network path as follows:

$$L_t = \sum_{l \le L} \sum_{c \in \mathcal{C}^l} k_c^l Lat_{cell} \tag{5}$$

where k_c^l is the probability of cell c being sampled from *l-th* layer. The details about the computation of k_c^l are presented in the supplementary materials.

2.3 Architecture Derivation

After the search, the architecture parameters $\mathcal{W}$ are decided. The final network can be derived by combining a group of paths formed by a sequence of cells from input to output. Note that different paths usually have overlapping segments in the network topology, which offers opportunities to further reduce the latency. Existing work MS-NAS constructs the network by finding a fixed number of paths with highest accumulated weights to maximize the accuracy but does not consider network latency. In fact, the weight-latency ratio of different paths varies significantly, so that carefully selecting the paths is important to reducing the network latency.

In RT-DNAS, we optimize the network path selection using genetic algorithm (GA) to build faster networks without significance in accuracy degradation. We define the following concepts in GA. *Gene:* a legal end-to-end path sampled from the backbone graph; *Individual:* a network formed by N_l paths; *Fitness:* the overall network latency of an individual. To keep the high accuracy, a gene pool is firstly built by top paths with high accumulated weights, and only the selected genes can exist during the evolution. We apply two operations to generate new graph, crossover and mutation. The crossover allows two individuals to exchange a portion of their genes and mutation replaces a random gene of an individual with another gene from the pool. To prevent duplication, the paths are sorted by their weights and the order of genes is maintained throughout the process.

3 Experiments

3.1 Experiment Setup

For performance evaluation, following existing works, we employ the data of the extended 2017 MICCAI ACDC challenge dataset made available by MSU-Net [20] with labels on all the frames in the training data. The dataset covers right ventricle (LV), myocardium (MYO), and left ventricle (LV) labeled by experienced radiologists from segmentation frames. There are totally 150 exams in this dataset each from a different patient, where the training set of 100 exams are employed to search the architecture. Further, 5-fold cross validation is performed to evaluate the resultant network using Dice score.

To implement RT-DNAS, we follow the guidelines presented in [25]. The skeleton is constructed by 10 layers of modules, three cells in each module, and three blocks in each cell. max-pooling and bilinear upsampling are used for contracting and expanding the feature size respectively, both by a factor of 2. We also adopt partial channel connection technique [24] to reduce memory cost by forcing 3/4 channels not going through the selected operations. We spend 40 epochs tuning the architectural parameters using an SGD optimizer with momentum of 0.9, learning rate from 0.01 to 0.001, and weight decay of 0.0003.

After all the parameters are fixed, networks consisting of different numbers of paths (N_l) are investigated and derived using GA method presented in Sect. 2.3. The initial population is set to 20, genes are randomly selected from a pool consisting of the top 10 paths, and the search loop runs for 100 generations. The evolution of the cell structures with highest probabilities as well as the final architecture obtained by RT-DNAS are visualized in the supplementary material.

We compare RT-DNAS with five baseline NAS frameworks. ICA-UNet [21] is a manually designed network that attains state-of-the-art performance on this dataset while satisfying real-time constraints. Hardware-aware NAS [26]

Table 1. Comparison of Dice, latency (LT, in ms) and throughput (TP, in FPS) of subnetworks formed by RT-DNAS with network level search carried out using the proposed GA algorithm (which considers real-time constraints) and the *Dijkstra* algorithm used in the original MS-NAS (which cannot consider real-time constraints). N_l is the number of consecutive paths involved.

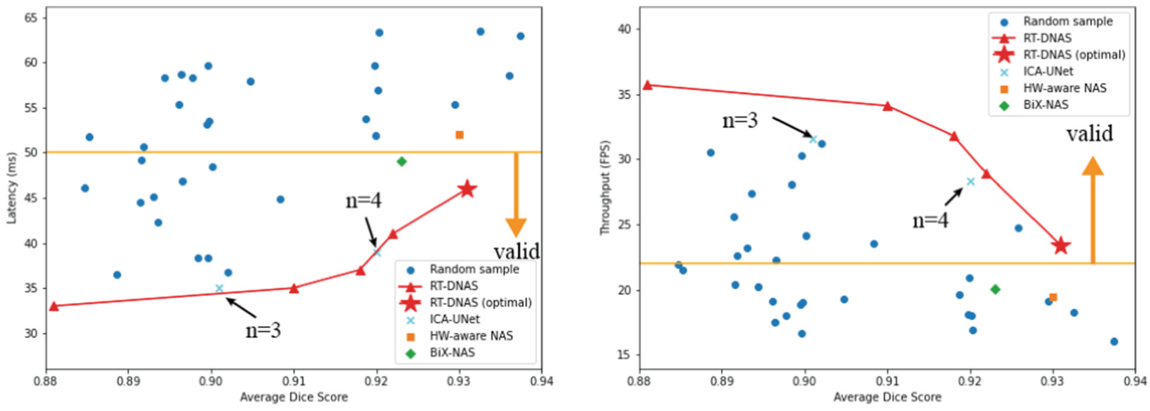
Methods	RV	MYO	LV	Average	LT	TP
Dijkstra ($N_l = 3$)	.865 ± .024	.843 ± .021	.877 ± .022	.849 ± .032	27	39.8
Dijkstra ($N_l = 5$)	.911 ± .033	.881 ± .028	.925 ± .019	.905 ± .020	40	32.0
Dijkstra ($N_l = 7$)	.928 ± .026	.882 ± .017	.943 ± .020	.918 ± .032	48	22.1
Genetic ($N_l = 3$)	.857 ± .034	.843 ± .020	.878 ± .015	.859 ± .027	26	40.4
Genetic ($N_l = 5$)	.910 ± .026	.889 ± .039	.932 ± .022	.910 ± .031	35	34.1
Genetic ($N_l = 7$)	.925 ± .023	.891 ± .019	.950 ± .016	.922 ± .028	41	28.9

Table 2. Comparison of Dice score, latency (LT, in ms) and throughput (TP, in FPS) of the state-of-the-art real-time caridac cine MRI segmentation method ICA-UNet (with two settings used in the original paper), and the best networks discovered by Hardware-aware NAS, BiX-NAS, and the proposed RT-DNAS. The latency cannot exceed 50 ms while throughput should be no less than 22 FPS. Violations of latency and throughput constraints are marked in red.

Methods	RV	MYO	LV	Average	LT	TP
ICA-UNet (n = 3) [21]	.900 ± .023	.869 ± .027	.934 ± .013	.901 ± .017	35	31.6
ICA-UNet (n = 4) [21]	.921 ± .017	.888 ± .034	.952 ± .011	.920 ± .019	39	28.3
HW-Aware NAS [26]	.922 ± .015	.910 ± .042	.958 ± .019	.930 ± .020	52	19.5
BiX-NAS [22]	.916 ± .021	.908 ± .012	.946 ± .033	.923 ± .021	49	20.1
MS-NAS [25]	.928 ± .011	.911 ± .024	.961 ± .025	.933 ± .023	55	18.8
DeU-NAS [4]	.961 ± .015	.927 ± .025	.971 ± .018	.953 ± .019	121	10.2
RT-DNAS ($N_l = 6$)	.924±.010	.890±.017	.939±.030	.918±.018	37	31.8
RT-DNAS ($N_l = 7$)	.925±.023	.891±.019	.950±.016	.922±.028	41	28.9
RT-DNAS ($N_l = 8$)	**.933±.018**	**.902±.029**	**.959±.011**	**.931±.019**	**46**	**23.4**

and BiX-NAS [22] are NAS frameworks for MRI segmentation with hardware performance objectives, where the former includes both accuracy and FLOPs, and the latter includes architectural optimization. They aim at reducing computation complexity and parameter number, but do not explicitly take latency/throughput as hard constraints. MS-NAS [25] and DeU-NAS [4] are two NAS frameworks driven by only accuracy. For reference, we calibrate their timing performance of the reported networks based on reproduced implementations.

All the networks are executed using CUDA/CuDNN [3]. All experiments, including latency and throughput profiling, run on a machine with 16 cores of Intel Xeon E5-2620 v4 CPU, 256G memory, and an NVIDIA Tesla P100 GPU.



(a) (b)

Fig. 2. Accuracy v.s. (a) latency and (b) throughput for randomly sampled architectures, ICA-UNet (n = 3, 4), the best architectures identified by hardware-aware NAS, BiX-NAS, and RT-DNAS. We can see that the Pareto frontier of the points in the valid region is formed by the architectures from RT-DNAS.

3.2 Results

We start with an ablation study to show the benefits of deploying the GA method instead of the *Dijkstra* algorithm used in the original MS-NAS in determining the network level structure. Based on the identified cell structures by RT-DNAS, we construct sub-networks using both *Dijkstra* and GA methods, and the performance of the resulting networks are summarized in Table 1. Comparing the two algorithms with the same number of consecutive paths N_l, it can be clearly observed that GA method gives higher throughput and lower latency, with slightly higher accuracy. This convincingly demonstrates the necessity and effectiveness of deploying GA in the network level search to handle the real-time constraints. It is also concluded that N_l is an effective tuning knob in balancing between the accuracy and timing performance: the more paths involved in the sub-network, the higher Dice score can be achieved, and the slower the network runs.

We collect the best networks obtained by RT-DNAS in terms of the tradeoff between accuracy and efficiency and compare with the baseline NAS approaches. There are two reported versions of ICA-UNet in [21] (with a hpyerparameter n equal to 3 and 4 respectively). As shown in Table 2, RT-DNAS ($N_l = 6$) has slightly longer latency and similar throughput compared with ICA-UNet (n = 3), but higher average Dice. Similar comparison can be observed between RT-DNAS ($N_l = 7$) and ICA-UNet (n = 4). Moreover, all these networks have a latency well below 50 ms and a throughput well above 22 FPS, suggesting additional room to enhance Dice. For hardware-aware NAS and BiX-NAS, although achieving higher average Dice, they violate the real-time constraint in terms of either latency or throughput. This echos our discussion earlier that simply considering network computation complexity or parameter number cannot guarantee that the resulting networks will meet the real-time constraints on latency and throughput. MS-NAS and DeU-NAS deliver even longer latency and lower throughput because they don't consider hardware performance during search. Finally, RT-DNAS (n = 8) yields the highest Dice in RV, MYO, and LV as well as overall average among all the networks that can satisfy real-time constraints. Specifically, RT-DNAS (n = 8) achieves 1.1% and 3.0% higher Dice compared with ICA-UNet (n = 4) and (n = 3), respectively.

Finally, in Fig. 2 we plot the average Dice v.s. latency/throughput for these networks as well as the randomly sampled architectures from the exploration space. From the figure we can see that a random search has a very slim chance to find any close-to-optimal architectures, due to the large exploration space. On the other hand, the architectures identified by RT-DNAS form the Pareto frontier of all the points in the valid region defined by maximum latency and minimum throughput, suggesting that they can indeed offer feasible designs with best tradeoff. Visualization of segmentation results by RT-DNAS (n = 8) along with the corresponding ground truth is included in the supplementary material.

4 Conclusion

In this work we proposed a latency-aware differentiable NAS framework for cine magnetic resonance image segmentation, named RT-DNAS. By incorporating hardware latency into the search objective, RT-DNAS is able to achieve the highest accuracy while perfectly meeting the real-time constraint, comparing with state-of-the-art manually and automatically designed architectures. It is demonstrated that using neural architecture search is a promising direction to real-time medical intervention using deep learning.

References

1. Annett, M., Ng, A., Dietz, P., Bischof, W.F., Gupta, A.: How low should we go? Understanding the perception of latency while inking. In: Graphics Interface 2014, pp. 167–174. AK Peters/CRC Press (2020)
2. Bosma, M., Dushatskiy, A., Grewal, M., Alderliesten, T., Bosman, P.A.: Mixed-block neural architecture search for medical image segmentation. In: SPIE Medical Imaging (2022)
3. Chetlur, S., et al.: cuDNN: Efficient primitives for deep learning. arXiv preprint arXiv:1410.0759 (2014)
4. Dong, S., et al.: DeU-Net 2.0: Enhanced deformable U-Net for 3D cardiac cine MRI segmentation. Medical Image Analysis **78**, 102389 (2022)
5. Fernandes, F.E., Yen, G.G.: Automatic searching and pruning of deep neural networks for medical imaging diagnostic. IEEE Trans. Neural Netw. Learn. Syst. **32**(12), 5664–5674 (2021)
6. Gaspar, T., Piorkowski, C., Gutberlet, M., Hindricks, G.: Three-dimensional real-time MRI-guided intracardiac catheter navigation. Eur. Heart J. **35**(9), 589–589 (2014)

7. He, Y., Yang, D., Roth, H., Zhao, C., Xu, D.: Dints: differentiable neural network topology search for 3d medical image segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 5841–5850 (2021)
8. Huang, Z., Wang, Z., Gu, L., et al.: Adwu-Net: adaptive depth and width U-Net for medical image segmentation by differentiable neural architecture search. In: Medical Imaging with Deep Learning (2021)
9. Iltis, P.W., Frahm, J., Voit, D., Joseph, A.A., Schoonderwaldt, E., Altenmüller, E.: High-speed real-time magnetic resonance imaging of fast tongue movements in elite horn players. Quant. Imaging Med. Surg. **5**(3), 374 (2015)
10. Jiang, W., et al.: Hardware/software co-exploration of neural architectures. IEEE Trans. Comput.-Aided Des. Integr. Circ. Syst. **39**(12), 4805–4815 (2020)
11. Li, Y., et al.: EDD: efficient differentiable DNN architecture and implementation co-search for embedded AI solutions. In: 2020 57th ACM/IEEE Design Automation Conference, DAC 2020. Proceedings - Design Automation Conference, Institute of Electrical and Electronics Engineers Inc., USA, July 2020
12. Lu, Z., et al.: Multi-scale and multi-level neural architecture search for low-dose ct denoising. arXiv preprint arXiv:2103.12995 (2021)
13. McVeigh, E.R., Guttman, M.A., Lederman, R.J., Li, M., Kocaturk, O., Hunt, T., Kozlov, S., Horvath, K.A.: Real-time interactive Mri-guided cardiac surgery: Aortic valve replacement using a direct apical approach. Mag. Resona. Med. **56**(5), 958–964 (2006)
14. Peng, C., et al.: Hypersegnas: bridging one-shot neural architecture search with 3d medical image segmentation using hypernet. arXiv preprint arXiv:2112.10652 (2021)
15. Radau, P.E., et al.: VURTIGO: visualization platform for real-time, MRI-guided cardiac electroanatomic mapping. In: Camara, O., Konukoglu, E., Pop, M., Rhode, K., Sermesant, M., Young, A. (eds.) STACOM 2011. LNCS, vol. 7085, pp. 244–253. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-28326-0_25
16. Radau, Z., et al.: VURTIGO: visualization platform for real-time, MRI-guided cardiac electroanatomic mapping. In: Camara, O., Konukoglu, E., Pop, M., Rhode, K., Sermesant, M., Young, A. (eds.) STACOM 2011. LNCS, vol. 7085, pp. 244–253. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-28326-0_25
17. Rogers, T., et al.: Transcatheter myocardial needle chemoablation during real-time magnetic resonance imaging: a new approach to ablation therapy for rhythm disorders. Circulat. Arrhyth. Electrophysiol. **9**(4), e003926 (2016)
18. Schaetz, S., Voit, D., Frahm, J., Uecker, M.: Accelerated computing in magnetic resonance imaging: real-time imaging using nonlinear inverse reconstruction. In: Computational and Mathematical Methods in Medicine 2017 (2017)
19. Vergara, C.R., et al.: Real-time magnetic resonance imaging-guided radiofrequency atrial ablation and visualization of lesion formation at 3 tesla. Heart Rhythm **8**(2), 295–303 (2011)
20. Wang, T., et al.: MSU-Net: multiscale statistical u-net for real-time 3d cardiac MRI video segmentation. In: Shen, S., et al. (eds.) MICCAI 2019. LNCS, vol. 11765, pp. 614–622. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-32245-8_68
21. Wang, T., et al.: ICA-UNet: ICA inspired statistical UNet for real-time 3D cardiac cine MRI segmentation. In: Martel, M.L., et al. (eds.) MICCAI 2020. LNCS, vol. 12266, pp. 447–457. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-59725-2_43

22. Wang, X., et al.: BiX-NAS: searching efficient bi-directional architecture for medical image segmentation. In: de Bruijne, M., et al. (eds.) MICCAI 2021. LNCS, vol. 12901, pp. 229–238. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-87193-2_22
23. Xu, S., Quan, H.: ECT-NAS: Searching efficient CNN-transformers architecture for medical image segmentation. In: 2021 IEEE International Conference on Bioinformatics and Biomedicine (BIBM), pp. 1601–1604. IEEE (2021)
24. Xu, Y., et al.: PC-darts: partial channel connections for memory-efficient architecture search. arXiv preprint arXiv:1907.05737 (2019)
25. Yan, X., Jiang, W., Shi, Y., Zhuo, C.: MS-NAS: multi-scale neural architecture search for medical image segmentation. In: Martel, A.L., et al. (eds.) MICCAI 2020. LNCS, vol. 12261, pp. 388–397. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-59710-8_38
26. Zeng, D., et al.: Towards cardiac intervention assistance: hardware-aware neural architecture exploration for real-time 3D cardiac cine MRI segmentation. In: Proceedings of the 39th International Conference on Computer-Aided Design, pp. 1–8 (2020)
27. Zhu, Y., Meijering, E.: Automatic improvement of deep learning-based cell segmentation in time-lapse microscopy by neural architecture search. Bioinformatics **37**(24), 4844–4850 (2021)