Scalable Neural Architecture Search for 3D Medical Image Segmentation

Sungwoong Kim[*1] SWKIM@KAKAOBRAIN.COM
Ildoo Kim[*1] ILDOO.KIM@KAKAOBRAIN.COM
Sungbin Lim[*1] SUNGBIN.LIM@KAKAOBRAIN.COM
Chiheon Kim[1] CHIHEON.KIM@KAKAOBRAIN.COM
Woonhyuk Baek[1] WBAEK@KAKAOBRAIN.COM
Hyungjoo Cho[2] JOYSQUARE@SNU.AC.KR
Boogeon Yoon[1] ERIC.YOON@KAKAOBRAIN.COM
Taesup Kim[1,3] TAESUP.KIM@UMONTREAL.CA
[1] *Kakao Brain, Pangyo, Seongnam, Gyeonggi, Republic of Korea*
[2] *Department of Transdisciplinary Studies, Seoul National University, Republic of Korea*
[3] *MILA, Université de Montréal, Canada*

Editors: Under Review for MIDL 2019

Abstract

In this paper, a neural architecture search (NAS) framework is formulated for 3D medical image segmentation, to automatically optimize a neural architecture from a large design space. For this, a novel NAS framework is proposed to produce the structure of each layer including neural connectivities and operation types in both of the encoder and decoder of a target 3D U-Net. In the proposed NAS framework, having a sufficiently large search space is important in generating an improved network architecture, however optimizing over such a large space is difficult due to the extremely large memory usage and the long run-time originated from high-resolution 3D medical images. Therefore, a novel stochastic sampling algorithm based on the continuous relaxation on the discrete architecture parameters is also proposed for scalable joint optimization of both of the architecture parameters and the neural operation parameters. This makes it possible to maintain a large search space with small computational cost as well as to obtain an unbiased architecture by reducing the discrepancy between the training-time and test-time architectures. On the 3D medical image segmentation tasks with a benchmark dataset, an automatically designed 3D U-Net by the proposed NAS framework outperforms the previous human-designed 3D U-Net as well as the randomly designed 3D U-Net, and moreover this optimized architecture is more compact and also well suited to be transferred for similar but different tasks.

Keywords: AutoML, Neural Architecture Search, Medical Image Segmentation.

1. Introduction

Recently, deep neural networks have been extensively used for medical image segmentation tasks (Ronneberger et al., 2015; Çiçek et al., 2016; Mortazi et al., 2017; Ciresan et al., 2012; Milletari et al., 2016a; Kamnitsas et al., 2016; Havaei et al., 2015; Yu et al., 2017; Kayalibay et al., 2017; Oktay et al., 2018; Isensee et al., 2018). However, such a method in general

[*] Contributed equally

relies on manual trial-and-error processes for making decisions on the network architecture, hyperparameters for training, and pre-/post-procedures. Due to being restricted to manual tuning, they would have limitations in performance improvement as well as fast transfer to related tasks. Currently, the same problem in the field of general deep learning has promoted the rapid development of automated machine learning (AutoML). Yet, in contrast to the recent intensive studies on the use of advanced AutoML algorithms such as neural architecture search (NAS) (Zoph et al., 2018; Liu et al., 2018a; Bender et al., 2018; Zoph and Le, 2017; Liu et al., 2018b; Pham et al., 2018; Zhang et al., 2018; Cai et al., 2018; Brock et al., 2018) and neural optimizer search (Bello et al., 2017; Alber et al., 2018; Wichrowska et al., 2017; Li and Malik, 2017; Andrychowicz et al., 2016) for general computer vision tasks, only a few naive AutoML approaches using simple hyperparameter optimization have been proposed for medical imaging tasks (Mortazi and Bagci, 2018; Naceur et al., 2018). Therefore, in this paper, we propose a novel NAS framework for AutoML in designing neural networks especially for 3D medical image segmentation.

Since both semantic as well as spatial information can be efficiently exploited through skip connections between an encoder and a decoder, a 3D U-Net has been popularly used in most state-of-the-art deep learning based algorithms for segmenting high-resolution 3D medical images (Çiçek et al., 2016; Milletari et al., 2016a; Yu et al., 2017; Kayalibay et al., 2017; Oktay et al., 2018; Isensee et al., 2018). However, a convolutional block for each layer in the 3D U-Net has been manually designed with various convolutional filter types, pooling types, skip-connections, and non-linear activation functions. Instead of using the suboptimally designed block, we propose to use a NAS framework to obtain an automatically optimized structure of the block, which is called a cell, for each layer in the 3D U-Net where all cell structures and the corresponding neural operation parameters (e.g. kernel weights) are simultaneously learned in an end-to-end manner. For this, four types of cells - encoder-normal cell, reduction cell, decoder-normal cell, expansion cell - are defined to compose the encoder as well as the decoder for the learned U-Net architecture, which is different from the use of two types of cells (normal cell and reduction cell) in previous NAS approaches for encoder-only networks (Zoph et al., 2018; Liu et al., 2018b; Pham et al., 2018). Here, it is noted that in NAS having a sufficiently large search space is important in generating an improved network architecture on a target task. However, optimizing over such a large space for this segmentation task is difficult due to the extreme memory usage and the long run-time when dealing with high-resolution 3D images. Moreover, NAS basically needs to jointly optimize not only the discrete architecture parameters but also the continuous operation parameters, which is so-called bi-level optimization (Liu et al., 2018b; Franceschi et al., 2018), and an exact bi-level optimization over this mixed domain(discrete and continuous) is also difficult, especially with this large search space associated with the 3D U-Net.

Therefore, in this work, a novel stochastic sampling algorithm is applied for bi-level optimization of the mixed parameters in the proposed NAS framework. This can not only search over a large design space but also lead to provide a consistent and unbiased architecture that avoids the retraining of suboptimal operation parameters from the obtained architecture. More specifically, the discrete architecture parameters corresponding to neural connections and operation types in each cell are defined as a set of one-hot discrete variables, and a continuous approximation using Gumbel-softmax (Jang et al., 2017; Maddison et al., 2016) is imposed on these discrete variables. This makes it possible to compute the gradients with respect to both of the approximated architecture variables and the neural operation parameters through a back-propagation and thereby allows to use a stochastic gradient descent (SGD) in bi-level optimization. Furthermore, during the SGD-based bi-level optimization, we utilize an iterative sampling of the candidate architecture, which simulates the test-time final architecture, based on the approximated continuous architecture variables by treating those as logits to provide a categorical distribution. This sampling procedure enables to reduce the computational burden of taking the entire connectivities and operations into account within an outrageously large network originated from the continuous relaxation. Moreover, it also reduces the discrepancy between the training-time and test-time architectures. Namely, the proposed differentiable NAS with stochastic sampling supports great scalability in terms of solvable large search space with small computational cost.

Experimental results on the benchmark 3D medical image segmentation dataset show that in comparison to the previous human-designed 3D U-Net, the network obtained by the proposed scalable NAS leads to better performances even with the less numbers of parameters and FLOPs (multiply-adds). It is furthermore shown that the found architecture from a task having large amounts of labeled data can be transferred to build a network for different segmentation tasks that have small amounts of labeled data and achieves better generalization performances.

To our best knowledge, this is the first work to exploit a complete NAS framework for automatically designing an architecture for the task of 3D medical image segmentation.

2. Related Works

NAS can be considered as one of meta-learning processes (Lemke et al., 2015; Vanschoren, 2018) in which a meta-controller performs a guided exploration on a given architecture space via evaluation of each candidate architecture in the inner loop (Zoph et al., 2018; Pham et al., 2018). Several recent works have focused on reducing the computational cost of this architecture evaluation by reusing the trained weights on different architectures (Bender et al., 2018; Liu et al., 2018b; Pham et al., 2018). Especially, they have sampled every candidate network from a single over-parametrized network, called an one-shot model, which allows to train only the one-shot model and directly evaluate any candidate network by inheriting this one-shot model's trained weights. Among them, DARTS (Liu et al., 2018b) have removed a meta-controller by continuous relaxation of the search space, which leads to simultaneously learn the structure parameters as well as the kernel weights by SGD-based bi-level optimization. Even though DARTS enables efficient SGD-based optimization, it still suffers from the large computational cost to handle all possible neural connectivities and operations in the whole large one-shot model. ProxylessNAS (Cai et al., 2018) have resolved this cost issue by sampling two operation types for each neural connection according to the multinomial distribution during the architecture training. However, it has still used a biased architecture during the training in that there is no guidance for real-valued operation gates (logits) representing the multinomial distribution to be converged to discrete one-hot variables standing for the final architecture at test-time. Hence, we use a stochastic architecture sampling based on the Gumbel-softmax (Jang et al., 2017; Maddison et al., 2016), that is a continuous and differentiable approximation of these one-hot variables, which makes the sampled architecture converged to be the final architecture during the training by gradually reducing the softmax temperature to 0. While the previous NAS approaches have been applied mostly to the tasks of image recognition and language modeling, Nekrasov et al. (2018) has recently adopted NAS for 2D image segmentation. However, they have optimized only the decoder architecture in an encoder-decoder framework with an RNN-based meta-controller trained by reinforcement learning.

Since Ronneberger et al. (2015) first introduced the U-Net for biomedical image segmentation, several modifications have been proposed. For example, Çiçek et al. (2016) has extended it with 3D convolutional kernels, and then Milletari et al. (2016a) has incorporated the residual blocks into the 3D U-Net. Moreover, Kayalibay et al. (2017) and Yu et al. (2017) have utilized multiple segmentation maps at different scales while Oktay et al. (2018) has adopted attention gates between an encoder and a decoder to simulate multi-stage cascaded convolutional neural networks (CNNs). Recently, Isensee et al. (2018) has introduced the nnU-Net that is able to dynamically adapt itself to any given segmentation task on the medical domain via non-architectural self-modifications based on the original U-Net. In (Mortazi and Bagci, 2018) the policy gradient algorithm automatically searches for the hyperparameters such as the number of filters, the filter size, and the pooling type for each layer for the 2D cine cardiac MR image segmentation while Naceur et al. (2018) incrementally optimized those hyperparameters as well as the number of layers for the 2D brain tumor segmentation. It is noted that unlike these architecture hyperparameter optimizations, we use the complete NAS to obtain the entire topology of the network architecture in this work.

3. Method

In this section, we first describe an architecture search space based on the U-Net-like network for 3D medical image segmentation, and then present a SGD-based bi-level optimization with the proposed stochastic sampling to simultaneously learn both of the architecture and the corresponding neural operation parameters.

3.1. Search Space for 3D Medical Image Segmentation

Following the idea of micro search space popularly used in the state-of-the-art NAS approaches (Liu et al., 2018b; Zoph et al., 2018; Pham et al., 2018), U-Net-like networks, which is composed of encoder and decoder layers, are designed as repeated encoder and decoder cells. The neural structure in each cell C is represented as a directed acyclic graph (DAG) (see Figure 1). Let $\mathcal{G} = (\mathcal{V}(C), \mathcal{E}(C))$ be the DAG where each node $i \in \mathcal{V}$ corresponds to an intermediate feature vector $\mathbf{x}^i$, and each directed edge $(i, j) \in \mathcal{E}$ stands for a connection between nodes i and j with a certain operation $o^{(i,j)}$ such that $\mathbf{x}^j = \sum_{(i,j)\in\mathcal{E}} o^{(i,j)}(\mathbf{x}^i)$. The output of a cell is a channel-wise concatenation of all the intermediate nodes. Here, a cell C is one of four cell-types - *encoder-normal* (C_{enc}), *reduction* (C_{red}), *decoder-normal* (C_{dec}), and *expansion* (C_{exp}) - such that $C \in \mathcal{C} = \{C_{\mathsf{enc}}, C_{\mathsf{red}}, C_{\mathsf{dec}}, C_{\mathsf{exp}}\}$, and the normal cells and resizing cells are stacked alternately with skip connections between the cells in the encoder and the cells in the decoder, layer-by-layer. Note that every cell takes two outputs

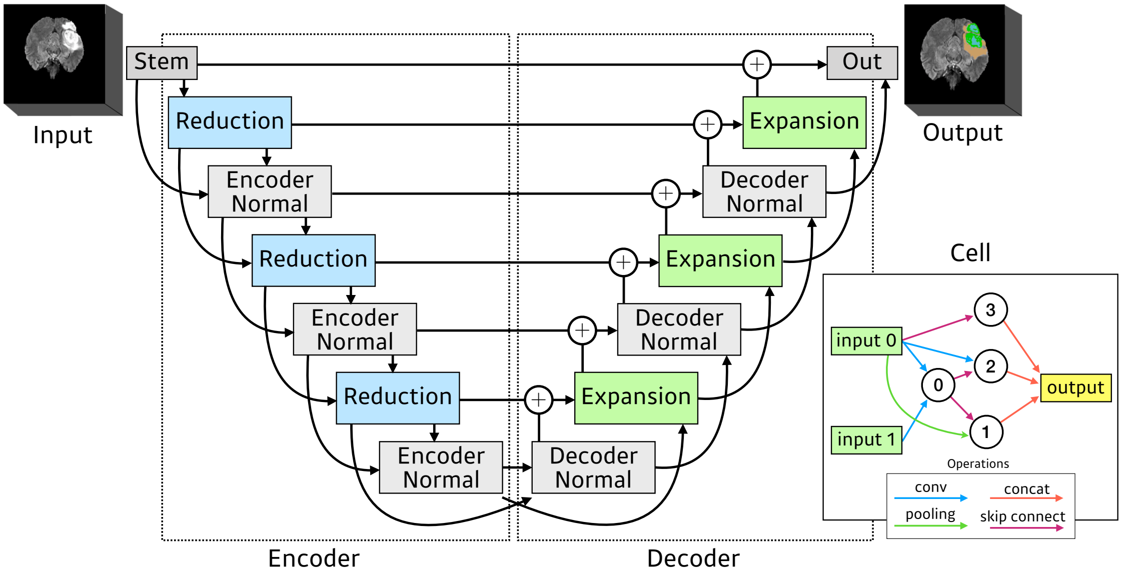

Figure 1: Architecture search space for 3D medical image segmentation tasks. Both encoder and decoder alternately stack normal cells and resizing cells. The directed arrows between cells indicate the forward paths. Each cell is represented as a DAG which receives two inputs and produces an output.

of the last previous two cells as inputs[1] except the first reduction cell which takes an output of the predefined first convolutional block, called a stem cell, and then duplicates it as two inputs. The segmentation output is obtained from the predefined last convolutional block, referred to as an out cell.

Since $o^{(i,j)} \in \mathcal{O}$ where $\mathcal{O}$ denotes the set of all candidate operations, the architecture search problem now amounts to find the best combination of all edge operations in the four cell-types. Basically, even the same type of cells can have different structures according to their layer levels. However, in this work, for simplicity, all cells that have a common type share a common structure regardless of layer levels. It is noted that a special *zero* operation is also one of the candidate operations to optimize the neural connectivities as well; *zero* means a lack of connection between two nodes.

[1] In the decoder, before used as one of inputs of the current cell, an output of the last previous cell is summed with an output of the encoder cell at the same level by skip connection.

3.2. Stochastic Bi-Level Optimization

We first represent the selected edge operation using the one-hot indicator vector, $\mathbf{z}^{(i,j)}$, as follows:

$$o^{(i,j)}(\mathbf{x}^i) = \sum_{o\in\mathcal{O}} z_o^{(i,j)} o(\mathbf{x}^i; \theta_o^{(i,j)}), \tag{1}$$

where $\mathbf{z}^{(i,j)} = \{z_o^{(i,j)} \mid o \in \mathcal{O}\}$, $\theta^{(i,j)} = \{\theta_o^{(i,j)} \mid o \in \mathcal{O}\}$, and $\theta_o^{(i,j)}$ denotes the parameter set of the operation o on edge (i,j), which means that the operation on each edge is differently learned even though the cells from different layers have the same structure, *i.e.* the same combination of operation types.

Then, finding the best cell architecture corresponds to solving the following bi-level optimization problem:

$$\begin{aligned} \min_{Z} \quad & \mathcal{L}_{\mathsf{val}}(\Theta^*(Z), Z) \\ \text{s.t.} \quad & \Theta^*(Z) = \operatorname*{argmin}_{\Theta} \mathcal{L}_{\mathsf{train}}(\Theta, Z), \end{aligned} \tag{2}$$

where $Z = \{\mathbf{z}^{(i,j)} \mid (i,j) \in \mathcal{E}(C), C \in \mathcal{C}\}$, $\Theta = \{\theta^{(i,j)} \mid (i,j) \in \mathcal{E}(C), C \in \mathcal{C}\}$, and $\mathcal{L}_{\mathsf{val}}$ and $\mathcal{L}_{\mathsf{train}}$ are validation loss and training loss, respectively. Note that this loss splitting is typically used in meta-learning processes including NAS for better generalization. This is a bi-level program in the mixed domain of continuous variables (Θ) and discrete variables (Z), which is hard to solve. DARTS (Liu et al., 2018b) and proxylessNAS (Cai et al., 2018) try to circumvent this difficulty by relaxing Z to a continuous operation-weight variables $\bar{Z}$ such that $\bar{z}_o^{(i,j)} \in [0,1]$ and $\sum_{o\in\mathcal{O}} \bar{z}_o^{(i,j)} = 1$ and making $\mathcal{L}_{\mathsf{val}}(\Theta, \bar{Z})$ be differentiable with respect to both of Θ and $\bar{Z}$. This allows to use a SGD-based optimization to obtain an approximate solution $(\Theta^*, \bar{Z}^*)$ and derive the final architecture from the relaxed variables $\bar{Z}^*$ by taking the operation with the highest weight on each edge.

One problem with this method is that the performance of the final architecture is inconsistent with the performance of the relaxed architecture since the relaxed architecture is not guaranteed to be converged to the final architecture. Hence, they necessarily have to retrain Θ from the scratch after obtaining the final network architecture. Moreover, applying this method directly to a large-scale task such as high-resolution 3D medical image segmentation is infeasible due to the extremely large memory usage and the long run-time during the training to compute the loss functions $\mathcal{L}_{\mathsf{val}}$ and $\mathcal{L}_{\mathsf{train}}$ as well as their gradients from the fact that the required resources are proportional to the number of nonzero entries in $\bar{Z}$, which scales with the number of candidate operations.

To overcome the aforementioned problems, we propose a modified optimization, called *stochastic bi-level optimization*, by first treating Z as random discrete variables and then replacing (2) as

$$\begin{aligned} \min_{\alpha} \quad & \mathbb{E}_{Z\sim P_\alpha}[\mathcal{L}_{\mathsf{val}}(\Theta^*(Z), Z)] \\ \text{s.t.} \quad & \Theta^*(Z) = \operatorname*{argmin}_{\Theta} \mathcal{L}_{\mathsf{train}}(\Theta, Z), \end{aligned} \tag{3}$$

where P_α is the discrete distribution on Z, parameterized by α. Since it is intractable to exactly compute $\nabla_\alpha \mathbb{E}_{Z\sim P_\alpha}[\mathcal{L}_{\mathsf{val}}(\Theta^*(Z), Z)]$, this gradient with respect to α is estimated by a continuous relaxation with sampling on Z in order to use the gradient-based bi-level optimization method in this work.

Algorithm 1: Gradient-based stochastic bi-level optimization

```
Initialize $\alpha$ and $\Theta$;
while not done do
    $\hat{Z} \longleftarrow \mathsf{GumbelSoftmaxSample}(\alpha, \tau)$;
    Update $\Theta$ by a gradient descent using $\nabla_{\Theta}\mathcal{L}_{\mathsf{train}}(\Theta, \hat{Z})$;
    Update $\alpha$ by a gradient descent using $\nabla_{\alpha}\mathcal{L}_{\mathsf{val}}(\Theta, \hat{Z})$;
    Anneal $\tau$;
end
```

Algorithm 2: $\mathsf{GumbelSoftmaxSample}(\alpha, \tau)$

```
for $(i, j) \in \mathcal{E}$ do
    $\epsilon_o^{(i,j)} \sim \mathsf{Gumbel}(0, 1), \quad o \in \mathcal{O}$;
    $\bar{\mathbf{z}}^{(i,j)} \longleftarrow \mathsf{Softmax}((\alpha^{(i,j)} + \epsilon^{(i,j)})/\tau)$;
    foreach pair $\{o_1, o_2\}$ in $\mathcal{O}$ do
        $q^{\{o_1,o_2\}} \longleftarrow \frac{\bar{z}_{o_1}^{(i,j)} + \bar{z}_{o_2}^{(i,j)}}{|\mathcal{O}|-1}$;
    end
    Sample $\{o_1, o_2\}$ with probability $q^{\{o_1,o_2\}}$;
    $\left(\hat{z}_{o_1}^{(i,j)}, \hat{z}_{o_2}^{(i,j)}\right) \longleftarrow \left(\frac{\bar{z}_{o_1}^{(i,j)}}{\bar{z}_{o_1}^{(i,j)} + \bar{z}_{o_2}^{(i,j)}}, \frac{\bar{z}_{o_2}^{(i,j)}}{\bar{z}_{o_1}^{(i,j)} + \bar{z}_{o_2}^{(i,j)}}\right), \quad \hat{z}_o^{(i,j)} \leftarrow 0, \quad o \notin \{o_1, o_2\}$;
end
return $\hat{Z}$;
```

3.3. Gumbel-Softmax Relaxation with Operation Sampling

The Gumbel-softmax reparametrization technique (Jang et al., 2017; Maddison et al., 2016) can approximate the above gradient by continuous relaxation as

$$\nabla_{\alpha}\mathbb{E}_{Z\sim P_{\alpha}}[\mathcal{L}_{\mathsf{val}}(\Theta^*(Z), Z)] \approx \mathbb{E}_{\epsilon\sim\mathsf{Gumbel}(0,1)}[\nabla_{\alpha}\mathcal{L}_{\mathsf{val}}(\Theta^*(\bar{Z}(\alpha, \epsilon; \tau)), \bar{Z}(\alpha, \epsilon; \tau)], \tag{4}$$

where continuously relaxed variables $\bar{Z}(\alpha, \epsilon; \tau) = \mathsf{Softmax}((\alpha + \epsilon)/\tau)$, τ denotes the temperature, and ϵ is α-independent random variables drawn from the Gumbel distribution. Here, the expectation in (4) is approximated with ϵ-sampling. It is noted that as $\tau \to 0$, the distribution of $\bar{Z}$ is identical to P_{α}, which means that by annealing τ we can enforce $\bar{Z}$ to be one-hot discrete variables Z during the training; the relaxed architecture is forced to be converged to the final architecture.

Algorithm 1 summarizes our stochastic bi-level optimization algorithm which alternately updates Θ and α by respective gradient descents. Here, note that in order to reduce the number of nonzero operation weights in $\bar{Z}$ and hence to reduce the computational cost, in each iteration during the training we again replace $\bar{Z}$ with $\hat{Z}$ by sampling two operations from the Gumbel-softmax and then rescaling the corresponding two operation weights to be summed to one with zero weights of the other operations on each edge, as shown in Algorithm 2.

Owing to our continuous relaxation based on the Gumbel-softmax with τ-annealing, the number of sampled operations on each edge is naturally reduced from two to one during the training. As a result, the proposed differentiable NAS with stochastic operation sampling is able to support improved scalability in terms of solvable large search space with small computational cost.

4. Experiments

Dataset The proposed scalable NAS (SCNAS) was evaluated on the three segmentation tasks of 3D MRI data, (1) brain tumor (484 labeled images, 3 classes), (2) heart (20 labeled images, 1 class), and (3) prostate (32 labeled images, 2 classes), from the Medical Segmentation Decathlon challenge (MSD, http://medicaldecathlon.com) where each task has different MRI sequences as well as different foreground classes, which is therefore suitable for evaluating the generalizability and transferability of the SCNAS.

Implementation Details We compared the SCNAS to the state-of-the-art architecture, 3D U-ResNet with the use of multiple segmentation maps (Kayalibay et al., 2017) and attention gates (Oktay et al., 2018), and the random architecture by random selection of edge-operations in each cell from the same architecture search space in the SCNAS. The set of operations $\mathcal{O}$ on each edge in the SCNAS consists of the following eight operations: $3 \times 3 \times 3$ convolutions, depthwise separable dilated $3 \times 3 \times 3$ convolutions with rate 2, 3 and 4, $3 \times 3 \times 3$ max and average 3D pooling, identity (skip connection), and zero. Here, we used the LeakyReLU-Conv-InstanceNorm for convolutional operations.

As shown in Figure 1, the whole network in the SCNAS is composed of 12 automatically designed cells, each of which has 4 nodes. This number of stacked cells is consistent with that of the 3D U-ResNet in terms of respective three times of downsampling and upsampling by a factor of 2. Here, all operations in the reduction cell in the SCNAS are of stride two while the expansion cells perform pre-upsampling for the inputs of the cell. Since the 3D U-ResNet in this evaluation was set to have 32 output channels in the first convolutional block, the number of output channels in the stem cell of the SCNAS was set to 32, and also similar to the 3D U-ResNet, the reduction and expansion cells in the SCNAS respectively double and halve the number of output channels of given inputs.

In both of the 3D U-ResNet and the SCNAS, patch-based training and inference were carried out such that each image was randomly cropped to the region of nonzero values with the predefined resolution during the training, while in testing, the prediction results were obtained by combining patch-based inference results with 50 percent overlap. Similar to Isensee et al. (2018), the predefined resolution for the input patch was set to $128 \times 128 \times 128$ for the tasks of brain tumors and heart while for the prostate task, the length of the z-axis was reduced to 24. Since even the same task provides 3D images with heterogeneous voxel spacings, the input images were first resized for all voxel spacings to be physically equal using the given meta-data, and then z-normalization was separately applied to each input channel. Note that unlike Isensee et al. (2018), any heuristic pre-/post-processing techniques including data augmentation, network-cascade, and prediction-ensemble were not adopted in this evaluation to solely examine the effects by the use of NAS in designing the network architecture.

Since the ground-truth labels for test images are not provided in the MSD dataset, the evaluation is conducted by 5-fold cross-validation (CV) on the training images with the average dice similarity coefficient (DSC) as the metric, and accordingly, we applied the well-known multi-class dice loss function (Milletari et al., 2016b; Isensee et al., 2018). With the ADAM optimizer, the 3D U-ResNet and the SCNAS models were trained for 300 epochs and 400 epochs, respectively, taking their convergences into consideration. The 3D U-ResNet was set with the batch size as 8, initial learning rate as 0.0001, and beta parameters for ADAM optimizer as $(0.9, 0.999)$ while in the SCNAS, with the batch size 1, the initial learning rates / beta parameters were as set to be 0.025 / $(0.1, 0.001)$ for training operation parameters Θ and 0.003 / $(0.5, 0.999)$ for training architecture parameters α. If a plateau for 20 epochs on the training loss was detected, the learning rate was reduced by a factor of 10. All experiments were conducted on V100 GPUs, and the implementation was done using PyTorch (Paszke et al., 2017).

Architecture Transfer Since the heart and prostate tasks only have 20 and 32 labeled MRI images, respectively, the 3D U-ResNet as well as the SCNAS can be prone to overfitting on the training set and hence to resulting in performance degradation on the validation set. Therefore, we transferred the optimized architecture obtained from the brain tumor task, which has 484 labeled MRI images, by the SCNAS into these two tasks having scarce data and retrained only the operation parameters on each task, in order to demonstrate that the SCNAS produces a more generalizable neural architecture for the similar tasks of 3D MRI image segmentation. Here, the transferred architecture came from the first CV fold in the brain tumor task.

Results Table 1 shows that the SCNAS produced better architectures than the (human-designed) 3D U-ResNet as well as the randomly designed 3D U-Net in terms of the overall performances on all three tasks. Especially, on the heart and prostate segmentation tasks, the transferred architecture from the brain tumor task achieved significantly better generalization performances. Note that the obtained architectures by the SCNAS have been also shown that the number of neural operation parameters and the computational complexity for output prediction (in terms of FLOPs) were significantly reduced compared to the 3D U-ResNet. We observed the performance degradation of the 3D U-ResNet when the number of initial output channels was halved. It is also noted that most previous NAS approaches retrained the neural operation parameters after completing architecture optimization because of the utilization of a biased architecture during the training, while the SCNAS simultaneously optimized both of the architecture parameters and neural operation parameters with an unbiased architecture and thereby removed the requirement of retraining. We conjecture that Isensee et al. (2018) might be benefit from complicated pre-/post-procedures and thus obtained slightly better performances than the SCNAS. Some example images and the corresponding segmentation outputs are included in Appendix A, and the details of the optimized cell architectures by the SCNAS are presented in Appendix B.

5. Conclusion

In this work, a complete NAS framework for automatically designing an architecture is proposed and demonstrated on the benchmark dataset of 3D medical image segmentation

Table 1: Mean DSC in Brain Tumor, Heart, and Prostate.

		Brain Tumor			
Model	GFLOPs, Params	Edema	Non-Enhancing	Enhancing	Average
3D U-ResNet	881, 7.6M	79.10 ± 1.80	**58.38** ± 1.29	77.37 ± 2.76	71.61
Random Search	152, 2.7M	**79.59** ± 1.28	57.97 ± 1.36	77.85 ± 1.35	71.80
SCNAS	129, 2.2M	79.42 ± 1.45	58.01 ± 1.46	**78.68** ± 1.80	**72.04**
		Heart	Prostate		
Model	GFLOPs, Params	Left Atrium	Peripheral	Transitional	Average
3D U-ResNet	870-163, 7.6M	89.60 ± 2.35	48.37 ± 1.44	79.17 ± 4.30	63.77
Random Search	104-18, 1.5M	89.14 ± 2.74	50.78 ± 1.22	79.58 ± 5.01	65.18
SCNAS	136-32, 3.0M	89.99 ± 1.32	49.70 ± 1.23	80.89 ± 3.19	65.30
SCNAS(transfer)	193-37, 4.2M	**90.47** ± 1.70	**53.81** ± 1.30	**82.02** ± 4.52	**67.92**

tasks. In the proposed framework, NAS is formulated as finding the optimal structure of four types of cells composing an encoder as well as a decoder, and both the architecture parameters and the neural operation parameters are learned by gradient descent in an end-to-end manner. We introduce a novel stochastic sampling algorithm which results in significant improvement in terms of the scalability suitable for handling high-resolution 3D medical images and also reduces the inconsistency of the train-time architecture against the final architecture, which leads to avoid the retraining of the operation parameters. Empirical evaluation demonstrates that the automatically optimized network via the proposed NAS outperforms the manually designed 3D U-Net. Moreover, the architecture learned from a task with the large number of training data is successfully transferred to different MRI segmentation tasks with the small number of data.

The analysis using more candidate operations and different cell structures of the same type at different layer levels are left for future research. In addition, the effects of including non-architectural procedures such as data augmentation, network-cascade, and prediction-ensemble in the proposed NAS framework need to be analyzed in future works. Another interesting research direction would be applying the NAS framework, either directly or by architecture transfer, to other medical modalities including CT, mammography, and X-ray.

Acknowledgments

We thank the Kakao Brain Cloud team for supporting to efficiently use GPU clusters for large-scale experiments.

References

Maximilian Alber, Irwan Bello, Barret Zoph, Pieter-Jan Kindermans, Prajit Ramachandran, and Quoc V. Le. Backprop evolution. In *ICML 2018 AutoML Workshop*, 2018.

Marcin Andrychowicz, Misha Denil, Sergio Gómez, Matthew W Hoffman, and David Pfau. Learning to learn by gradient descent by gradient descent. In *NIPS*, 2016.

Irwan Bello, Barret Zoph, Vijay Vasudevan, and Quoc V. Le. Neural optimizer search with reinforcement learning. In *ICML*, 2017.

Gabriel Bender, Pieter-Jan Kindermans, Barret Zoph, Vijay Vasudevan, and Quoc Le. Understanding and simplifying one-shot architecture search. In *ICML*, 2018.

Andrew Brock, Theodore Lim, James M Ritchie, and Nick Weston. Smash: one-shot model architecture search through hypernetworks. In *ICLR*, 2018.

Han Cai, Ligeng Zhu, and Song Han. Proxylessnas: Direct neural architecture search on target task and hardware. *arXiv preprint arXiv:1812.00332*, 2018.

Özgün Çiçek, Ahmed Abdulkadir, Soeren S. Lienkamp, Thomas Brox, and Olaf Ronneberger. 3d u-net: Learning dense volumetric segmentation from sparse annotation. *MICCAI, Springer, LNCS*, 9901:424–432, 2016.

Dan Ciresan, Alessandro Giusti, Luca M. Gambardella, and Jürgen Schmidhuber. Deep neural networks segment neuronal membranes in electron microscopy images. In *NIPS*, 2012.

Luca Franceschi, Paolo Frasconi, Saverio Salzo, Riccardo Grazzi, and Massimilano Pontil. Bilevel programming for hyperparameter optimization and meta-learning. In *ICML*, 2018.

Mohammad Havaei, Axel Davy, David Warde-Farley, Antoine Biard, Aaron C. Courville, Yoshua Bengio, Chris Pal, Pierre-Marc Jodoin, and Hugo Larochelle. Brain tumor segmentation with deep neural networks. *CoRR*, abs/1505.03540, 2015. URL http://arxiv.org/abs/1505.03540.

Fabian Isensee, Jens Petersen, Andre Klein, David Zimmerer, Paul F. Jaeger, Simon Kohl, Jakob Wasserthal, Gregor Koehler, Tobias Norajitra, Sebastian Wirkert, and Klaus H. Maier-Hein. nnu-net: Self-adapting framework for u-net-based medical image segmentation. 2018.

Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. In *ICLR*, 2017.

Konstantinos Kamnitsas, Christian Ledig, Virginia F. J. Newcombe, Joanna P. Simpson, Andrew D. Kane, David K. Menon, Daniel Rueckert, and Ben Glocker. Efficient multi-scale 3d CNN with fully connected CRF for accurate brain lesion segmentation. *CoRR*, abs/1603.05959, 2016. URL http://arxiv.org/abs/1603.05959.

Baris Kayalibay, Grady Jensen, and Patrick van der Smagt. Cnn-based segmentation of medical imaging data. *CoRR*, abs/1701.03056, 2017. URL http://arxiv.org/abs/1701.03056.

Christiane Lemke, Marcin Budka, and Bogdan Gabrys. Metalearning: a survey of trends and technologies. *Artificial Intelligence Review*, 44(1), 2015.

Ke Li and Jitendra Malik. Learning to optimize neural nets. *CoRR*, abs/1703.00441, 2017. URL http://arxiv.org/abs/1703.00441.

Chenxi Liu, Barret Zoph, Jonathon Shlens, Wei Hua, Li-Jia Li, Li Fei-Fei, Alan L. Yuille, Jonathan Huang, and Kevin Murphy. Progressive neural architecture search. In *ECCV*, 2018a.

Hanxiao Liu, Karen Simonyan, and Yiming Yang. Darts: Differentiable architecture search. *arXiv preprint arXiv:1806.09055*, 2018b.

Chris J Maddison, Andriy Mnih, and Yee Whye Teh. The concrete distribution: A continuous relaxation of discrete random variables. *arXiv preprint arXiv:1611.00712*, 2016.

Fausto Milletari, Nassir Navab, and Seyed-Ahmad Ahmadi. V-net: Fully convolutional neural networks for volumetric medical image segmentation. *CoRR*, abs/1606.04797, 2016a. URL http://arxiv.org/abs/1606.04797.

Fausto Milletari, Nassir Navab, and Seyed-Ahmad Ahmadi. V-net: Fully convolutional neural networks for volumetric medical image segmentation. In *3D Vision (3DV), 2016 Fourth International Conference on*, pages 565–571. IEEE, 2016b.

Aliasghar Mortazi and Ulas Bagci. Automatically designing cnn architectures for medical image segmentation. In *MLMI*, 2018.

Aliasghar Mortazi, Rashed Karim, Kawal S. Rhode, Jeremy Burt, and Ulas Bagci. Cardiacnet: Segmentation of left atrium and proximal pulmonary veins from mri using multi-view cnn. *MICCAI, Springer, LNCS*, 10434:377–385, 2017.

Mostefa Ben Naceur, Rachida Saouli, Mohamed Akil, and Rostom Kachouri. Fully automatic brain tumor segmentation using end-to-end incremental deep neural networks in mri images. *Computer Methods and Programs in Biomedicine, Elsevier*, 166:39–49, 2018.

Vladimir Nekrasov, Hao Chen, Chunhua Shen, and Ian Reid. Fast neural architecture search of compact semantic segmentation models via auxiliary cells. *arXiv preprint arXiv:1810.10804*, 2018.

Ozan Oktay, Jo Schlemper, Loïc Le Folgoc, Matthew C. H. Lee, Mattias P. Heinrich, Kazunari Misawa, Kensaku Mori, Steven G. McDonagh, Nils Y. Hammerla, Bernhard Kainz, Ben Glocker, and Daniel Rueckert. Attention u-net: Learning where to look for the pancreas. In *MIDL*, 2018.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

Hieu Pham, Melody Y Guan, Barret Zoph, Quoc V Le, and Jeff Dean. Efficient neural architecture search via parameter sharing. In *ICML*, 2018.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. *MICCAI, Springer, LNCS*, 9351:234–241, 2015.

Joaquin Vanschoren. Meta-learning: A survey. *arXiv preprint arXiv:1810.03548*, 2018.

Olga Wichrowska, Niru Maheswaranathan, Matthew W. Hoffman, Sergio Gomez Colmenarejo, Misha Denil, Nando de Freitas, and Jascha Sohl-Dickstein. Learned optimizers that scale and generalize. In *ICML*, 2017.

Lequan Yu, Xin Yang, Hao Chen, Jing Qin, and Pheng Ann Heng. Volumetric convnets with mixed residual connections for automated prostate segmentation from 3d mr images. In *AAAI*, 2017.

Xinbang Zhang, Zehao Huang, and Naiyan Wang. You only search once: Single shot neural architecture search via direct sparse optimization. *arXiv preprint arXiv:1811.01567*, 2018.

Barret Zoph and Quoc V. Le. Neural architecture search with reinforcement learning. In *ICLR*, 2017.

Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V. Le. Learning transferable architectures for scalable image recognition. In *CVPR*, 2018.

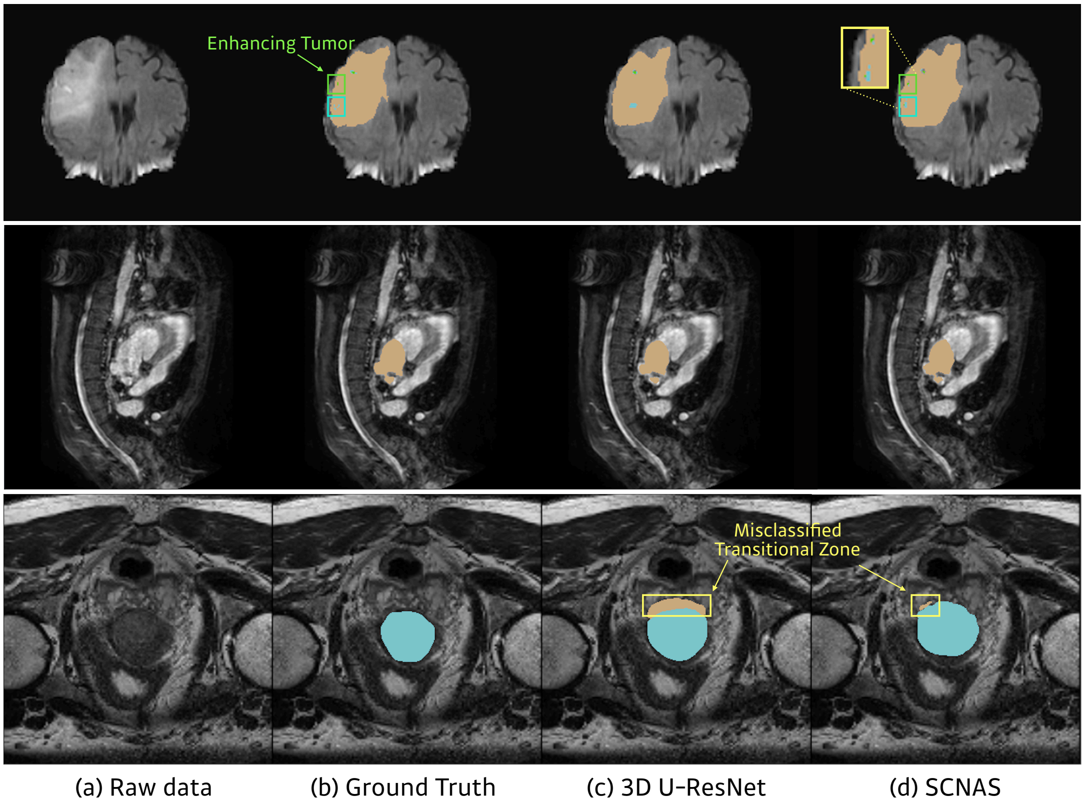

Figure 2: (a) Raw data. (b) Ground Truth. Segmentation results of (c) 3D U-ResNet (d) SCNAS in Brain Tumor (top), Heart (middle), and Prostate (bottom) from the MSD dataset. The prediction results on the heart and prostate data are obtained from transferred networks which are trained on brain tumor data.

In this appendix, we provide examples of segmentation predictions by the 3D U-ResNet and proposed method. For a qualitative assessment, we compare the two results with ground truth. Additionally, samples of cell architectures found by the SCNAS are illustrated in the sequent section.

Appendix A. Segmentation Samples

Figure 2 shows samples of 3D segmentation results of 3D U-ResNet and proposed method for each MSD dataset. The above samples demonstrate that the architectures found by SCNAS predict more accurately than 3D U-ResNet. Especially, the architecture found on brain tumor images can be transferred well to the dataset of different MRI segmentation tasks.

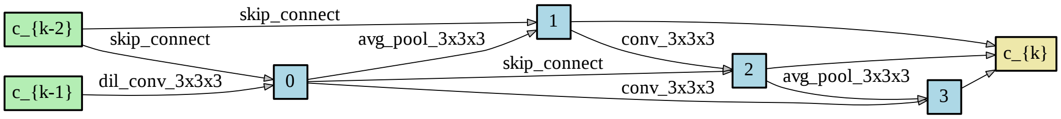

Figure 3: Reduction Cell

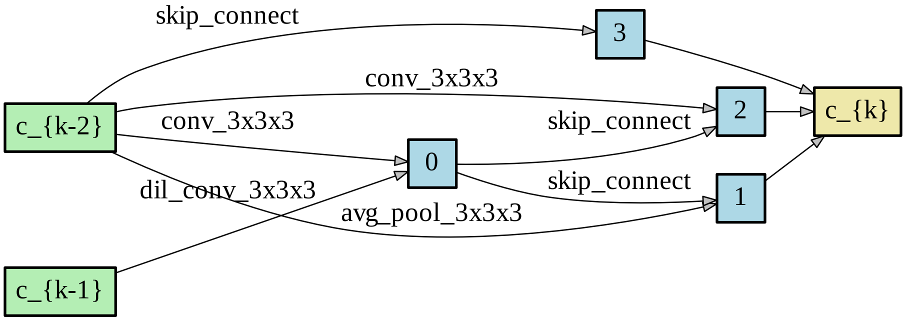

Figure 4: Encoder Normal Cell

Appendix B. Cell Architectures

Figure 3-6 show the samples of cell architectures which SCNAS found in the first CV fold experiment on the Brain Tumor dataset at the convergence.

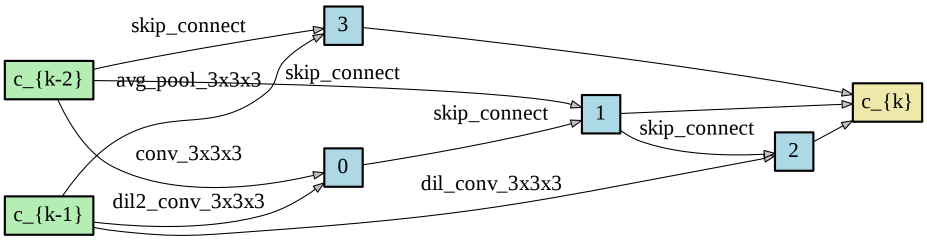

Figure 5: Expansion Cell

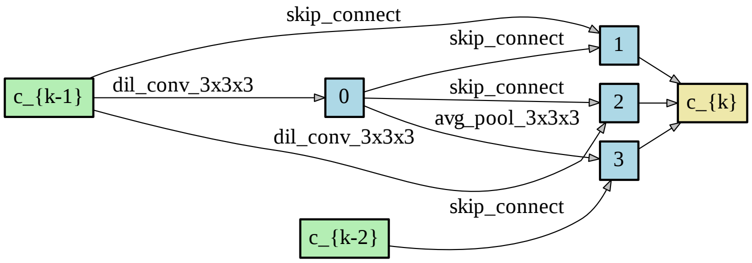

Figure 6: Decoder Normal Cell