Bioinformatics, 37(24), 2021, 4844–4850
doi: 10.1093/bioinformatics/btab556
Advance Access Publication Date: 30 July 2021
Original Paper

OXFORD

Bioimage informatics

Automatic improvement of deep learning-based cell segmentation in time-lapse microscopy by neural architecture search

Yanming Zhu and Erik Meijering *

School of Computer Science and Engineering, University of New South Wales, Sydney, NSW 2052, Australia

*To whom correspondence should be addressed.

Associate Editor: Jinbo Xu

Received on January 28, 2021; revised on July 18, 2021; editorial decision on July 25, 2021; accepted on August 4, 2021

Abstract

Motivation: Live cell segmentation is a crucial step in biological image analysis and is also a challenging task because time-lapse microscopy cell sequences usually exhibit complex spatial structures and complicated temporal behaviors. In recent years, numerous deep learning-based methods have been proposed to tackle this task and obtained promising results. However, designing a network with excellent performance requires professional knowledge and expertise and is very time-consuming and labor-intensive. Recently emerged neural architecture search (NAS) methods hold great promise in eliminating these disadvantages, because they can automatically search an optimal network for the task.

Results: We propose a novel NAS-based solution for deep learning-based cell segmentation in time-lapse microscopy images. Different from current NAS methods, we propose (i) jointly searching non-repeatable micro architectures to construct the macro network for exploring greater NAS potential and better performance and (ii) defining a specific search space suitable for the live cell segmentation task, including the incorporation of a convolutional long short-term memory network for exploring the temporal information in time-lapse sequences. Comprehensive evaluations on the 2D datasets from the cell tracking challenge demonstrate the competitiveness of the proposed method compared to the state of the art. The experimental results show that the method is capable of achieving more consistent top performance across all ten datasets than the other challenge methods.

Availabilityand implementation: The executable files of the proposed method as well as configurations for each dataset used in the presented experiments will be available for non-commercial purposes from https://github.com/291498346/nas_cellseg.

Contact: erik.meijering@unsw.edu.au

Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 Introduction

Live cell segmentation has received increasing attention in past years due to its key importance for further progress in biological research (Meijering, 2012; Xing and Yang, 2016). However, live cells in time-lapse microscopy sequences usually exhibit complex spatial structures and temporal behaviors, which make their segmentation a challenging task. In the past years, many researchers have made substantial efforts for tackling this task and achieved promising results (Dimopoulos *et al.*, 2014; Kong *et al.*, 2015). Recently, with the huge success of deep learning in various image processing problems, deep neural networks have been proposed for microscopy cell segmentation (Al-Kofahi *et al.*, 2018; Araújo *et al.*, 2019; Huang *et al.*, 2020; Jalali *et al.*, 2021).

Among the proposed networks, U-Net (Ronneberger *et al.*, 2015) is one of the most renowned due to its demonstrated effectiveness and efficiency. In fact, after this, the U-shaped network has become the *de facto* standard architecture for cell segmentation tasks. Several variants and improvements on U-Net have been made by researchers for better performance. For example, Drozdzal *et al.* (2016) proposed using both long and short skip connections for a U-shaped network to benefit the segmentation. Hollandi *et al.* (2020) proposed combining both Mask R-CNN and U-Net to predict the segmentation and thus improve the accuracy of the nucleus segmentation. Arbelle and Raviv (2019) proposed an integration of convolutional long short term memory (CLSTM) and U-Net to exploit temporal information to support the cell segmentation decisions. Long (2020) proposed an enhanced U-Net with a modified encoded

branch for cell nucleus segmentation in a low-resources computing scenario. Despite the impressive performance of these popular networks, their architectures were designed manually by experts for months or even years, and further improving them by hand is very time-consuming and labor-intensive, limiting the rate of progress in the field. Also, designing such networks requires a large amount of professional knowledge and expertise in the field, and thus traps the average researchers without such capability. In addition, a growing demand for increasingly more complex architectures has severely challenged researchers.

To tackle the above issues, neural architecture search (NAS), which can automatically search the optimal architecture for a given task, has emerged in recent years (Elsken *et al.*, 2019; Hutter *et al.*, 2019). NAS can be seen as a subfield of automated machine learning and its research focuses on three main aspects: search space, search strategy and performance estimation strategy. The search space defines which architectures can be represented in principle. Generally, it includes the selection of basic operations (BOs) used to build a block (micro structure) and the backbone architecture used to define the outer network (macro structure). Current works can be divided into two categories based on the definition of search space: (i) searching repeatable micro structure while keeping the macro structure fixed (Cai *et al.*, 2018b; Liu *et al.*, 2018; Zela *et al.*, 2019), and among them, the NASNet (Zoph *et al.*, 2018) is a representative; and (ii) jointly search repeatable micro structure and macro structure (Liu *et al.*, 2019; Yan *et al.*, 2020) for exploring more architectural variations. The search strategy details how to explore the search space, and the methods can be divided into three categories: (i) evolutionary algorithms (Lu *et al.*, 2019; Xie and Yuille, 2017), (ii) reinforcement learning-based methods (Cai *et al.*, 2018a; Zoph *et al.*, 2018) and (iii) gradient-based methods (Brock *et al.*, 2018; Liu *et al.*, 2018). The performance evaluation refers to the process of (i) estimating the performance of the candidate architectures to select an optimal architecture that achieves high predictive performance and (ii) evaluating the optimal architecture for final performance.

Since NAS was proposed, it has mainly solved natural image and language tasks. Concerning medical image segmentation tasks, only few works have been proposed (Mortazi and Bagci, 2018; Yang *et al.*, 2019). The current practice in this field is to respectively search repeatable micro structures for down and up blocks and construct a U-shaped network. For example, Weng *et al.* (2019) proposed a NAS-U-Net based on the U-Net for segmenting 2D prostate MRI (magnetic resonance imaging), liver CT (computed tomography) and nerve ultrasound images. Zhu *et al.* (2019) proposed a V-NAS, which is a DARTS-style (Liu *et al.*, 2018) differentiable NAS U-Net, for lung and pancreas 2D CT image segmentation. Kim *et al.* (2019) proposed a scalable NAS for 3D medical image segmentation based on a 3D U-Net. Wang and Biswal (2020) proposed a NAS solution for MRI gliomas image segmentation based on a 3D U-Net. There are also several works designed for non-U-shaped networks such as NAS based on adversarial network (Dong *et al.*, 2019) and deep belief network (Qiang *et al.*, 2019). Generally, the underlying idea of these methods is introducing common BOs (e.g. convolution, pooling, etc.) prevalent in current NAS methods to define the search space and searching repeatable micro architectures to construct the macro network. This limits the search space and potential of NAS to find more efficient architectures. Also, employing these methods for live-cell segmentation is likely not optimal, as they focus on spatial information and do not exploit temporal information. The recently proposed nnU-Net (Isensee *et al.*, 2021) is a self-configuring segmentation method inspired by U-Net, which has achieved promising performance in international biomedical segmentation competitions. However, unlike NAS, which aims to find an optimal architecture, it focuses on the non-architectural aspects in the segmentation methods and aims to design an automatic pipeline for given architectures.

Therefore, in this article, we propose a NAS-based solution for time-lapse microscopy cell segmentation with specifically defined search space and non-repeatable micro structures. Building on and significantly extending our preliminary conference report (Zhu and Meijering, 2020), we design four new BO sets for the micro architecture searching, which incorporate operations suitable for the time-lapse microscopy cell segmentation task. For example, we incorporate a CLSTM network to better explore the temporal information in time-lapse sequences. Also, different from the current NAS methods, we propose to jointly search non-repeatable micro structures to construct the macro network. This allows different layers of the macro network to have their own focuses, and thus allows the macro network to better achieve the exploration and fusion of multiscale features. We conduct experiments on all 2D cell tracking challenge (CTC) datasets (Ulman *et al.*, 2017) to comprehensively evaluate the performance of our method. Experimental results demonstrate the competitiveness of the proposed method compared to the state of the art. The searched networks achieve more consistent top performance on the ten datasets than the other challenge methods.

The contributions of this work are summarized as follows. First, we are the first to extend NAS to time-lapse microscopy cell segmentation. Second, we propose to jointly search non-repeatable micro structures to construct the macro network, which augments and complements the much-studied repeatable one. Third, we define a novel search space, which incorporates four types of candidate operation sets and searchable skip connections. And fourth, we show experimentally that the proposed NAS approach is capable of optimizing the architecture to improve the performance of a given macro network.

2 Materials and methods

In this section, we firstly describe our specifically defined search space. Then, our search strategy followed by the performance estimation strategy are presented.

2.1 Search space

We propose to jointly search non-repeatable micro architectures to construct the macro network. For the macro network, we still follow the routine of constructing a U-shaped network due to its success and transferability in biomedical image segmentation (Section 2.1.1). For the micro architecture, we propose a novel search space suitable for the time-lapse microscopy cell segmentation task (Section 2.1.2).

2.1.1 Macro network

In essence, U-shaped networks are composed of mutually connected down-sampling and up-sampling blocks. The down blocks are responsible for feature embedding which compresses the resolution and extracts target sensitive information, and the up blocks are responsible for mixing the embedded features with the outputs of horizontally corresponding down blocks to recover the position information for predicting the segmentation. In our NAS network, the macro network still follows this approach. Yet unlike the traditional U-Net, our macro network (i) has n down blocks and $n+2$ up blocks, which is determined by our task; (ii) the n down blocks and the first n up blocks are symmetrical, and there is no additional convolutional layer in the middle of the network; and (iii) the skip connection is included in the architecture search of the up block, so it is not a simple concatenation but an automatically searched operation.

Figure 1 illustrates the schematic of our NAS network. Its input is a time-lapse microscopy cell image and the outputs are the segmentation and markers images, the latter of which contain cell blobs that can be used to further improve the segmentation in case of adjacent cells in the image (see Supplementary Section S1 for more details). These two outputs are generated by blocks consisting of only a convolutional layer and a softmax or sigmoid layer. Ahead of the down blocks, there is a block used to generate the inputs for the following down blocks. It is added due to the required two inputs for each down or up block. These three blocks are fixed during the NAS. Therefore, for our NAS network, n down blocks and $n+2$ up blocks need to be searched. Note that these down and up blocks are jointly searched to have their own architectures rather than sharing

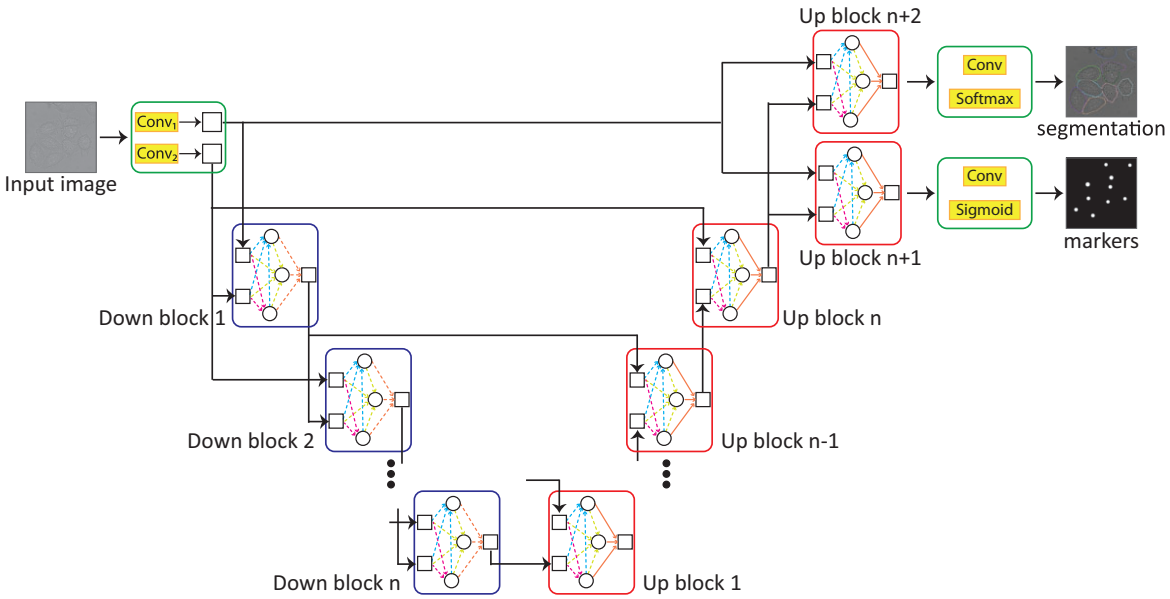

Fig. 1. Schematic of the proposed NAS network. The macro architecture is a U-shaped network, where the blue and red rectangles represent the down and up blocks to be searched respectively. Each down or up block has two inputs and one output (rectangles). The dashed line arrows within the blocks represent possible forward data paths and each possible path corresponds to a set of operations to be searched. The first green rectangle ahead of the down blocks consists of two convolutional layers followed by a group normalization layer and is used to generate the inputs for the subsequent down blocks. The last two green rectangles after the up blocks consist of only a convolutional layer and a softmax or sigmoid layer and are used to generate the output segmentation and markers. The solid line arrows represent the forward data paths

a same architecture. For the down block, its two inputs are defined as the outputs of the previous and pre-previous down blocks, and for the up block, its two inputs are defined as the output of the previous up block and the output of the horizontally corresponding down block.

2.1.2 Micro search space

To explain the search space, we start with the definition of a set of BOs which are the fundamental elements in the down and up blocks. The selection of BOs is important because it directly affects the NAS performance and efficiency. Based on the principle of no redundancy and reduced parameters, in this article, we propose to (i) limit all convolution operations to a size of 3×3 pixels and pooling operations to a size of 2×2 pixels, (ii) use convolution operations with a stride of 2 pixels to achieve the down and up operations, (iii) incorporate depthwise-separable convolution (Chollet, 2017) operations to reduce network parameters and (iv) incorporate shuffle convolution (Zhang *et al.*, 2018) operations to reduce computation cost. Also, considering the particularity of the time-lapse microscopy cell segmentation task that the sequences contain both spatial and temporal properties, we incorporate CLSTM operations to exploit cell dynamics. In addition, for better cell segmentation performance, we adopt the atrous convolution (Chen *et al.*, 2018) operation intentionally proposed for image segmentation and the squeeze-and-excitation (Hu *et al.*, 2018) operation aimed to suppress redundant features while enhance useful features. Overall, we design four sets of BOs for the micro architecture search, as summarized in Table 1.

Different BO sets have different BOs, which are designed based on their own functions. For example, the temporal BO set is used to capture the temporal information of cell sequences and thus contains only the CLSTM operation. The down BO set is used to halve the dimension of feature maps and thus contains various down-sampling operations. The same underlying idea goes for up and normal BO sets. The identity operation in the normal BO set refers to only group normalization and ReLU calculations, which aims to further reduce network parameters. Note that all BOs in down and up blocks are followed by group normalization and ReLU activation. The reasons for choosing group normalization are: (i) it has better performance than batch normalization (Wu and He, 2018) and (ii) it is suitable for segmentation tasks that have small batch size.

Table 1. Four BO sets designed for the micro architecture search

BO set type	BO candidates
Temporal (T)	CLSTM
Down (D)	1. Max-pooling
	2. Average-pooling
	3. Down conv
	4. Down atrous conv
	5. Down depthwise-separable conv
	6. Down squeeze-and-excitation
Up (U)	1. Up conv
	2. Up atrous conv
	3. Up depthwise-separable conv
	4. Up squeeze-and-excitation
Normal (N)	1. None
	2. Identity
	3. Conv
	4. Atrous conv
	5. Shuffle conv
	6. Depthwise-separable conv
	7. Squeeze-and-excitation

Note: We named them as T, D, U and N, respectively.

Before describing the down and up blocks, we first define the fundamental computing unit (CU) in them. A CU is constructed by all BO candidates from a BO set. Figure 2 exhibits its structure and we can see it refers to a fusion of multiple BOs. For each BO_i in a CU, there is a parameter w_i whose softmax transformation is assigned to it as a weight. This parameter controls the contribution of the BO to the CU output, and therefore, the CU output is a weighted sum of all the BOs. During the search, the weights of BOs that contribute more to the CU output will be increased and the weights of BOs that contribute less to the CU output will be reduced. Thus, the NAS can determine the optimal BO by selecting the largest weight. We name these parameters as BO parameters.

In our NAS network, there are four types of CUs (named as T, D, U and N CUs respectively) which are corresponding to the four BO sets (T, D, U and N BO sets). For example, the U CU contains the

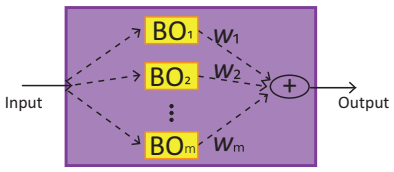

Fig. 2. The structure of the CU where BO_i is selected from a BO set, w_i is its weight and m is the number of BOs in the set

four BOs in the U BO set. The same underlying ideas go for T, D and N CUs. In the following description of the down and up blocks, for conciseness, we omit the structure of a CU and only use a purple rectangle labeled with CU type (T, D, U and N) to represent it.

As mentioned above, our NAS jointly searches non-repeatable micro architectures to construct the macro structure. This means each down and up block has its own architecture. Yet to avoid an overly large search space, they share the same basic structures. Figure 3 illustrate the basic structures for down and up blocks respectively. Each block has two inputs, three nodes and one output. The inputs are the feature maps learned by the previous layers. The nodes are clusters of CUs, represented as circles in the figure. The number of CUs in these three nodes are in ascending order because in addition to the two inputs, the output of the previous nodes will be input to the following nodes. For the down block, its output is the concatenation of the outputs of three D CUs connecting to the three nodes. For the up block, its output is the concatenation of the outputs of three nodes.

The dashed lines represent possible forward data paths connecting to CUs. The NAS takes control of selecting paths for each node and selecting BOs for each CU. We define the search space for down and up blocks as follows. For the down block, since we expect it to extract both spatial and temporal characteristics and embed the target sensitive features, we define its BOs search from a space composed of T, D and N BO sets. Specifically, the BOs corresponding to the two inputs are taken from the T BO set, the BOs corresponding to the block output are from the D BO set, and all the other BOs are from the N BO set. For the up block, since its responsibility is to decode the embedded features and search for the connection operation

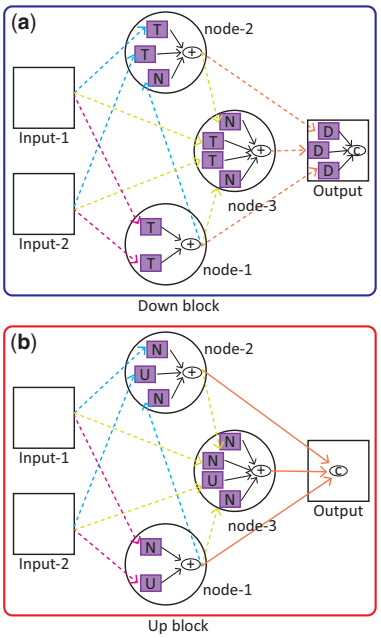

Fig. 3. Diagrams of basic block structures: (**a**) the down block and (**b**) the up block. The circles represent the nodes. The rectangles labeled with T, D, U and N, represent the T, D, U and N CUs, respectively. The + symbol represents the sum and the c represents the concatenation

with high-level features in the horizontally corresponding down block, we define its BOs search from a space composed of U and N BO sets. Specifically, the BOs corresponding to the input from the previous up block are taken from the U BO set and all the other BOs are from the N BO set. This way, the information transfer operation achieved by the skip connections in traditional methods is automatically searched. Also, by including the T BO set in the search space for the down block, our network can encode spatial-temporal features, which is especially suitable for segmentation in time-lapse sequences. In addition, there are fewer BOs in the up blocks compared to the down blocks, which reduces the size of the search space and simplifies the search.

2.2 Search strategy

The search process aims to find the optimal micro architecture of each block. More specifically, it decides (i) which forward data path is selected for each node and (ii) which BO is selected for each CU. In our work, we require each node to have two paths for the searched network and each CU will keep only one BO. For the BO selection, the NAS learns the BO parameters and the BO with the highest w_i will be kept. Regarding the path selection, current methods (Wang and Biswal, 2020; Weng *et al.*, 2019) also use the highest BO parameters w_i for making decision. This is feasible when the number of BOs in the BO sets is more or less the same, but when the number of BOs in different BO sets is imbalanced, the NAS tends to select the path with a small number of BOs because such BO weights tend to be higher than those of BOs in a large BO set. Therefore, we propose to assign a weight w_i for each CU to tackle this issue. We name these parameters as CU parameters. This way, the path can be selected based on the CU parameter corresponding to this path.

Overall, in this article, there are two classes of parameters: architecture parameters (including BO and CU parameters) and kernel parameters (the values of the filters). Therefore, we choose to use a gradient-based method to alternately optimize the two classes of parameters. For each class of parameters, there is a separate optimizer and the two optimizers worked sequentially in each iteration. Specifically, when training the architecture parameters, the kernel parameters are fixed, and vice versa. Once the training of architecture parameters is finished, the optimal architecture can be derived by pruning redundant BOs and paths. Finally, in order to save GPU memory, we use the method proposed in (Cai *et al.*, 2018b) as our search strategy.

2.3 Performance estimation strategy

The performance estimation strategy aims to evaluate the performance of the architecture candidates without using a standard training and validation process so as to reduce the computation cost. In our work, during the NAS, we divide the training dataset into about 60% training set and 40% validation set, and the performance is estimated on a subset of the validation set. Specifically, 60 batches of size 8 samples from the validation set. Then, after obtaining the optimal architecture, we train it from scratch using the entire training dataset and report its performance on the test dataset.

3 Results

In this section, we firstly present the implementation details of our NAS setup. Then, the datasets and metrics used for performance evaluation are described. Finally, we show the searched networks and report the segmentation performance on benchmark datasets.

3.1 Experimental settings

The configurations of our NAS are empirically set as follows. The number of the down blocks n is set to be three, and thus the number of up blocks is five. Therefore, the total size of the search space is in the order of 10^{10} possible network configurations and the parameter size is about 0.29 M. The number of kernels (channels) in the feature map of each BO is set to 16 and we do not double the number when halving or doubling the resolution of the feature map. Therefore, the

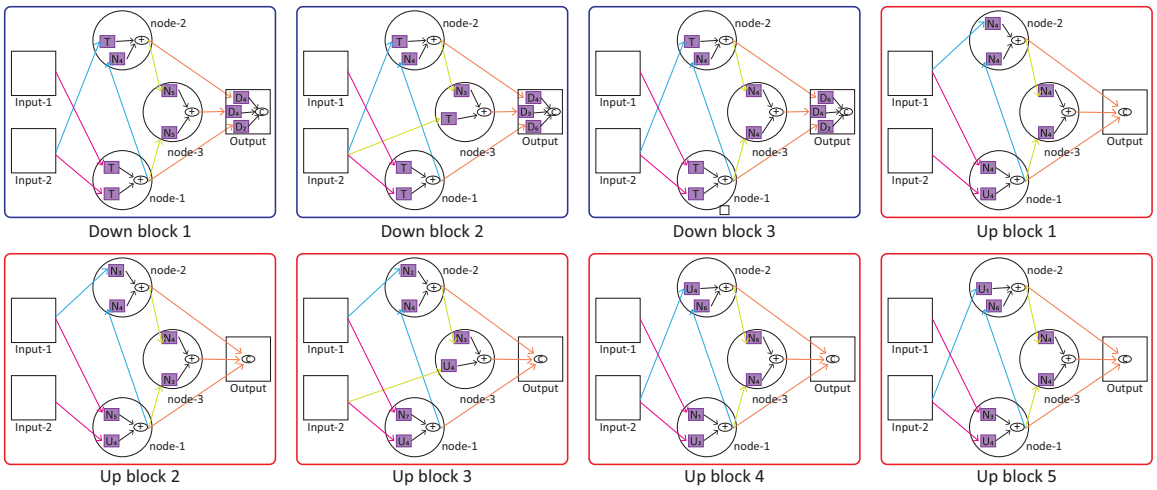

Fig. 4. The architectures found by NAS for down (blue rectangle) and up (red rectangle) blocks for dataset DIC-C2DH-HeLa. The digit following a letter represents the BO number in the BO set corresponding to the letter. For example, N_4 represents the atrous convolution

Table 2 Segmentation results and ranks of our NAS-produced networks on the ten 2D CTC datasets compared with the state-of-the-art (top-1)

Method	State-of-the-Art	Our Networks	
	OP_{CSB} (#); SEG (#); DET (#)	OP_{CSB}; SEG; DET	Ranks/count
Dataset			
BF-C2DL-HSC	0.905 (1); 0.818 (1); 0.995 (our)	0.893; 0.792; 0.995	3; 3; 1/14
BF-C2DL-MuSC	0.878 (1); 0.777 (1); 0.982 (2)	0.805; 0.644; 0.966	10; 11; 6/14
DIC-C2DH-HeLa	0.925 (3); 0.870 (3); 0.979 (3)	0.912; 0.863; 0.960	3; 4; 3/27
Fluo-C2DL-Huh7	0.843 (our); 0.752 (our); 0.935 (our)	0.843; 0.752; 0.935	1; 1; 1/5
Fluo-C2DL-MSC	0.761 (4); 0.687 (our); 0.876 (4)	0.760; 0.687; 0.832	2; 1; 3/32
Fluo-N2DH-GOWT1	0.952 (2); 0.938 (5); 0.980 (6)	0.948; 0.933; 0.963	2; 2; 5/43
Fluo-N2DH-SIM+	0.905 (7); 0.832 (7); 0.983 (4)	0.887; 0.807; 0.967	5; 6; 10/38
Fluo-N2DL-HeLa	0.954 (8); 0.923 (8); 0.994 (1)	0.951; 0.917; 0.984	3; 3; 13/41
PhC-C2DH-U373	0.959 (3); 0.927 (3); 0.991 (3)	0.954; 0.927; 0.982	3; 2; 7/31
PhC-C2DL-PSC	0.859 (1); 0.743 (1); 0.975 (1)	0.847; 0.733; 0.962	2; 2; 5/33

Note: The (arbitrary) numbers in parentheses indicate different methods. It can be seen that the top-1 performances are achieved by different methods, including ours. The count in the last column is the total number of competing methods.

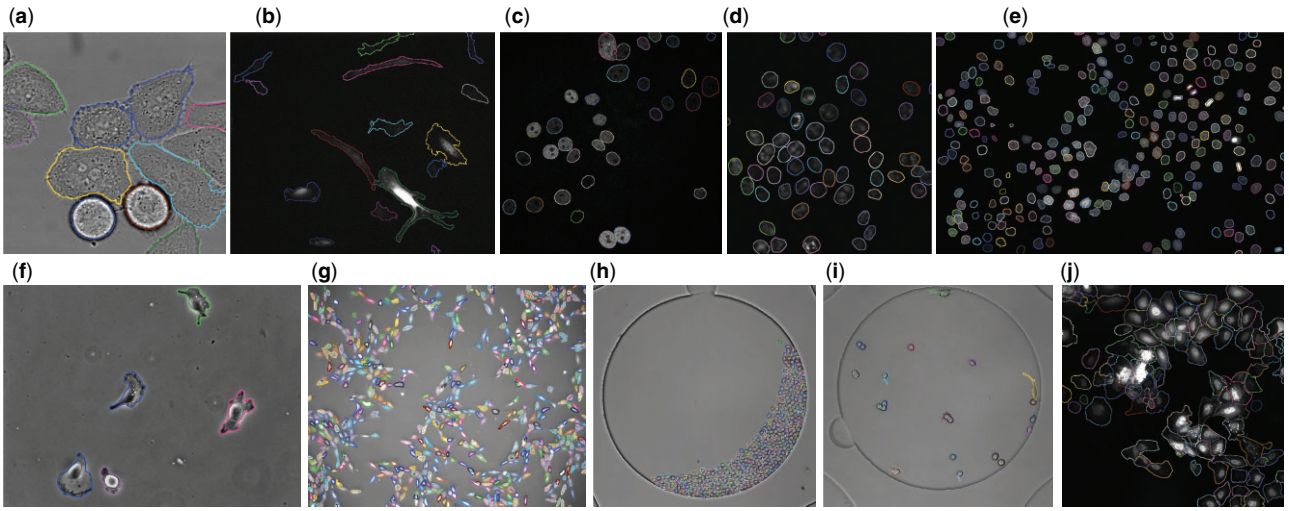

Fig. 5. Segmentation results (colored contours) of our NAS-produced networks on the ten 2D CTC datasets. (**a**) DIC-C2DH-HeLa, (**b**) Fluo-C2DL-MSC, (**c**) Fluo-N2DH-GOWT1, (**d**) Fluo-N2DH-SIM+, (**e**) Fluo-N2DL-HeLa, (**f**) PhC-C2DH-U373, (**g**) PhC-C2DL-PSC, (**h**) BF-C2DL-HSC, (**i**) BF-C2DL-MuSC and (**j**) Fluo-C2DL-Huh7

output of each node has 16 channels because it is the sum of the BOs, and the output of each block has 48 channels because it is the concatenation of the outputs of three nodes. The input to the NAS network is randomly cropped patches of size of 256×256 pixels from original images. We also randomly augment the data for increasing variability by (i) random horizontal and vertical flip, (ii) random 90° rotation, (iii) random sequence reverse ([*T*,*T*-1,...,2,1]) and (iv) random affine and elastic transformations. The batch size is set to be 8 and the NAS is stopped when the architecture does not change for 60 epochs or when the maximum number of epochs 300 is reached. The unroll length for the CLSTM is 2. We use Adam (Kingma and Ba, 2015) optimizer with learning rate 0.0003 and weight decay 0.0001 to learn both the architecture and kernel parameters. The experiments are run using four NVIDIA V100 GPUs (see Acknowledgements for used resources and Supplementary Section S4 for running times).

The loss functions used to train the model are as follows. As shown in Figure 1, the network generates two outputs, namely the segmentation and markers images. The segmentation image has two (or four) channels (depending on the dataset as detailed in Supplementary Table S1) representing the probability of each pixel belonging to the background and cell regions (and touch and gap regions). The markers image is a single-channel map representing the probability of each pixel belonging to a cell marker. The ground-truth training images for both are obtained from the annotated cell instance segmentation and tracking labels provided with the CTC datasets (see Supplementary Section S2 for details). Using these, we employ the classical weighted cross-entropy loss function (Ronneberger *et al.*, 2015) to penalize the segmentation, and a binary cross-entropy loss function (Arbelle and Raviv, 2019) to penalize the markers, and the final loss is set to be the sum of the two.

3.2 Datasets and metrics

We use all ten 2D CTC benchmark datasets to comprehensively evaluate the performance of our method. These datasets are time-lapse image sequences of moving and dividing cells, recorded using fluorescence, bright-field, phase-contrast or differential interference contrast (DIC) microscopy. The videos cover a wide range of cell types, spatial and temporal resolutions and signal-to-noise ratios, which makes the evaluation of our method more persuasive. For each sequence, there are two datasets: one contains both original image data and reference annotations for training and another without annotations for testing. The performance metrics of the cell segmentation benchmark (OP_{CSB}, SEG and DET) are used to evaluate the segmentation performance. For full details about the datasets and the performance metric, we refer to Ulman *et al.* (2017).

3.3 Performance evaluation

Application of our proposed NAS method produces architectures for each dataset. As an example, see Figure 4 for the DIC-C2DH-HeLa dataset (Supplementary Section S3 shows the architectures for the other datasets). To comprehensively evaluate the performance of our method, we conducted experiments on the ten 2D CTC datasets. We submitted our segmentations of the test sets to the CTC evaluators to compare with the state-of-the-art methods and we received the evaluation results from them. Table 2 reports the OP_{CSB}, SEG and DET results and the CTC ranks of our method in the latest round (May 2021). As can be seen from Table 2 and the leaderboard on the CTC website, our method ranks among the top-3 for eight datasets and among the top-10 for the remaining two according to OP_{CSB}. It is worth noting that so far, no method has achieved such consistent top performance on all ten datasets, which demonstrates the effectiveness of the proposed method. Our method achieves better results for the SEG metric than for DET. This is not surprising, as our method was optimized specifically for accurate segmentation of cells and their boundaries, not for detection. Our method performs least optimal on the BF-C2DL-MuSC dataset due to systematic over-segmentation. Representative examples of our final segmentation results for the ten 2D CTC datasets are shown in Figure 5. We note that our method achieved remarkable results especially on the comparatively more challenging datasets, such as PhC-C2DL-PSC, Fluo-C2DL-MSC and Fluo-C2DL-Huh7, showing complex cell shapes and many closely adjacent cells.

4 Discussion

In this article, we proposed a NAS-based solution for time-lapse microscopy cell segmentation. Different from the current NAS methods, we jointly searched non-repeatable micro structures to construct the macro network which augments the much-studied repeatable one. This allows different layers of the macro network to have their own focuses and thus enable the network to better explore and fuse multiscale features. Also, we defined a specific search space for the time-lapse microscopy cell segmentation task, including the incorporation of CLSTM for extracting spatiotemporal information, the atrous convolution for better segmentation, etc.

Our NAS is inclined to select the atrous convolution operation (including atrous, down atrous and up atrous), which accounts for the largest proportion, to form the architectures. This operation rarely appeared in previous NAS works or did not appear so frequently in their searched networks. This can be explained from the following two aspects: (i) the atrous convolution was intentionally designed for tackling image segmentation issues by enlarging the receptive field to enable each convolution capture information at different scales while many of the current NAS works focus on image classification and natural language processing; (ii) despite the achievements of NAS applied in image segmentation, the operations and architectures required for different tasks are different. This supports our claim that defining a specific search space suitable for the time-lapse sequence is important. We also conclude that our NAS tended to select the depthwise-separable convolution for all the three types of operations (down, up, normal). Squeeze-and-excitation is often selected as part of the down and up operation, but hardly as the normal operation. This makes sense, as it can strengthen the features of important channels and weaken the features of unimportant channels, which is meaningful for the down and up sampling process. Also, the NAS hardly selected the max-pooling, down convolution and none operations, which is consistent with current experience in manual network design.

Comprehensive evaluations on the benchmark datasets from the CTC demonstrate the competitiveness of the proposed method compared to the state of the art. The results show that the method can achieve more consistent top performance across all ten datasets than the other methods. Altogether our findings show the great potential of NAS to yield improved neural networks for a wide range of cell segmentation problems. Although we have focused on 2D segmentation, the proposed method may be extended to 3D segmentation using slice-by-slice processing, and we also aim to develop a fully 3D implementation in the future.

Financial Support: none declared.

Conflict of Interest: none declared.

Acknowledgements

This research was undertaken with the assistance of resources and services provided by the National Computational Infrastructure (NCI) which is supported by the Australian Government.

References

Al-Kofahi,Y. *et al.* (2018) A deep learning-based algorithm for 2D cell segmentation in microscopy images. *BMC Bioinformatics*, **19**, 365.

Araújo,F.H. *et al.* (2019) Deep learning for cell image segmentation and ranking. *Comput. Med. Imaging Graph.*, **72**, 13–21.

Arbelle,A. and Raviv,T.R. (2019) Microscopy cell segmentation via convolutional LSTM networks. In: *16th International Symposium on Biomedical Imaging (ISBI 2019)*, pp. 1008–1012, Venice, Italy.

Brock,A. *et al.* (2018) SMASH: one-shot model architecture search through hypernetworks. In: International Conference on Learning Representations (ICLR), Vancouver, Canada.

Cai,H. *et al.* (2018a) Efficient architecture search by network transformation. In: *AAAI Conference on Artificial Intelligence*, Vol. 32, Louisiana, USA.

Cai,H. *et al.* (2018b) ProxylessNAS: direct neural architecture search on target task and hardware. *In: International Conference on Learning Representations,* Vancouver, Canada.

Chen,L.-C. *et al.* (2018) Deeplab: semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFS. *IEEE Trans. Pattern Anal. Mach. Intell.*, **40**, 834–848.

Chollet,F. (2017) Xception: deep learning with depthwise separable convolutions. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1251–1258, Honolulu, Hawaii.

Dimopoulos,S. *et al.* (2014) Accurate cell segmentation in microscopy images using membrane patterns. *Bioinformatics*, **30**, 2644–2651.

Dong,N. *et al.* (2019) Neural architecture search for adversarial medical image segmentation. In: *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 828–836, Shenzhen, China.

Drozdzal,M. *et al.* (2016) The importance of skip connections in biomedical image segmentation. In: *Deep Learning and Data Labeling for Medical Applications*. Springer, Cham, Athens, Greece, pp. 179–187.

Elsken,T. *et al.* (2019) Neural architecture search: a survey. *J. Mach. Learn. Res.*, **20**, 1–21.

Hollandi,R. *et al.* (2020) nucleAIzer: a parameter-free deep learning framework for nucleus segmentation using image style transfer. Cell Syst., 10, 453-458.

Hu,J. *et al.* (2018) Squeeze-and-excitation networks. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7132–7141, Utah, USA.

Huang,C. *et al.* (2020) Segmentation of cell images based on improved deep learning approach. *IEEE Access*, **8**, 110189–110202.

Hutter,F. *et al.* (2019) *Automated Machine Learning: Methods, Systems, Challenges*. Springer Nature, Berlin.

Isensee,F. *et al.* (2021) nnU-Net: a self-configuring method for deep learning-based biomedical image segmentation. *Nat. Methods*, **18**, 203–211.

Jalali,Y. *et al.* (2021) ResBCDU-Net: a deep learning framework for lung CT image segmentation. *Sensors*, **21**, 268.

Kim,S. *et al.* (2019) Scalable neural architecture search for 3D medical image segmentation. In: *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 220–228, Shenzhen, China.

Kingma,D.P. and Ba,J. (2015) Adam: a method for stochastic optimization. *In: International Conference for Learning Representations,* San Diego, USA.

Kong,J. *et al.* (2015) Automated cell segmentation with 3D fluorescence microscopy images. In: *12th International Symposium on Biomedical Imaging (ISBI)*, pp. 1212–1215, NY, USA.

Liu,C. *et al.* (2019) Auto-deeplab: hierarchical neural architecture search for semantic image segmentation. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 82–92, California, USA.

Liu,H. *et al.* (2018) Darts: differentiable architecture search. *In: International Conference on Learning Representations,* Vancouver, Canada.

Long,F. (2020) Microscopy cell nuclei segmentation with enhanced U-Net. *BMC Bioinformatics*, **21**, 8–12.

Lu,Z. *et al.* (2019) Nsga-net: neural architecture search using multi-objective genetic algorithm. In: *ACM Genetic and Evolutionary Computation Conference*, pp. 419–427, NY, USA.

Meijering,E. (2012) Cell segmentation: 50 years down the road. *IEEE Signal Process. Mag.*, **29**, 140–145.

Mortazi,A. and Bagci,U. (2018) Automatically designing CNN architectures for medical image segmentation. In: *International Workshop on Machine Learning in Medical Imaging*, pp. 98–106, Granada, Spain.

Qiang,N. *et al.* (2019) Neural architecture search for optimizing deep belief network models of fMRI data. In: *International Workshop on Multiscale Multimodal Medical Imaging*, pp. 26–34, Shenzhen, China.

Ronneberger,O. *et al.* (2015) U-Net: convolutional networks for biomedical image segmentation. In: *International Conference on Medical Image Computing and Computer-assisted Intervention*, pp. 234–241, Munich, Germany.

Ulman,V. *et al.* (2017) An objective comparison of cell-tracking algorithms. *Nat. Methods*, **14**, 1141–1152.

Wang,F. and Biswal,B. (2020) Neural architecture search for gliomas segmentation on multimodal magnetic resonance imaging. *arXiv, 2005.06338*.

Weng,Y. *et al.* (2019) NAS-Unet: neural architecture search for medical image segmentation. *IEEE Access*, **7**, 44247–44257.

Wu,Y. and He,K. (2018) Group normalization. In: *European Conference on Computer Vision (ECCV)*, pp. 3–19, Munich, Germany.

Xie,L. and Yuille,A. (2017) Genetic CNN. In: *IEEE International Conference on Computer Vision (ICCV)*, pp. 1379–1388, Venice, Italy.

Xing,F. and Yang,L. (2016) Robust nucleus/cell detection and segmentation in digital pathology and microscopy images: a comprehensive review. *IEEE Rev. Biomed. Eng.*, **9**, 234–263.

Yan,X. *et al.* (2020) MS-NAS: multi-scale neural architecture search for medical image segmentation. In: *International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, pp. 388–397, Lima, Peru.

Yang,D. *et al.* (2019) Searching learning strategy with reinforcement learning for 3D medical image segmentation. In: *International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, pp. 3–11, Shenzhen, China.

Zela,A. *et al.* (2019) Understanding and robustifying differentiable architecture search. *In:* International Conference on Learning Representations, Vol. 3, pp. 7, Addis Ababa, Ethiopia.

Zhang,X. *et al.* (2018) Shufflenet: an extremely efficient convolutional neural network for mobile devices. In: *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6848–6856, Utah, USA.

Zhu,Y. and Meijering,E. (2020) Neural architecture search for microscopy cell segmentation. In: *International Workshop on Machine Learning in Medical Imaging*, pp. 542–551, Lima, Peru.

Zhu,Z. *et al.* (2019) V-NAS: neural architecture search for volumetric medical image segmentation. In: *International Conference on 3D Vision (3DV)*, pp. 240–248, Quebec, Canada.

Zoph,B. *et al.* (2018) Learning transferable architectures for scalable image recognition. In: *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 8697–8710, Utah, USA.