Multi-stage Multi-recursive-input Fully Convolutional Networks for Neuronal Boundary Detection

Wei Shen[1,2], Bin Wang[1], Yuan Jiang[1]*, Yan Wang[2], Alan Yuille[2]
[1] Key Laboratory of Specialty Fiber Optics and Optical Access Networks, Shanghai University
[2] Department of Computer Science, Johns Hopkins University
wei.shen@t.shu.edu.cn,{wangbin418,jy9387}@outlook.com,{wyanny.9,alan.l.yuille}@gmail.com

Abstract

In the field of connectomics, neuroscientists seek to identify cortical connectivity comprehensively. Neuronal boundary detection from the Electron Microscopy (EM) images is often done to assist the automatic reconstruction of neuronal circuit. But the segmentation of EM images is a challenging problem, as it requires the detector to be able to detect both filament-like thin and blob-like thick membrane, while suppressing the ambiguous intracellular structure. In this paper, we propose multi-stage multi-recursive-input fully convolutional networks to address this problem. The multiple recursive inputs for one stage, i.e., the multiple side outputs with different receptive field sizes learned from the lower stage, provide multi-scale contextual boundary information for the consecutive learning. This design is biologically-plausible, as it likes a human visual system to compare different possible segmentation solutions to address the ambiguous boundary issue. Our multi-stage networks are trained end-to-end. It achieves promising results on two public available EM segmentation datasets, the mouse piriform cortex dataset and the ISBI 2012 EM dataset.

1. Introduction

A central theme of neuroscience is to understand how a brain's functions are related to its neuronal structure [23]. This is difficult because of the small size of neurons and the extremely high packing density of the neuropil, e.g., neurons densely packed axons and dendrites [13]. Recent advances in high-throughput serial section electron microscopy (EM) have made possible the imaging of large volumes of neuronal tissue at high resolution, allowing neuroscience experts to reconstruct neuronal circuits and study the interconnections of neurons [38]. However, analysis of a large number of EM images by expert annotators is laborious and even impractical [13], which drives the demand for efficient automated neuronal circuit reconstruction approaches.

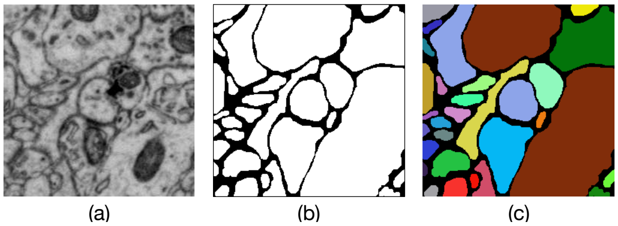

Figure 1. Neuronal structure segmentation: an EM image (a) and the ground truths for its neuronal boundary detection result (b) and segmentation result (c), respectively.

Serial section EM produces a stack of 2D images by cutting sections of brain tissue. Due to the anisotropic resolutions of in-plane and out-of-plane, most neuronal circuit reconstruction approaches follow the following pipeline: (1) neuronal boundary detection on each 2D image, (2) neuronal structure segmentation based on the 2D boundary map, and (3) linking the neuronal segments across 2D images into a 3D reconstruction result.

This paper focuses on neuronal boundary detection on 2D images from serial section EM, also called membrane detection, which is the essential first step of the reconstruction pipeline. The challenges of this problem, as can be seen from Fig. 1, mainly lie in the following aspects: (1) The variation of the membranal thickness is large, as thin as a filament to as thick as a blob. (2) The noise of EM acquisition makes the membrane contrast to be low, inducing some membranal boundaries are even invisible. (3) The presence of confounding structures, such as mitochondria and vesicles, also increases the difficulty in membrane detection.

Driven by the rapid process of deep neural networks, more and more deep learning based methods have been proposed for neuronal boundary detection on EM images, achieving considerable progress [8, 5, 11, 28, 21]. Howev-

*equal contribution with the second author er, most of them only focus on how to improve detection performance by using the deep learning strategies shown to be effective on general computer vision problems, like image classification [17, 37, 12], semantic segmentation [25] and boundary/symmetry detection [35, 42, 36]. For example, using fully convolutional networks enables holistic image training instead of patch-by-patch training [5]; deeply supervised learning hierarchical representations [5]; using residual structures rather than plain structures [11, 28]. Although these strategies indeed improve membrane detection results, they lack an interpretation for this problem, i.e., how can membranes be detected and intracellular structures be suppressed meanwhile, even the contrasts of intracellular structures to context are much higher than those of membranes?

In this paper, we propose multi-stage multi-recursive-input fully convolutional networks (M^2FCN) for neuronal boundary detection. The architecture of M^2FCN is shown in Fig. 2. The whole net consists of multiple stages, where each stage generates multiple side outputs by imposing supervision at different levels [20, 42] of the sub-net in this stage, and all these side outputs are concatenated with the original image to serve as the inputs for the next stage.

From a neurobiological prospective, this network architecture is biologically-plausible. First, as explained in [21], this recursive framework is in accord with the interplay process, named "countercurrent disambiguating process" [7], between the primary and higher visual cortical areas (V1 and V4, respectively) in monkeys' brains, found by examining monkeys' performances on contour detection tasks. The latter stages of our networks act as V4 to detect the overall "contour" of neuronal boundaries and feed top-down influence to the early stages, which acts as V1, to enhance the activation on neuronal boundaries while suppressing those on intracellular structures. Second, as pointed out in [13], the ambiguous neuronal boundaries make segmentation difficult, and this issue can only be resolved when explicitly comparing the different possible segmentation solutions, which is what the human visual system may compute when inspecting this situation. We can roughly think that of the multiple recursive inputs, computed at the different levels in the previous stage, provide different possible segmentation solutions for training of the next stage. We show that using multiple recursive inputs is very important in our experiments, as it leads to much better results than using a single recursive input.

From the deep learning view, our networks take several advantages of the latest deep learning strategies to facilitate the learning of a neuronal boundary detection model, including (1) holistic image training and prediction benefited from using the architecture of fully convolutional neural networks, (2) supervised multi-scale feature learning at each level of each stage and (3) learning multi-stages in an end-to-end fashion, different from previous recursive approaches [39, 21, 14, 9] which learn a series of classifiers stepwise. Using multiple recursive inputs rather than one is a major difference between our networks and [21]. Note that, the multiple recursive inputs for one stage are the multiple side outputs computed at the levels having different scales (receptive field sizes). In general, a side output with a small scale has better ability than one with larger scale for detecting a thin neuronal boundary between cluttered neurons. Conversely, a side output computed at a large scale can suppresses the false predictions on intracellular structures by using more image context. Therefore, these multiple recursive inputs provide richer information than one for the learning of next stage. We show that introducing these strategies in our networks can improve the performance of neuronal boundary detection.

We verify our networks on two public available EM segmentation datasets. One is the mouse piroform cortex EM dataset [21], a sizable and important EM dataset, contains 4 stacks of EM images of mouse piriform cortex, covering 460 images for training and 168 images for testing. We analyze alternative designs in our network architecture on this dataset and show it outperforms the state-of-the-arts [21]. The other is the dataset used for ISBI 2012 EM segmentation challenge [29], covering 30 images for training and 30 images for testing. Our networks can achieve a comparable result to the state-of-the art [28, 10] on this dataset.

In summary, our main contributions includes (1) end-to-end multi-stage networks architecture, in which each stage generates multiple side outputs by imposing supervision on different levels and feeds them into the next stage as the multiple recursive inputs. (2) using multiple recursive inputs rather than single recursive inputs can not only boost the performance of neuronal boundary detection, but also be biologically-plausible. (3) our networks achieve promising results on two public available EM segmentation datasets.

2. Related Work

Segmenting EM images of neural tissue is an important step to understand the circuit structure and the function of the brain [21, 28]. Early work of this topic needs to impose experts' knowledge. For example, users need to label intracellular regions to allow graph cut segmentation, and also correct segmentation errors afterwards [41]. To reduce the amount of human labor required [13], automatic neuron segmentation became an active research direction, which follows the pipeline that detects neuronal boundaries by machine learning algorithms [19, 15, 18, 31] and then applies post-processing algorithms, such as watershed [26, 40, 43], hierarchical clustering [27, 24] and graph cut [16] algorithms, to boundary maps to obtain neuron segments. But early methods, which are based on hand-crafted features, tend to fail when the membrane is ambiguous.

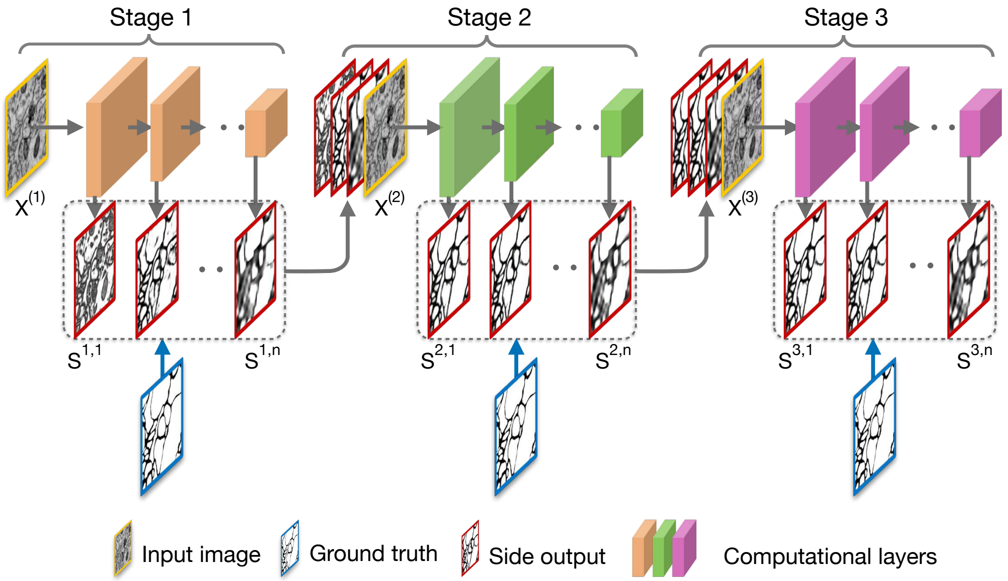

Figure 2. The architecture of the proposed M^{2}FCN. The multiple side outputs of one stage are concatenated with the original image along the image channel dimension, and are fed into the next stage. It makes the side outputs learned at the next stage approach ground truth.

With the rapid development of deep networks, segmentation and classification on medical images have undergone a vast revolution, from analyzing specific features of neuronal boundaries to designing fully automatic algorithm without experts' knowledge. Recent deep learning based neuron segmentation methods are well suited to deal with ambiguous neuronal boundaries in EM images. One of the earliest works made by Ciresan *et al.* [8] used a succession of max-pooling convolutional networks as a pixel classifier, which estimated the probability of a pixel being a membrane. This method won the ISBI 2012 EM segmentation challenge [1]. Ronneberger *et al.* [29] presented a U-net structure with contracting paths, which captures multi-contextual information. Compared with the work in [8], the U-net replaces pooling operations by upsampling operators, which propagates context information from multiple feature channels to higher resolution layers. Fully convolutional networks (FCNs) [25] led to a breakthrough on semantic segmentation. With deep supervision and side outputs on FCN, a holistically-nested edge detector (HED) [42] was then proposed to solve the ambiguity in edge and object boundary detection. Due to the great success of these methods, Chen *et al.* [5] presented a deeply supervised contextual network to fuse the multi-level side-outputs to segment the membrane. Deep residual network (ResNet) [12] addresses the degradation (of training accuracy) problem by optimizing the residual mapping, which ranked the 1st on ILSVRC 2015 image classification challenge [30]. This inspired fully residual convolutional neural networks with nested short and long skip connections for membrane segmentation, proposed by Quan *et al.* [28] and Fakhry *et al.* [11]. Most of the recent deep learning based methods were motivated by the novel network architectures proposed in the computer vision community. Although these methods achieve improvements on membrane detection, they lack interpretation and comprehensive analysis. Our proposed multi-stage multi-recursive-input fully convolutional networks (M^{2}FCN) not only considers the advantages of the latest deep network architectures but also is biologically plausible.

The recursive training framework has been applied to many computer vision tasks, such as image labeling [39], instance segmentation [22], human pose estimation [34], and face alignment [9, 3]. However, they trained the recursive framework stepwise and only used one single recursive input. There are two image segmentation methods [32, 33] also fed multi-recursive inputs into next stage in the recursive training framework. But, their strategies to generate the multi-recursive inputs are different from ours. In the first one [32], the multi-recursive inputs for one stage were obtained by applying a series of Gaussian filters to the single output of the previous stage, but ours are the multiple outputs supervised at different levels in a deep network. The second one [33] downsampled an original input image into multiple input images with different resolutions and obtained the multi-recursive inputs from these multiple input images, but ours are computed from the same input image by using the hierarchy of a deep net. In addition, the multiple stages in their methods were trained in a stepwise manner. On the contrary, we embed the recursive learning in a deep network and first learn it in an end-to-end fashion.

Our work is related to the recursively trained network proposed in [21], which trains a Very Deep 2D (VD2D) network first, then a Very Deep 2D-3D (VD2D3D) network initialized with learned 2D representations from VD2D network is trained to generate the boundary map. There are two important differences between the networks in [21] and our method. (1). We use multiple recursive inputs with different receptive field sizes to incorporate multi-level contextual boundary information learned from the previous stage, while VD2D3D only uses the single output of VD2D as its recursive input. (2). We train our network in an end-to-end fashion to co-enhance the learning ability (e.g., detect membranes while suppress intracellular structure) of all stages, while [21] sequentially learns deep networks. Benefited from end-to-end training and multiple recursive inputs, our networks can achieve better performance than VD2D3D while only using 2D EM images.

3. Multi-stage Multi-recursive-input Fully Convolutional Networks

3.1. Overview

Fig. 2 illustrates the proposed network architecture. Our networks consist of multiple sequential stages, where each stage is a sub-net built based on fully convolutional networks with multiple side outputs [42]. Each sub-net consists of multiple levels, each of which is composed of several combinations of one convolutional layers followed by one ReLU layer and a final pooling layer. Each side output layer is connected to the last convolutional layer of each level, which is composed of a 1×1 convolutional layer and a deconvolutional layer to ensure the resolution of each side output is the same as the input original image. A sigmoid layer is applied to each side output layer to generate a neuronal boundary map having values belonging to $[0, 1]$.

Note that, the multiple side outputs of one stage are concatenated with the original image along the image channel dimension and fed into the next stage. By feeding multiple side outputs to the next stage, the side outputs of next stage can obtain the information from all scales, so that even low level side outputs in the next stage can capture large objects. While in a single stage network, each side output only corresponds to a certain scale, e.g., low level side outputs cannot suppress large intracellular structures, and high level side outputs cannot locate thin membranes accurately. This is like how a human visual system may compare different possible segmentation solutions and select the right scale from them [13]. The multiple stages in our networks can be trained in an end-to-end fashion, which enable the side outputs in a previous stage to receive feedback from those of the next stage, like V4 in the human visual system gives the top-down influence to V1.

3.1.1 Sub-net Architecture

We adopt the well-known HED network [42] as our default sub-net, which is converted from VGG-16 net [37], having 5 levels, with strides 1, 2, 4, 8 and 16, respectively, and receptive field sizes 5, 14, 40, 92, 196, respectively. Each side output layer is connected to the last convolutional layer of each level, i.e., conv1_2, conv2_2, conv3_3, conv4_3, conv5_3, respectively.

3.2. M^{2}FCN for Neuronal Boundary Detection

Now we formulate our approach for neuronal boundary detection as a per-pixel classification problem. Given a raw input EM image $X = (x_j, j = 1, \ldots, |X|)$, where index j is over the image spatial dimensions of image X, the goal is to predict the neuronal boundary map $\hat{Y} = (\hat{y}_j, j = 1, \ldots, |X|)$, where $\hat{y}_j \in \{0, 1\}$ denotes the predicted label for each pixel x_j, i.e., if x_j is predicted as a boundary pixel, $\hat{y}_j = 0$; otherwise, $\hat{y}_j = 1$. We learn an M^{2}FCN, which consists of M stages and each stage has N levels, to address this problem. Next, we introduce the training and testing phases of our approach respectively.

3.2.1 Training Phase

Since our networks use holistic image training, we consider each training image independently. Suppose that we are given a training batch with one 2D EM image X and its corresponding ground truth neuronal boundary map Y, our goal is to supervise the multiple side outputs at different level to approach the ground truth map Y. Let $S^{m,n} = (s_j^{m,n}, j = 1, \ldots, |X|)$ be the side output at the n-th level of the m-th stage. Since the side outputs of one stage will be fed to the next stage as the inputs, we express the input to the m-stage by

$$X^{(m)} = X \oplus S^{m-1,1} \oplus \ldots \oplus S^{m-1,n} \tag{1}$$

where $\oplus$ is the the concatenation operation along the image channel dimension and we define $X^{(1)} = X$. Let $\mathbf{W}^m$ be the network parameters for the sub-net in the m-th stage and $\mathbf{w}^{m,n}$ be the parameters for the n-th side output in the m-th stage, we define a cross-entropy loss function for this side output by

$$\begin{aligned} &\ell^{m,n}(\mathbf{W}^m, \mathbf{w}^{m,n}; X^{(m)}, Y) = \\ &-\beta \textstyle\sum_{j \in |B|} \log(1 - \sigma(s_j^{m,n})) \\ &-(1-\beta) \textstyle\sum_{j \in |\bar{B}|} \log(\sigma(s_j^{m,n})). \end{aligned} \tag{2}$$

This loss function (Eqn.2) is computed over all pixels in the training image X, where $|B|$ and $|\bar{B}|$ denote the boundary and non-boundary ground truth label sets respectively, $\sigma(\cdot)$ is the sigmoid function and $\beta = |\bar{B}|/|B|$ is a positive/negative class-balancing weight [42] to eliminate the bias between boundary and non-boundary ground truths in training (in a typical 2D EM image, most of the ground truth is non-boundary). To supervise all the side outputs in our network, we define a loss function by

$$\mathcal{L}_s(\mathbf{W}, \mathbf{w}; X, Y) = \sum_{m=1}^{M} \sum_{n=1}^{N} \alpha_{m,n} \ell^{m,n}(\mathbf{W}^m, \mathbf{w}^{m,n}; X^{(m)}, Y), \tag{3}$$

where $\mathbf{W} = (\mathbf{W}^m, m = 1, \ldots, M)$, $\mathbf{w} = (\mathbf{w}^{m,n}, m = 1, \ldots, M, n = 1 \ldots, N)$ and $\alpha_{m,n}$ is a loss weight for each side output.

To obtain a fused output $S^{f,m}$ for m-stage , we use a 1×1 convolutional layer to fuse its side outputs:

$$S^{f,m} = \sum_{n=1}^{N} h_{m,n} S^{m,n}, \tag{4}$$

where $\mathbf{h} = (h_{m,n}, m = 1, \ldots, M, n = 1 \ldots, N)$ is the fusion weight. Similar to Eqn. 2, a class-balanced cross-entropy loss function is defined for the fused output of m-stage:

$$\ell^{f,m}(\mathbf{W}^m, \mathbf{w}^m; X^{(m)}, Y) = \\ -\beta \sum_{j \in |B|} \log(1 - \sigma(s_j^{f,m})) - (1-\beta) \sum_{j \in |\bar{B}|} \log(\sigma(s_j^{f,m})). \tag{5}$$

where $\mathbf{w}^m = (\mathbf{w}^{m,n}, n = 1 \ldots, N)$. The loss function for all the fused outputs is

$$\mathcal{L}_f(\mathbf{W}, \mathbf{w}, \mathbf{h}; X, Y) = \\ \sum_{m=1}^{M} \alpha_{f,m} \ell^{f,m}(\mathbf{W}^m, \mathbf{w}^m; X^{(m)}, Y), \tag{6}$$

where $\alpha_{f,m}$ is a loss weight for each stage.

All these parameters, $\mathbf{W}, \mathbf{w}, \mathbf{h}$, are optimized simultaneously by standard back-propagation:

$$(\mathbf{W}, \mathbf{w}, \mathbf{h})^* = \arg\min(\mathcal{L}_s(\mathbf{W}, \mathbf{w}; X, Y) \\ +\mathcal{L}_f(\mathbf{W}, \mathbf{w}, \mathbf{h}; X, Y)). \tag{7}$$

3.2.2 Testing Phase

In the testing stage, given an image X, by considering the m-th sub-net as a function F_m, we sequentially obtain the side outputs of the m-th sub-net by

$$(S^{m,n}, n = 1 \ldots, N) = F_m(X^{(m)}, \\ (\mathbf{W}^1)^*, \ldots, (\mathbf{W}^m)^*, (\mathbf{w}^{m,1})^*, \ldots, (\mathbf{w}^{m,N})^*). \tag{8}$$

The fusion output of m-stage is obtained by

$$S^{f,m} = \sum_{n=1}^{N} h^*_{m,n} S^{m,n}. \tag{9}$$

We use the fused output of the last stage $S^{f,M}$ as the final output of our network, then the unified boundary probability map is given by $\hat{Y} = \sigma(S^{f,M})$.

3.2.3 Initialization for Multi-stage Training

The proposed network is very deep (our 3-stage network with 5 levels has totally 48 layers). It is known that, training such a deep network from scratch is not easy. Here, we adopt a simple strategy to initialize the multi-stage network. First, we train a single stage network. Then, this network is used to initialize the first stage of our network, while the rest of the network is randomly initialized. The initialization for the first stage provides high-confidence recursive inputs to the consecutive stage, which facilitates the training procedure.

4. Experimental Results

In this section, we discuss our designs for network architectures and training strategies, and compare our performance with other competitors.

4.1. Experiment Setting

The hyper parameters of our networks include: the mini-batch size (1), the loss weight for each side-output (1), the momentum (0.9), and the weight decay (2×10^{-4}). We set the base learning rate to 1e-8 and pre-train a single stage network by 20,000 iterations. Then we use this pre-trained single stage network to initialize the first stage of our multi-stage network, and reduce the base learning rate to 1e-9 and train it by 10,000 iterations.

4.2. Mouse Piriform Cortex Dataset

The images of mouse piriform cortex dataset [21] were collected from the piriform cortex of an adult mouse, which contains 4 stacks of EM images. We use the same training-testing split in [21], i.e., stack2, stack3 and stack4 are for training and stack1 is for testing.

4.2.1 Evaluation Metric

To evaluate membrane segmentation performances, we follow the protocol used in [21], where the segmented membrane is measured by the Rand F-score:

$$V^{Rand}_{Fscore} = \frac{2V^{Rand}_{merge} V^{Rand}_{split}}{V^{Rand}_{merge} + V^{Rand}_{split}}, \tag{10}$$

where V^{Rand}_{merge} and V^{Rand}_{split} are Rand merge score and Rand split score respectively, and defined by:

$$V^{Rand}_{merge} = \frac{\sum_{ij} n_{ij}^2}{\sum_i (\sum_j n_{ij})^2}, \quad V^{Rand}_{split} = \frac{\sum_{ij} n_{ij}^2}{\sum_j (\sum_i n_{ij})^2}, \tag{11}$$

where n_{ij} denote the number of voxels in the i-th segment of the proposal segmentation and j-th segment of the ground truth segmentation. V^{Rand}_{merge} and V^{Rand}_{split} are close to 1 when there are fewer merge and split errors, respectively. To calculate the Rand score, we obtain the neuronal segmentation based on the boundary map by applying the same modified graph-based watershed algorithm [43] as in [21]. We use the default setting in [21] and report our best Rand F-score V^{Rand}_{Fscore}.

4.2.2 Data Augmentation

Data augmentation is a standard way to generate sufficient training data for learning a "good" deep network. We rotate the images to 4 different angles (0°, 90°, 180°, 270°) and flip them with different axis (up-down, left-right, no flip), then resize images to 3 different scales (0.8, 1.0, 1.2), totally leading to an augmentation factor of 36.

4.2.3 Alternative Design Discussion

We use the pre-trained single stage network as the baseline and discuss some possible alternative designs for network architectures and training strategies. The results of these alternative designs are summarized in Table. 1. To simplify description, we denote each alternative design by "AD" plus an index.

The role of multiple stages. Since our networks consist of multiple stages, it's necessary to see whether the performance can be improved by adding stage by stage. Due to the limitation of GPU memory, the deepest network we can train is a 3-stage network with sub-nets of 5 levels (Sec. 3.1.1). We compare the performance between a 1-stage network (AD_I), a 2-stage network (AD_VI) and a 3-stage network (AD_VII). Note that, the AD_I has the same architecture as the pre-trained single stage network. As shown in Table. 1, the performance of AD_I is almost the same as that of the pre-trained single stage network, which indicates that training a single stage network by more iterations cannot improve the performance considerably. The AD_VI and the AD_VII achieve 1.39% and 1.86% performance improvements compared with the baseline, respectively, which shows that our multi-stage training is effective for neuronal boundary detection. A qualitative comparison between these three networks, i.e, 1-stage (AD_I), 2-stage (AD_VI) and 3-stage (AD_VII) networks, is given in Fig. 3. Observed that, the false detections on intracellular structures such as mitochondria and vesicles can be reduced (indicated by red arrows) by training a network with more stages.

The role of multiple recursive inputs. As we stated in the Sec. 1, using multiple recursive inputs for the multiple stage training is crucial for our framework. To evaluate this, we test two alternative network architectures, which only uses the side output of the 5-th level (AD_III) and the 4-th level (AD_IV) in each stage as the recursive input for the next stage respectively. As shown in Table. 1, only using single recursive input results in a significant performance drop, 4.56% decrease for AD_III and 2.1% decrease for AD_IV. We illustrate the side outputs of the second stages of AD_III and AD_VI in Fig. 4, where we see that the side outputs of the former one only response to large scale objects, while even the side outputs of the latter one can capture objects of different scales, thanks to the multiple recursive inputs.

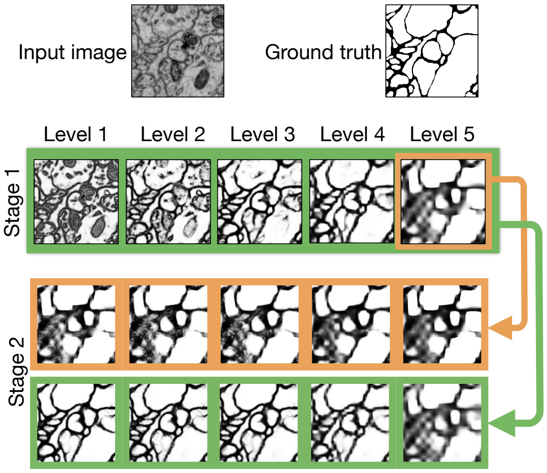

Figure 4. The side outputs of the second stages learned by single recursive input (AD_III) and multiple recursive inputs (AD_VI).

Stepwise or end-to-end training. One difference between our multi-stage training framework to others [39, 21, 14, 9] is we train these multiple stages in an end-to-end fashion, not stepwise. To verify which way is better, we also train a 2-stage network stepwise, by simply fixing the parameters of the first stage in this 2-stage network (AD_V). As can be seen from Table. 1, training the 2-stage network stepwise leads to a considerable performance decrease ($0.9819 \rightarrow 0.9762$). During end-to-end training, previous stages are influenced by next stages. We visualize the fused outputs (after watershed) of the first stages of (AD_V) and (AD_VI), respectively, in Fig. 5, where we see the latter one leads to better segmentation results (indicated by red arrows). Therefore, we conclude that training in an end-to-end way is better.

The range of the receptive field sizes in each sub-net. The receptive field sizes of the 5 levels in each sub-net range from 5 to 196. Such a wide range of receptive field sizes ensure the side outputs to be able to capture small neuronal boundaries while suppress relative big intracellular structures. To show this wide range of receptive field sizes is important for neuronal boundary detection, we use an alternative network architecture for each sub-net, which is obtained by removing the last level from the default sub-net, i.e., a 2-stage network with 4-level sub-net (AD_II). As we expected, as shown in Table. 1, removing the last level in each sub-net leads to performance decrease.

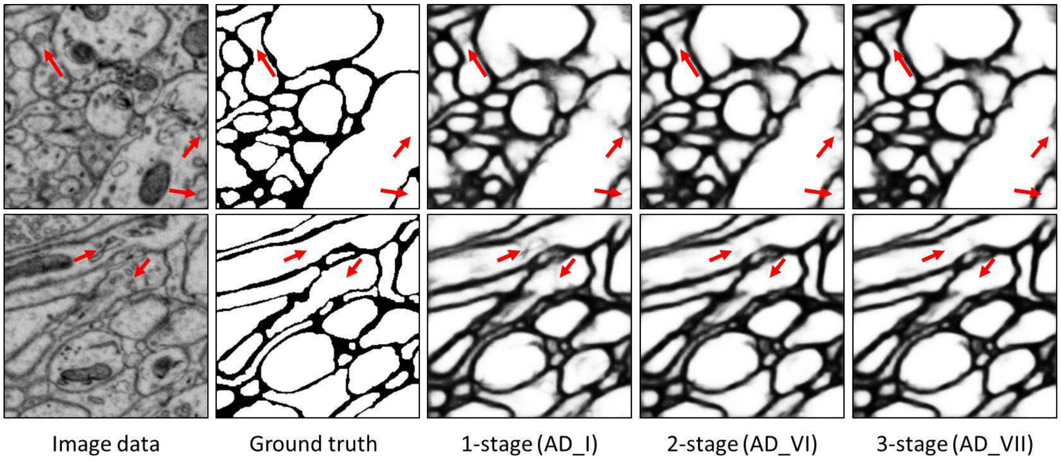

Figure 3. Qualitative comparison between 1-stage (AD_I), 2-stage (AD_VI) and 3-stage (AD_VII) networks. Red arrows indicate suppressed false detections by training more stages.

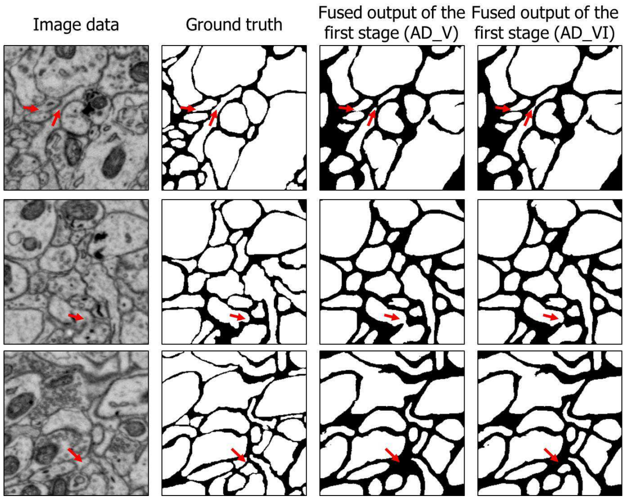
Image data
Ground truth
Fused output of the first stage (AD_V)
Fused output of the first stage (AD_VI)

Figure 5. The comparison between the fused outputs (after watershed [43]) of the first stages of a 2-stage network trained stepwise (AD_V) and end-to-end (AD_VI). The latter one leads to better segmenting results (indicated by red arrows).

Table 1. Performance of alternative designs for network architectures and training strategies.

Alternative Design	V^{Rand}_{Fscore}
pre-trained single stage, 5-level (baseline)	0.9680
AD_I: 1-stage, 5-level	0.9688
AD_II: 2-stage, 4-level, end-to-end, multi-recursive-input	0.9739
AD III: 2-stage, 5-level, end-to-end, single-recursive-input (level 5)	0.9410
AD_IV: 2-stage, 5-level, end-to-end, single-recursive-input (level 4)	0.9656
AD_V: 2-stage, 5-level, stepwise, multi-recursive-input	0.9762
AD_VI: 2-stage, 5-level, end-to-end, multi-recursive-input	0.9819
AD_VII: 3-stage, 5-level, end-to-end, multi-recursive-input	**0.9866**

4.2.4 Performance Comparison

Now we compare our networks (3-stage with 5 levels) with other competitors on the mouse piriform cortex dataset. The quantitative results are summarized in Table 2 and the precision (rand merge)-recall (rand split) curves are illustrated in Fig. 6. As can be seen, our method can maintain a high precision even when it achieves a high recall, thanks to the multi-stage training which suppresses false detections of boundaries on intracellular structures while enhancing neuronal boundaries. The current state-of-the-art method on the mouse piriform cortex dataset is VD2D3D [21], which is also a recursive training framework. But it trains two stages stepwise, i.e., train the first one, a 2D convolutional network, then uses its output as the recursive input for the second one, a 3D convolutional networks. The experimental results show that with the end-to-end multi-stage training and multi-recursive-inputs, our 2D 2-stage network can achieve better performance than a 2D-3D network. Note that, as VD2D3D already obtained a high Rand F-score, our method achieves around 1.5% improvement on it, which is meaningful. Such a low error obtained on a large EM image dataset is important for neuron reconstruction. Some neuron segmentation results obtained by applying the watershed algorithm [43] to our boundary maps are shown in Fig. 7.

4.3. ISBI 2012 EM segmentation dataset

Most of current neuronal boundary detection methods are evaluated on the public dataset of ISBI 2012 EM segmentation challenge [29]. The training data of this dataset is a set of 30 consecutive images (512 × 512 pixels) from a

Table 2. Neuronal boundary detection performance comparison between different methods. The values of V^{Rand}_{merge} and V^{Rand}_{split} correspond to the best Rand F-score V^{Rand}_{Fscore}.

Method	V^{Rand}_{merge}	V^{Rand}_{split}	V^{Rand}_{Fscore}
N4 [8]	0.9619	0.9010	0.9304
VD2D [21]	0.9771	0.9174	0.9463
VD2D3D [21]	0.9891	0.9555	0.9720
M^{2}FCN (1 stage)	0.9576	0.9802	0.9688
M^{2}FCN (2 stage)	0.9759	0.9880	0.9819
M^{2}FCN (3 stage)	0.9917	0.9815	**0.9866**

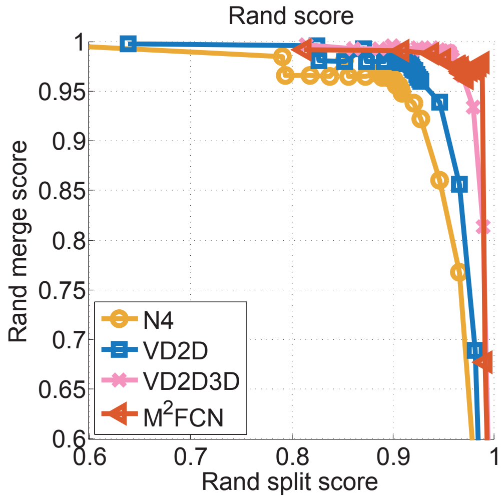

Figure 6. Evaluation of neuronal boundary detection methods by precision (rand merge)-recall (rand split) curves.

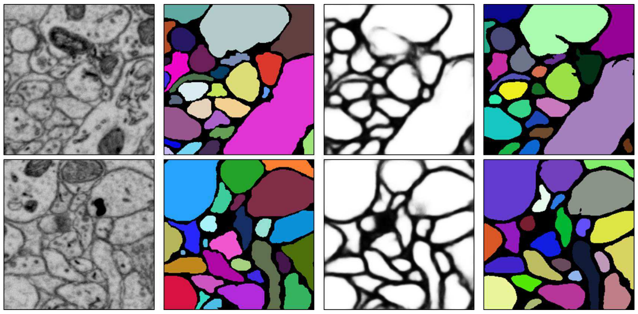

Figure 7. Neuron segmentation examples. From left to right: the original image, the ground truth segmentation, our boundary map and the segmentations obtain by applying the watershed algorithm [43] to our boundary maps.

serial section Transmission Electron Microscopy (ssTEM) dataset of the Drosophila first instar larva ventral nerve cord [4]. The testing data of this dataset also contains 30 consecutive EM images of the same resolution. The ground truth boundary maps of the training images are made publicly available to enable participants to develop their algorithm, while the ground truth boundary maps of the test images are kept by the organizers. Although this challenge is over, it is still open for submissions. The performance of the new submissions will be reported on the leader board of this challenge. There are over 70 results listed on the leader board, but not all of them have published papers. We summarized some leading quantitative results reported in published papers in Table 3. Note that, many state-of-the-art methods apply post-processing or average multiple trained models to boost the performance, such as PolyMtl [10], FusionNet [28] and CUMedVision [5]. Our method, a two-stage network using a single trained model without post-processing, can achieve 0.9780 Rand F-score, which is comparable with the state-of-the-art methods and better than CUMedVision [5], a one-stage HED. But, CUMedVision used post-processing and averaged 6 trained models to improve the result. This comparison shows the effectiveness of our multi-stage learning. IAL IC [2] is a post-processing method, which can be applied to our result to improve our performance. Besides, as both FusionNet and PolyMtl are built on ResNet [12], we can also replace the sub-net in our model by such a powerful network to gain improvement.

Table 3. Comparison to published entries on the ISBI 2012 EM dataset [29]. For full ranking of all submitted methods, please refer to the challenge website: `http://brainiac2.mit.edu/isbi_challenge/leaders-board-new`.

Method	V^{Rand}_{Fscore}
PolyMtl [10]	0.9806
M^{2}FCN (ours)	0.9780
FusionNet [28]	0.9780
IAL IC [2]	0.9773
CUMedVision [5]	0.9768
FCN+LSTM [6]	0.9754
Unet [29]	0.9727

5. Conclusion

We present multi-stage multi-recursive-input fully convolutional networks for neuronal boundary detection. In the proposed architecture, the multiple side outputs learned at different scales in one stage, are fed into the next stage. This provides the ability to detect neuronal boundaries while suppressing false predictions on intracellular structures. Extensive analysis on two public EM segmentation datasets, the mouse piriform cortex dataset and the ISBI 2012 EM dataset, verifies the advantages of our network architecture.

Acknowledgement. This work was supported in part by the National Natural Science Foundation of China No. 61672336 and No. 61303095, in part by “Chen Guang” project supported by Shanghai Municipal Education Commission and Shanghai Education Development Foundation No. 15CG43, and in part by NSF CCF-1231216.

References

[1] I. Arganda-Carreras, S. C. Turaga, D. R. Berger, D. Cireşan, A. Giusti, L. M. Gambardella, J. Schmidhuber, D. Laptev, S. Dwivedi, J. M. Buhmann, T. Liu, M. Seyedhosseini, T. Tasdizen, L. Kamentsky, R. Burget, V. Uher, X. Tan, C. Sun, T. D. Pham, E. Bas, M. G. Uzunbas, A. Cardona, J. Schindelin, and H. S. Seung. Crowdsourcing the creation of image segmentation algorithms for connectomics. *Frontiers in Neuroanatomy*, 9:142, 2015.

[2] T. Beier, B. Andres, U. Köthe, and F. A. Hamprecht. An efficient fusion move algorithm for the minimum cost lifted multicut problem. In *ECCV*, pages 715–730, 2016.

[3] X. Cao, Y. Wei, F. Wen, and J. Sun. Face alignment by explicit shape regression. In *Proc. CVPR*, pages 2887–2894, 2012.

[4] A. Cardona, S. Saalfeld, S. Preibisch, B. Schmid, A. Cheng, J. Pulokas, P. Tomancak, and V. Hartenstein. An integrated micro- and macroarchitectural analysis of the drosophila brain by computer-assisted serial section electron microscopy. *PLoS Biol*, 8(10):e1000502, 2010.

[5] H. Chen, X. Qi, J. Cheng, and P. Heng. Deep contextual networks for neuronal structure segmentation. In *Proc. AAAI*, pages 1167–1173, 2016.

[6] J. Chen, L. Yang, Y. Zhang, M. S. Alber, and D. Z. Chen. Combining fully convolutional and recurrent neural networks for 3d biomedical image segmentation. In *NIPS*, pages 3036–3044, 2016.

[7] M. Chen, Y. Yan, X. Gong, C. D. Gilbert, H. Liang, and W. Li. Incremental integration of global contours through interplay between visual cortical areas. *Neuron*, 82(3):682–694, 2014.

[8] D. C. Ciresan, A. Giusti, L. M. Gambardella, and J. Schmidhuber. Deep neural networks segment neuronal membranes in electron microscopy images. In *Proc. NIPS*, pages 2852–2860, 2012.

[9] P. Dollár, P. Welinder, and P. Perona. Cascaded pose regression. In *Proc. CVPR*, pages 1078–1085, 2010.

[10] M. Drozdzal, G. Chartrand, E. Vorontsov, L. Di-Jorio, A. Tang, A. Romero, Y. Bengio, C. Pal, and S. Kadoury. Learning normalized inputs for iterative estimation in medical image segmentation. *CoRR*, abs/1702.05174, 2017.

[11] A. Fakhry, T. Zeng, and S. Ji. Residual deconvolutional networks for brain electron microscopy image segmentation. *IEEE Trans. Med. Imaging*, 36(2):447–456, 2017.

[12] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proc. CVPR*, pages 770–778, 2016.

[13] M. Helmstaedter. Cellular-resolution connectomics: challenges of dense neural circuit reconstruction. *Nature methods*, 10(6):501–507, 2013.

[14] E. Jurrus, A. R. C. Paiva, S. Watanabe, J. R. Anderson, B. W. Jones, R. T. Whitaker, E. M. Jorgensen, R. Marc, and T. Tasdizen. Detection of neuron membranes in electron microscopy images using a serial neural network architecture. *Medical Image Analysis*, 14(6):770–783, 2010.

[15] V. Kaynig, T. J. Fuchs, and J. M. Buhmann. Geometrical consistent 3d tracing of neuronal processes in sstem data. In *Proc. MICCAI*, pages 209–216, 2010.

[16] N. Krasowski, T. Beier, G. W. Knott, U. Koethe, F. A. Hamprecht, and A. Kreshuk. Improving 3d EM data segmentation by joint optimization over boundary evidence and biological priors. In *ISBI*, pages 536–539, 2015.

[17] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Proc. NIPS*, pages 1106–1114, 2012.

[18] R. Kumar, A. V. Reina, and H. Pfister. Radon-like features and their application to connectomics. In *CVPR Workshops*, pages 186–193, 2010.

[19] D. Laptev, A. Vezhnevets, S. Dwivedi, and J. M. Buhmann. Anisotropic sstem image segmentation using dense correspondence across sections. In *Proc. MICCAI*, pages 323–330, 2012.

[20] C. Lee, S. Xie, P. W. Gallagher, Z. Zhang, and Z. Tu. Deeply-supervised nets. In *Proc. AISTATS*, 2015.

[21] K. Lee, A. Zlateski, A. Vishwanathan, and H. S. Seung. Recursive training of 2d-3d convolutional networks for neuronal boundary prediction. In *Proc. NIPS*, pages 3573–3581, 2015.

[22] K. Li, B. Hariharan, and J. Malik. Iterative instance segmentation. In *Proc. CVPR*, pages 3659–3667, 2016.

[23] J. W. Lichtman and W. Denk. The big and the small: Challenges of imaging the brains circuits. *Science*, 334(6056):618–623, 2011.

[24] T. Liu, C. Jones, M. Seyedhosseini, and T. Tasdizen. A modular hierarchical approach to 3d electron microscopy image segmentation. *Journal of Neuroscience Methods*, 226:88–102, 2014.

[25] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *Proc. CVPR*, pages 3431–3440, 2015.

[26] L. Najman and M. Schmitt. Geodesic saliency of watershed contours and hierarchical segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 18(12):1163–1173, 1996.

[27] J. Nunez-Iglesias, R. Kennedy, T. Parag, J. Shi, and D. B. Chklovskii. Machine learning of hierarchical clustering to segment 2d and 3d images. *PLoS ONE*, 8(8), 2013.

[28] T. M. Quan, D. G. C. Hildebrand, and W. Jeong. Fusionnet: A deep fully residual convolutional neural network for image segmentation in connectomics. *CoRR*, abs/1612.05360, 2016.

[29] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *Proc. MICCAI*, pages 234–241, 2015.

[30] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 2015.

[31] M. Seyedhosseini, R. Kumar, E. Jurrus, R. Giuly, M. Ellisman, H. Pfister, and T. Tasdizen. Detection of neuron membranes in electron microscopy images using multi-scale context and radon-like features. In *Proc. MICCAI*, pages 670–677, 2011.

[32] M. Seyedhosseini, M. Sajjadi, and T. Tasdizen. Image segmentation with cascaded hierarchical models and logistic

disjunctive normal networks. In *Proc. ICCV*, pages 2168–2175, 2013.

[33] M. Seyedhosseini, M. Sajjadi, and T. Tasdizen. Image segmentation with cascaded hierarchical models and logistic disjunctive normal networks. In *Proc. ICCV*, pages 2168–2175, 2013.

[34] W. Shen, K. Deng, X. Bai, T. Leyvand, B. Guo, and Z. Tu. Exemplar-based human action pose correction and tagging. In *Proc. CVPR*, pages 1784–1791, 2012.

[35] W. Shen, X. Wang, Y. Wang, X. Bai, and Z. Zhang. Deepcontour: A deep convolutional feature learned by positive-sharing loss for contour detection. In *Proc. CVPR*, pages 3982–3991, 2015.

[36] W. Shen, K. Zhao, Y. Jiang, Y. Wang, Z. Zhang, and X. Bai. Object skeleton extraction in natural images by fusing scale-associated deep side outputs. In *Proc. CVPR*, pages 222–230, 2016.

[37] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

[38] O. Sporns, G. Tononi, and R. Kötter. The human connectome: A structural description of the human brain. *PLOS Computational Biology*, 1(4):245–251, 2005.

[39] Z. Tu. Auto-context and its application to high-level vision tasks. In *Proc. CVPR*, 2008.

[40] M. G. Uzunbas, C. Chen, and D. N. Metaxas. Optree: A learning-based adaptive watershed algorithm for neuron segmentation. In *Proc. MICCAI*, pages 97–105, 2014.

[41] N. Vu and B. S. Manjunath. Graph cut segmentation of neuronal structures from transmission electron micrographs. In *Proc. ICIP*, pages 725–728, 2008.

[42] S. Xie and Z. Tu. Holistically-nested edge detection. In *Proc. ICCV*, pages 1395–1403, 2015.

[43] A. Zlateski and H. S. Seung. Image segmentation by size-dependent single linkage clustering of a watershed basin graph. *CoRR*, abs/1505.00249, 2015.