Food and Drug Administration ILL

ILLiad TN: 236520

Borrower: GZU

Lending String: GZM,*HNT,SCE

Patron:

Journal Title: Bioinformatics

Volume: 33 **Issue:** 16
Month/Year: Aug 15 2017 **Pages:** 2555-62

Article Author: Zeng T, Wu B, Ji S.

Article Title: DeepEM3D: approaching human-level performance on 3D anisotropic EM image segmentation

Imprint: Oxford : Oxford University Press, ©1998-

ILL Number: 182544996

Call #:

Location:

Need Date: 10/28/2017

Special Instructions:

PDF
Maxcost: 20IFM

Shipping Address:
University of Wisconsin - La Crosse
Murphy Library ILL
1631 Pine Street
La Crosse, WI 54601-3792

Fax: 608.785.8806
Email: illoffice@uwlax.edu

Bioinformatics, 33(16), 2017, 2555–2562
doi: 10.1093/bioinformatics/btx188
Advance Access Publication Date: 30 March 2017
Original Paper

Bioimage informatics

DeepEM3D: approaching human-level performance on 3D anisotropic EM image segmentation

Tao Zeng, Bian Wu and Shuiwang Ji*

School of Electrical Engineering and Computer Science, Washington State University, Pullman, WA 99164, USA

*To whom correspondence should be addressed.
Associate Editor: Robert Murphy

Received on October 23, 2016; revised on February 24, 2017; editorial decision on March 25, 2017; accepted on March 25, 2017

Abstract

Motivation: Progress in 3D electron microscopy (EM) imaging has greatly facilitated neuroscience research in high-throughput data acquisition. Correspondingly, high-throughput automated image analysis methods are necessary to work on par with the speed of data being produced. One such example is the need for automated EM image segmentation for neurite reconstruction. However, the efficiency and reliability of current methods are still lagging far behind human performance.
Results: Here, we propose DeepEM3D, a deep learning method for segmenting 3D anisotropic brain electron microscopy images. In this method, the deep learning model can efficiently build feature representation and incorporate sufficient multi-scale contextual information. We propose employing a combination of novel boundary map generation methods with optimized model ensembles to address the inherent challenges of segmenting anisotropic images. We evaluated our method by participating in the 3D segmentation of neurites in EM images (SNEMI3D) challenge. Our submission is ranked #1 on the current leaderboard as of Oct 15, 2016. More importantly, our result was very close to human-level performance in terms of the challenge evaluation metric: namely, a Rand error of 0.06015 versus the human value of 0.05998.
Availability and Implementation: The code is available at https://github.com/divelab/deepem3d/
Contact: sji@eecs.wsu.edu
Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 Introduction

Brain functions are governed by information flow through the interconnected circuits of neurons. Investigating the components and network structure of this circuit is one of the top priorities in neuroscience research (Lichtman and Denk, 2011; Peng *et al.*, 2015). The goal of EM-based connectomics is to map neuronal circuits at single-cell level. This consists of two parts; namely generating high-resolution 3D images of neural tissue and reconstruction of corresponding circuits using these images. While the progress in EM imaging allows high-throughput acquisition of high-resolution neural images at the unprecedented scale, the speed of image analysis is lagging behind, forming a bottleneck to high-throughput connectomics (Lichtman and Denk, 2011). In early work on neuron circuit reconstruction that focused on organisms with simple neural systems such as *C. elegans*, the labeling was mainly done manually. As the focus extended to more complex organisms, image resolution and size scaled up, and even semi-automated method became a burden for human intervention. Despite attempts to fully automate this process, current methods are not able to achieve the required accuracy. As a result, reconstruction still relies heavily on human labor for proofreading of machine-segmented results. A breakthrough for automatic dense EM reconstruction is in urgent need, and better machine learning algorithms are considered the most likely approach toward this goal (Helmstaedter *et al.*, 2013; Jain *et al.*, 2010).

A common approach to generating EM images is to do serial section imaging that slices and scans brain tissues and stacks the images together as volume data. The EM scanning can achieve very high resolutions of the surface (XY-axes) of each slice, while slicing resolutions in the Z-direction is generally 10 times lower (Briggman and

Bock, 2012), which results in anisotropic 3D image stacks. Although recent advances in EM techniques have enabled isotropic 3D images, we focus on anisotropic 3D EM image in this work, as the majority of existing datasets of interest are anisotropic, and anisotropic EM imaging can produce sizable images relatively efficiently. Reconstructing neuronal circuits from such volume is very challenging. This is primarily because of the anisotropic nature of the data in which adjacent image slices are often misaligned due to imperfect sectioning of tissues or errors in alignment algorithms. In addition, processing large 3D volume is very time-consuming and computationally costly.

Typically, neurite reconstruction involves two steps: namely, generation of a membrane probability map from EM images and segmentation of neuronal processes based on this map. Early studies applied methods such as random forest (Kaynig *et al.*, 2015) to obtain probability maps. Recently, there has been an increasing member of cases that use deep convolutional neural networks (DCNN) (Berning *et al.*, 2015; Ciresan *et al.*, 2012; Turaga *et al.*, 2010). For the segmentation step, previous methods usually first generated over-segmented regions to form a hierarchical region graph before merging algorithms (Liu *et al.*, 2014) were applied to determine whether regions should be merged. For anisotropic 3D EM data, the segmentation is usually done on 2D slices one by one, and then these 2D regions are grouped across multiple sections into geometrically consistent 3D neuronal objects. Problems with such approaches are that domain knowledge is required to extract region features (Nunez-Iglesias *et al.*, 2013) and building hierarchical region graphs is computationally intensive.

It has been shown that putting more effort into learning a better membrane predictor will simplify post-processing (Farabet *et al.*, 2013), thereby leading to dramatic increases in speed and improved applicability of methods. In that spirit, we argue that algorithms providing highly accurate predictions of boundary probability maps require only simple post-processing, thereby reducing both computation costs and requirements on domain knowledge.

In earlier work (Fakhry *et al.*, 2016), we developed deep learning methods for segmenting 2D EM images (Arganda-Carreras *et al.*, 2015). Accurate segmentation of 3D anisotropic EM images requires not only accurate segmentation of 2D slices but also making consistent predictions across slices. This is further complicated by the fact that alignment of 2D slices is a challenging task, and thus segmentation methods have to deal with misalignment problems. In this work, we have developed a set of methods for accurate segmentation of 3D anisotropic EM images. Specifically, we propose a pipeline for segmenting 3D EM images with minimum computation costs for post-processing, yet producing very promising results. Our approach in 3D EM neurite reconstruction has achieved near-human-level accuracy. To the best of our knowledge, this is the first time an almost automatic neurite reconstruction has reached this level of performance. The result was achieved with simple human interventions at only two steps: namely, tuning a single threshold parameter in watershed and correcting misalignment of one slice.

The main contributions of this work are: (i) We proposed a novel deep network model for 3D neurite boundary detection. (ii) We developed a pipeline for segmenting image stacks which requires far less computation during post-processing and can be easily generalized to other datasets. (iii) We developed an ensemble strategy to deal with anisotropy and misalignment problems, which are commonly seen in 3D EM data. (iv) Our method represents the first computational approach that is close to human-level accuracy in 3D EM neurite segmentation.

2 Deep learning for anisotropic 3D EM images

Here we present a deep learning method for anisotropic 3D EM image segmentation. Our method is able to produce highly accurate 3D neurite boundary probability maps, thereby requiring only a simple watershed method to do segmentation. We applied our method to the anisotropic 3D images from the 3D segmentation of neurites in EM images (SNEMI3D) (http://brainiac2.mit.edu/SNEMI3D/) challenge. We created new 3D boundaries of neurites from segmentation labels in training data. We then used these labels and raw image stacks to train our deep models. To tackle problems resulting from the anisotropic resolution, we derived a series of architecture variants differing in the number of input slices, which allowed us to observe the effects of anisotropy in various scales. Upon inference, we used an ensemble strategy to effectively fuse the outputs of variants in order to integrate useful cross-slice information, while suppressing the cross-slice noise introduced by anisotropy. Finally, simple 3D watershed was used to produce final 3D segmentation.

2.1 Overview of key technical innovations

Traditionally, classifiers trained on handcrafted features have been used to merge over-segmented regions during post-processing (Farabet *et al.*, 2013; Jain *et al.*, 2011). This process usually requires a lot of computation and makes the whole system not end-to-end trainable. In order to simplify the post-processing in a way that relies only on the watershed algorithm for final segmentation, two issues need to be addressed.

First, the watershed algorithm is expected to connect geometrically non-separated areas. Thus a tiny broken point on the boundary between two regions can lead to a false merge. In other words, post-processing by watershed is sensitive to false negative boundary predictions. On the other hand, this method is robust to false positive boundaries. As long as no enclosed region is formed, it will not result in over-segmented regions.

The anisotropic property of 3D EM data is another challenge for 3D segmentation that uses simple post-processing. Compared to XY-directions, the boundary locations do not change smoothly along the Z-axis, leading to high probability of broken boundary in Z-direction. In addition, 3D EM image volume is comprised of stacked section images in which tissues are sectioned slice by slice across Z-direction. Such an image acquisition method has inherent problems of misalignment or deformation.

Taken together, our proposed method requires probability maps to be very accurate in terms of recall while allowing some degree of tolerance in precision metric (tolerance for false positives). Hence, reducing false broken boundaries while attempting to keep false boundary rates unchanged is the key to high-quality segmentation. To this end, we applied several techniques at multiple steps in the pipeline, aiming at increasing the boundary precision metric while still minimizing the negative effect caused by a possible lower recall metric.

First, we applied a boundary enhancement technique to the training stack. This generated several boundaries varying in thickness, which were used as labels to train multiple deep models. The idea was that a wide boundary helps maintain its continuity but may tend to reduce the size of small regions or even remove them. By contrast, a thin boundary is able to maintain the size of small regions but is at risk of resulting in a broken boundary in a stack that contains misaligned or deformed slices.

Second, we designed a novel deep architecture that incorporates inception and residual learning techniques in the object abstraction path, and skip connection and pyramid context growing techniques in the resolution recovery path. We constructed several models variants in order to have them trained on different sizes of slices in the Z-direction with labels of various thickness. Careful combination of probability maps produced by these models can enhance predicted boundaries that take advantage of 3D information while reducing anisotropic effects, resulting in high quality boundary maps. In the following sections, we describe these techniques in detail and show the improvement they make.

2.2 SNEMI3D dataset

The SNEMI3D challenge data consisted of two image stacks used for training and testing, respectively. Each stack contained one hundred 1024×1024 slices. The serial section scanning electron microscopy (ssSEM) technique was used to generate these image stacks by sectioning tissue into slices across the Z-dimension. This yielded anisotropic 3D images with high-resolution in the XY-plane and low resolution along the Z-dimension. The image resolution was $6 \times 6 \times 30$ nm/voxel which covers a micro-cube of approximately $6 \times 6 \times 3$ microns (Kasthuri *et al.*, 2015). There were 400 neurites in the training stack that had been labeled consistently across the 100 slices. Some neurites were split into several segments in some slices, but consistent labeling across the Z-direction was still required. This increased the complexity of 3D segmentation significantly. The labels of the test stack were not available to challenge participants, and segmentations submitted by participants were evaluated by challenge organizers. In our experiment, we used the top 20 slices for validation and the remaining 80 slices for training. The adapted Rand error, which was used by challenge organizers to assess the segmentation results, was defined as: 1– the maximal F-score of the Rand index (excluding the zero component of the original labels). Details on this metric were described in (Liu *et al.*, 2014). The ground truth labels were based on the consensus between two human expert raters, and the differences between them were listed as human values on the challenge leader board. It is important to note that the training and test stacks were from two different sites within neural systems. The morphology and organization of tissues in these two stacks were quite different, posing a major challenge for fully automated methods. We consistently observed that methods and parameters that worked well on a validation set (held out from the training stack) were not guaranteed to work on the test stack. Therefore, we only report results from the test stack.

2.3 Boundary label generation

In order to train classifiers for identifying neurite boundaries, a common approach is to use the provided membrane labels. However, in the SNEMI3D dataset, such membrane labels did not form closed boundaries for all neurites. Since we rely heavily on accurate boundaries to simplify the post-processing step, the quality of the deep model is of critical importance. A false broken boundary in the training labels is harmful, as it may lead to trained networks that predict more false negatives. This would consequently result in the false merging of regions, as the simplified post-processing is unable to correct the broken section. Thus, instead of directly using the provided membrane labels, we generated our own boundary labels from the segmentation labels of the training dataset. Suppose B is the binary boundary label map we want to generate, and we have $B(x, y, z) = 0$ if $S_{x,y,z} = 1$, and 1 if $S_{x,y,z} > 1$. Here $B = 1$ denotes boundary and 0 denotes neurites, $S_{x,y,z}$ is the number of distinct labels in the local cube centered at (x, y, z). This basically says, if pixels in the 3D kernel centered at (x, y, z) contain more than one distinct segmentation label, pixel at (x, y, z) of the boundary label map is marked 1. In this study, three different kernels were used: 5^3 cube, 3^3 cube, and 6-connected neighborhood along each axis (up-down, left-right, front-back). These are denoted as $c5$, $c3$ and $l3$, respectively. We used a sliding window search to mark all pixels as either boundary or neurite.

2.4 Convolution neural networks for dense prediction

Convolution neural networks (CNNs) have been successful in solving many visual tasks in recent years (LeCun *et al.*, 1998). CNNs in these tasks build hierarchical representations of the input in a bottom-up fashion. This is mainly achieved by applying convolutional layers followed by pooling in alteration to increase contextual information of each unit while reducing its spatial resolution. The increase of contextual information is achieved by an increased receptive field size as well as an increased number of feature maps. All together, they compute object-level representations at higher layers.

In dense prediction, the task is to label each pixel in the image. It thus needs to preserve spatial resolution. Therefore, this task implicitly requires solving the conflicting goals of preserving resolution and augmenting context at the same time. Fully convolutional networks (FCN), which extends the architecture used in image classification tasks by adding a resolution recovery path, was proposed to efficiently generate such dense predictions (Long *et al.*, 2015). The resolution recovery path in FCN uses bilinear kernels or learned transpose-convolution (deconvolution) kernels to recover the spatial resolution from multiple intermediate network layers, which are then aggregated to a yield final prediction. Such aggregation is also referred to as skip connection (Pinheiro *et al.*, 2016). FCN-based models have been widely used in EM image analysis tasks (Fakhry *et al.*, 2017), yielding significant improvements over traditional methods.

For the purpose of 3D EM image segmentation, it is natural to extend the 2D FCN architecture to 3D so the 3D contextual information can be incorporated. The example in Figure 4 clearly shows the advantage of using 3D contextual information. One major obstacle to a fully 3D FCN is the high computational and memory costs related to 3D convolution. More importantly, the anisotropic property in many 3D EM images may cause problems if we simply treat all three dimensions the same in 3D convolution (Chen *et al.*, 2016a). In light of these challenges, we tackled both problems jointly by designing a 3D-2D hybrid architecture for FCN-based models (Lee *et al.*, 2015).

2.5 Inception and residual module structure

In deep learning, the network's depth is an important factor. However, more layers may also lead to overfitting and gradient vanishing problems. Size of the receptive field, or size of the viewing field that the network can take information from, is another important factor that affects network performance. A larger receptive field includes more information. But as the receptive field grows larger, it also takes in more unrelated information (noise) if the object of interest does not fully occupies the field. Considering that objects in a real task may vary in size, a fix-size receptive field may not be sufficient.

Inception networks (Szegedy *et al.*, 2015) try to solve the above problem by using stacked inception modules. The inception module allows information to pass through kernels of multiple sizes (which correspond to multiple receptive field sizes) in a parallel manner. The downside of this approach is that suing multiple kernels increases the number of network parameters and computation costs.

Inception networks alleviate this problem by applying 1×1 convolution to reduce the number of input feature maps prior to convolution with large kernels. This effectively reduces model complexity yet achieves superior performance.

Another recent advance in deep learning is residual learning (He *et al.*, 2016), which introduces shortcut paths that skip one or several processing layers. The shortcut connections are equivalent to performing simple identity mapping, requiring a minimum of additional computation. The processing layers in the block parallel to the shortcut are considered to learn the residual between input and output feature maps, instead of projecting to output directly. Such residual learning is considered to be much easier than learning the original functions and is able to mitigate the gradient vanishing problem. As a result, residual learning enables the training of very deep networks.

2.6 Multi-scale context aggregation

In natural images, an object in view can vary in size depending on its physical size as well as the viewing distance to it. Similar to the benefit of using multi-scale receptive fields at the module level, providing multi-scale context information at the whole network level can improve semantic segmentation. Recently, dilated convolution has been developed to allow larger receptive fields without increasing the number of parameters. Yu and Koltun (2016) applied a series of dilated convolution filters with progressively larger dilation ratios from the bottom to the top layers, giving rise to very large receptive fields at the top layers. Chen *et al.* (2016b) proposed a spatial pyramid multi-dilated convolutional module to extract features from various receptive fields in parallel. These results were then combined together element-wise as outputs. We employed parallel operations similar to pyramid dilated convolution. Instead of convolution, though, we applied multiple dilated deconvolutions to aggregate multi-scale contexts at each upsampling layer, which were then combined to generate output feature maps.

2.7 CNN architecture design

The focus of this work is to deal with 3D volumetric data with anisotropic resolutions. In designing dense CNN architecture for 3D EM neurite segmentation, the following objectives were considered necessary to improve the quality of probability maps. (i) high efficiency in terms of both training and test time, (ii) aggregation of multi-scale context information needed for classification, (iii) using Z-directional information while being able to minimize the adversarial effect of noise. As illustrated in Figure 1, at each resolution level, we incorporated residual techniques into the inception modules by connecting identity projections from the bottom layer to the top layer, similar to Inception-V4 (Szegedy *et al.*, 2016). As stated in Section 2.5, use of inception-residual blocks benefits training time and allows multi-scale information to be processed at multiple layers, leading to very efficient hierarchical feature representations. We initially attempted to incorporate the original architecture in Szegedy *et al.* (2016), including inception-residual and inception-reduction blocks as modules of the bottom-up path in our model, as it achieved top performance in image classification tasks. Only the first two layers in the stem block were modified to allow 3D convolution, and skip connections were added as a resolution recovery path. The expectation was that the entire structure would fully use representational power similar to that in image classification. However, we found that simple reproduction of the original architecture of 2 Inception I blocks, 5 Inception II blocks and 3 Inception III blocks was prone to overfitting when applied to EM image segmentation. To overcome this problem, we limited the number of parameters in the model by reducing the number of inception-residual blocks. We found that a model consisting of only one of each type of inception-residual block was robust against overfitting (Supplementary Fig. S1). In the top-down spatial recovery path, we applied skip connections similar to those found in FCN but with added pyramid dilated deconvolution to further enhance the representational power of multi-scale information. We call this dense CNN architecture 'DeepEM3D-Net' throughout the rest of this paper. To address the anisotropy problem in 3D, the network was designed to have only two 3D convolution layers to integrate 3D information in the early stages and perform 2D convolution for the rest of following layers. This ensures minimum computation costs while taking advantage of 3D context information. Such early 3D fusion design allows the network to incorporate information in the Z-direction but mainly handle information in XY-directions. In addition, we derived a few variants of architecture to accept different numbers of image slices in the Z-direction. We expected that combining the outputs of these variants could help minimize the adversarial effect of noise caused by misalignment and anisotropy in 3D data.

At the very highest-level, the architecture bears some similarity with the U-net architecture (Ronneberger *et al.*, 2015). However, there are some key differences. Specifically, there are inception and residual modules in the encoding path, and each module is different in design. We also do not employ a symmetrical design between the encoding and decoding paths. Instead, skip connections from different encoding stages are upsampled to the same size and merged in one step. This design works well and saves memory during training.

3 The proposed segmentation pipeline

3.1 Data augmentation

Data augmentation can increase the effective size of training data, alleviating the overfitting problem. It is commonly used in training CNNs for image classification tasks. We augmented both training and test data by applying several spatial transformations on the input image. The transformations were combinations of horizontal and vertical mirroring, rotations by +90, -90 and 180 degrees in the

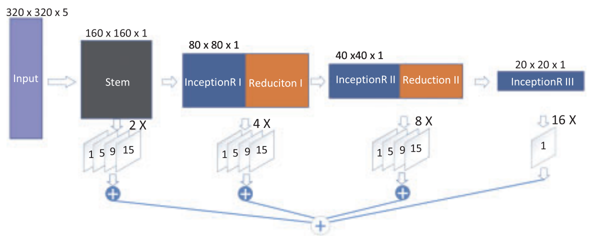

Fig. 1. The architecture of DeepEM3D-Net. The networks are designed for a dense segmentation task, which consisted of a bottom-up path (upper components) and a resolution recovery path (bottom components). The former consisted of stem, several inception-residual (InceptionR) modules, and spacial reduction modules. The later comprised several skip connections, which contained dilated deconvolution layers that recovered the spatial resolution of feature maps to the size of the networks's original inputs. Note that each of the first 3 skip connections contains 4 dilated connections to allow multiscale receptive fields, whereas the last skip connection (16 X) contains only 1 deconvolution layer intended to limit the number of parameters since the last layer of the bottom-up path contains over 2000 feature maps. Details for each module are given in the Supplementary Figure S1 (Color version of this figure is available at *Bioinformatics* online.)

XY-plane, and flip/horizontal mirroring in Z-direction, resulting in a total of 16 variants. For training, all 16 transformations were used. For testing, after given the 16 transformed data to the networks, we applied reverse transformations to each probability map. We took the average of predictions from each variation as the final output of the boundary probability map.

3.2 Model training and ensemble

In our study, we trained 5 dense CNN models with various boundary labels that were described in Section 2.3. These models differed only in the first two layers in order to take in different numbers of slices in Z-direction. Two models took $h \times w \times 5$ input and were trained with boundary labels $c5$ and $c3$, where h and w denote input size in X- and Y-directions, respectively. We used 320×320 for training and 1024×1024 at test time. The other three models take either $h \times w \times 1$, $h \times w \times 3$, $h \times w \times 5$ inputs and were trained with boundary label $l3$. The models that accepted 3 and 5 slices performed 3D convolutions in the first two layers and 2D convolutions in the remaining layers. Models that took only 1 slice performed 2D convolutions in all layers. During training, we randomly cropped $320 \times 320 \times d$ patches from 16 variants of raw image volumes, where the sizes of d were 1, 3 or 5 depending on the corresponding models. To tackle the problem of imbalanced class between boundary and non-boundary (neurite), we introduced class weights into the softmax loss layer by adding weights that were inversely proportional to class ratio. We trained models with a mini-batch size of 5 and with 50k iterations for each model. We used a base learning rate of 0.01 and stochastic gradient descent with a momentum of 0.9 and polynomial decay. The weights were initialized using Gaussian distribution with a standard deviation of 0.01. The cross-entropy loss was used as our training objective. During testing, we combined these models to improve the final segmentation. In Section 4, we show optimizing the combination of boundary probability maps from multiple models resulted in improved 3D EM segmentation.

3.3 Probability map generation

Entire image stack was processed in a sliding scheme during testing as shown in Figure 2. The input moved 1 slice along the Z-dimension each time and fed one or several full size neighboring images (1024×1024) to each of the models as described in Section 3.2. The number of slices fed includes 1, 3 or 5 based on the corresponding model configuration. Each model yielded 1 slice of the probability map spatially aligned with the center input slice. As a result, the entire stack of probability map was produced one by one. In this sliding process, each slice took only about 5 seconds to produce on GPU. The entire stack took less then 10 minutes to be produced. For each CNN model, all 16 variants of the raw image stack were used to produce 16 prediction stacks, which were reverse-transformed back to raw spatial order for voxel-wise averaging. The total time to predict the probability maps of an ensemble of 3 models with 16 variants of data on a single GPU was about 6 hours. In our experiment, we used 6 GPUs in parallel, and this reduced the total prediction time to 1 hour.

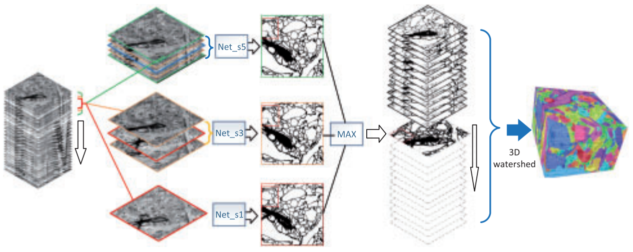

Fig. 2. The pipeline for producing 3D segmentation. Here we show 3 probability maps, which were combined using the element-wise maximum operation before the watershed algorithm was used to generate final segmentation. The actual combinations of multiple outputs in the experiment were described in Section 4. Note that the probability maps shown here are inverse probability maps for better visualization. The element-wise maximum was performed on the original probability maps, where the boundary value was close to 1 (white color) (Color version of this figure is available at *Bioinformatics* online.)

3.4 Post-processing

The 5 averaged output probability maps of each network model were further combined. Since our post-processing is relatively sensitive to false negative errors while robust to false positive errors in predicted boundary probability map, we combined the 5 probability maps by taking the maximum voxel value instead of average, as is done in most ensemble methods. This ensured boundary continuity to the maximum degree and minimized the probability of broken boundaries.

After obtaining the combined probability map stack, we smoothed the probability maps with a Gaussian kernel of size 6×6 and standard deviation of 1 on each 2D slice. Then, the 3D watershed algorithm was directly applied to the entire smoothed stack to generate the final segmentations. Since we trained the models using boundary labels that are connected in 3 perpendicular directions (Section 2.3), we used 6-connected neighborhood (up-down, left-right, front-back in 3D space) for the catchment basin search. Watershed algorithms construct regions from adjacent catchment basins determined by local minima, which typically lead to over-segmentation. To address this issue, the H-minima transform technique (Soille, 1999) was used to suppress undesired minima, thus reducing over-segmentation. The H-minima transform is controlled by a depth threshold. A large threshold leads to less over-segmentation. To compute an optimal depth threshold, we performed a grid search for this parameter on the training data and empirically fine-tuned it on the test data by visual inspection. Specifically, we focused on visualizing over-merged regions that were incorrectly segmented. Such merging is mainly due to high H-minima threshold that locally smooths out true boundaries, leading to incorrectly merged neighboring regions when watershed algorithm is applied. Once we observed over-merged regions, we slightly lowered the threshold by steps of 0.02 until over-merged segmentation was corrected. More details are provided in the supplemental document.

In contrast to prior methods that used complicated post-processing, we did not apply any post-processing after the 3D watershed. By putting more effort into boundary prediction, we were able to obtain high-quality cell membrane probability maps. At the same time, this strategy dramatically saved post-processing time. The 3D watershed involves only 1 parameter and does not rely on hand-crafted features.

4 Experimental results

4.1 Deep model selection

In order to identify CNN architectures suitable for 3D EM segmentation, we began by evaluating three CNN models. First, we designed the preliminary version of DeepEM3D-Net, termed pDeepEM3D-Net, which applied deconvolution in the upsampling layers instead of the multiple pyramid dilated deconvolution kernels. We then employed multiple pyramid dilated kernels, which led to DeepEM3D-Net. These two models were compared with residual deconvolutional networks (RDN) (Fakhry *et al.*, 2017), which was also designed for EM image segmentation. Ensemble models usually outperform single models. However, fully training multiple deep models is also time-consuming, so we performed an early stop during training, after 15K iterations, and evaluated these three models' performance on the test dataset using a single model (no data augmentation) trained on boundary *c5s5*. We used identical post-processing parameters for all models to produce the final segmentations. Table 1 shows that pDeepEM3D-Net yielded better segmentation than RDN. During the experiment, we also noticed that RDN's accuracy with the validation dataset started to drop after 15K iterations, which is an indication of overfitting. In contrast, pDeepEM3D-Net was able to improve accuracy after 15K iterations. Note that with addition of pyramid dilated deconvolution kernels to pDeepEM3D-Net, the DeepEM3D-Net further improved performance. Taken together, we thereby chose DeepEM3D-Net as our base architecture for further experiments. To handle the anisotropic problem, we constructed several variants of DeepEM3D-Net that were slightly modified to accept different numbers of slices. We implemented our models based on the Caffe library. We remodeled the library to enable fast 3D dense prediction on GPU, and designed a customized data cropping layer that was able to randomly crop patches from 3D raw data for the purpose of dense image-based training. In addition, we modified the cross-entropy loss layer to support a weighted version that could deal with class imbalance problems.

4.2 Approaches to improve performance

To fully evaluate the effects of boundary thickness and usefulness of information provided in the Z-direction, we trained 5 variants of DeepEM3D-Net, described in Section 2.7. Each DeepEM3D-Net variant was trained for 50K iterations, which took 4–5 days each on a single GPU. After close inspection of the dataset, we noticed that the training and test stacks were very different in terms of neurite morphology. Thus, the fine-tuned threshold for H-minima that performed well on the training stack did not produce a similar performance on the test stack. Therefore, instead of reporting the validation data result, we report only the test data results in the following.

The performance of different ensemble strategies is summarized in Table 2. We first used the *s5c5*model, which accepted a patch of 5 slices and was trained with label *c5*, to produce one probability map. This yielded a Rand error of 0.0906 on the test image stack. We noticed that the predicted probability maps produced by *s5c5*contained some broken boundaries. From both the raw image stack and the produced probability maps, we could see that boundaries in the slices above and below were well-aligned with each other. The middle slice, however, shifted dramatically and was inconsistent with any of its neighboring slices. This indicated the occurrence of misalignment in stacking. Such misalignment appears from occa-

Table 1. Segmentation performance achieved by different deep models

Architecture	Rand errors	# of trainable parameters
DeepEM3D-Net	0.0912	18M
pDeepEM3D-Net	0.0928	16M
RDN	0.0940	24M

Note: pDeepEM3D-Net uses only the original deconvolution for upsampling.

Table 2. Segmentation performance achieved by different combinations of networks

Global boundary	Local boundary	Rand error
s1l3, *s3l3*, *s5l3*	*s5c3*	0.0601
s1l3, *s3l3*, *s5l3*	*s5c5*	0.0648
s1l3	*s5c5*	0.0791
s5c5, *s1l3*	*NA*	0.0814
s5c5	*NA*	0.0906

Note: *s#* indicates the number of slices that the network accepts; *l#* denotes the type of boundary labels used for training. 'Global boundary' indicates that the boundaries were used for the entire test stack, while 'local boundary' indicates the application to partial slices in the test stack. More details are given in the text. Local boundary itself means that such boundary was only applied to a subset of slices that were visually inspected to have large deformations. There was only one slice that had major broken boundaries in the testing image stack. The bottom two rows use global boundary only and are the test stack results without correction for image misalignment.

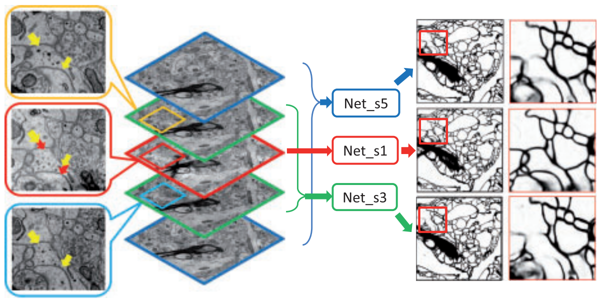

Fig. 3. Examples of boundary predictions in a misaligned image stack. First column: Magnified raw images corresponding to original image slices in the second column. Red arrows in the middle slice point to shifted boundaries from the supposedly aligned locations indicated by the yellow arrows. Second column: Example of a consecutive raw image stack. Third column: 3 networks that accept 5(blue), 3(green), or 1(red) slice(s) as input and produce one slice of prediction corresponding to the central slice. Fourth column: Predicted slices from 3 networks. Fifth column: Magnified images from predictions in the fourth column. Note that the network (Net-s1) using only one input slice yields better boundaries than do those using multiple slices (Color version of this figure is available at *Bioinformatics* online.)

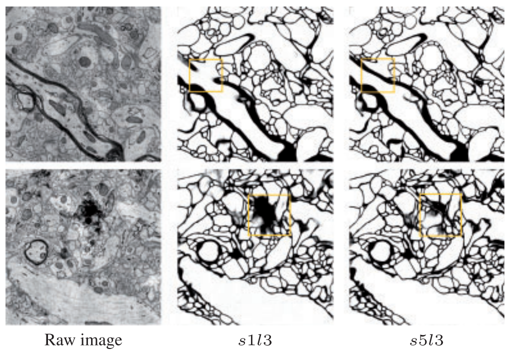

Fig. 4. Examples of predicted boundary in well-aligned image stack. The first column represents raw image in test stack. The second and third columns are the predictions of *s*1l3 and *s*5l3 CNN models, which take 1 and 5 slice(s) as input, respectively. Note that by using Z-direction context information, *s*5l3 model yields better boundaries than *s*1l3, which uses only X-Y plane context information (yellow box) (Color version of this figure is available at *Bioinformatics* online.)

sionally, and one example could be seen in Figure 3, which is in contrast to Figure 4. We only targeted misaligned slices that could result in broken boundaries in probability maps generated by models that take more than one slice as input (*s*3 and *s*5). Specifically, we identified misaligned slices by visually examining major broken boundaries on the probability map, and comparing these with neighboring slices. Practically, we found only one such misaligned slice from the testing stack. But due to the 3D connectivity of neurites, even small broken boundaries lead to incorrect merging of large regions. As a result, correcting such small error is very beneficial to overall segmentation quality. CNN architectures designed to take advantage of information from multiple slices are also prone to the noise brought about by those adjacent slices. To address this broken boundary problem, we considered using probability maps produced by the *s*1*l*3model, which accepts single slice input and produces probability maps that are independent of neighboring slices. Given that the properties of *s*1*l*3and *s5c5*are complimentary in dealing with anisotropic 3D data, we combined these two probability maps by taking voxel-wise maximums. This combination was able to reduce the Rand error to 0.0814.

Although a thick boundary is beneficial for segmentation in large regions as it enhances boundary continuity, it hampers segmentation in small regions. Given that the above *s5c5*model was trained on thick boundaries and consequently produced thick boundaries, we examined the effect of this factor on the test dataset. Specifically, instead of combining *s5c5*with *s*1*l*3, we used *s*1*l*3throughout the entire stack. We then combined the probability maps of *s5c5*and *s*1*l*3by taking their maximum values in slices that were identified as misaligned. Such local enhancement is based on the fact that misaligned slices can have large spatial displacements such that even successful boundary prediction in 2D slices can still result in broken boundaries in the Z-direction. We can see that such combination improved the segmentation and yielded a Rand error of 0.0791. In the experiment, we applied 10 slices of *s5c5*as local boundaries centered on the misaligned slice. Even though such local boundary involved manual observation to some degree, we found that the resulting performance was robust to the location of local boundaries. For example, by applying 5, 7, 10, or 13 slices of local boundary with sliding variants of 3–5 slices aligned to a slice that contain broken boundaries, we still achieved similar results.

Next, we added the probability maps of *s*1*l*3and *s*1*l*5globally into the above ensemble configuration. These two probability maps were obtained by using information in the Z-direction slices and training with thin boundary labels *l*3. This further improved the Rand error to 0.0648. Finally, we managed to improve performance in small regions while maintaining boundary connectivity in large regions. This was achieved by replacing thick *c*5 boundaries with slightly thinner *c*3 boundaries on those misaligned slices. As a result, we achieved a Rand error of 0.06015, which is very close to the human value of 0.05998. The current SNEMI3D challenge leader board as of October 15, 2016, is given in Table 3.

Table 3. The SNEMI3D challenge leader board as of October 15, 2016

Group	Rand error
human values	0.05998
DIVE (our team)	0.06015
IAL	0.06561
IAL_deprecated	0.08039
Team Gala	0.10041
SCI	0.10829

5 Conclusions

3D EM neurite reconstruction plays an important role in connectome studies. It is key to accurately segmenting raw neuronal EM images so that each single neurite is delineated by a unique label in 3D space. Generating boundary probability maps followed by post-processing is a widely used approach for this reconstruction task. Most traditional post-processing methods are time consuming and require domain knowledge, which forms a bottleneck for both speed and accuracy in the pipeline. We proposed a novel deep architecture for obtaining very accurate boundary probability maps, thus enabling a simple post-processing paradigm for final segmentation. Our model uses the power of inception and residual structures in the bottom-up path to efficiently integrate image information, and combines skip connection techniques with pyramid multi-scale contexture aggregation in the top-down path to produce dense prediction. Also, to maximally exploit the advantage of this model, we trained multiple deep models variants that accepted different numbers of input slices and predicted boundaries of different thickness. The ensemble strategy for boundary enhancement in probability maps produced by these models was the key to obtaining high quality segmentation, as well as to suppressing noise in Z-direction alignment. Given that misalignment is a common problem in anisotropic EM datasets, we proposed to overcome such problems by enhancing local boundary probability maps. In our setting visual examination to identify misalignment required very little human labor, it nevertheless limited full automation of the entire pipeline. Overcoming this limitation will be one of our key future goals.

Altogether, the results are exciting. To the best of our knowledge, our approach is the first nearly automatic 3D neurite reconstruction method that has achieved near-human-level performance. In addition, we believe that our simplified post-processing approach can generalize well to other similar dense prediction problems. In the future, we plan to apply similar end-to-end learning approaches, such as recurrent convolutional models, to 3D space for such tasks. We expect to remove the need for post-processing and further increase the generalizability of our methods.

Acknowledgement

We gratefully acknowledge the support of NVIDIA Corporation, which donated the Tesla K40 GPU used for this research.

Funding

This work was supported in part by National Science Foundation grant DBI-1641223.

Conflict of Interest: none declared.

References

Arganda-Carreras,I. *et al.* (2015) Crowdsourcing the creation of image segmentation algorithms for connectomics. *Front. Neuroanat.*, **9**, 142.

Berning,M. *et al.* (2015) SegEM: Efficient image analysis for high-resolution connectomics. *Neuron*, **87**, 1193–1206.

Briggman,K.L. and Bock,D.D. (2012) Volume electron microscopy for neuronal circuit reconstruction. *Curr. Opin. Neurobiol.*, **22**, 154–161.

Chen,H. *et al.* (2016a) Voxresnet: Deep voxelwise residual networks for volumetric brain segmentation. *arXiv Preprint arXiv*, *1608.05895*.

Chen,L.-C. *et al.* (2016b) Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *arXiv Preprint arXiv*, *1606.00915*.

Ciresan,D. *et al.* (2012) Deep neural networks segment neuronal membranes in electron microscopy images. In: *Advances in Neural Information Processing Systems*, pp. 2843–2851.

Fakhry,A. *et al.* (2016) Deep models for brain EM image segmentation: novel insights and improved performance. *Bioinformatics*, **32**, 2352–2358.

Fakhry,A. *et al.* (2017) Residual deconvolutional networks for brain electron microscopy image segmentation. *IEEE Trans. Med. Imaging*, **36**, 447–456.

Farabet,C. *et al.* (2013) Learning hierarchical features for scene labeling. *IEEE Trans. Pattern Anal. Mach. Intell.*, **35**, 1915–1929.

He,K. *et al.* (2016) Deep residual learning for image recognition. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Helmstaedter,M. *et al.* (2013) Connectomic reconstruction of the inner plexiform layer in the mouse retina. *Nature*, **500**, 168–174.

Jain,V. *et al.* (2010) Machines that learn to segment images: a crucial technology for connectomics. *Curr. Opin. Neurobiol.*, **20**, 653–666.

Jain,V. *et al.* (2011) Learning to agglomerate superpixel hierarchies. In: *Advances in Neural Information Processing Systems*, pp. 648–656.

Kasthuri,N. *et al.* (2015) Saturated reconstruction of a volume of neocortex. *Cell*, **162**, 648–661.

Kaynig,V. *et al.* (2015) Large-scale automatic reconstruction of neuronal processes from electron microscopy images. *Med. Image Anal.*, **22**, 77–88.

LeCun,Y. *et al.* (1998) Gradient-based learning applied to document recognition. *Proc. IEEE*, **86**, 2278–2324.

Lee,K. *et al.* (2015) Recursive training of 2d-3d convolutional networks for neuronal boundary prediction. In: *Advances in Neural Information Processing Systems*, pp. 3573–3581.

Lichtman,J.W. and Denk,W. (2011) The big and the small: challenges of imaging the brain's circuits. *Science*, **334**, 618–623.

Liu,T. *et al.* (2014) A modular hierarchical approach to 3D electron microscopy image segmentation. *J. Neurosci. Methods*, **226**, 88–102.

Long,J. *et al.* (2015) Fully convolutional networks for semantic segmentation. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3431–3440.

Nunez-Iglesias,J. *et al.* (2013) Machine learning of hierarchical clustering to segment 2D and 3D images. *PloS ONE*, **8**, e71715.

Peng,H. *et al.* (2015) BigNeuron: large-scale 3D neuron reconstruction from optical microscopy images. *Neuron*, **87**, 252–256.

Pinheiro,P.O. *et al.* (2016) Learning to refine object segments. *arXiv Preprint arXiv*, *1603.08695*.

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. Springer.

Soille,P. (1999) *Morphological Image Analysis: Principles and Applications*. Springer-Verlag New York, Inc. Secaucus, NJ, USA.

Szegedy,C. *et al.* (2015) Going deeper with convolutions. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–9.

Szegedy,C. *et al.* (2016) Inception-v4, inception-resnet and the impact of residual connections on learning. *arXiv Preprint arXiv*, *1602.07261*.

Turaga,S.C. *et al.* (2010) Convolutional networks can learn to generate affinity graphs for image segmentation. *Neural Comput.*, **22**, 511–538.

Yu,F. and Koltun,V. (2016) Multi-scale context aggregation by dilated convolutions. In: *Proceedings of the International Conference on Learning Representations*.

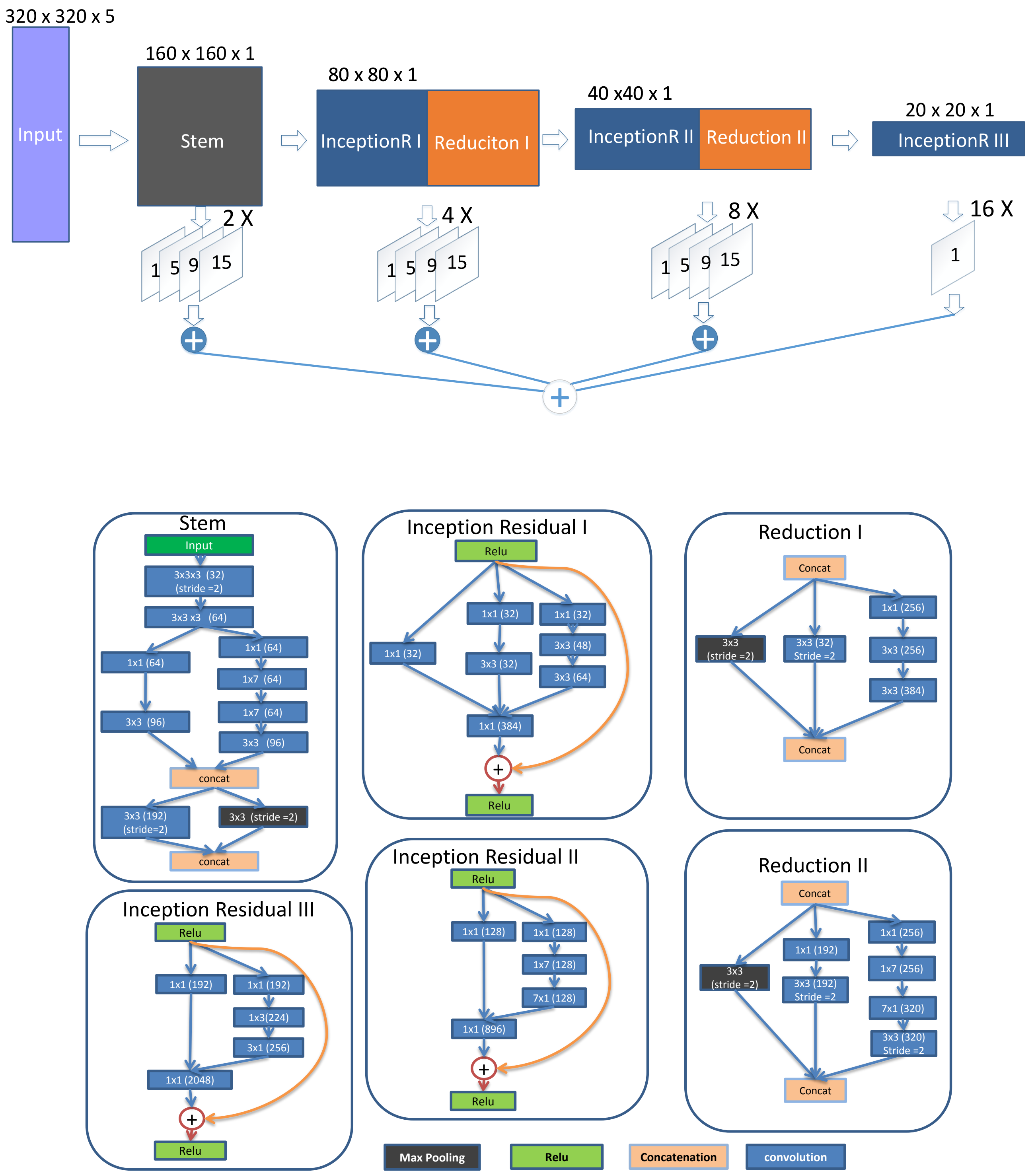

Fig. S1. The detailed architecture of DeepEM3D-Net. Top: the network consists of 6 modules in the bottom-up path, 3 pyramid dilated deconvolutions and 1 regular deconvolution in the resolution recovery path. The number in feature map of deconvolution indicates the dilation rate of deconvolution kernels. Bottom: configuration details of the 6 modules in the bottom-up path. The circled plus sign denotes the element-wise summation. Note that padding of 1 was used prior to carrying out 3 by 3 convolution in order to keep the size of feature map unchanged.

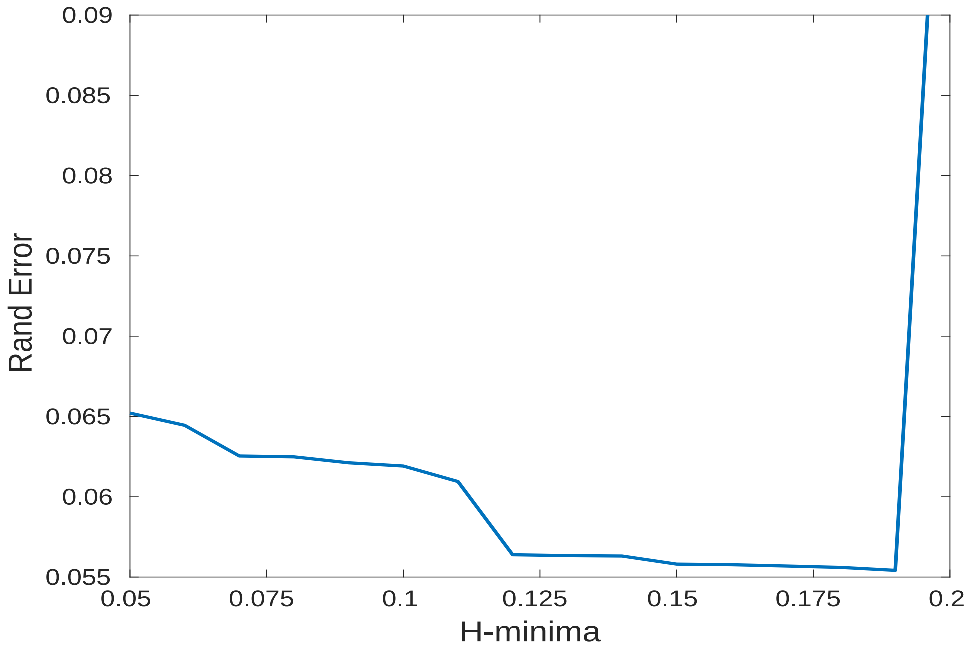

Fig. S2. The curve of H-minima values against Rand error of segmentation on the validation set. Note that the Rand error gradually reduces to an optimal value around 0.18 as H-minima increases. Continually increasing H-minima after that resulted in dramatic increase in Rand error. This is due to broken boundary caused by large H-minima value, resulting in incorrect merge of large regions.