# BOOTSTRAPPING 3D ELECTRON MICROSCOPY SEGMENTATION DATASETS

*Matthew Guay*

National Institutes of Health
National Institute of Biomedical Imaging and Bioengineering
Bethesda, MD, 20892

## ABSTRACT

Biomedical electron microscopy (EM) image segmentation has received a great deal of interest from the computer vision and machine learning research communities, driven by the challenge of automating construction of 3D models of cells and cellular structures. Despite high-profile work on neural membrane segmentation benchmarks for connectomics, biologists in other areas generally still lack practical tools for accelerating 3D modeling of features in EM image volumes. Here, we demonstrate how to bootstrap a large segmentation dataset using a convolutional neural network training, inference, and correction scheme tailored to "growing" a label dataset quickly from a small manually-labeled seed region. Our bootstrapping approach yields a $100\times$ acceleration for creating EM image segmentation labels.

***Index Terms*—** deep learning, automated segmentation, electron microscopy, serial block-face imaging

## 1. INTRODUCTION

Electron microscopy (EM) allows biomedical researchers to investigate the structure of biological matter at the nanoscale, and recent advances in 3D imaging methods such as serial block face scanning electron microscopy (SBF-SEM)[1] are capable of rapidly producing gigavoxel and larger datasets. Such datasets give scientists the raw materials needed to create 3D structural models of organelles, cells, and tissues across unprecedented spatial volumes, but building 3D models at scale requires automated tools to assist in segmenting image features. Image segmentation tasks for biological EM can be extremely challenging to automate, and fully-automated solutions with sufficient accuracy for research projects are still a matter of intense research in the biomedical computer vision community.

The work in this paper builds upon a popular line of biomedical semantic segmentation research into U-Nets[2], convolutional encoder-deocder neural networks capable of segmenting large image patches simultaneously. This network motif has been expanded upon greatly, including 3D variants tailored to volumetric imaging[3].

With these algorithms, great strides have been made towards high-quality, fully-automated, robust segmentation algorithms, but turn-key solutions do not yet exist. There are challenge competitions and popular benchmark datasets where researchers may compete for the best task metrics, but using neural segmentation tools to accelerate concrete biomedical EM research applications remains difficult. Now, software such as ImageJ and Arivis are developing plugins or features to incorporate neural segmentation into polished image processing and annotation workflows, but work remains for creating flexible and robust segmentation tools. While we await the holy grail of accessible, robust, and highly accurate general-purpose EM segmentation algorithms, there is much that can be done with the current generation of semantic segmentation neural networks to dramatically accelerate the creation of segmentation label datasets.

In this paper, we show how to bootstrap a segmentation dataset from a single large image volume with limited manual labor, using a novel U-Net variant and training procedure adapted to the process. In five bootstrapping iterations, a single researcher was able to segment all human platelet cell membrane in a $1200 \times 1100 \times 22$ SBF-SEM volume with only 9 hours of manual labeling effort, 3.5 of which were needed to label an $800 \times 800 \times 1$ initial seed region completely by hand. In the final bootstrap iteration, we observed up to a $113\times$ acceleration in membrane labeling overall and up to a $364\times$ acceleration in contour correction. We provide resources for interested individuals to try out this method on their own data, using only free, open-source tools.

## 2. METHODS

### 2.1. Data

This study used data prepared for a previous project from a human platelet sample as part of a collaborative effort between the National Institute of Biomedical Imaging and Bioengineering (NIBIB), NIH and the University of Arkansas for Medical Sciences. The resin-embedded, heavy metal-stained tissue sample was imaged using a Gatan 3View® SBF-SEM to produce a $2000 \times 2000 \times 250$ image volume with $x - y$ resolution of 10nm and $z$ resolution of 50nm. See Figure 1

DRAFT

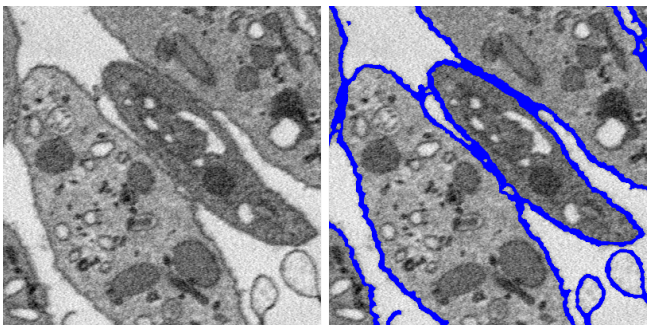


**Fig. 1**. A sample $400 \times 400$ image region showing cross sections of human platelet cells, and an overlay of ground-truth cell membrane labels on the image.

for an example of the data and ground-truth cell membrane labels.

### 2.2. Software

For manual segmentation and manual correction of algorithmic segmentations, we used the free, open-source GNU Image Manipulation Program, though any image processing software with basic draw, paint, region selection, and layering would suffice. All algorithmic tools were written in Python using the PyTorch library.

### 2.3. Bootstrapping

We have developed a method to bootstrap a mostly-automated segmentation system that begins with manual labeling of a small amount of training data. Label bootstrapping was done iteratively, training a neural network on the initial "seed" data, using it to segment an image region near the seed region, correcting that segmentation, performing another round of network training using the enlarged labeled dataset, and so on.

The goal of this bootstrapping technique is to mitigate the problems caused by networks trained on small datasets overfitting to the data and generalizing poorly. When producing a neural network intended for general use on arbitrary data within some class, this overfitting is a problem. However, for large-volume images like those produced by SBF-SEM, one can progressively segment an expanding contiguous region within the image while only encountering new data that looks very similar to the data in the training set. In this way, one can avoid the problem of needing a large amount of initial training data. After several **bootstrapping iterations** (BI), one can label a large data region and get a high-quality, more generalizable segmentation algorithm with only a modest amount of time spent manually labeling the seed region and manually correcting algorithm output.

For this study, we performed five bootstrap iterations, recording the time required for manual labeling, manual label correction, network training, and network inference. Since label data expansion was generally done by segmenting $800 \times 800$ 2D image regions stacked sequentially along the $z$ axis, these regions are referred to here as $z$-slices, with the initial one being $z = 0$. A summary of the techniques used for each BI follows.

**BI 0**: Manual labeling of the $z = 0$ $z$-slice.

**BI 1**: We preprocessed image data by applying an unsharp filter and median filter. We trained a standard 2D U-Net, with instance normalization layers before each pooling or upsampling layer, with a $404 \times 404$ input window shape, using ADAM with elastic deformation for data augmentation. We used the trained network to segment the $z = 1$ $z$-slice, and corrected its output.

**BI 2**: We used the same preprocessing and network training setup as BI 1. We used the trained network to segment $z$-slices 2-4.

**BI 3**: We used a U-Net with depthwise separable convolutions and LeakyReLU activations instead of the standard ReLU. The training loss function for all BIs was a weight decay-regularized cross-entropy function, summed over all voxels in the training region, weighted per-voxel by the class frequency of the label class it contained (cell membrane vs. background). BI 3 introduced a new weighting term to the loss function - **prediction error weighting**. Since each successive BI's training dataset is built by correcting the output of a previous BI's segmentation predictions, we have information about regions where networks are likely to make errors. These regions can be upweighted in the training loss calculation for future BI's. The trained network was used to segment $z$-slices 5-9. Since $z$-slices are larger than the input size of the networks, they are segmented by stitching together windows. We found that setting a high overlap between successive windows and averaging the predictions led to an improvement in quality, at the cost of increased computation time.

**BI 4**: Now with 10 $z$-slices in the training region, we were able to switch to 3D networks. We used a **Thin 3D U-Net**, with a window shape of $302 \times 302 \times 3$. The "thin" distinction refers to the fact that no pooling or upsampling was done along the $z$ axis, and all convolutions were zero-padded, so that the input, output, and all intermediate data tensors had length 3 along the $z$ axis. We reverted to standard convolutions due to the significantly increased computation time for 3D depthwise separable convolutions, but retained the LeakyReLU activations. The prediction error weighting was modified to only upweight false negative regions, as these are far more time consuming to correct. When segmenting new data, an ensemble of 5 networks with the same architecture and training routine but different random initializations were used. The $z$-slices 0-9 were resegmented to an $x - y$ size of $1200 \times 1100$, and 25 $z$-slices from $z = -5$ through $z = 19$ were segmented, in addition to 3 $800 \times 800$ $z$-slices far from the training region. This latter region (`eval-far`) and $z$-slices -5 through -3 (`eval-near`) were used as validation regions during a final round of training to gauge how

DRAFT

performance differs between data near the training region and data further from the training region.

All networks were small enough to fit on a single NVIDIA GTX 1080 GPU for training and inference, though a cluster of NVIDIA Tesla V100's was used to accelerate training of multiple network variants as well.

We validated our bootstrapping hypothesis by comparing the performance of our BI 4 net on `eval-near` and `eval-far` after training instances on a variable number of $z$-slices, from 3 to the full 22 available. We computed the percentage of false negative regions in each validation $z$-slice. In particular, the false negatives that are costly to correct are the large contiguous missing regions, mostly seen at points of contact between two cells. Therefore, we computed our false negative percentages after excluding connected components with area smaller than a given threshold. We call this statistic **FN-n** in the following text to indicate a false negative percentage counting only regions with area greater than or equal to $n$.

## 3. RESULTS

Table 1 summarizes the contour correction timing results. The progression of algorithmic segmentation quality can be seen in Figure 2 (a)-(b). Overall, between BI 0 and BI 4 we observe on average a $\mathbf{131}\times$ **decrease** in time needed for contour correction. The prototype workflow used in this study incurred additional labeling costs for false positive removal which were fixed at about 2 minutes per $z$-slice. This can be automated by combining the cell membrane segmentation network with a full cell segmentation network, using its predictions to label the image connected components partitioned by the cell membrane labels, and deleting any detected regions fully within cells, but even including the additional 2 minutes per slice we realize on average a $\mathbf{67}\times$ **decrease** in labeling time.

Each network instance took approximately 12 hours to train on an NVIDIA Tesla V100, and segmented new data at approximately 1 megavoxel per minute on an NVIDIA GTX 1080, though this decreased linearly when doing ensemble segmentation by sequentially computing ensemble element segmentations. By dovetailing training and inference while manual label correction took place, they ended up not being a bottleneck in the process. Proper distributed or multi-GPU computation and optimizations to the training procedure would also decrease these times significantly, should they become a bottleneck.

These numbers speak for themselves in terms of accelerating label creation, but we also present results concerning our bootstrapping hypothesis. We observe a significant difference in false negative rates between algorithmic segmentations of the `eval-near` and `eval-far` regions. Figure 3 shows the mean FN-401 values for BI4 network instances trained on varying numbers of $z$-slices. Interestingly, when very small training datasets are used, performance appears to be similarly poor for both regions, but as the training dataset size increases more significant differences are observed. While in absolute numbers all of these percentages are small, less than $1\%$, the relative difference is important. See Figure 2 (c)-(d) for an example of the difference in validation performance for a BI 4 network trained on 9 $z$-slices. As raw algorithmic accuracy approaches $100\%$, a small difference in the number of contours to correct has a magnified effect on total time required for segmentation. Despite its generally good accuracy, the correction of the BI 4 ensemble's output for `eval-far` to create ground-truth labels took on average 6.0 minutes per megavoxel, $2.4\times$ longer than the average time indicated in the BI 4 row of Table 1.

| BI | # MVox | Mean (m) | Min (m) | Max (m) |
|---|---|---|---|---|
| 0 | 0.64 | 328 | 328 | 328 |
| 1 | 0.64 | 31.3 | 31.3 | 31.3 |
| 2 | 1.92 | 27.3 | 25.2 | 29.4 |
| 3 | 3.2 | 20.7 | 16.6 | 24.8 |
| 4 | **33** | **2.5** | **0.9** | **3.9** |

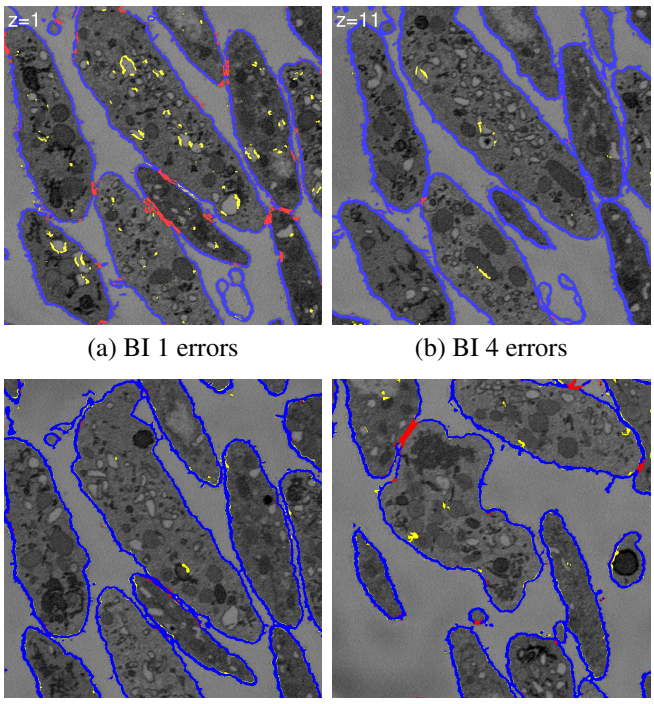
**Table 1**. Summary of contour correction times per bootstrap iteration (BI). Column 2 indicates the number of megavoxels (MVox) segmented per BI. Columns 3-4 indicate the mean, min, and max manual contour correction times in minutes per megavoxel of image volume, for correction times which were recorded per 2D $z$-slice in the segmentation region.

## 4. CONCLUSION

The goal of this study was to determine an effective strategy to practically accelerate the creation of high-quality semantic segmentation labels for biomedical EM. We hypothesized that, for very large image volumes such as those created by SBF-SEM, the spatial correlations between nearby image regions would allow one to effectively bootstrap a segmentation label dataset by iteratively training neural networks on progressively larger contiguous regions of data. We have demonstrated that our bootstrapping hypothesis is consistent with our observations, and we have demonstrated that with this strategy we can realize $100\times$ acceleration of label creation. We have also shown that it can be done using only free, open-source tools and small networks capable of running on commodity compute hardware, so that interested researchers can get started immediately with the approach. The code used for this project is available at https://www.github.com/leapmanlab/isbi2021.

These results, while promising, still offer substantial room for improvement. Integrating cell membrane segmentation with additional feature segmentation would allow false positive removal to be automated to remove a significant time cost. Moreover, while there is value in making this approach

DRAFT




(a) BI 1 errors (b) BI 4 errors

(c) `eval-near` errors (d) `eval-far` errors

**Fig. 2**. (a)-(b) Comparison of label errors for example $z$-slices from BI 1 and BI 4. (c)-(d) Comparison of label errors bewteen `eval-near` and `eval-far` for a BI 4 network trained on 9 $z$-slices. False negatives are red, false positives are yellow.

accessible through the use of free software, proper integration with commercial microscopy annotation software for 3D annotation and correction is another potential source of acceleration. The iterative nature of the bootstrapping process in this study reflects the relatively crude tools used to do it - integrated into a proper active learning environment, correction times for successive $z$-slices would decrease significantly, since errors between successive slices were highly correlated. Our lab is pursuing all of these lines of development, but we are pleased to share this project in its current stage so that some of the much-hyped potential of deep learning can be converted to tangible advances for biomedical EM practitioners.

## 5. COMPLIANCE WITH ETHICAL STANDARDS

The data used in this study contains no personally identifiable information and was made available for open access by NIBIB as a U.S. government creative work.

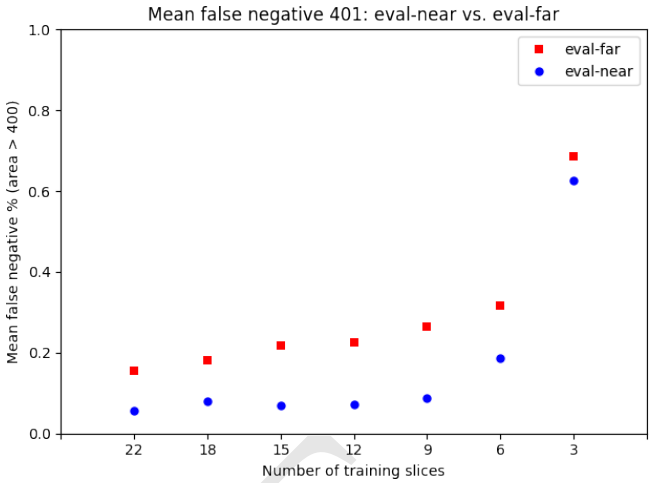



**Fig. 3**. Performance comparison on the `eval-near` and `eval-far` validation regions. The BI 4 network was trained on a varying number of $z$-slices, from 3-22, as indicated on the $x$-axis. For each slice count, we trained 10 network instances, varying only the random number generator seeds controlling weight initialization and training data augmentation, to account for variances in final validation metrics due to small training dataset sizes.

## 6. ACKNOWLEDGMENTS

This work was supported by the intramural program of NIBIB, NIH. The author has no relevant financial or non-financial interests to disclose.

This work utilized the computational resources of the NIH HPC Biowulf cluster. (https://hpc.nih.gov).

The author thanks Dr. Richard Leapman of NIBIB and Prof. Brian Storrie of the University of Arkansas for Medical Sciences for their helpful discussions and feedback.

## 7. REFERENCES

[1] Winfried Denk and Heinz Horstmann, "Serial block-face scanning electron microscopy to reconstruct three-dimensional tissue nanostructure," *PLoS Biol*, vol. 2, no. 11, pp. e329, 2004.

[2] Olaf Ronneberger, Philipp Fischer, and Thomas Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.

[3] Özgün Çiçek, Ahmed Abdulkadir, Soeren S Lienkamp, Thomas Brox, and Olaf Ronneberger, "3d u-net: learning dense volumetric segmentation from sparse annotation," in *International conference on medical image computing and computer-assisted intervention*. Springer, 2016, pp. 424–432.

DRAFT