Dense cellular segmentation using 2D-3D neural network ensembles for electron microscopy

Matthew Guay[1,+,*], Zeyad Emam[1,2,+], Adam Anderson[1,2], Maria Aronova[1], and Richard Leapman[1]

[1]National Institute of Biomedical Imaging and Bioengineering, NIH, Bethesda, 20892, USA
[2]University of Maryland, College Park, 20740
[*]matthew.guay@nih.gov
[+]these authors contributed equally to this work

ABSTRACT

Modern biological electron microscopy produces nanoscale images from biological samples of unprecedented volume, and researchers now face the problem of making use of the data. Image segmentation has played a fundamental role in EM image analysis for decades, but challenges from biological EM have spurred interest and rapid advances in computer vision for automating the segmentation process. In this paper, we demonstrate dense cellular segmentation as a method for generating rich, 3D models of tissues and their constituent cells and organelles from scanning electron microscopy images. We describe how to use ensembles of 2D-3D neural networks to compute dense cellular segmentations of cells and organelles inside two human platelet tissue samples. We conclude by discussing ongoing challenges for realizing practical dense cellular segmentation algorithms.

Biomedical researchers use electron microscopy (EM) to image the structure of cells, organelles, and their constituents at the nanoscale. New EM instruments such as serial block-face scanning electron microscopes (SBF-SEM)[1] and focused ion-beam scanning electron microscopes (FIB-SEM)[2] use automated serial sectioning techniques to rapidly produce gigavoxel image volumes and beyond. This rapid growth in throughput challenges traditional image analytic workflows for EM, which relies on trained humans to identify salient image features. High-throughput EM offers to revolutionize systems biology by providing nanoscale structural detail across macroscopic tissue regions, but analyses of such datasets in their entirety will be infeasibly expensive and time-consuming until analytic bottlenecks are automated.

A fundamental component of common EM image anaylsis workflows is *segmentation*, which groups image voxels together into labeled regions that correspond to image content. For semantic segmentation, each voxel is assigned an object classification label, such as `cell` or `mitochondrion`. In this paper we introduce a *dense* semantic segmentation task, illustrated in Figure 1, which seeks to segment the entirety of an image volume with multiple granular class labels. Dense semantic segmentation is vital for systems biologists seeking to create semantically-rich 3D models of cells and subcellular structure interconnected within tissue environments.

Manually performing dense segmentation tasks for EM volumes is tedious and infeasible at scale for new high-throughput microscopes. However, automating dense segmentation for EM is challenging due to the image complexity of biological structures at the nanoscale. Current state-of-the-art computational solutions use a variety of convolutional neural network architectures to solve problems on a per-task basis, and automation difficulty is task-dependent. An image with little noise and high contrast between features may be accurately segmented with simple methods such as thresholding[3], while accurate segmentation of complex images with multiscale features, noise, and textural content remains an open problem for many tasks of interest to biomedicine. The field of connectomics has spurred a number of advances in biomedical segmentation in its pursuit of accurate neural circuit tracing[4], but there are few solutions for other analysis problems facing systems biologists, and working with 3D data presents challenges for common compute hardware to properly exploit 3D spatial context during classification.

In this paper, we introduce a new 3D biomedical segmentation algorithm based on neural networks with separate

2D and 3D prediction modules, building off of existing work in volumetric image segmentation[5] to solve a dense segmentation task for a platelet SBF-SEM dataset. We also demonstrate a simple ensembling technique for improving the performance of segmentation networks. The resulting segmentation algorithms outperform existing 3D segmentation networks trained on dense segmentation of our platelet data. We also highlight a challenge for validating segmentation network design choices - the wide distribution of trained architecture performance results due to random initialization - and offer a simple solution to make better-informed conclusions on segmentation network architecture design.

Prior Work

In recent years, *deep convolutional neural networks* (CNN) became the de facto algorithms on most computer vision tasks[6–8]. The field of 2D segmentation of natural images has received enormous attention, with major companies releasing sophisticated trained models in the pursuit of solutions to vision problems of economic importance[9–12].
In comparison, work in biomedical image segmentation has been comparatively modest, with more work done in medical imaging such as CT [**cite**] than in microscopy. A seminal contribution from this area was the U-Net[7], which spawned numerous encoder-decoder variants sporting architectural improvements[13–15]. The proliferation of volumetric data in biomedicine has also spurred developments in 3D segmentation[5,16–22], though the field suffers from a lack of high-quality benchmark datasets for use with testing different architectural choices. For this paper, we adapted existing U-Net, Deeplab, and DeepVess[22] architectures to our segmentation task as a baseline for our new results.

Methods

SBF-SEM image volumes were obtained from identically-prepared platelet samples from two humans. Lab members manually segmented portions of each volume into seven classes to analyze the structure of the platelets. The labels were used for the supervised training of candidate network architectures, as well as baseline comparisons. Each candidate architecture was trained multiple times with different random initializations. The best-performing instances were ensembled together to produce the final segmentation algorithms used in this paper.

Data collection

This study used datasets prepared from human platelet samples 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 platelet samples were imaged using a Zeiss 3View SBF-SEM. The Patient 1 dataset, the main one used in this study, is a $250 \times 4000 \times 4000$ voxel image with a lateral resolution of 10 nm and an axial resolution of 50 nm, from a sample volume with dimensions $12.5 \times 40 \times 40 \mu\text{m}^3$.
We assembled labeled training, evaluation, and test datasets from manually-segmented regions of the platelet datasets. Lab members created initial manual segmentations using Amira[23]. For the test dataset, segmentations were reviewed by subject experts and corrected. The training image was a $50 \times 800 \times 800$ subvolume of the Patient 1 dataset spanning the region $81 \leq z \leq 130$, $620 \leq x \leq 1419$, $1073 \leq y \leq 1872$ in 0-indexed notation. The evaluation image was a $24 \times 800 \times 800$ subvolume of the Patient 1 dataset spanning the region $100 \leq z \leq 123$, $620 \leq x \leq 1419$, $200 \leq y \leq 999$. The test image was a $121 \times 609 \times 400$ subvolume of the Patient 2 dataset spanning the region $0 \leq z \leq 120$, $460 \leq x \leq 1068$, $308 \leq y \leq 707$. The training and evaluation labels covered the entirety of their respective images, while the test labels covered a single cell contained within the test image. The labeling schema divides image content into seven classes: `background`, `cell`, `mitochondrion`, `canalicular vessel`, `alpha granule`, `dense granule`, and `dense granule core`.

Neural architectures and ensembling

The Patient 1 and Patient 2 datasets were binned by 2 in x and y and aligned. For each of the training, evaluation, and testing procedures, the respective image subvolumes were normalized to have mean 0 and standard deviation 1 before further processing.

The highest-performing network architecture in this paper, 2D-3D+3x3x3, is a composition of a 2D U-net-style encoder-decoder and 3D convolutional spatial pyramid, with additional 3x3x3 convolutions at the beginning of convolution blocks in the encoder-decoder. All convolutions are zero-padded to preserve array shape throughout the network, allowing deep architectures to operate on data windows with small z-dimension. A ReLU activation follows each convolution. All convolution and transposed convolutions use bias terms. The architecture is fully specified as a diagram in Figure 2(a). Additionally, several baseline comparison networks and three 2D-3D+3x3x3 ablation networks were also tested in this paper and are described in Validation and Performance Metrics.

To build a 2D-3D network, one can adapt a 2D U-net-style encoder-decoder module to work on 3D data by recasting 2D 3x3 convolutions as 1x3x3 convolutions, and 2D 2x2 max-pooling and transposed convolution layers as 1x2x2 equivalents. In this way, a 3D input volume can be processed as a sequence of independent 2D regions in a single computation graph, and the 2D and 3D modules can be jointly trained end-to-end. Intermediate 2D class predictions $\hat{x}_{2D}$ are formed from the 2D module output, and the 2D output and class predictions are concatenated along the feature axis to form an input to a 3D spatial pyramid module. The 3D module applies a 1x2x2 max pool to its input to form a two-level spatial pyramid with scales 0 (input) and 1 (pooled). The pyramid elements separately pass through convolution blocks, and the scale 1 block output is upsampled and added to the scale 0 block output with a residual connection to form the module output. 3D class predictions $\hat{x}_{3D}$ are formed from the 3D module output, and the final segmentation output $\hat{\ell}$ of the algorithm is a voxelwise argmax of the 3D class predictions. To build a 2D-3D+3x3x3 network, we inserted 3x3x3 convolution layers at the beginning of the first two convolution blocks in the 2D encoder and the last two convolution blocks in the 2D decoder.

Given a collection of networks' 3D class predictions, one can form an ensemble prediction by computing a voxelwise average of the predictions and computing a segmentation from that. Ensembling high-quality but non-identical predictions can produce better predictions[24], and there is reason to think that more sophisticated ensembles could be constructed from collections of diverse neural architectures[25], but in this paper we use a simple source of differing predictions to boost performance: ensembles of identical architectures trained from different random initializations. The sources of randomness in the training procedure are examined more thoroughly in Validation and Performance Metrics, but in our experiments this variation produced a small number of high-performing network instances per architecture with partially-uncorrelated errors.

Network training

Assume a network predicting classes $C = \{0, \ldots, 6\}$ for each voxel in a shape-(o_z, o_x, o_y) data window Ω containing $N = o_z o_x o_y$ voxels $\{v_i\}_{i=1}^{N}$. The ground-truth segmentation of this region is a shape-(o_z, o_x, o_y) array ℓ such that $\ell(v) \in C$ is the ground-truth label for voxel v. A network output prediction is a shape-$(7, o_z, o_x, o_y)$ array $\hat{x}$ such that $x_v \triangleq \hat{x}(:, v)$ is a probability distribution over possible class labels for voxel v. The corresponding segmentation $\hat{\ell}$ is the per-voxel arg max of $\hat{x}$. Inversely, from ℓ one may construct a shape-$(7, o_z, o_x, o_y)$ per-voxel probability distribution x such that $x_v(i) = 1$ if $i = \ell(v)$ and 0 if not, which is useful during training.

We trained our networks as a series of experiments, with each experiment training and evaluating 1 or more instances of a fixed network architecture. Instances within an experiment varied only in the random number generator (RNG) seed used to control trainable variable initialization and training data presentation order. In addition to the main 2D-3D+3x3x3 architecture, there were three ablation experiments - No 3x3x3 Convs, No Multi-Loss, No 3D Pyramid - and five baseline experiments - Original U-Net, 3D U-Net Thin, 3D U-Net Thick, Deeplab + DRN, and Deeplab + ResNet101. Instances were trained and ranked by evaluation dataset MIoU. Experiments tracked evaluation MIoU for each instance at each evaluation point throughout training, and saved the final weight checkpoint as well as the checkpoint with highest eval MIoU. In this work we report eval MIoU checkpoints for each instance. The 2D-3D+3x3x3 experiment and its ablations trained 26 instances for 40 epochs (33k steps). The Original U-Net experiment trained 500 instances for 100 epochs (180k steps). The 3D U-Net Thin experiment trained 26 instances for 100 epochs (29k steps), and the 3D U-Net Thick experiment trained 26 instances for 100 epochs (30k steps). The Deeplab + DRN and Deeplab + ResNet101 experiments trained 1 instance each for 40 epochs (4k steps). Due to poor performance and slow training times of the Deeplab models, we deemed it unnecessary to train further instances. Networks were trained on NVIDIA GTX 1080 and NVIDIA Tesla P100 GPUs.

This subsection details the training of the 2D-3D+3x3x3 network. Baseline and ablation networks were trained identically except as noted in Validation and Performance Metrics. All trainable variables were initialized from Xavier uniform distributions. Each instance was trained for 40 epochs on shape-$(5,300,300)$ windows extracted from the training volume, and output a shape-$(7,5,296,296)$ class prediction array. The number of windows in each epoch was determined by a window spacing parameter which determined the distance along each axis between the top-back-left corners of each window, here $(2,100,100)$, resulting in 828 windows per epoch. An early stopping criterion halted the training of any network that failed to reach a mean intersection-over-union (MIoU) of 0.3 after 10 epochs.

Networks were trained using a regularized, weighted sum of cross-entropy functions. The network has a set Θ trainable variables divided into four subsets: Θ_{2D} for variables in the 2D encoder-decoder module, Θ_{3D} for variables in the 3D spatial pyramid module, the single 1x1x1 convolution variable $\{\theta_{2DP}\}$ which produces intermediate 2D class predictions $\hat{x}_{2D}$ from the encoder-decoder's 64 output features, and the single 1x1x1 convolution variable $\{\theta_{3DP}\}$ which produces the final 3D class predictions $\hat{x}_{3D}$ from the spatial pyramid's 64 output features. Predictions are compared against ground-truth labels as

$$\begin{aligned} L(x,\hat{x}_{3D},\hat{x}_{2D};\Theta) = \frac{1}{N}\sum_{i=1}^{N}\left[\mathcal{W}\otimes\mathcal{H}\left(x,\hat{x}_{3D}\right)\right]_i + \frac{c_{2D}}{N}\sum_{i=1}^{N}\left[\mathcal{W}\otimes\mathcal{H}\left(x,\hat{x}_{2D}\right)\right] \\ + \lambda_{2D}\sum_{\theta\in\Theta_{2D}}\|\theta\|_2^2 + \lambda_{3D}\sum_{\theta\in\Theta_{3D}}\|\theta\|_2^2 + \lambda_P\left(\|\theta_{2DP}\|_2^2 + \|\theta_{3DP}\|_2^2\right), \end{aligned} \tag{1}$$

where $\lambda_{2D} = 1\times10^{-4.7}$ and $\lambda_{3D} = 1\times10^{-5}$ are L^2 regularization hyperparameters for the variables in Θ_{2D} and Θ_{3D}, $\lambda_P = 1\times10^{-9}$ is an L^2 regularization hyperparameter for the predictor variables θ_{2DP} and θ_{3DP}, and $c_{2D} = 0.33$ is a constant that weights the importance of the intermediate 2D class predictions in the loss function. $\mathcal{H}(x,\hat{x})$ is the voxelwise cross-entropy function, i.e.,

$$\mathcal{H}(x,\hat{x})_v \triangleq H(x_v,\hat{x}_v) = -\sum_{j=1}^{7} x_v(j)\log\left[\hat{x}_v(j)\right] = -x_v(\ell_v)\log\left[\hat{x}_v(\ell_v)\right].$$

$\mathcal{W}$ is a shape-$(5,296,296)$ array of weights; its Kronecker product with $\mathcal{H}$ produces a relative weighting of the cross-entropy error per voxel. This weighting strategy is based generally on the approach in (Ronneberger et al., 2015)[7]:

$$\mathcal{W} = w + \mathcal{W}_{cb} + \mathcal{W}_{ep}.$$

The initial $w = 0.01$ is a constant that sets a floor for the minimum weight value, W_{cb} is a class-balancing term such that $W_{cb,i} \propto 1/N_i$, where N_i is the number of occurences in the training data of ℓ_i. W_{ep} is an edge-preserving term that upweights voxels near boundaries between image objects and within small 2D cross-sections. In (Ronneberger et al., 2015) this is computed using morphological operations. We used a sum of scaled, thresholded diffusion operations to approximate this strategy in a manner that requires no morphology information. The weight file used in this paper is available with the rest of the platelet dataset at `leapmanlab.github.io/dense-cell`.

We employed data augmentation to partially compensate for the limited available training data. Augmentations were random reflections along each axis, random shifts in brightness ($\pm12\%$) and contrast ($\pm20\%$), and elastic deformation as in (Ronneberger et al., 2015). For elastic deformation, each 800x800 $x-y$ plane in the shape-$(50,800,800)$ training data and label arrays was displaced according to a shape-$(800,800,2)$ array of 2D random pixel displacement vectors, generated by bilinearly upsampling a shape-$(20,20,2)$ array of iid Gaussian random variables with mean 20 and standard deviation 0.6. During each epoch of training, a single displacement map was created and applied to the entire training volume before creating the epoch's batch of input and output windows. Training used the ADAM optimizer with learning rate 1×10^{-3}, $\beta_1 = 1-1\times10^{-1.5}$, $\beta_2 = 1-1\times10^{-2.1}$, and $\varepsilon = 1\times10^{-7}$. Training also used learning rate decay with an exponential decay rate of 0.75 every $1\times10^{3.4}$ training iterations.

Validation and performance metrics

The performance metric used in this work is mean intersection-over-union (MIoU) between ground-truth image segmentation ℓ's 7 labeled sets $\{L_j = v \in \Omega \,|\, \ell(v) = j\}_{j \in C}$ and predicted segmentation's $\hat{\ell}$ labeled sets $\{\hat{L}_j = v \in \Omega \,|\, \hat{\ell}(v) = j\}_{j \in C}$. Given two sets A and B,

$$\mathrm{IoU}(A,B) = \frac{|A \cap B|}{|A \cup B|}.$$

Then for segmentations ℓ and $\hat{\ell}$ with their corresponding labeled sets,

$$\mathrm{MIoU}(\ell,\hat{\ell}) = \frac{1}{7}\sum_{j \in C} \mathrm{IoU}(L_j,\hat{L}_j).$$

More generally, for a subset of labels $D \subseteq C$, one can compute the MIoU over D, or $\mathrm{MIoU}^{(D)}$, as

$$\mathrm{MIoU}^{(D)}(\ell,\hat{\ell}) = \frac{1}{|D|}\sum_{j \in D} \mathrm{IoU}(L_j,\hat{L}_j).$$

Here we are concerned with MIoU's over two sets of labels: $\mathrm{MIoU}^{(all)}$ over the set of all 7 class labels, and $\mathrm{MIoU}^{(org)}$ over the set of 5 organelle labels 2-7. Our network validation metrics were $\mathrm{MIoU}^{(all)}$ and $\mathrm{MIoU}^{(org)}$ on the evaluation dataset, and $\mathrm{MIoU}^{(org)}$ on the test dataset. Test data uses $\mathrm{MIoU}^{(org)}$ because the labeled region is a single cell among several unlabeled ones, and restricting validation to the labeled region invalidates MIoU stats for the background and cell classes (0 and 1). We include eval $\mathrm{MIoU}^{(org)}$ to quantify how performance drops between a region taken from the physical sample used to generate the training data, and a new physical sample of the same tissue system.

Using this procedure, the performance of the 2D-3D+3x3x3 network was compared against three ablations and five baseline networks. The three ablations each tested one of three features that distinguish the 2D-3D+3x3x3 network in this paper from similar baselines. The first, 2D+3x3x3 No 3x3x3 Convs, replaces the 3x3x3 convolutions in the net's encoder-decoder module with 1x3x3 convolutions that are otherwise identical. With this ablation, the network's encoder-decoder loses any fully-3D layers. The second, 2D+3x3x3 No Multi-Loss, modifies the loss function in Equation (1) by removing the term involving $\hat{x}_{2D}$ but otherwise leaving the architecture and training procedure unchanged. This ablation tests whether it is important to have auxiliary accuracy loss terms during training. The third ablation, 2D-3D+3x3x3 No 3D Pyramid, removes the 3D spatial pyramid module and 3D class predictor module from the network architecture, so that $\hat{x}_{2D}$ is the network's output. Correspondingly, the loss term involving $\hat{x}_{3D}$ is removed from Equation (1).

We implemented five baseline networks by adapting common models in the literature to our platelet segmentation problem. Three of these were 2D - The original U-Net[7] as well as two Deeplab variants[10,11] using a deep residual network (DRN) backbone and a ResNet101 backbone[8], minimally modified to output 7 class predictions. The original U-Net used $(572,572)$ input windows and $(388,388)$ output windows, while the Deeplab variants used $(572,572)$ input and output windows. The two 3D networks were fully-3D U-Net variants adapted on the 3D U-Net in (Çiçek et al., 2016)[26] - 3D U-Net Thin and 3D U-Net Thick. The variants used same-padding, had three convolutions per convolution block, and two pooling operations in the encoder for convolution blocks at three spatial scales. The 3D U-Net Thin network used $(5,300,300)$ input windows and $(5,296,296)$ output windows, and pooling and upsampling operations did not affect the z spatial axis. The 3D U-Net Thick network used $(16,180,180)$ input windows and $(16,180,180)$ output windows, and pooled and upsampled along all three spatial axes.

To determine whether one architecture is superior to another, trained instances are compared with each other. However, sources of randomness in the training process induce a distribution of final performance metric scores across trained instances of an architecture, so that a single sample per architecture may be insufficient to determine which is better. While expensive, a collection of instances can be trained and evaluated to empirically approximate the performance distribution for each architecture. In this way, better inferences may be made about architecture design choices. Figure 4 shows the empirical performance distributions for the 26 trials of the 2D-3D+3x3x3 architecture and its three ablations, as well as the 26 trials of the 3D U-Net and 500 trials of the 2D Original U-Net.

A final point of comparison was drawn between algorithm performance and network performance by computing performance metrics for two independent human segmentations of a single test region. For this comparison, two trained lab members were tasked with producing segmentations of a cell from the Patient 2 dataset, and a cell from the Patient 3 dataset. The $\text{MIoU}^{(org)}$ of the segmentations were computed between the humans, to be compared with $\text{MIoU}^{(org)}$ results from networks on the test dataset.

Reproducibility

All data used in this experiment, as well as examples showing how to download and train neural networks used in this paper on the platelet segmentation task, can be found at `leapmanlab.github.io/dense-cell`.

Results

Our results are outlined in Table 1 along with Figures 3 and 4. Table 1 lists several networks and ensembles of networks along with their performance metrics: $\text{MIoU}^{(all)}$ refers to the mean IoU score across all classes, while $\text{MIoU}^{(org)}$ refers to the mean IoU score over the organelle classes only, not considering the background and cell body; *Eval* refers to the evaluation data consisting of a subset of Patient 1's data which is held-out during training; *Test* refers to the test data consisting of a single cell from patient 2. We consider performance on the test data to be the best indicator of the algorithm's performance as it shows its ability to generalize across different samples.

Our Top-4 2D-3D+3x3x3 model performs best on the test data scoring 44.6% $\text{MIoU}^{(org)}$ while the next best ensemble only achieves 42.1% $\text{MIoU}^{(org)}$. Our best model significantly improves segmentation results compared to all baseline networks. We also significantly outperform the best ensemble of 2D U-Nets, improving the test $\text{MIoU}^{(org)}$ by 20%. Our ablation analysis confirms our conjectures about the importance of 3D context input to the network, and the importance of 3x3x3 convolutions over 1x3x3 convolutions for generalization performance. The latter do not capture correlations along the z spatial dimension, likely contributing to their poorer performance. Ablation analysis also indicates that removing either the multi-loss training setup or the 3D spatial pyramid module from the 2D-3D+3x3x3 architecture carries significant performance penalties.

We have also tried using the work in[18], namely their DeepVess network, on our data;, however, DeepVess performed poorly, and learned to assign a single class (background) to the entire output patch. We believe there may be two reasons behind DeepVess' poor performance: (1) Unlike U-Net and Deeplab networks, the DeepVess network is designed with very small input patches in mind; small patches do not contain enough context for the network to distinguish between objects. (2) DeepVess's last layer consists of a fully-connected operation with a single hidden layer containing 1024 neurons, therefore any attempt to input significantly larger patches would imply increasing the number of neurons in the last layer, but fully-connected layers do not scale well and the network quickly outgrows GPU memory.

We display the output segmentations of the algorithms in table 1 in figure 3 for a qualitative assessment. We experimented with simple post-processing to remove small connected components from the segmentation (similar to the techniques used here[18]). However, we found those techniques did not improve $\text{MIoU}^{(org)}$ results and did not look better qualitatively. We believe this is because platelet cells have complex multiscale correctly-labeled regions, and removing small connected components can remove correct regions as well, especially if they are only partly labeled, or are assigned multiple labels, or are covered by multiple connected components by noisy algorithmic segmentations.

In Figure 4 we experiment with various weight initialization random seeds to determine the robustness of various models to the weight initialization scheme. In order to determine whether one architecture choice is superior to another, the outputs of different trained networks are compared with each other. However, sources of randomness in the training process (initialization of trainable weights from a Xavier uniform distribution, and the random presentation order of training data elements) induce a distribution of final performance metric scores. These scores are random variables, and a single sample per architecture may be insufficient to determine which is better. By empirically approximating the distribution for each architecture, better inferences may be made about architecture design choices. For this figure, multiple instances of the same architecture (26 for 2D-3D and fully-3D nets, 500 for

the U-Net) were trained under identical conditions, varying only random number generation seeds. The resulting distributions support the conclusions that 2D-3D networks outperform their 2D and fully-3D counterparts.

Finally, we can provide context for the $\text{MIoU}^{(org)}$ numbers for each network by comparing two human segmentations of two cells similar to the test dataset. For the first cell, the two humans had a $\text{MIoU}^{(org)}$ of 0.54 relative to one another. For the second cell, the $\text{MIoU}^{(org)}$ was 0.57. These results indicate two things: (1) The process for human segmentation in complex biological datasets may need further improvement in order to create gold-standard dense semantic segmentations for research use. (2) While algorithmic segmentation errors may differ from human segmentation errors, overall quality approaches or exceeds human labelers when applied to new regions within a volume that supplied training data, but does not yet meet human quality when generalizing to new datasets. This suggests that further work remains to boost algorithm performance, but in the near-term algorithmic suggestions may be productively integrated into human annotation tools to boost human productivity for semi-supervised labeling of large datasets.

Discussion

In this work, we argued that dense semantic labeling of 3D EM images for biomedicine is an image analysis method with revolutionary potential for systems biology. We demonstrated that while challenges exist for both human and algorithmic labelers, automated methods are approaching the performance of trained humans, and may soon be integrated into annotation software for enhancing the productivty of humans segmenting large datasets. We introduced a new hybrid 2D-3D convolutional neural network architecture and demonstrated that it outperforms baseline networks on our platelet segmentation task, and that the novel architecture features are responsible for performance improvements. We do this by training multiple network instances per architecture and comparing the resulting trained network performance distributions, in order to account for variation introduced by randomness in the training procedure. By highlighting the biomedical dense semantic labeling task, building neural architectures and ensembles to better solve this task for a challenging dataset, and suggesting a way forward for developing better algorithms to enable this method to be useful in practice, we hope to spur the research and development of techniques to make large-scale EM imaging more useful to systems biology as a field.

References

1. Denk, W. & Horstmann, H. Serial Block-Face Scanning Electron Microscopy to Reconstruct Three-Dimensional Tissue Nanostructure. *PLoS Biol.* **2**, DOI: 10.1371/journal.pbio.0020329 (2004).

2. Dunn, D. & Hull, R. Reconstruction of three-dimensional chemistry and geometry using focused ion beam microscopy. *Appl. Phys. Lett.* **75**, 3414–3416 (1999).

3. Kurita, T., Otsu, N. & Abdelmalek, N. Maximum likelihood thresholding based on population mixture models. *Pattern recognition* **25**, 1231–1240 (1992).

4. Li, P. H. *et al.* Automated reconstruction of a serial-section em drosophila brain with flood-filling networks and local realignment. *bioRxiv* 605634 (2019).

5. Lee, K., Zlateski, A., Ashwin, V. & Seung, H. S. Recursive Training of 2d-3d Convolutional Networks for Neuronal Boundary Prediction. In Cortes, C., Lawrence, N. D., Lee, D. D., Sugiyama, M. & Garnett, R. (eds.) *Advances in Neural Information Processing Systems 28*, 3573–3581 (Curran Associates, Inc., 2015).

6. Krizhevsky, A., Sutskever, I. & Hinton, G. E. ImageNet Classification with Deep Convolutional Neural Networks. In Pereira, F., Burges, C. J. C., Bottou, L. & Weinberger, K. Q. (eds.) *Advances in Neural Information Processing Systems 25*, 1097–1105 (Curran Associates, Inc., 2012).

7. Ronneberger, O., Fischer, P. & Brox, T. U-Net: Convolutional Networks for Biomedical Image Segmentation. In *LNCS*, vol. 9351, 234–241, DOI: 10.1007/978-3-319-24574-4_28 (2015).

8. He, K., Zhang, X., Ren, S. & Sun, J. Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 770–778, DOI: 10.1109/CVPR.2016.90 (2016). ISSN: 1063-6919.

9. Kirillov, A., He, K., Girshick, R., Rother, C. & Dollár, P. Panoptic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 9404–9413 (2019).

10. Chen, L.-C., Papandreou, G., Schroff, F. & Adam, H. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587* (2017).

11. Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K. & Yuille, A. L. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis machine intelligence* **40**, 834–848 (2017).

12. Redmon, J., Divvala, S., Girshick, R. & Farhadi, A. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 779–788 (2016).

13. Milletari, F., Navab, N. & Ahmadi, S.-A. V-net: Fully convolutional neural networks for volumetric medical image segmentation. In *2016 Fourth International Conference on 3D Vision (3DV)*, 565–571 (IEEE, 2016).

14. Long, J., Shelhamer, E. & Darrell, T. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 3431–3440 (2015).

15. Badrinarayanan, V., Kendall, A. & Cipolla, R. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE transactions on pattern analysis machine intelligence* **39**, 2481–2495 (2017).

16. Chen, H., Dou, Q., Yu, L., Qin, J. & Heng, P.-A. VoxResNet: Deep voxelwise residual networks for brain segmentation from 3d MR images. *NeuroImage* **170**, 446–455, DOI: 10.1016/j.neuroimage.2017.04.041 (2018).

17. Chen, S., Ma, K. & Zheng, Y. Med3d: Transfer Learning for 3d Medical Image Analysis. *arXiv:1904.00625 [cs]* (2019). ArXiv: 1904.00625.

18. Haft-Javaherian, M. *et al.* Deep convolutional neural networks for segmenting 3d in vivo multiphoton images of vasculature in Alzheimer disease mouse models. *PLoS ONE* **14**, DOI: 10.1371/journal.pone.0213539 (2019).

19. Roth, H. R. *et al.* An application of cascaded 3d fully convolutional networks for medical image segmentation. *Comput. Med. Imaging Graph.* **66**, 90–99, DOI: 10.1016/j.compmedimag.2018.03.001 (2018).

20. Wang, G. *et al.* DeepIGeoS: A Deep Interactive Geodesic Framework for Medical Image Segmentation. *IEEE Transactions on Pattern Analysis Mach. Intell.* **41**, 1559–1572, DOI: 10.1109/TPAMI.2018.2840695 (2019). ArXiv: 1707.00652.

21. Zhu, Z., Xia, Y., Shen, W., Fishman, E. & Yuille, A. A 3d Coarse-to-Fine Framework for Volumetric Medical Image Segmentation. In *2018 International Conference on 3D Vision (3DV)*, 682–690, DOI: 10.1109/3DV.2018.00083 (2018). ISSN: 2378-3826.

22. Fu, H., Xu, Y., Lin, S., Wong, D. W. K. & Liu, J. Deepvessel: Retinal vessel segmentation via deep learning and conditional random field. In *International conference on medical image computing and computer-assisted intervention*, 132–139 (Springer, 2016).

23. Stalling, D., Westerhoff, M., Hege, H.-C. *et al.* Amira: A highly interactive system for visual data analysis. *The visualization handbook* **38**, 749–67 (2005).

24. Krogh, A. & Vedelsby, J. Neural network ensembles, cross validation, and active learning. In *Advances in neural information processing systems*, 231–238 (1995).

25. Guay, M., Emam, Z., Anderson, A. & Leapman, R. Designing deep neural networks to automate segmentation for serial block-face electron microscopy. In *2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018)*, 405–408 (IEEE, 2018).

26. Çiçek, Ö., Abdulkadir, A., Lienkamp, S. S., Brox, T. & Ronneberger, O. 3d u-net: learning dense volumetric segmentation from sparse annotation. In *International conference on medical image computing and computer-assisted intervention*, 424–432 (Springer, 2016).

List of figures

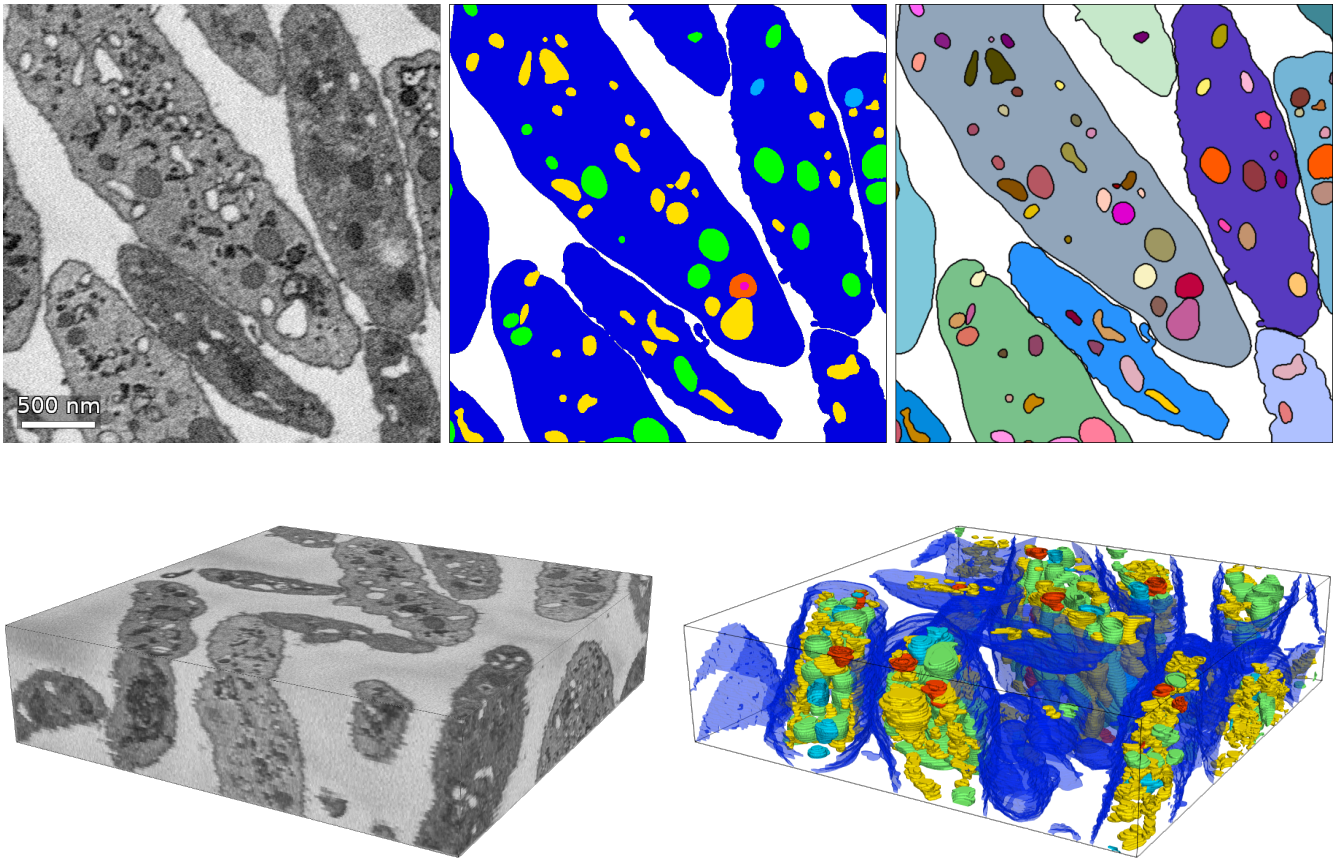
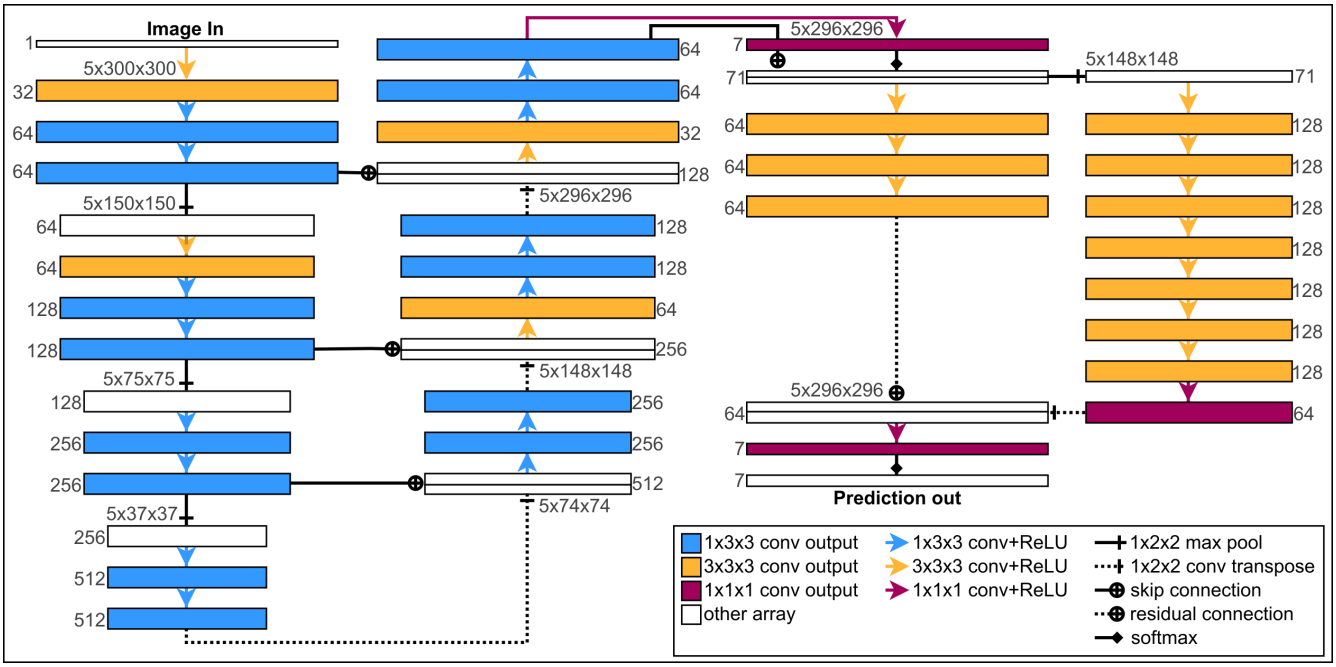


Figure 1. Overview of SBF-SEM imaging and dense cellular segmentation. (**Top**) Lateral (*x*-*y*) view of the platelet training dataset, its ground-truth segmentation into 7 semantic classes, and a digital cell model built from the semantic segmentation. (**Bottom**) 3D views of the training dataset and its ground-truth segmentation.



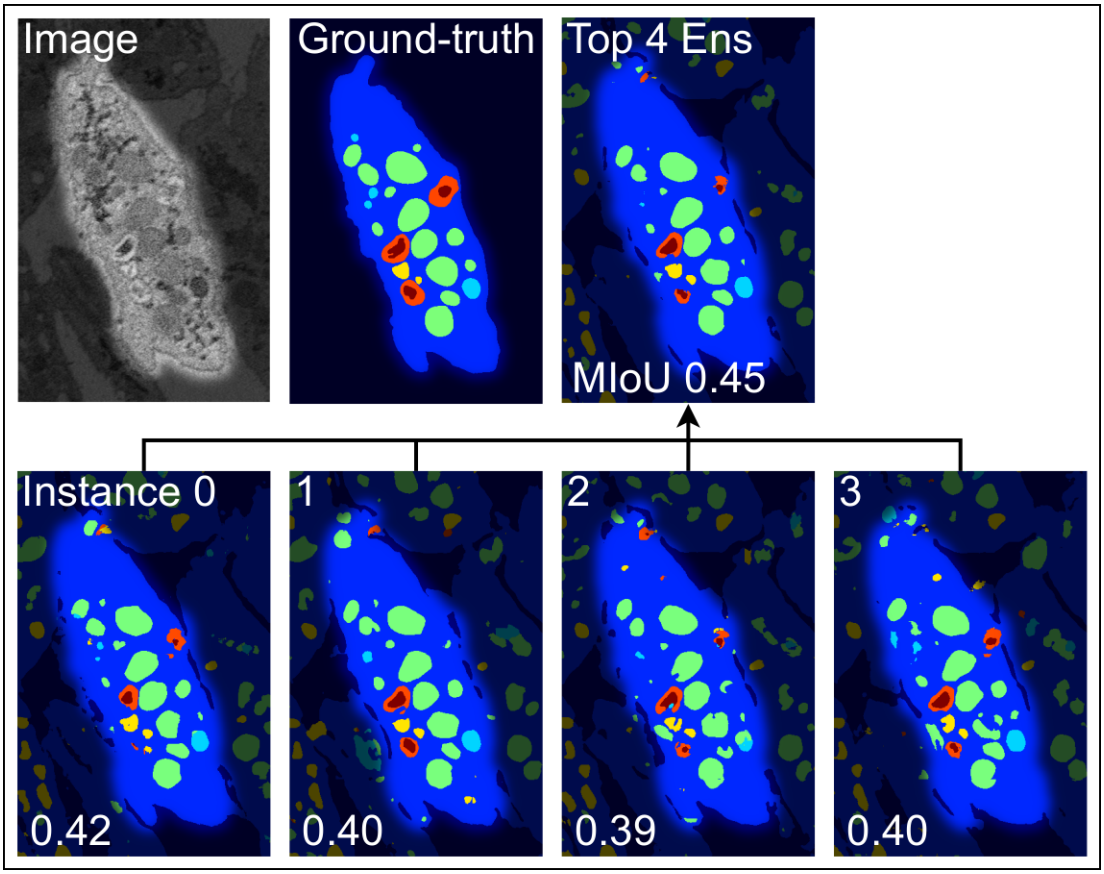


Figure 2. (**Top**) Diagram of the 2D-3D+3x3x3 network architecture. The outputs of a 2D encoder-decoder (left two columns) are connected to a 3D convolutional spatial pyramid (right two columns) to produce a prediction of per-voxel probability distributions across a 3D spatial window. The first two encoder convolution blocks and the last two decoder convolution blocks begin with a 3x3x3 convolution to add additional 3D context into the 2D initial module. (**Bottom**) Illustration of initialization-dependent performance of trained segmentation networks, and exploiting it for ensembling. In the first row, an example image of a cell and its ground-truth segmentation is compared with the result of an ensemble of the top 4 trained 2D-3D+3x3x3 network instances, which has an $\text{MIoU}^{(org)}$ of 0.45. The second row shows the segmentations from the individual instances and their $\text{MIoU}^{(org)}$ scores. The ensemble shows an 8.3% improvement in $\text{MIoU}^{(org)}$ over the best single network alone.

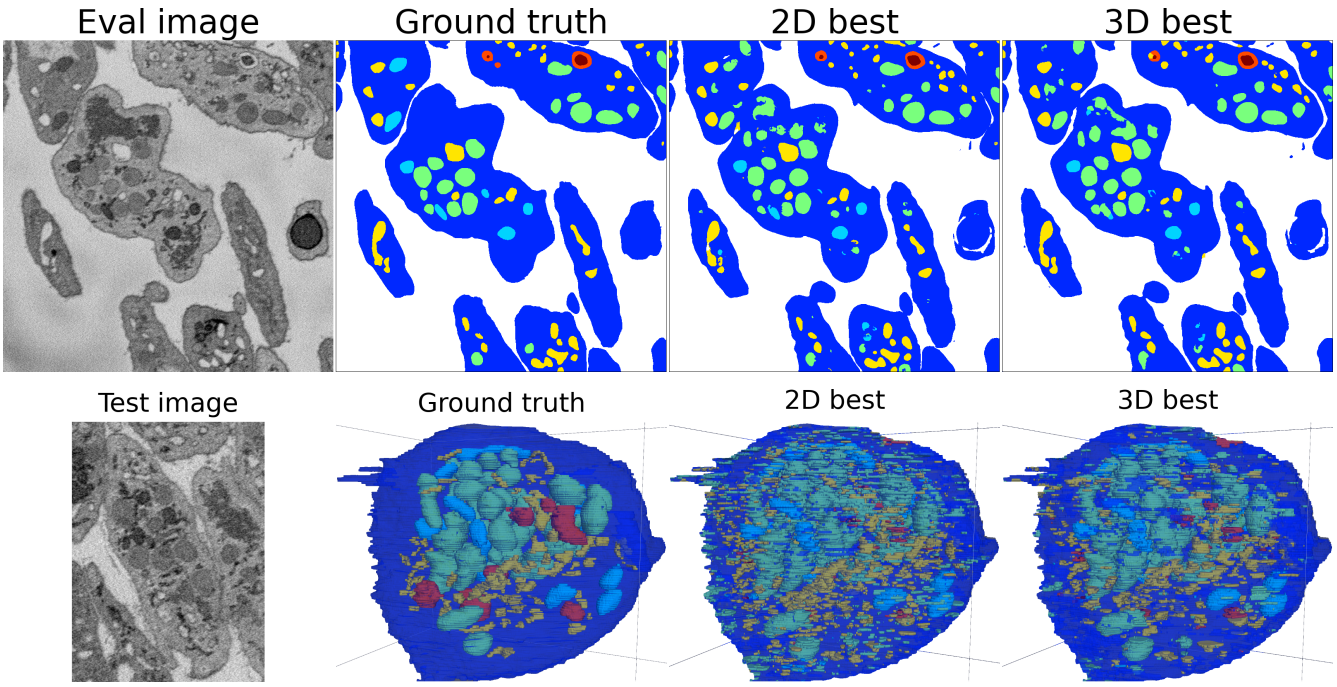


Figure 3. Visualizations of segmentations created by the best 2D and 3D ensembles. (**Top**) Comparison of segmentations on 2D cross-sections of the evaluation data volume, a subvolume from the patient 1 platelet dataset. Both 2D and 3D ensembles perform similarly on organelle segmentation, making similar mistakes in locations where organelles not within the classification schema are encountered. The 3D ensemble is better at separating cells in regions of near contact between neighbors. (**Bottom**) Comparison of 3D renderings of segmentations of the test volume, a cell from the patient 2 platelet dataset. Testing networks on the same biological system from a different physical sample helps gauge how robust they are to image variations due to preparation differences.

	Eval MIoU$^{(all)}$	Eval MIoU$^{(org)}$	Test MIoU$^{(org)}$
Top-4 2D-3D+3x3x3	0.686	0.595	**0.446**
Top-5 No 3x3x3 Convs	**0.690**	**0.601**	0.419
Top-3 No Multi-Loss	0.633	0.524	0.338
Top-3 No 3D Pyramid	0.681	0.590	0.421
Top-5 Original U-Net	0.663	0.562	0.371

(a) Ensembles of Networks

2D-3D+3x3x3 (10.3M)	0.665	0.568	0.417
No 3x3x3 Convs (9.9M)	0.667	0.571	0.358
No Multi-Loss (10.3M)	0.652	0.550	0.355
No 3D Pyramid (7.9M)	0.646	0.542	0.376

(b) Single 2D-3D+3x3x3 Network and Ablations

Original U-Net (31.0M)	0.626	0.515	0.334
3D U-Net Thick (2.1M)	0.496	0.348	0.314
3D U-Net Thin (2.0M)	0.613	0.502	0.280
Deeplab + DRN (40.7M)	0.511	0.368	0.159
Deeplab + ResNet101 (59.3M)	0.501	0.361	0.174

(c) Baseline Networks

Table 1. Segmentation algorithm results summary showing mean intersection-over-union (MIoU) across all classes (MIoU$^{(all)}$) on evaluation data and MIoU across organelle classes (MIoU$^{(org)}$) on evaluation and test data. The patient 2 dataset from which the test data is taken contains only a small number of labeled cells among unlabeled ones; we use MIoU$^{(org)}$ to measure test performance since restricting the MIoU stat to labeled regions invalidates `background` and `cell` class statistics. (**a**) Results for the best ensemble from each architecture tested. A top-k ensemble averages the predictions of the best k trained networks as judged by MIoU$^{(all)}$ on the evaluation dataset. (**b**) Results for the best single network from each architecture class. Trainable parameter counts are in parentheses. (**c**) Results from baseline comparison networks. Trainable parameter counts are in parentheses

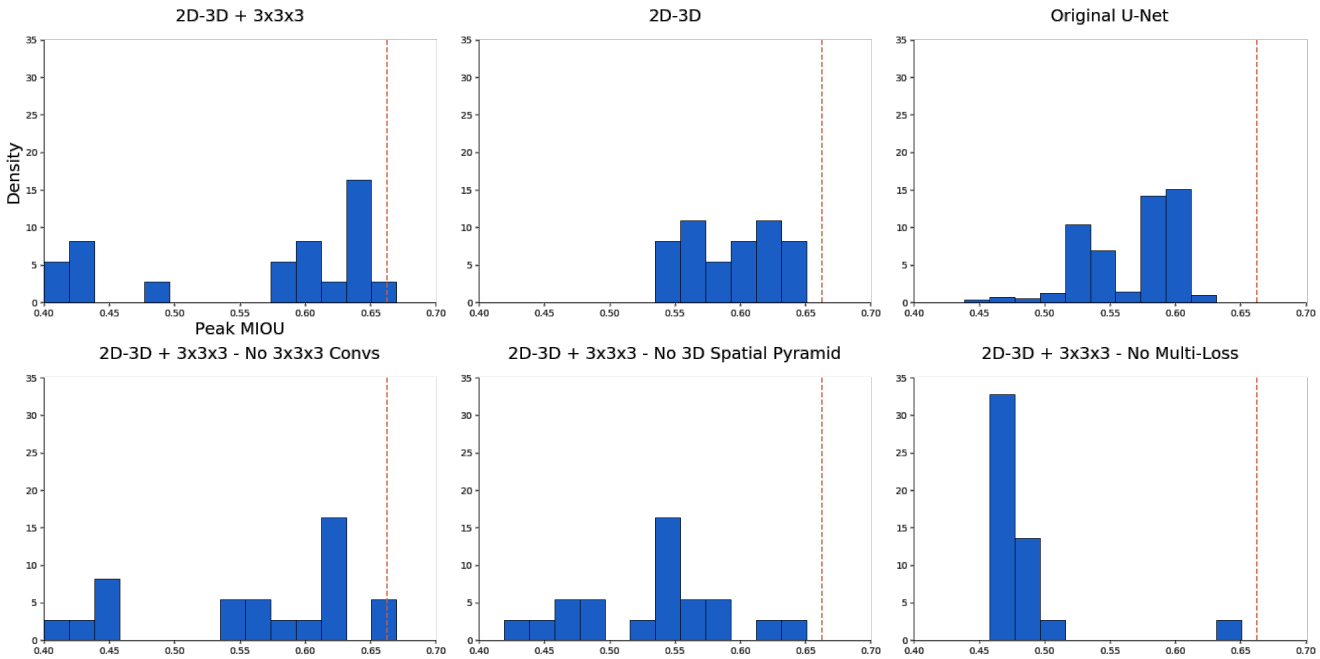
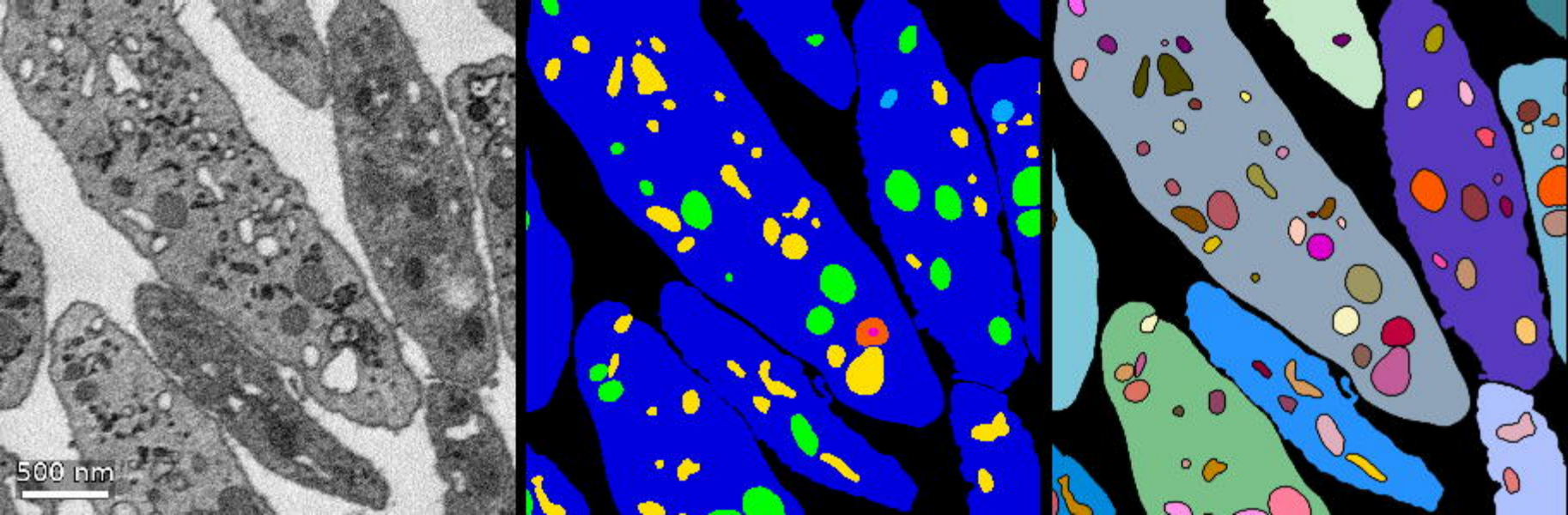
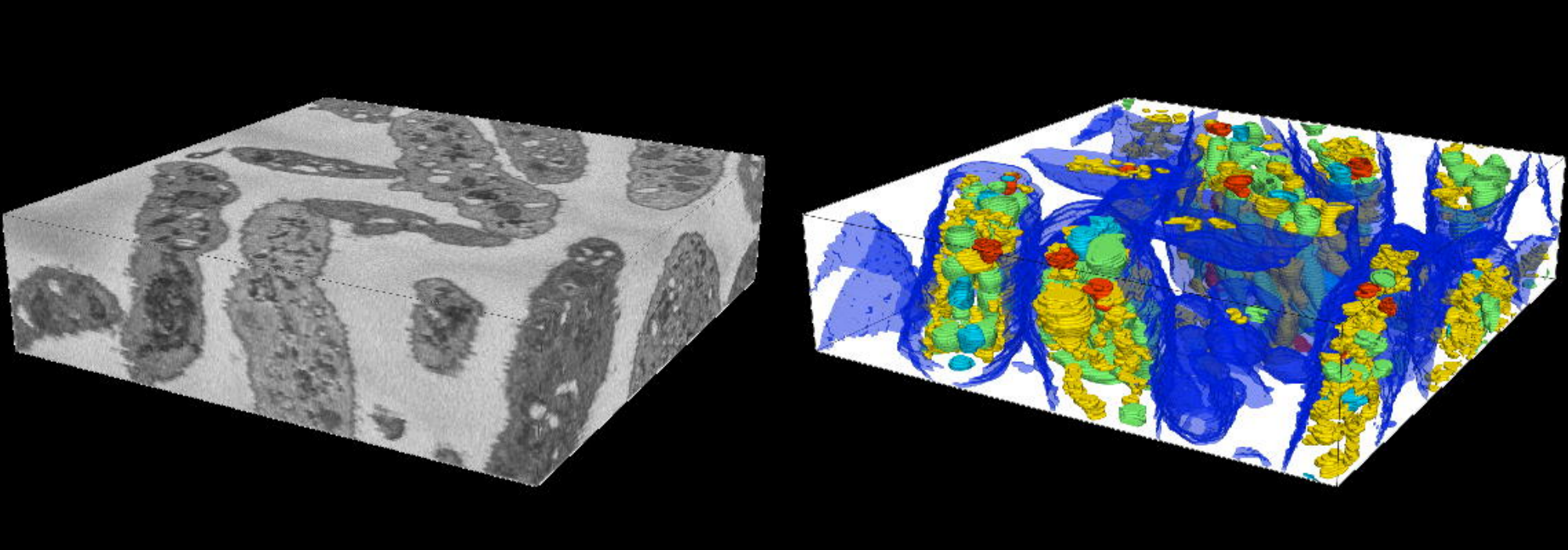


Figure 4. Normalized histograms of peak MIoU on the evaluation data volume for each of the architectures examined in this paper. In order to determine whether one architecture choice is superior to another, the outputs of different trained networks are compared with each other. However, sources of randomness in the training process (initialization of trainable weights from a Xavier uniform distribution, and the random presentation order of training data elements) induce a distribution of final performance metric scores. These scores are random variables, and a single sample per architecture may be insufficient to determine which is better. By empirically approximating the distribution for each architecture, better inferences may be made about architecture design choices. For this figure, multiple instances of the same architecture (26 for 2D-3D nets, 500 for the U-Net) were trained under identical conditions, varying only random number generation seeds. The resulting distributions support the conclusions that 2D-3D networks outperform the 2D U-Net and that multi-loss training is necessary for 2D-3D architectures.


500 nm



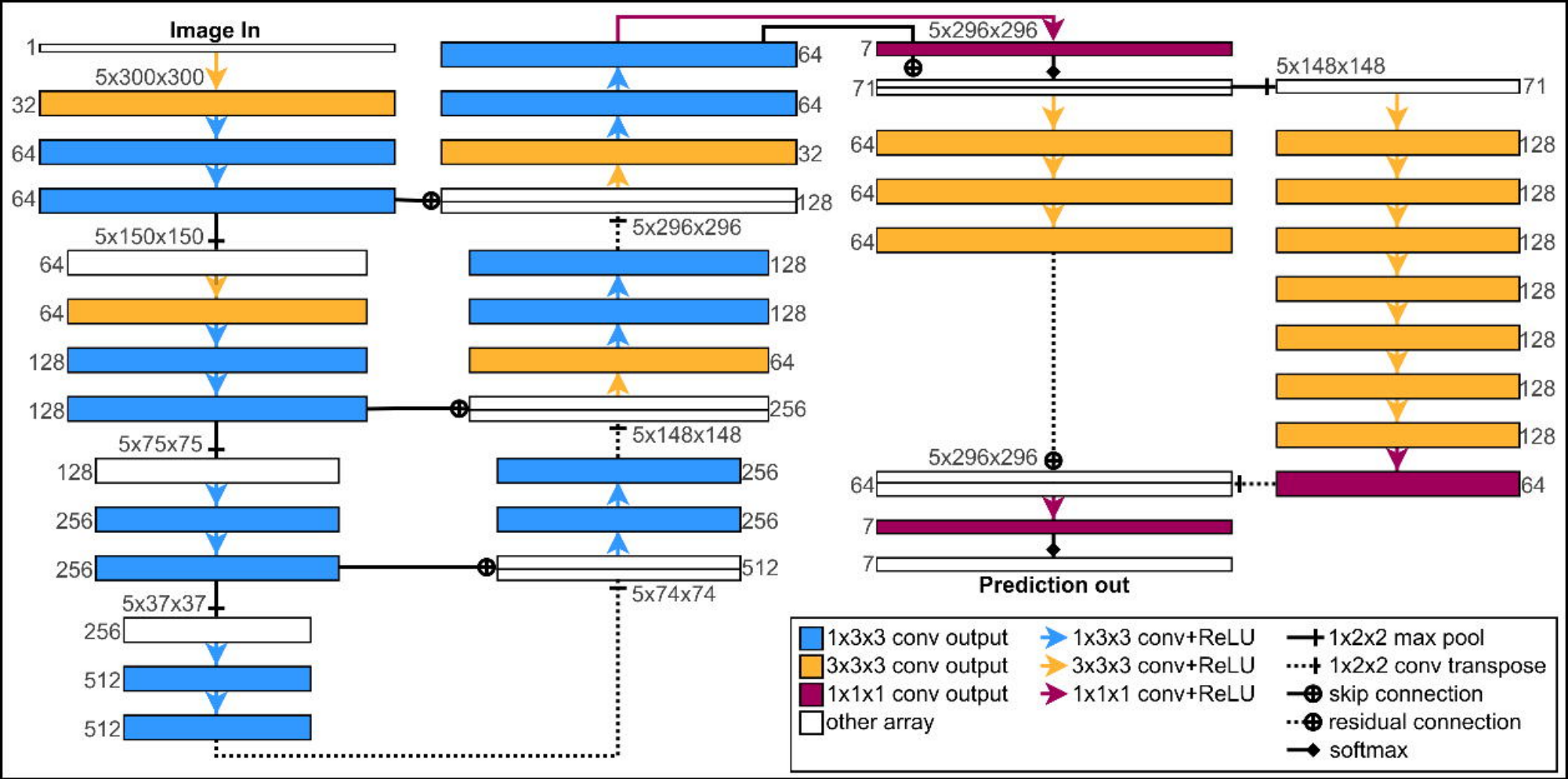


Image In
1
5x300x300
32
64
64
5x150x150
64
64
128
128
5x75x75
128
256
256
5x37x37
256
512
512
64
64
32
128
5x296x296
128
128
64
256
5x148x148
256
256
512
5x74x74
7
5x296x296
71
5x148x148
71
64
64
64
128
128
128
128
128
128
5x296x296
64
64
7
7
Prediction out
1x3x3 conv output
3x3x3 conv output
1x1x1 conv output
other array
1x3x3 conv+ReLU
3x3x3 conv+ReLU
1x1x1 conv+ReLU
1x2x2 max pool
1x2x2 conv transpose
skip connection
residual connection
softmax

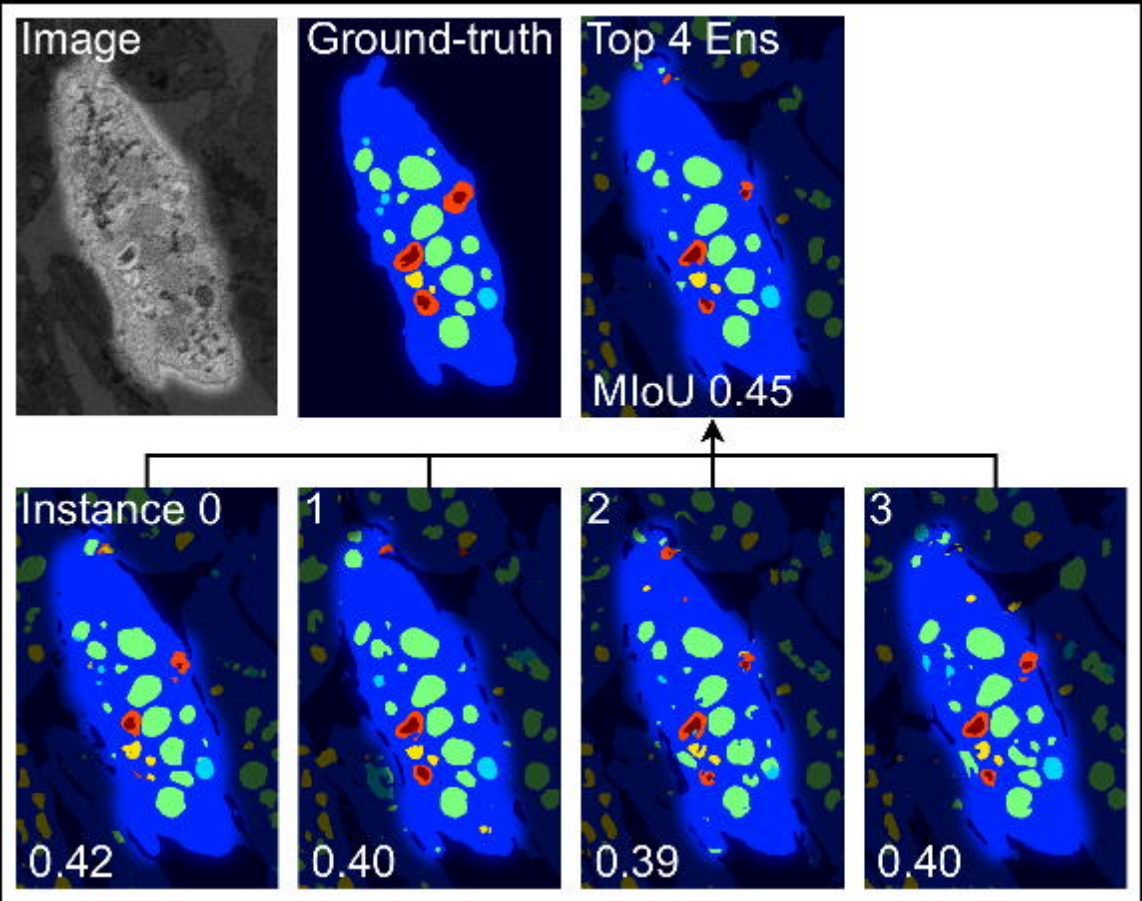
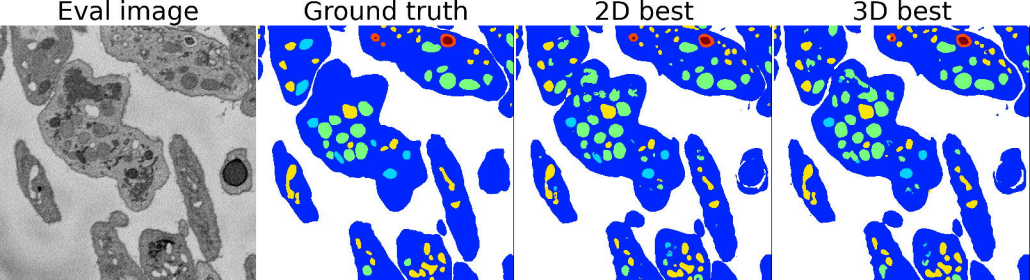
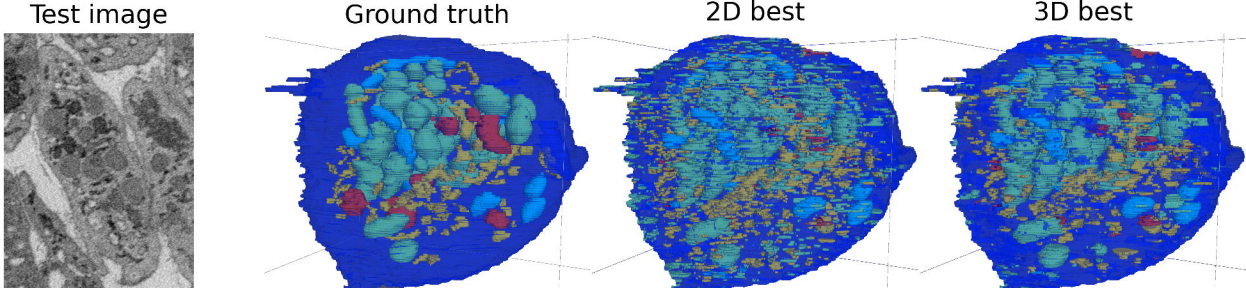


Image
Ground-truth
Top 4 Ens
MIoU 0.45
Instance 0
0.42
1
0.40
2
0.39
3
0.40


Eval image
Ground truth
2D best
3D best


Test image
Ground truth
2D best
3D best