Enabling Scalable Multicolor Connectomics Through Expansion Microscopy

by

Jeremy Wohlwend

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Engineering in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2017

Author . . . .
Department of Electrical Engineering and Computer Science
May 24, 2017

Certified by . . . .
Edward S. Boyden
Associate Professor
Thesis Supervisor

Accepted by . . . .
Chris J. Terman
Chairman, Department Committee on Graduate Theses

Enabling Scalable Multicolor Connectomics Through Expansion Microscopy

by

Jeremy Wohlwend

Submitted to the Department of Electrical Engineering and Computer Science
on May 24, 2017, in partial fulfillment of the
requirements for the degree of
Master of Engineering in Electrical Engineering and Computer Science

Abstract

Decades of work in computational power and high resolution imaging have made it possible to observe specific neurons and their connections at a microscopic level. This level of resolution is known as the connectomic level. While we have been able to observe the brain at such resolution for some time thanks to electron microscopy (EM), the analysis of the resulting data has proven to be a very challenging task, mainly because it relies only on membrane contrast to differentiate cells. Expansion microscopy (ExM) offers an exciting alternative to EM by expanding physical brain tissue, with the added benefits that regular optical microscopes can be used for image acquisition. This will enable a faster, cheaper, and multicolor approach to connectomics, and may have the potential to realistically tackle a nervous system as large and complex as a mammalian brain. Currently, the study and the development of Expansion Microscopy are limited by the rate at which experiments run, which can take weeks, and sometimes months. In a previous project, we described a pipeline to create synthetic ExM images (SimExM) and discussed the importance of ground truth data in experimenting with segmentation algorithms. One of the main bottlenecks in creating a full connectome is the need for these ground truth annotations which require a substantial amount of manual work. This work explores various reconstruction strategies based on ExM. We evaluate their dependency on manual annotations and their overall performance, and contribute, as such, to the development of a scalable connectomics pipeline.

Thesis Supervisor: Edward S. Boyden
Title: Associate Professor

Acknowledgments

Professor Edward Boyden for allowing me to work in a stimulating and exciting environment, and for asking me the right questions, at the right times. Adam Marblestone, who has been a true mentor to me the past two years. Nicholas Barry, Daniel Goodwin, Amauche Emenari, Young Gyu Yoon, David Rolnick and the rest of the Synthetic Neurobiology group, for without their help, this work would not have been possible. The MIT course 6 department for an amazing 5 years. Finally, my family, for their everlasting support.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The main focus of connectomics is to make use of bio-engineering techniques and computer power to acquire and process high resolution brain images in an attempt to produce a comprehensive map of the brain's neural circuits [21]. This can only be accomplished when observing the brain at nanoscale resolution, where it becomes possible to trace out the 3-dimensional morphology of the neurons present in the volume [12]. Knowing where the neurons reside in space allows one to learn about their types, shapes, and ultimately study what composes the brain with nanoscale precision. Most importantly, these reconstructions can generate maps of neural circuits, which play a crucial role in understanding the brain's computations. Mapping the brain could lead to unprecedented advances in neuroscience, artificial intelligence, and likely many other fields [23].

This work investigates the effects of different labeling and imaging strategies on the mapping process. The first section provides a brief overview of the state of Connectomics research and Expansion Microscopy. The second section covers SimExm, a software for simulated Expansion Microscopy experiments. The third section describes the reconstruction strategies and the model architectures. The fourth section describes our results, which are discussed in the fifth section, alongside some thoughts on the future of connectomics.

1.1 Connectomics

There is currently no scalable solution to generate precise maps of the nervous system. Most methods in the field involve electron microscopy (EM), a slow and expensive process that has been at the center of connectomic research from its birth. EM provides great resolution but has failed at scaling. For instance, one of the largest annotated datasets of EM brain images was recently published by the Lichtman Lab at Harvard, and consists of ”only” 1500 cubic microns [12]. For the rest of this paper we refer to ”segmentation” as the process of assigning each pixel in the image to a the cell it belongs to, or to extra cellular space. EM analysis relies solely on membrane contrast, which makes the segmentation difficult, even for a trained human. Results obtained through fully automated algorithms or machine learning still require intensive proofreading and, therefore, EM remains far from ideal [9, 17].

1.2 Expansion Microscopy (ExM)

Expansion microscopy offers an exciting alternative, but its application to connectomics still needs to be fleshed out [4]. ExM consists of the physical expansion of brain tissue uniformly in all directions, which allows for the use of optical microscopes for high resolution imaging, out-weighing the constraints imposed by the diffraction limit of light. The specimen is labeled pre-expansion and then embedded in a water swellable gel which binds to the label proteins in the sample. The tissue is then washed while the targeted fluorescent proteins remain bounded to the gel. The water swellable nature of the gel allows it to expand in all directions uniformly and with minimal distortion, thus improving on the resolution of the imaged sample and avoiding the limitations imposed by the diffraction limit of light. In contrast to EM, which imposes grayscale imaging, ExM makes use of light microscopy for high resolution imaging, thus adding color labeling. The use of multiple color channels should help solidify reconstruction process, and the use of cheaper and faster optical microscopes will make ExM a more scalable solution than EM.

1.3 A scalable approach?

Expansion microscopy was only published very recently, in 2015, and most work done so far has attempted to make the expansion process better, with higher magnification and minimal distortion [3]. Attempts to apply the technique to connectomics are still in the early stages. Expansion microscopy data is only available in limited quantity, and the question of whether the data is "good enough" for segmentation is often hard to answer. This makes it hard to test image processing methods, and limits progress on the computational end of connectomics research. To address this issue, we developed SimExM, a software for expansion microscopy simulations. SimExM uses annotated electron microscopy images to simulate the biological labeling and imaging process involved in Expansion Microscopy. Because it provides ground truth for the synthetic data, SimExM enables quantitative measurement of the accuracy of segmentation algorithms, and makes it possible to train supervised models, in a similar fashion to what is done in electron microscopy.

The decades of work on electron microscopy have accumulated a vast resource of processing and segmentation methods to efficiently reconstruct neuron morphologies [12, 19]. These usually rely heavily on supervised learning, and similar algorithms can be used for the analysis of ExM. However, the addition of color, which increases the amount of information in the image, may also enable methods which don't rely as strongly on training data. The best strategy remains an open question and the amount of ground truth required will be an important question to answer. In particular, obtaining such ground truth currently involves having a human expert manually annotate the images, assigning each pixel in the image to a cell, or to background. A key factor is ensuring that this amount of manual work does not become a bottleneck.

Building a scalable process thus requires finding a segmentation algorithm that only needs few training examples, as in a semi-supervised approach or develop a fully unsupervised segmentation. This work explores how different strategies impact the amount of ground truth needed, the required expansion factor, and the overall complexity of the process.

Chapter 2

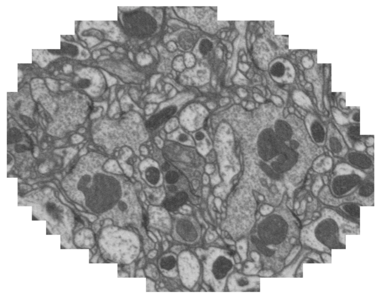
SimExM

2.1 Description

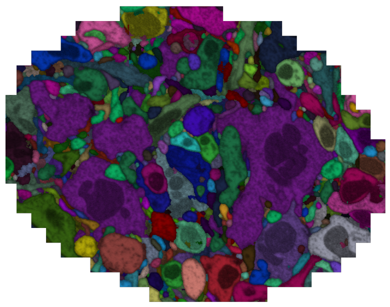
SimExM is a software, fully written in Python, which allows flexible expansion microscopy simulations, under different labeling and imaging conditions. It is designed to be highly flexible, and to reproduce the basics of the biological labeling, and optical imaging under confocal microscopy. It produces simulated image stacks, and the corresponding ground truth labels on a per channel basis. A simulation run consists of three steps: data processing, labeling and imaging. The software is described in detail in the next section, followed by a set of example use cases.

2.1.1 Data processing

SimExM takes as input segmented fluorescent or electron microscopy image stacks and distributes fluorophore proteins across the cells or cell regions present in the volume. The ground truth images must be formatted as follows: the value of each voxel in the volume is a cell id, assigning that voxel to the cell that it belongs to. An example input is shown in Figure 2-1. We then simulate the basics of what we understand of protein labeling on a per-cell basis. The data is loaded in a way that the set of voxels can be accessed from a per cell basis. SimExM also allows to specify which region of the cell to label, should the annotation be present in the ground truth.



(a) Electron Microscopy image.



(b) Ground Truth, cell annotation

Figure 2-1: Left: Electron Microscopy image with a pixel resolution of 8 nanometers. Right: Manually annotated EM image. Each neuron is labeled with a different color. The right image is an example of ground truth data, which is used as the basis for creating synthetic images. Images taken from the Janelia Group[10].

2.1.2 Labeling

One labeling approach, which can be coupled to ExM, is known as the "Brainbow" method. This is a process where cells are genetically targeted and express a single color given some uniform probability distribution over a predefined color space [2]. Simply put, different cells are likely to have different colors on their surface, and this can be simulated with flexibility in SimExM. Each fluorophore targets a portion of the cells in the volume as defined by the input labeling density. The number of fluorophores usually determines the number of channels in the output image. To accurately reproduce channel leaking, and the dominance of certain fluorescent proteins over others, SimExM uses a database of fluorophores and their attributes, storing the excitation and emission wavelengths, as well as the quantum yield and the extinction coefficient for each of the following fluorophore type: Alexa350, Alexa790, ATTO390, ATTO425, ATTO430LS, ATTO465, ATTO488, ATTO490LS, ATTO550, ATTO647N, ATTO700. The set of labeling parameters for the labeling module can be found in Table 2.1. Once the fluorescent proteins are distributed across the volume, the antibody amplification is simulated by amplifying and replacing the fluorescent proteins using a Gaussian distribution centered at their original locations.

2.1.3 Imaging

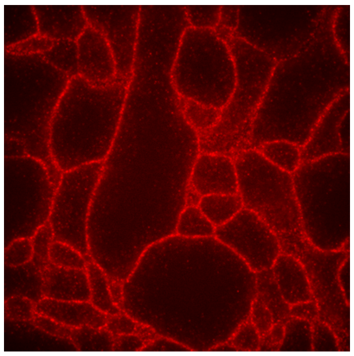
Next, the labeled volume is passed to the optics module whose role is to simulate the imaging process through optical microscopes. SimExM allows control over noise, lasers, filters, and other microscope parameters. The optics module starts by computing the mean photon count associated with each laser / fluorophore pair. The number of photons per protein is then sampled from a Poisson distribution with the above mean. Once the photon counts are grouped by channels, the module generates a theoretical point spread function and performs a 3D convolution with the location of the fluorescent proteins in the volume. The point spread function follows the approximation proposed by Richards and Wolf [22]. Finally, Gaussian noise is added to the volume, which is then normalized to the [0, 255] range. The full set of parameters for the labeling and optics modules can be found in the appendix.

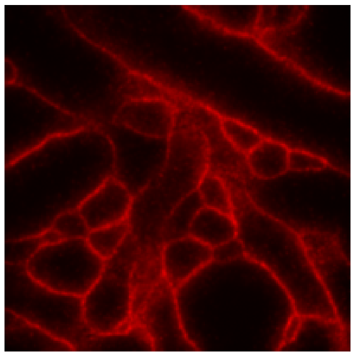
2.2 Use cases

We show here some example use cases of SimExM. The ground truth datasets used for these simulations are public and were taken from the Lichtman Lab, at Harvard University [12], through the ISBI challenge [1]. These simulated stacks are used to train and evaluate the models described in Chapter 3. The full set of parameters used in producing these stacks can be found in the appendix as well.

2.2.1 Brainbow

The following figures show the Brainbow labeling at three different levels of expansion, three degrees of sparsity and three different number of color channels. In all figures, the 20x single membrane stain is used as baseline for comparison. Figure 2-2, shows slices of the volume at 4x, 10x, and 20x expansion. Figure 2-3, shows the volume with 10%, 50% and 100% of labeled cells. Figure 2-3, shows cells labeled with one, two and three orthogonal fluorophores, respectively. All stacks use the same protein density, the same optics parameters and the same amount of baseline noise.


(a) 20X


(b) 10X

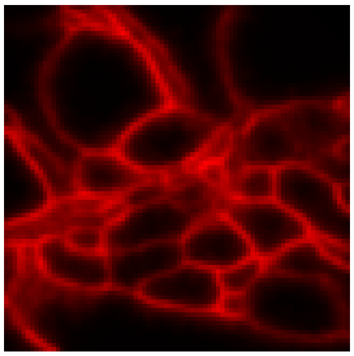
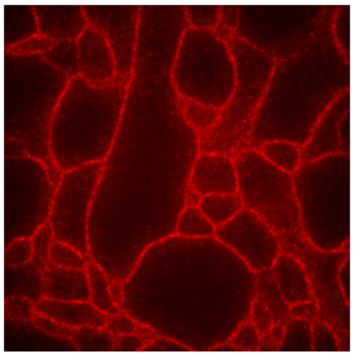
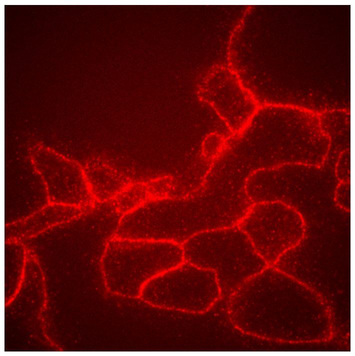

(c) 4X

Figure 2-2: Single membrane stain at different expansion levels


(a) High density


(b) Medium density

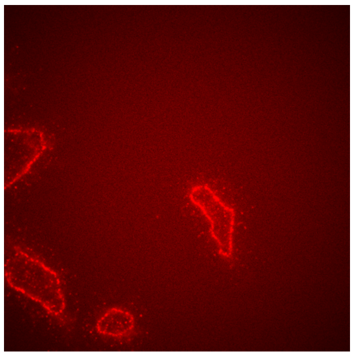
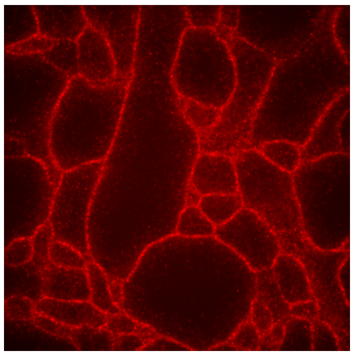
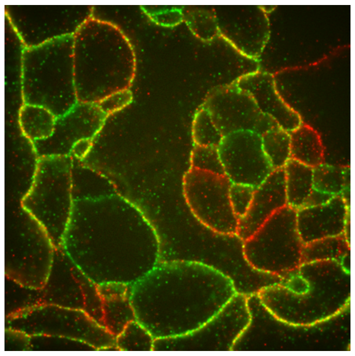

(c) Low density

Figure 2-3: Low, medium and high labeling density


(a) Single channel


(b) Two channels

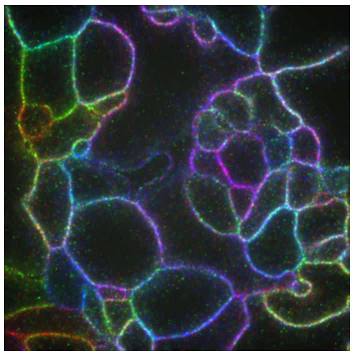

(c) Three channels

Figure 2-4: One, two and three color stains over the whole volume

2.2.2 Sparse-dense

Another interesting labeling approach is to use a different labeling density in different channels. For instance, in the two stacks below, all neurons are labeled in the first channel (red), while only a few are labeled in the second (green, once as a membrane stain and once as a cytosolic stain). These datasets are particularly interesting from a semi-supervised point of view. Because single neuron reconstruction is an easy task, one can imagine having an automated algorithm to segment a single neuron within a sub-volume, which could create training labels automatically from the sparse channel.

These object masks can then be used to train the model to produce a segmentation by training exclusively on the dense channel, in an attempt to generalize to all neurons in the volume. One possible issue with this method is the lack of negative examples for the boundary detection. However, this strategy fits particularly well with flood filling networks (see 3.1.3) as they do not focus on detecting edges, but compute whole object masks. Assuming enough cells are reconstructed in the sparse channel, it should be possible to train a model that generalizes well.

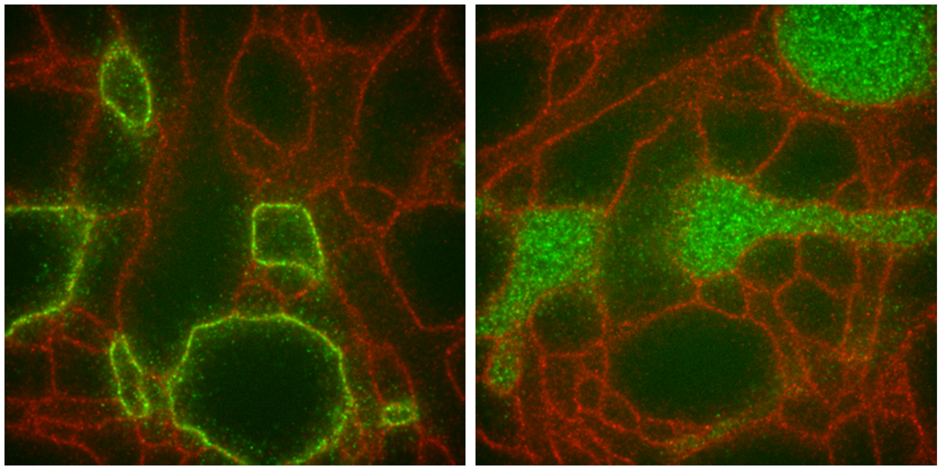


Figure 2-5: Sparse-dense datasets. The first channel (red) is a membrane stain on all cells. The second channel (green) is a membrane stain for the left image and a cytosolic stain for the right image, on a few neurons in the volume ($\approx 10\%$).

2.2.3 Rosetta Brain

Another crucial aspect of connectomics is the detection of synaptic connections between neurons, and the strength of these connections. This is difficult to estimate. With ExM, there is the hope of being able to stain synapses in an orthogonal channel, which should prove a lot more reliable than post synaptic density detection in electron microscopy images, though that remains to be tested further [3]. In fact, it may be difficult, if not impossible to segment spatially close synapses without expansion. In the figure below, we show a simulated estimate of how synapses might look if all stained with their own color, at different expansion levels. Note that this is a simulation and should be interpreted as such. From looking at the images, it seems difficult to distinguish synaptic terminals from each other below 3x or 4x expansion.

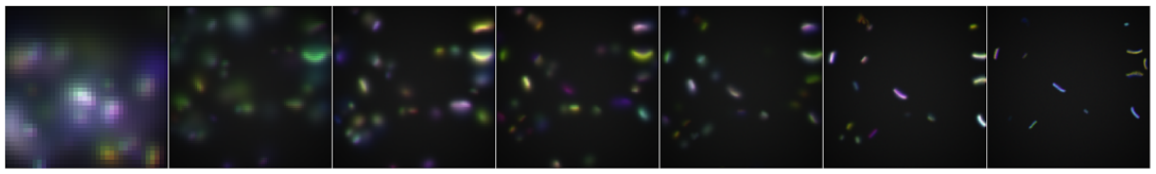


Figure 2-6: Multicolor synapse stain at different levels of expansion. From left to right: 1x, 2x, 3x, 4x, 5x, 10x, 20x.

The Rosetta Brain paper [18] explains a potential implementation of RNA bar-codes to identify cells (and potentially their synapses) using multicolor dots to label voxels to specific cells. Each cell produces a unique RNA bar-code. This bar-code is then transcribed in a way that each A, T, G, C base is associated with a different fluorescent protein (in practice the process is a little more complex and involves pairs of bases). As the bar-code is transcribed, a chemical lock is put in place to ensure that the microscope has the time to take a snapshot of the sample. This results in a sequence of images, where the bar-codes change color, effectively reproducing their RNA bar-code sequence in color space. These bar-codes can be used to uniquely identify a cell, in multiple locations. One of the current goals is to force these bar-codes to travel to the synaptic terminal to reliably identify connections between cells.

We use SimExM to simulate this process. In particular, we use 4 possible colors (as in 4 bases), and bar codes of length 15. The bar-codes take the form of 400nm dots in post-expansion space. In the following images, most bar codes target the synaptic terminals but a few are spread around the rest of cell body. Synaptic terminals are stained in yellow, while the 4 color bar-codes use the red, green, blue and magenta channels. The same image is shown at two expansion levels: 1x and 4x, and at two different steps (A, T, G, C base) in the bar-code sequence. While dots can be seen easily in both the 1x and 4x expansions, only the 4x expansion allows to distinguish between two different dots sitting at the same synaptic terminals. As real data is not currently available for this task, this work does not explore the related computational questions further.

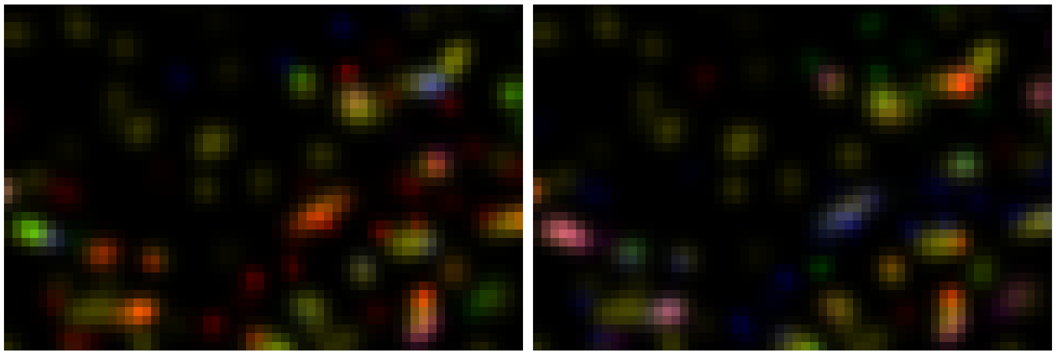


Figure 2-7: Rosetta brain at 1x expansion. Left: 1st base, Right: 2nd base

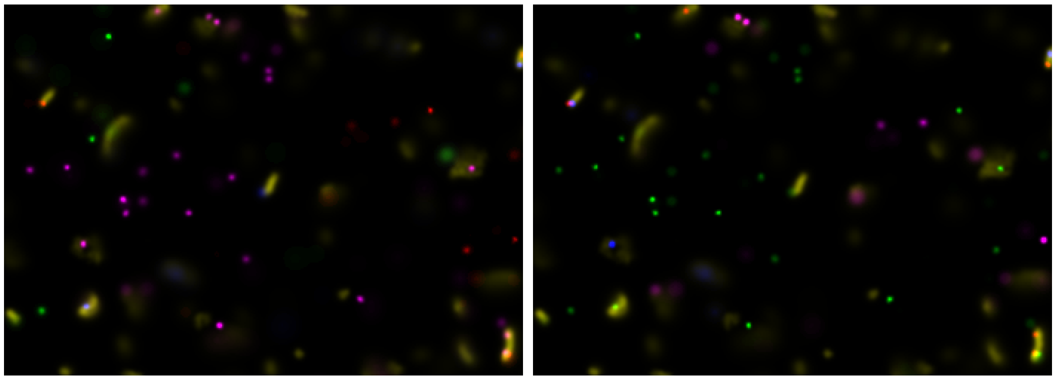


Figure 2-8: Rosetta brain at 4x expansion Left: 1st base, Right: 2nd base.

Chapter 3

Cell Morphology Reconstruction

Segmentation refers to the process of assigning each voxel in a given volume a label which indicates what neuron the voxel is on. With a segmented volume, we can then try to infer which cells are connected to each other, as well as learn about the shapes and types of neurons composing the volume. In general, most segmentation algorithms adopt one of two strategies:

1. Most segmentation approaches consist of a boundary detection step, followed by a pixel agglomeration algorithm based on these boundaries. The watershed algorithm, in particular, is used in both [6, 15].

2. More recently, there has been work in trying to build end to end models producing the segmentation directly as opposed to first detecting the cell boundaries. This is an active area of research in semantic segmentation, with the recent appearance of fully convolutional networks and autoencoder designs [16, 20, 5]. Currently, the only application of such designs to connectomics are Flood-Filling Networks [11], which are described in section 3.1 3.

Both strategies can be used with unsupervised or supervised learning. With the current state of the field, it is difficult to imagine a fully unsupervised algorithm being able to account for the stochasticity in biological samples. Using machine learning is promising but requires computational power, and a large amount of annotated data. The task is thus to find a form of data that is easily segmentable. We call *easily segmentable*, a dataset for which the ratio of reconstruction accuracy to amount of training data is high. From an unsupervised learning perspective, a dataset is considered easily segmentable if the segmentation accuracy matches more or less the segmentation accuracy obtained with supervised learning.

3.1 Supervised segmentation

The two strategies are outlines below. In the first section, we describe a convolutional neural network architecture for boundary prediction. The output of this network is then passed through a watershed to produce the final segmentation. The basics of the watershed algorithm are covered in section 3.1.2. Finally, we describe the Flood-Filling network architecture, current state of the art in connectomics [11].

3.1.1 Boundary detection

We set up the boundary detection learning task by using a pixel classifier. For some pixel p_i, the goal is to predict if p_i belongs to the same object as its neighbors, or if it doesn't (in which case it is considered a boundary pixel). The feature vector is a square window around the pixel, thus providing the local information needed in detecting edges. In short, this is a simple binary classification task. For a single pixel p_i with label y_i and predicted value a_i, the pixel-wise cross entropy loss if given by:

$$L(y_i, a_i) = -y_i log(a_i) - (1 - y_i) log(1 - a_i)$$

Note that the data is usually highly unbalanced as the number of negative examples greatly exceeds the number of positives. To account for the unbalance, we modify the loss function by adding a weight on the positive examples. For this task we use the N4 convolutional network model, from Ciresanet al. [6]. The network can be extended to account for 3D information as well [15], but we use the simple 2D case as a baseline.

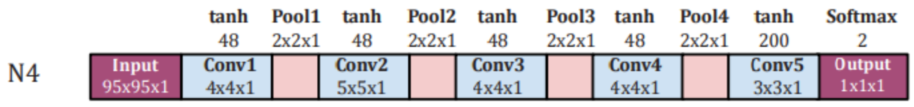


Figure 3-1: The N4 architecture. Image taken from [15]

The network consists of 4 convolutional layers, each followed by a maximum pooling layer. Then the output is flattened and passed through two fully connected layers, and a softmax activation which computes the probability of belonging to each of the two classes. In our experiments, we modify the architecture slightly by using ReLU activation functions. All weights are initialized using Xavier initialization [7]. The network is trained on mini-batches of image patch, using the Adam optimizer [13] and the loss is averaged over the mini-batch. Then, inference consists of feeding the network each pixel in the test image, thus producing a heat map of where edges are located. It is usually preferable to process the output of the network with a smoothing kernel, and simple thresholding, to remove artifacts,and facilitate the watershed.

3.1.2 Watershed Segmentation

Once the boundary prediction is obtained, it's passed on to the watershed algorithm. The watershed uses seed locations, and expands them using the gradient of the image to guide the flow of the expansion. When two super-pixels (group of pixels) collide, they are either merged together, if they hold the same label, or the expansion halts at that location.

Thus, one issue with the watershed is to determine the initial seed points – as it turns out, this problem also affects Flood filling network. Fortunately, the current work on RNA bar-codes [18], as described in section 2.2.3 has the potential to fill that gap. In our segmentations, we assume that such bar-codes are available. We run our segmentations using a single bar-code per cell per slice in a volume of size $3.2\mu \times 3.2\mu \times 0.24\mu$.

3.1.3 Flood-filling networks

Flood-Filling networks, as opposed to the previously mentioned method, produce the segmentation directly, without going through a boundary detection step. The network architecture is shown below.

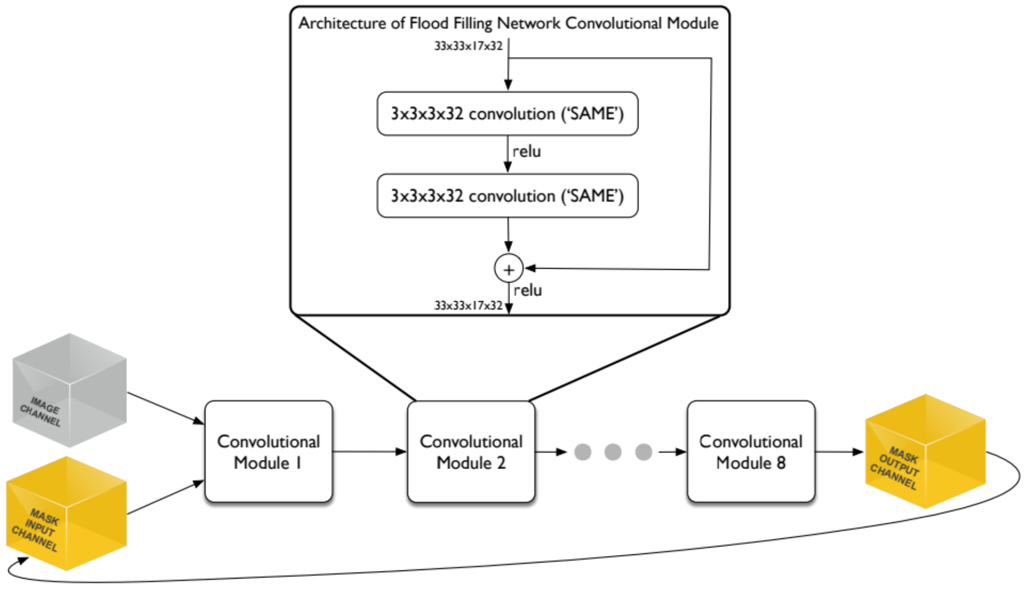


Figure 3-2: Flood-Filling Network architecture. Image taken from [11]

The network takes as input a sub-volume of data, and an object mask, and outputs an updated object mask. The target mask is a binary volume where only voxels belonging to the object covering the center voxel are given. The network tries to reconstruct objects one at a time, and the mask is updated iteratively. Computing object masks for all seed points (given at least one seed per cell) produces the final segmentation. During training, the field of view (FoV) of the network is restricted to pre-specified sub-volumes, and the loss is weighted with respect to the number of active voxels in the sub-volume. An active voxel is defined here as a voxel belonging to the same object as the center voxel in the FoV.

3.2 Sparse-dense learning

While Flood-filling networks provide great accuracy, they require a large amount of training data. We would like to tune the chemistry in a way that either removes human annotations for the pipeline or reduces them effectively. Generating gold-standard ground truth is difficult, but in a very sparse setting, with for instance a single cell in the second channel, it may be possible, depending on the quality of the data, to reconstruct the cell to gold standard quality, and use its object mask to train a flood filling network to generalize to all cells in the volume. One could even imagine using the sparse channel as the object masks directly, though the noise might be too important without any preprocessing.

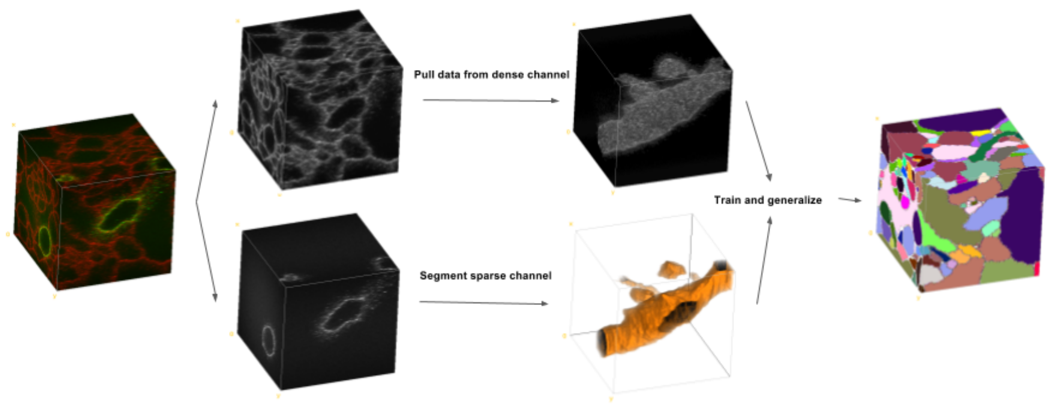


Figure 3-3: The sparse-dense learning framework.

We first produce a segmentation of the sparse channel. Then, we sample sub-volumes from the segmented object. Again we bias the network to focus on small ambiguous regions by adding a weight on the sample inversely proportional to the number of active voxels in the sub-volume. The network architecture is the same, but only the first (red) channel is fed for training and inference.

Chapter 4

Results

This section covers the segmentation results using the watershed, N4, and Flood-filling network models, on the simulated stacks from section 2. We first run three baseline experiments based on the Brainbow stacks. In particular, we consider the effect of expansion, the number of color channels, and the proportion of cells labeled in the volume on the segmentation accuracy. In our experiments we assume that we have access to at least one bar-code per cell per slice in the test volume of size $3.2\mu \times 3.2\mu \times 0.24\mu$. This would be the equivalent of having a human point at each cell in the volume, for example. All baselines experiments are run on the first two models: a simple preprocessing + watershed, and the boundary classifier + watershed, as they don't require large training times. Each use the 20x expansion, single channel membrane stain dataset as baseline for comparison with other datasets.

We then proceed to train a flood-filling network on the same 20x expansion, single channel membrane stain dataset, and compare its performance to the baseline models. Finally, we show that flood filling networks can be used in a sparse-dense learning task, where labels are created automatically from the sparse channel, given a robust single cell reconstruction algorithm, and attempt to generalize the segmentation to all cells in the volume by using a single neuron to generate training examples.

4.1 Segmentation Accuracy

Many metrics can be used to measure segmentation accuracy. For all experiments we evaluate the accuracy using two of these: the pixel error, and the rand error, the latter being more robust to small spatial shifts than the former. Given an output segmentation S, and a ground truth segmentation G, the two errors are computed as follows:

1. Pixel error: the proportion of voxels having the same label in S and in G is given by:

$$P.E \quad = \quad \frac{1}{N} \sum_{p_S, p_G \in S \times G} \mathbb{I} p_S = p_G \tag{4.1}$$

2. Rand error: we start by first counting a) the number of pairs of pixels which are in the same object in both S and G, and b) the number of pairs of pixels that are not in the same object in both S and G. The rand error is then given by:

$$R.E \quad = \quad 1 - \frac{a+b}{\binom{n}{2}} \tag{4.2}$$

4.2 Baselines

We use the standalone watershed algorithm and the N4 boundary predictor followed by the watershed as baseline models. The N4 models were trained over a 100000 examples, with a batch size of 100. The learning rate was set to 0.0001 and no regularization penalty used. The trained models were then used to segment a small test sub-volume. The predicted boundary map were passed through a 3D median filter of radius 2 pixels, and the watershed algorithm, producing the final segmentation. Due to the stochasticity in assigning initial seed locations, the results were averaged over 10 runs of the watershed for each experiment.

4.2.1 Expansion

We first look at the effect of expansion on the segmentation accuracy using the 4x, 10x, and 20x stacks, and a single channel dense membrane stain. Note that the N4 model was modified to only a reduced window size of 47 on the smallest 4x dataset due to the size of the training volume.

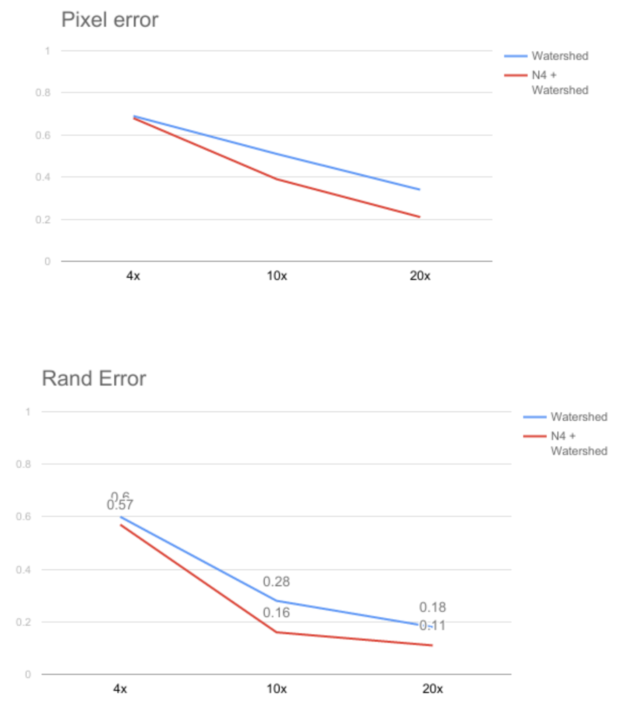


Figure 4-1: Segmentation accuracy, at 4x, 10x and 20x expansion. Top: pixel error, bottom: rand error.

As expansion increases, the pixel error drops significantly. Note however, that at 10x expansion, the rand error is almost as low as for the 20x data. Looking into the images reveals that, as expected, the 10x model fails in some of the smaller regions. Higher expansion thus improves on the segmentation accuracy, but the gain seems to diminish as the expansion factor increases.

4.2.2 Sparsity

Next, we look at the effect of sparsity on the segmentation accuracy. The low density, average density, and high density stacks contain 10%, 50% and 100% of cells labeled, respectively. The loss was adjusted in training to compensate for the drop in positive examples in the lower density stacks.

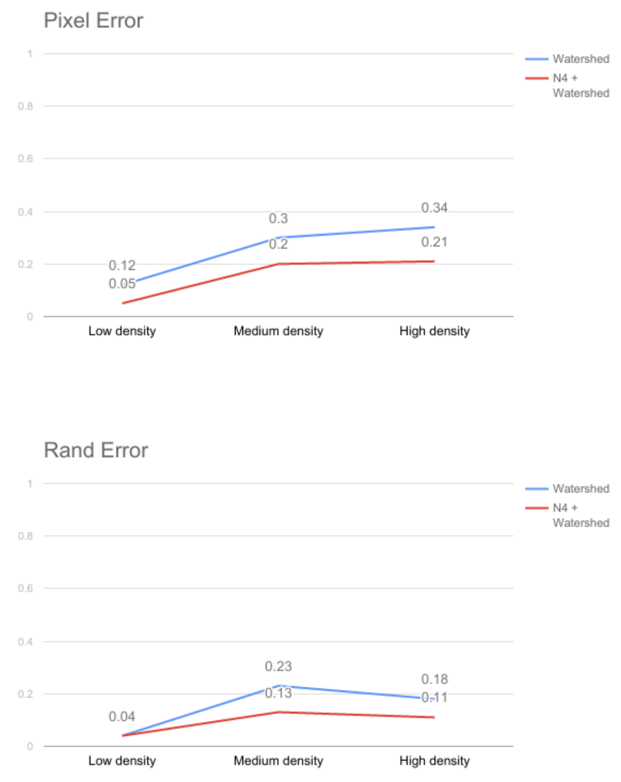
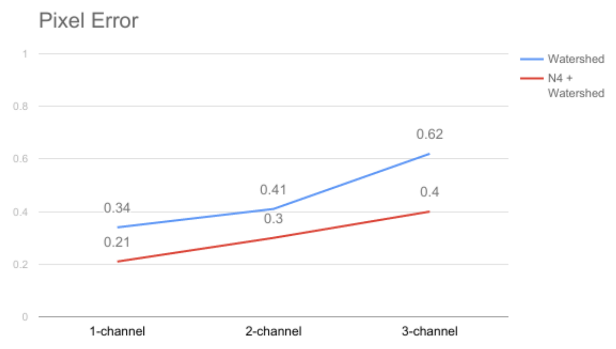


Figure 4-2: Segmentation accuracy, at low, medium and high labeling density. Top: pixel error, bottom: rand error.

Here, very low density provided close to perfect reconstruction for both the watershed and the N4 predictor. Interestingly, The models only perform marginally better on the mid-level density stack compared to the high density stack. This can be explained in part by the fact that although the potential for merging errors increase with the number of cells, having a very dense labeling also allows for more seed points which restrict how large these merging errors might be.

4.2.3 Color channels

Finally, we consider the impact of adding color channels on the segmentation accuracy. For each expansion level and each model, we compute the pixel error, rand error on 1-channel, 2-channel, and 3-channel image stacks. The networks are modified slightly to account for the multiple input channels.



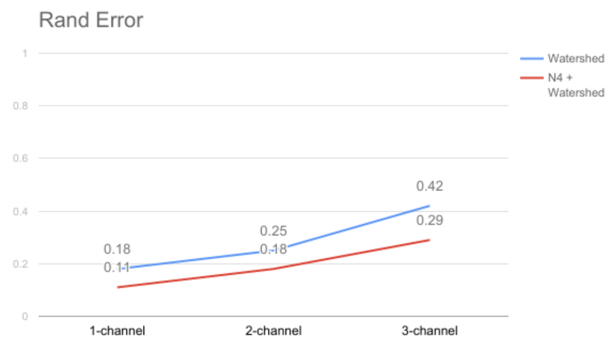


Figure 4-3: Segmentation accuracy, using 1, 2, and 3 color channels. Top: pixel error, bottom: rand error.

As the number of colors used in a stain increases, the accuracy decreases. This may be a surprising result at first but can be explained by the limited amount of gained information (at least given knowledge of bar code locations), and the non negligible amount of added noise. Adding color channels can be beneficial but repeating information in a different color does not help segmentation unless there are no bar-codes in the volume.

4.3 Flood-filling

We train the flood filling network (FFN) with asynchronous gradient descent. We use a learning rate of 0.0001 and the Adam optimizer. We train over 3000 training examples, and make 10 passes over the dataset. Each example is a sub-volume, which is explored using a smaller field of view (FoV) and the model's weights are updated after every move of the field of view. The shape of the field of view and the training sub-volumes were changed to (33, 33, 11) and (51, 51, 17) to match the anisotropy of the dataset which has resolution 8x8x25 nm.

Once training was completed, we ran the inference procedure using RNA bar-code locations as seed points. For simplicity, we represented these bar-codes as random voxels in the cell object. We removed the split decision biasing and lowered the movement threshold *tmove* from 0.9 to 0.75 after training, thus giving the network more freedom to explore the volume. We averaged the segmentation outputs over 10 runs, and take the mode value over each voxel in the volume. We also prohibited the network from writing on another cell's mask, and randomize the order in which cells are segmented. We first present results on the baseline 20x dense membrane stain dataset. We then show that the network can be trained in a sparse-dense fashion, using a single object to sample the training examples from.

4.3.1 Bar-codes

We compare the performance of the Flood filling network as the number of bar-codes per cell increases. We compare the segmentation errors using a single bar-code per cell, a bar-code per cell per slice (which would be the equivalent of having a human point at each cell in an series of images), and sparse cell filling bar-codes, where 10% of cell's voxels are bar-coded. We run the model on a testing volume of shape $(400 \times 400 \times 17)$ allowing a single FoV move in the Z-direction. We then evaluate the segmentation on the same 10 slices as the previous sections. Figure 4-4 shows the pixel and rand errors on the various bar-code initializations. Figure 4-5 shows the output segmentations for the first slice in the testing volume for the best model.

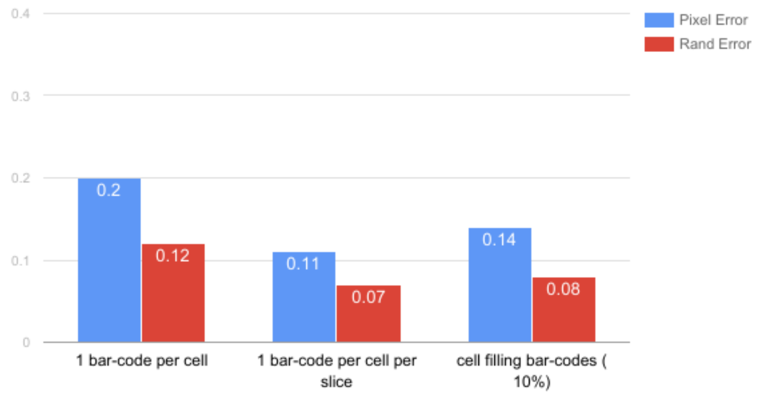


Figure 4-4: Segmentation error for the three bar-code initializations.

We obtain our best accuracy when using a bar-code for each cell and each slice in the volume, and observe a significant improvement on previous models. Using a single bar-code per cell caused some for some cells not to be explored at all. We believe that this is mainly results from the small number of examples used in training (3000). Given more time, and computational power, we expect the model to perform significantly better, especially when using few seed locations. Adding more bar-codes beyond the one per slice did not improve the results.

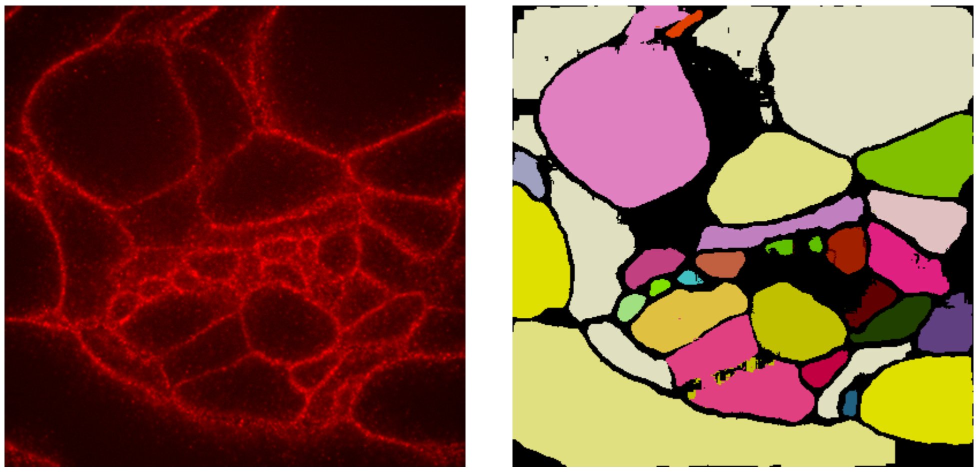


Figure 4-5: Left: the raw image. Right: the segmentation produced by the FFN.

In most places the segmentation is very accurate. However, we notice a few cells missing from the segmentation. This could be a result of a bad initial bar-code location, and the small number of training examples, especially in smaller regions.

4.3.2 Sparse-dense

Next we use a single neuron in the volume as training data, and assume that this data can be easily reconstructed from the sparse channel. In practice, the sparse channel is likely to have degraded signal but the hope is that in a sparse enough environment (ex: single neuron), the amount of signal needed to reconstruct the cell accurately is small. Furthermore, manually annotating a sparse dataset is considerably less time consuming than annotating a dense stack. The model can therefore be be trained easily in both a semi-supervised and a fully unsupervised fashion.

In our experiment, we use the largest cell in the volume in the sparse channel, ensuring that we have enough training examples, given the small size of the dataset. The model was trained over 3000 examples for 10 successive rounds, and evaluated over the same (400 x 400 x 17) sub-volume used above. The segmentation error for all four models discussed in this paper are shown in Figure 4-6, for comparison. As we saw earlier, the FFN performs significantly better than the other models, producing a rand error of 0.07 with only 3000 training examples, which is considerably less than the number of examples used in training the N4 model, though that increase in training examples is compensated by a simpler and quicker model. Thus, although the number of training examples differed across models, all were trained roughly for the same amount of time.

The sparse-dense network produced a pixel error of 0.13 and a rand error of 0.11, only slightly worse than the regular FFN. This is a very promising result, and there is hope that the accuracy of the FFN can be matched by training the network for longer and with a few more cells. It is particularly encouraging in that, even with a few examples, the sparse dense network still performs significantly better than the N4 boundary predictor, state of the art in the field back in 2012.

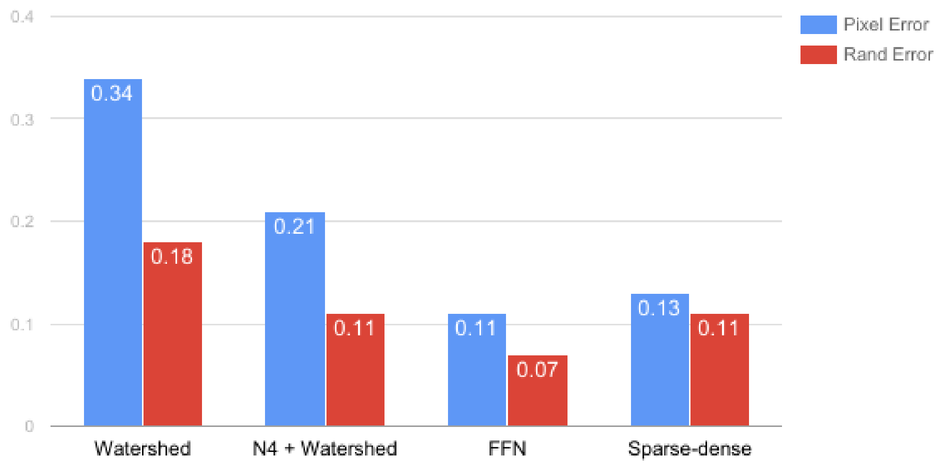


Figure 4-6: Segmentation error on four models: simple watershed, N4 + watershed, FFN, and sparse-dense FFN.

Chapter 5

Discussion

While obtaining good segmentation results on simulated data do not necessarily imply similar performance on real data, it can be a good indicator of the effects of some of the most basic data transformations, such as expansion or sparsity. A large part of this work is indeed motivated by the lack of annotated expansion microscopy images, as simulations form a convenient baseline before experimenting on real datasets. We discuss these baseline results, and their implication in the following sections.

5.1 Expansion

The first experiment shows that expansion is necessary in obtaining robust segmentations, but that as expansion increases, the noise to signal ratio increases as well. For instance, in our experiments, the models performed fairly similarly on the 10x and 20x datasets. The 4x dataset, however, resulted in a large drop in performance. Expansion is increasingly important in smaller regions, which is crucial to connectomics, and why 20x should be preferred. The Rosetta brain experiment in section 2 underlines the importance of expansion further. Under 4x expansion, it seems difficult, if not impossible, to resolve synaptic terminals, and while individual synapses are better seen at 4x expansion, the 20x image ensures that the pre and post synaptic terminals can be identified. As another example, the recent paper on iterative expansion microscopy shows synaptic terminals under 20x expansion [3].

The argument for 20x expansion also derives from electron microscopy. Recent studies [12, 15, 11] use voxel resolutions of 6x6x30, 8x8x40 and 10x10x20 nanometers respectively. The question of whether it can be done at lower resolutions is open, but the pragmatic answer is that 20x is likely to be successful, not only because of its very high resolution, but also because considerable effort has already been put into tuning the procedure. In fact, given a 40x objective, 6500nm output pixels, 500nm focal depth and 20x expansion, one achieves a resolution of:

$$\frac{6500}{40 \cdot 20} \times \frac{6500}{40 \cdot 20} \times \frac{500}{20} \quad = \quad 8 \times 8 \times 25 \; nm \tag{5.1}$$

which matches EM resolution. Higher levels of expansion may be beneficial but as the amount of data increases as the cube of the expansion factor, there is a trade off in computational complexity. An average male human brain has a size of 1260 cubic centimeters which is the equivalent of $1260 \cdot 10^{21}$ cubic nanometers. Assuming we image a whole brain at 8×25 nm resolution, and that each voxel contains at least 4 bytes, imaging a whole brain would require more storage than is available in all hard drives currently in the world combined. Once a segmentation is obtained, however, the data can be stored as simple skeletons, which significantly reduces the space complexity of the process.

Marblestone et al. [17] argue that mapping a mouse brain with electron microscopy would be a multi billion dollar project, that could span over many years, or even decades. Confocal microscopy may help in reducing the time and expenses, but there is little doubt that mapping a whole mammalian brain will require a large computational and financial effort, regardless of the method. Another interesting approach would be to image the sample at different resolutions, using 20x expansion only on regions that may be more difficult to segment. For instance the RNA barcodes could be resolved at 4x expansion, before the second round of expansion, and their computed coordinates could be upsampled to match the shape of 20x volume.

5.2 Sparsity

The proportion of cells labeled inside a volume has a large impact on *segmentability*. While a very low density usually results in very accurate reconstruction, very dense volumes, when coupled with RNA bar-codes can turn out to be easier to segment, as they offer less room for large merge errors. In our experiments, the models performed similarly on the dense dataset than on the mid density, confirming the hypothesis. The low density dataset (with roughly 10% of cells labeled) resulted in a close to perfect segmentation on all models. We thus consider the sparse dataset easily segmentable as the supervised algorithm did not improve on the results obtained with a simple watershed.

One strategy, which particularly underlines the advantages of using a sparse dataset, is the sparse-dense learning framework described in section 3.2. Flood filling networks can be trained on the object masks computed from the sparse channel. However, this raises the question of how sparse to make the second channel. Accuracy is essential if we try to create ground truth labels from the sparse channel, thus favoring lower densities. However, there is the risk that not enough training data can be sampled from the sparse channel or that these examples aren't comprehensive enough, resulting in a model that fails to generalize. In that case, we may need to stain additional cells, while ensuring that the performance remains the same. We compared the distribution of training example in terms of the their proportion of active voxels and found that the samples taken from a single neuron more or less match their distribution over the whole volume. However, further research on a real dataset will be needed to confirm this hypothesis and tune for the right degree of sparsity.

In our simple experiment, the sparse dense model is able to perform almost as well as the model trained on the whole volume. While these results need to be verify on a real dataset of the same kind, it is promising to see that the model is able to generalize on a per cell basis. One could even imagine targeting specific cells through heuristics, in an attempt to form a training dataset that is a better representation of the underlying distribution of the sub-volumes in the sample.

5.3 Color channels

Our experiments also showed that adding color channels does not improve segmentation accuracy if the added channels fails to contribute new information about cell morphologies. In fact, adding redundant color channels resulted in worse performance due to the added noise. Using orthogonal color channels is one the main benefits of ExM, but these are only useful if the information is "orthogonal" as well. One question, which will be addressed in future work is whether adding a cytosolic stain, or an extra cellular stain can help segmentation, by correcting for errors in the other channels, or if the gain in information is too low to account for the added noise.

Another interesting question regarding color channels, is the number of orthogonal fluorescent proteins required in the different segmentation frameworks. From these experiments, we believe that using a sparse-dense 20x sample coupled with RNA bar-codes would be the most viable strategy from both the computational and biological point of views. This would require 2 channels for the membrane stains, 4 channels for the bar-codes, and up to 1 or 2 other channels for to stain synaptic terminals. This would require in total of 7 or 8 orthogonal channels, which may turn into an interesting problem of itself.

Chapter 6

Conclusion

We studied in this work the application of Expansion Microscopy (ExM) to connectomics, analyzed computational bottlenecks and identified a scalable strategy for segmentation based on sparse-dense learning and RNA bar-codes. We showed that factors such as sparsity, expansion or the number of color channels have a large impact of segmentation accuracy. Adding redundant color channels, for instance, does not help segmentation, while still increasing the noise in the data. Expansion increases accuracy but faces reduced signal quality as the sample grows, resulting in an log-like curve. We also made interesting observations when varying the degree of sparsity, showing that a dataset with average density can be harder to segment than a dense dataset. We applied three models in our experiments including a boundary predictor followed by a watershed, and the recent Flood-Filling networks (FFN). We showed that FFN's perform particularly well and that the model fits in the sparse-dense framework, which could help reduce the amount of human labor significantly. From the results of our experiment, we feel that ExM offers great promises from a computational point of view, and expect important advances in the next few years, as the chemistry around bar-codes and expansion gets further calibrated.

6.1 Summary of Contributions

This work contributes to the study of the application of Expansion Microscopy (ExM) to connectomics. In particular, the main contributions of this thesis are:

- A software: SimExM for flexible simulations of ExM experiments. The software is public, open source and can be found at github.com/jwohlwend/SimExm.
- An implementation of two state of the art supervised models for cell segmentation: the N4, and Flood-Filling networks. The FFN implementation is also public and can be found at github.com/jwohlwend/Flood-Filling-Networks.
- A study of the effects of expansion, sparsity, and adding color channels on segmentation accuracy, using synthetic images and baseline models
- A description, and initial results of a semi-supervised sparse-dense learning framework, which can help reduce the amount of required human annotations.

6.2 Future work

There are two main avenues for future work. First, there is a crucial need of real data on which to run experiments. Simulations can be insightful but their results are meaningless without real data to run on. In particular, we would like to create a 20x sparse-dense dataset with a full dense red channel and a very sparse green channel containing one or very few neurons. This dataset would be used for single cell reconstruction first, and the generated labels would then be used to attempt sparse-dense learning. Secondly, there is still work to be done in improving Flood filling networks, and the current state of the art in image segmentation. In particular, we would like to further investigate the use of RNA bar-codes, and determine if a fully unsupervised approach is feasible or, alternatively, if using only the bar-codes as labeled data would be sufficient to produce robust segmentations based on other unsupervised learning methods such as variational autoencoders, or generative adversarial networks [14, 8].

Appendix A

SimExM Parameters

Table A.1: Simulation Parameter List

Labeling Parameter	Unit	Range
Labeling density	Percentage	0 - 1
Probability of infection	Probability	0 - 1
Antibody amplification	Dimensionless	1 - ∞
Fluorophore noise	Percentage	0 - 1
Fluorophores	Fluorophore object	From list
Expansion Parameter	Unit	Range
Factor	Dimensionless	1.0 - ∞
Optics Parameter	Unit	Range
Laser wavelengths	Nanometers	300 - 800
Laser powers	MilliWatts	40 - 60
Laser percentages	Percentage	0- 100
Filters	Nanometers	300 - 800
Baseline noise	Percentage	0 - 100
Exposure time	Seconds	0 - ∞
Numerical aperture	Dimensionless	0 - 2.0
Refractory index	Dimensionless	1.0 - 2.0
Objective efficiency	Percentage	0 - 1.0
Detector efficiency	Percentage	0 - 1.0
Focal plane depth	Nanometers	300 - 800
Objective factor	Dimensionless	1 - 100
Pixel size	Nanometers	1 - ∞
Pinhole Radius	Micrometers	0 - ∞

Table A.2: Simulation stacks parameters. Bold font highlights the parameters that were varied in the experiments.

Labeling Parameter	Unit	Value
Labeling density	**Percentage**	**0.1, 0.5, 1.0**
Probability of infection	Probability	1.0
Antibody amplification	Dimensionless	10.0
Fluorophore noise	Percentage	0.4
Fluorophores	**Fluorophore object**	**ATTO488, ATTO550, ATT0700**

Expansion Parameter	Unit	Value
Factor	**Dimensionless**	**4, 10, 20**

Optics Parameter	Unit	Value
Laser wavelengths	**Nanometers**	**488, 550, 700**
Laser powers	MilliWatts	50
Laser percentages	Percentage	0.25
Filters	**Nanometers**	**[438, 538], [500, 600], [650, 750]**
Baseline noise	Percentage	100
Exposure time	Seconds	0.1
Numerical aperture	Dimensionless	1.15
Refractory index	Dimensionless	1.33
Objective efficiency	Percentage	0.8
Detector efficiency	Percentage	0.6
Focal plane depth	Nanometers	500
Objective factor	Dimensionless	40
Pixel size	Nanometers	6500
Pinhole Radius	Micrometers	0.5

Bibliography

[1] I Arganda-Carreras, HS Seung, A Vishwanathan, and D Berger. 3d segmentation of neurites in em images challenge–isbi 2013, 2013.

[2] Dawen Cai, Kimberly B Cohen, Tuanlian Luo, Jeff W Lichtman, and Joshua R Sanes. Improved tools for the brainbow toolbox. *Nat Meth*, 10(6):540–547, 06 2013.

[3] Jae-Byum Chang, Fei Chen, Young-Gyu Yoon, Erica E Jung, Hazen Babcock, Jeong Seuk Kang, Shoh Asano, Ho-Jun Suk, Nikita Pak, Paul W Tillberg, et al. Iterative expansion microscopy. *Nature Methods*, 2017.

[4] Fei Chen, Paul W. Tillberg, and Edward S. Boyden. Expansion microscopy. *Science*, 347(6221):543–548, 2015.

[5] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs. *arXiv preprint arXiv:1412.7062*, 2014.

[6] Dan Ciresan, Alessandro Giusti, Luca M Gambardella, and Jürgen Schmidhuber. Deep neural networks segment neuronal membranes in electron microscopy images. In *Advances in neural information processing systems*, pages 2843–2851, 2012.

[7] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Aistats*, volume 9, pages 249–256, 2010.

[8] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[9] Moritz Helmstaedter, Kevin L Briggman, and Winfried Denk. High-accuracy neurite reconstruction for high-throughput neuroanatomy. *Nat Neurosci*, 14(8):1081–1088, 08 2011.

[10] Janelia Group, John Hopkins University.

[11] Michał Januszewski, Jeremy Maitin-Shepard, Peter Li, Jörgen Kornfeld, Winfried Denk, and Viren Jain. Flood-filling networks. *arXiv preprint arXiv:1611.00421*, 2016.

[12] Narayanan Kasthuri, Kenneth Jeffrey Hayworth, Daniel Raimund Berger, Richard Lee Schalek, Jos Angel Conchello, Seymour Knowles-Barley, Dongil Lee, Amelio Vzquez-Reina, Verena Kaynig, Thouis Raymond Jones, Mike Roberts, Josh Lyskowski Morgan, Juan Carlos Tapia, H. Sebastian Seung, William Gray Roncal, Joshua Tzvi Vogelstein, Randal Burns, Daniel Lewis Sussman, Carey Eldin Priebe, Hanspeter Pfister, and Jeff William Lichtman. Saturated reconstruction of a volume of neocortex. *Cell*, 162(3):648 – 661, 2015.

[13] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[14] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[15] Kisuk Lee, Aleksandar Zlateski, Vishwanathan Ashwin, and H Sebastian Seung. Recursive training of 2d-3d convolutional networks for neuronal boundary prediction. In *Advances in Neural Information Processing Systems*, pages 3573–3581, 2015.

[16] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3431–3440, 2015.

[17] Adam H Marblestone, Evan R Daugharthy, Reza Kalhor, Ian D Peikon, Justus M Kebschull, Seth L Shipman, Yuriy Mishchenko, David A Dalrymple, Bradley M Zamft, Konrad P Kording, et al. Conneconomics: the economics of large-scale neural connectomics. *Biorxiv*, page 001214, 2013.

[18] Adam H Marblestone, Evan R Daugharthy, Reza Kalhor, Ian D Peikon, Justus M Kebschull, Seth L Shipman, Yuriy Mishchenko, Je Hyuk Lee, Konrad P Kording, Edward S Boyden, et al. Rosetta brains: A strategy for molecularly-annotated connectomics. *arXiv preprint arXiv:1404.5103*, 2014.

[19] Yuriy Mishchenko. Automation of 3d reconstruction of neural tissue from large volume of conventional serial section transmission electron micrographs. *Journal of Neuroscience Methods*, 176(2):276 – 289, 2009.

[20] Hyeonwoo Noh, Seunghoon Hong, and Bohyung Han. Learning deconvolution network for semantic segmentation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1520–1528, 2015.

[21] Jeffrey M. Perkel. Life science technologies: This is your brain: Mapping the connectome. *Science*, 339(6117):350–352, 2013.

[22] B. Richards and E. Wolf. Electromagnetic diffraction in optical systems. ii. structure of the image field in an aplanatic system. 253(1274):358–379, 1959.

[23] S. Seung. *Connectome: How the Brain's Wiring Makes Us Who We Are.* Houghton Mifflin Harcourt, 2012.