UNIVERSITY OF CALIFORNIA SAN DIEGO

3Nsemble: Improved Electron Microscopy Image Segmentation Performance with Stacked Generalization

A thesis submitted in partial satisfaction of the requirements for the degree of Master of Science

in

Electrical Engineering (Intelligent Systems, Robotics and Control)

by

Shubha Bhaskaran

Committee in charge:

Professor Mark Ellisman, Chair
Professor Pamela Cosman, Co-Chair
Professor Ramesh Rao

2020

University of California San Diego

2020

DEDICATION

I dedicate this thesis to my family and close friends whose support and encouraging words made this possible. There are not enough words to describe the love that I have for my parents who were there with me to celebrate all the small wins and navigate the tough situations this year has put us in. My parents constantly offered their unwavering support by listening to my research banter and providing their inputs. My amazing sister spent countless hours tirelessly working with me to ensure I had the resources that I needed to move forward and be ready for my defense in time. My brother in law and niece have been so patient with me as I rain checked jam sessions and play dates. My cousin Vasav provided me with his invaluable input on the technical aspects of my work, spent time hearing my thoughts and helped me organize my results for presentation. I am incredibly fortunate to have received all this support and I look forward to spending well-deserved time with my family.

I want to thank my friends like Swetha and Kokila who spent their precious time imparting some of their domain knowledge to help me polish and gain confidence in my work. I also want to thank my dear friends Sammy, Meena, Winnie, Shagun, Roumen, Steve and Naveen who were always ready to talk and provide a well needed break from the research frenzy.

I am forever grateful to Prof. Mark Ellisman and Dr. Matthias Haberl for the opportunity to work on such exciting research. My success is due to their immeasurable support and feedback.

EPIGRAPH

It is the harmony of the diverse parts, their symmetry, their happy balance; in a word it is all that introduces order, all that gives unity, that permits us to see clearly and to comprehend at once both the ensemble and the details.

Henri Poincaré

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

I sincerely thank Professor Mark Ellisman for his support as the chair of my committee. His guidance proved indispensable in understanding the significance and extent of this research and it was a great pleasure to be able to work with him.

I thank Professor Pamela Cosman and Professor Ramesh Rao for their support as committee members and for their insight and feedback which was crucial to my success.

Additionally, I would like to express my gratitude for Dr. Matthias Haberl, without whom this research would not have materialized. His support even from miles away was immeasurable.

I would also like to recognize NCMIR and my fellow EM Analysis research mates who have been wonderful to work with and have provided me with valuable input and inspiration throughout this journey.

This thesis is co-authored with Dr. Matthias Haberl and Prof. Mark Ellisman. The thesis author was the primary author of this paper.

ABSTRACT OF THE THESIS

3Nsemble: Improved Electron Microscopy Image Segmentation Performance with Stacked Generalization

by

Shubha Bhaskaran

Master of Science in Electrical Engineering (Intelligent Systems, Robotics and Control)

University of California San Diego, 2020

Professor Mark Ellisman, Chair
Professor Pamela Cosman, Co-Chair

Deep neural networks are widely successful for many tasks of image analysis, including image segmentation. Ensemble strategies are generally used with deep neural networks not only to enhance the performance but also to improve the robustness of predictions. In particular, robustness is currently a limiting factor for image segmentation networks. Here we propose 3Nsemble which uses stacked generalization with a trained meta-classifier to improve image segmentation of Electron Microscopy (EM) image data. This research, using neurobiology data, has shown highly accurate automated segmentations of organelles that greatly benefits the study of connectomics and moves us closer to understanding the brain and brain disorders.

We compare prediction performance of a trained meta-classifier against simple averaging. The additional costs of training and applying the meta-classifier is outweighed by the benefit of improved performance. The results show improvement in performance metrics with the trained predictions and most notably saw at least a 12% increase in Intersection Over Union (IOU) score.

Introduction

As part of the BRAIN initiative, extensive work has been done to further the understanding of the human brain and brain disorders [1]. A major milestone in achieving this, is to be able to recreate a complete map of the mouse brain [2]. As a result, expert scientists are working towards producing highly accurate imaging and segmentation tools to map the connections of the brain. This is no small feat, as there are numerous challenges to imaging and understanding the brain as put forth by Jeff W. Lichtman and Winfried Denk [3]. Consequentially, researchers in the field need more robust and accessible computational tools to analyze information-rich and diverse data without resorting to manual labeling. Additionally, keeping such tools freely available online enables scientists, who do not have the computational resources, easy access to quick segmentation of 3D microscopy data.

These challenges put forth have propelled the development of more accurate, efficient and robust tools for analyzing microscopy data. A recently developed cloud based application, CDeep3M, makes highly accurate image segmentation widely accessible to the scientific community for multiple microscopy modalities [4]. This work proposes improvements to the existing CDeep3M prediction method.

Key terminology that is used throughout the following sections is defined below for clarification and convenience:

- **Segmentation:** Labeling of objects in image data to extract meaning and aid in a variety of applications.

- **Ensemble Learning:** A broad machine learning technique that combines diverse models to enhance prediction performance and robustness. There are many ensemble techniques and strategies that are discussed briefly in sections 1.5 and 1.6.

- **3Nsemble:** An improved version of CDeep3M that uses a trained meta-classifier to produce better predictions.
- **Meta-classifier:** Used in ensemble systems to combine individual model hypotheses to form final classification or prediction.
- **Averaged prediction:** Segmentation produced by using an averaging strategy to combine predictions in an ensemble system.
- **Trained prediction:** Segmentation produced by using a trained neural network to combine predictions in an ensemble system.
- **Anisotropic:** Data that has varying in resolution in different axes.
- **Frame (fm):** A 2D image that makes up one plane in a volume stack of images.

Chapter 1

Background

This research is an interdisciplinary exploration of methods to further enhance our analysis tools in neuroscience and computer vision. In this study, we propose improvements to image analysis methods of microscopic images. In order to understand the research problem at hand, we must explore topics in neurobiology, microscopy imaging, image segmentation, and deep learning using ensemble learning.

Although the methods described in this paper apply to all types of images in general, this research was conducted using Electron Microscopy data specifically.

1.1 Electron Microscopy

Electron Microscopy (EM) is an imaging method that utilizes an electron beam to capture high resolution images of biological and non-biological samples. EM imaging is particularly useful in neuroscience for imaging the brain and is one of the imaging techniques that provides the highest resolution imaging with increasingly fast acquisition times.

The development of 3D EM techniques make it possible to visualize blocks of tissue samples. Volume EM (VEM) allows scientists to process large volumes of data while retaining dense low level sub-cellular information. This is very useful in the area of connectomics where the goal is to understand and model the structural and functional interconnections of the brain [5]. VEM data has the resolution needed to image synapses and synaptic vesicles involved in the transmission of neuronal messages. This low level information is crucial to understanding the higher level structure and behavior of neurons. [6]. In combination with the ability to process large volumes of data, VEM plays a forefront role in the journey to understand the brain.

There are two main types of EM, Transmission EM (TEM) and Scanning EM (SEM). TEM uses a projection of transmitted electrons passed through a thin sample to create the image. On the other hand, SEM uses a raster scanning pattern over the sample surface and produces the image from reflected electrons. Each type has it's own strengths and weaknesses which need to be considered when choosing a method for a particular scientific investigation. Specifics on the different types of EM and existing imaging technology are detailed in Briggman and Bock's article [7]. In this study the datasets were obtained using SEM imaging.

1.2 Challenges with Electron Microscopy Datasets

EM data can face certain challenges that need to be addressed in relation to our study on computational image analysis methods. The biggest issue is the need for automated analysis tools to efficiently work with the increasing amount of image data due to high-throughput VEM [8, 6]. Additionally, a lot of existing EM data is fundamentally complicated because of the anisotropic property of image stacks [9, 6]. This property stems from the serial section imaging approach in which the x-y resolution is much higher than in the z plane where the slices occur. This impacts 3D segmentation since structures can change shape and size between slices that is not captured in the image. Thus it is a challenge to accurately segment the continuation of structures between layers. Despite recent advancements in technology that aim to solve this problem, the majority of existing data suffers from this property. Additionally, accessibility to cutting edge technology that addresses this issue is difficult for the majority of research labs.

Another challenge is in the nature of preparing samples, which does not guarantee a standardized signal to noise ratio (SNR) or intensity range in the images. This means that the segmentation must be robust to varying noise and intensities between individual sample volumes.

There is also the importance of retaining structural integrity during imaging so as to get the most accurate representation of the sample in the images. Images of sub-cellular structures are especially difficult to segment because texture and intensities can be very similar for different structures [10].

Current research focuses on minimizing these challenges via improved imaging and reconstruction methods. Kornfeld does an excellent job of taking stock of recent advances in VEM that are bringing us closer to modeling complex neural systems. He details the evolution from hybrid expert-machine approaches to the now highly successful Flood Fill Networks (FFN) [11]. Of interest to us is the ensemble of Convolutional Neural Networks (CNNs) to address a key aspect in connectomics: segmentation.

1.3 Image Segmentation

Image segmentation is the task of separating and labeling different parts of an image to extract meaningful information. In this work we focus on semantic segmentation where we treat all instances of the object as one class. Since the goal of connectomics is to map the neuronal structures and circuits of the brain at a sub-cellular level, methods to detect and label structures is crucial. These structures are difficult to segment since they are densely packed and their image textures are very similar [10]. Manual labeling and reconstruction is possible for very simplistic neural systems; however, with increasing complexity such methods become futile, thus motivating the need for highly accurate automated segmentation tools.

The challenge with developing segmentation tools for information-dense images is two-part. The first is generating highly accurate 2D segmentations of each image plane. This can be done appropriately using trained deep learning networks. The second part involves using these 2D segmentations to interpolate between layers and reconstruct 3D structures. This is typically performed by techniques such as region growing or flood filling networks [12, 13]. These techniques primarily use deep learning to produce highly accurate, automated segmentations.

To address the aforementioned challenges in VEM data and 3D segmentation, researchers developed a hybrid 2D-3D deep learning model [6]. This hybrid model strategically combines 2D and 3D contextual information for each image plane. In fact, this strategy has achieved near human level performance for 3D neurite segmentation. Recent systems which have adopted this approach show significant improvement in prediction accuracy [14, 15]. One major result of this work is the creation of CDeep3M.

1.4 CDeep3M

CDeep3M is an open source program that employs deep learning techniques to perform 2D segmentation on VEM data. This program is available with pretrained weights for multiple microscope modalities and organelles, and aims to streamline the process of automated image segmentation for all researchers. It consists of three networks that utilize a different number of image planes of the sample volume to produce independent probability maps that are combined to compute the final segmentation [4]. This unification of diverse networks, also known as an ensemble system, broadens the applicability of CDeep3M to various segmentation tasks and imaging modalities.

1.5 Ensemble Learning

Ensemble learning is a powerful and widely used approach that is based upon a very intuitive idea: leverage the strengths of individual networks to improve final predictions. This divide and conquer approach is beneficial since perfect data does not exist. There are numerous types of ensemble learning, most notably Bagging, Boosting and Adaboost. Polikar's IEEE article provides a thorough explanation of ensemble learning strategies and their advantages [16].

In his article, Polikar explains that the success of ensemble systems lies in the diversity of classifiers used and the method of combining each classifier's prediction. Diversity in the classifiers can be obtained in a variety of ways, for example using different architectures or different feature vectors for training. Likewise, there is no single method for combining the classifier outputs. Polikar refers to this component of the ensemble system as the combination rule, which can be non-trainable or trainable. A commonly used deterministic combination rule is to simply compute the mean or a weighted average; however, there may be a trained rule that is better at learning the optimum combination of classifiers. Nonetheless, ensemble methods have shown a lot of success in situations when there is too much or too little data or when the problem at hand is quite complex. The reason for the success of ensemble learning can be understood through the statistics involved in machine learning algorithms.

In traditional learning algorithms the goal is to find the single best approximation of an unknown function that fits the data out of a set of possible hypotheses. There are some drawbacks to this single solution approach that can be classified into the statistical, computational or representation problem [17]. The statistical problem deals with the case in which a large space of possible hypotheses increases the probability of multiple solutions having the same accuracy, leading to a algorithm with high variance. The computational problem deals with computational variance in which the available computation methods are not guaranteed to find the best solution every time. The representation problem arises when there is not a good solution to the problem in the space, i.e. the network suffers high bias. These problems are visualised in Figure 1.1.

To prevent these situations, ensemble learning finds a set of independent hypotheses. Then depending on the implementation, the best hypothesis is chosen or combined with the other top hypotheses to produce the final solution. This strategy improves the bias and variance of the overall learning algorithm.

The choice of combination rule used can make a significant difference in the final predictions depending on the nature of the dataset and task. Currently CDeep3M utilizes an averaging strategy which weighs each network's output equally. The next section details the ensemble strategy and trained combination rule that is the basis of this study.

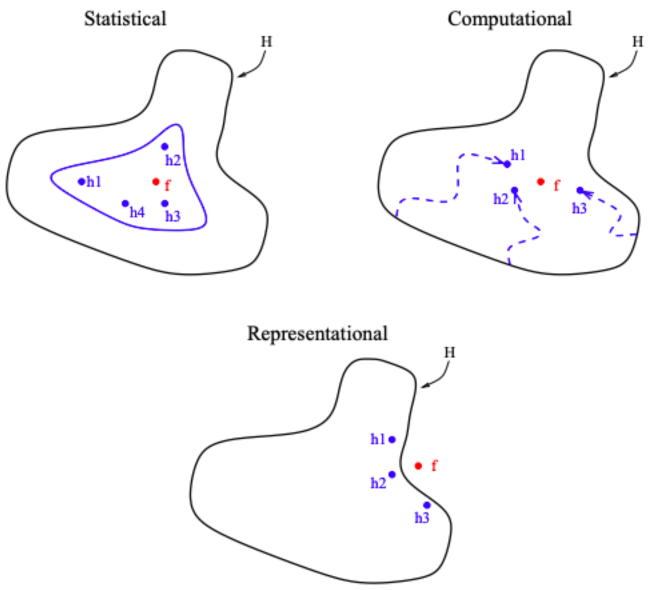


Figure 1.1. Visualization of statistical problems encountered in traditional machine learning algorithms [17]

1.6 Stacked Generalization

Stacked generalization or ”Stacking” is an ensemble technique used to combine the results of multiple classifiers into a final prediction. This technique was originally put forth by Wolpert in 1992 with the key idea of generalizing a model’s prediction accuracy by ”stacking” levels of neural networks [18].

The first level, level 0, consists of the base classifiers which are trained on different sets of the training data. Level 1 is the meta classifier which uses cross validation to train on the level 0 hypotheses. The mathematical background of how this works is explained thoroughly in Wolpert’s paper. This strategy of training on the level 0 probability maps leads to an unbiased final prediction since the meta-classifier never sees the original data [16, 19]. A useful diagram detailing this technique is shown in Figure 1.2.

Previous studies of stacking has shown that the best results are obtained when the meta-classifier uses class probabilities as it’s training data rather than classification labels [20]. The success of this method in preventing overfitting and increasing generalization accuracy has led to

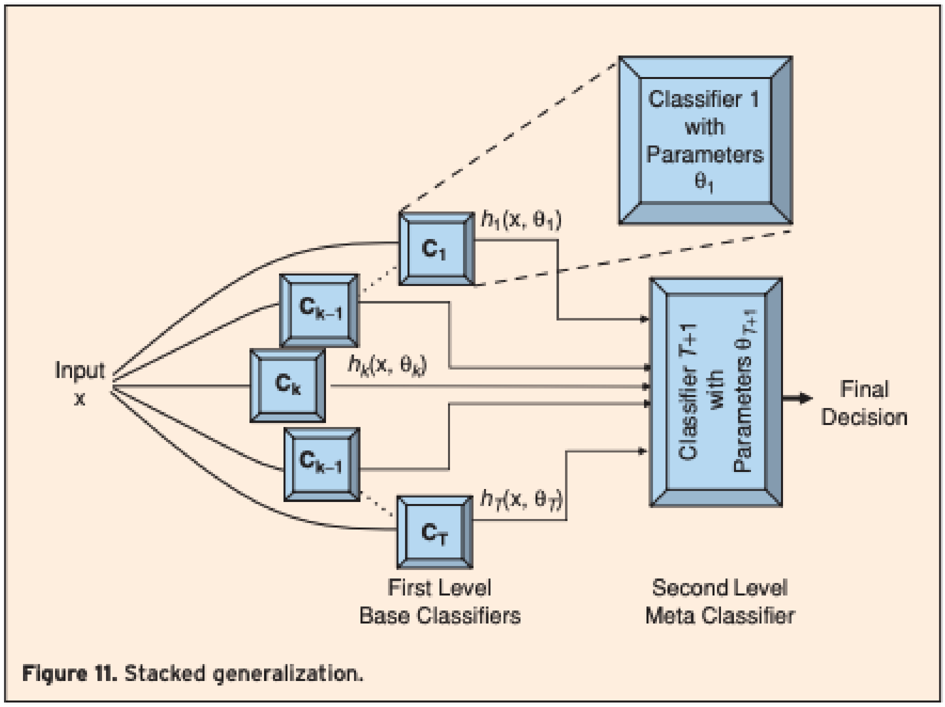

Classifier 1 with Parameters θ1
h1(x, θ1)
C1
Ck−1
Input x
Ck
hk(x, θk)
Ck−1
CT
hT(x, θT)
Classifier T+1 with Parameters θT+1
Final Decision
First Level Base Classifiers
Second Level Meta Classifier
Figure 11. Stacked generalization.

Figure 1.2. Diagram detailing stacked generalization ensemble technique from Polikar's article [16]. $C_1,..,C_T$ make p the first level of classifiers which output hypothesis $h_1,...,h_T$. C_{T+1} is the meta-classifier that uses the hypotheses to train and produce the final prediction.

it being coined a ”super learner” [21]. Choosing the right network for training the meta-classifier is crucial. For image segmentation tasks, CNNs and it's variants have been extremely successful.

1.7 U-Net

Computer visual analysis performance has drastically improved in the last decade with the help of deep learning. Convolutional neural networks (CNNs), have played a big part in this [22]. Due to its success, there have been numerous developments to tailor CNNs for biomedical image segmentation tasks; most notably U-Net and MaskR-CNN [23, 24].

U-Net is a very popular network used for semantic segmentation of biomedical images. The fully convolutional network architecture achieves a U-shape with a contracting path for feature extraction and expanding path for localization [23]. The network was originally used on EM data for segmentation of neuronal structures and achieved impressive results. It was also successfully used on light microscopy images for cell segmentation. Since then, the network has been used and adapted to numerous applications.

1.8 Challenges

Although there are many benefits to using deep learning, a significant drop in prediction performance can be easily experienced with less ideal conditions. With smaller datasets it is easy to overfit the network during training, which results in the model to poorly generalize. To combat this we can use regularization techniques such as weight decay, dropout, cross-validation, early stopping, simplifying model complexity, etc.

Deep learning networks are also hard to understand because of their black-box nature. For example, it is increasingly evident that deep networks are prone to adversarial attacks [25]. Usually in order to combat this, attempts are made to resist the injection of noise (e.g. by training with noisy data), blurred and distorted images, but in the field of biomedical imaging an extremely wide range of distortions and differences in images can occur, and training for each of them is to date not possible and will not be the most appropriate solution. Thus the need to ensure robustness of these networks is pressing.

Another challenge is tuning hyperparameters. There is a lot of literature on best practice and we followed Leslie Smith's guidelines closely to ensure the best possible training [26]. Even with these guidelines, there is not one single way to approach hyperparameter tuning, and there involves some extent of experimentation in this aspect.

1.9 Research Aims

In order to combat the challenges discussed previously, our research introduces 3Nsemble as an improved version of CDeep3M. 3Nsemble uses a trained meta-classifier to predict segmentations of 2D images rather than an averaging strategy. 3Nsemble implements a trained U-Net as out meta-classifier in order to improve prediction performance. Our work also tries to show that ”stacking” increases the generalization and robustness of the system.

To summarize, our main aims of this project are:

1. **Introduce trained combination rule to CDeep3M in the form of a meta-classifier to enhance prediction performance.**
2. **Demonstrate that ”stacking” increases the generalization and robustness of the system.**

We hypothesize that our improved system, 3Nsemble, produces better predictions than CDeep3M because it is able to weigh each base classifier’s hypothesis appropriately. The averaging strategy employed by CDeep3M can be thrown off by small errors in alignment and noise whereas a trained meta-classifier can deal with such errors more effectively.

1.10 Research Significance

The previous sections summarized some of the key components and challenges that currently exist in the pipeline to mapping a brain model. Our research focuses on improving image segmentation prediction by using deep learning and ensemble techniques. Improvements in EM image segmentation will enable scientists to expand their understanding of neuronal functions.

There have been many related studies that show success of deep learning with ensemble techniques for biomedical image segmentation tasks [27, 28, 29]. In particular to stacking and trainable classifiers, there are a great number of studies catalogued such as [19], which demonstrated stacking for membrane protein classification, and [27], which used an ensemble of random forests to classify blood vessels. Studies such as [30] demonstrated general robustness of deep learning ensemble strategies and [28] have demonstrated the success of these methods for small biomedical datasets.

Many published works on membrane and mitochondria segmentations in EM images exist thus far [9, 8, 31]. There has also been a lot of progress in segmenting synapses, such as in

[32, 10]; however, there is less literature on individual vesicle segmentation methods. Recent work highlights the importance and proposes machine learning methods to address this [33]. Our work also contributes to this need by demonstrating that stacking can greatly improve vesicle segmentation performance.

In addition, making highly accurate, efficient and relevant segmentation tools to the scientific community allows faster development and progress towards larger community goals. Overall, the contribution of this research is interdisciplinary and stretches beyond connectomics. It is clear that this work can be applied to other image segmentation applications beyond biomedical research. The presentation of our work details the data and methods used to implement our experiments (Chapter 2), our results (Chapter 3). Following this is a discussion covering the key takeaways from this study (Chapter 4) and a conclusion summarizing significance and future work (Chapter 5).

Chapter 2

Datasets and Methods

2.1 Datasets Used in this Study

The images used in this research are Electron Microscopy (EM) images of mitochondria, membrane and synaptic vesicles of mice and rat brains that were collected and imaged by experts in the field. The mitochondria and vesicles were collected at the National Center of Microscopy and Image Research (NCMIR) and can be found in the Cell Image Library (CIL) at http://www.cellimagelibrary.org/home. The membrane image stack is publicly available and is used in the SNEMI3D membrane segmentation challenge [34].

For our study, each dataset consists of the raw EM image stack, corresponding labels, and CDeep3M base classifier probability maps. The raw images are annotated by expert scientists with the help of IMOD, a software that allows contour drawing to create ground truth labels, for performance evaluation. These images and labels are fed into CDeep3M's 1fm, 3fm and 5fm networks to produce 3 probability maps which are saved in the dataset to implement the different combination rules.

A total of five datasets were used in this study. All images were obtained using Scanning Electron Microscopy (SEM). The following subsections describe each in more detail. A summary of dataset specifications are provided in Table 2.1 and sample images from each dataset are provided in Figure 2.1.

2.1.1 Mitochondria Datasets

There are two mitochondria datasets used in this research, Mitochondria and Mitochondria2, which were sampled from different areas of the mouse brain.

The Mitochondria dataset was collected from a mouse hippocampus, and was imaged using serial block-face scanning electron microscopy (SBEM) as described in Nature Methods 2018 [4]. The training data was 80 image planes of 1024x1024 resolution with corresponding ground truth labels. 15% of the images were reserved for testing and the remaining were split into training and validation appropriately.

The Mitochondria2 dataset includes 20 image planes of 500x500 pixel resolution collected from a mouse suprachiasmatic nucleus (SCN). 25% of the data was reserved for testing and the rest were used in training and validation.

2.1.2 Membrane Dataset

The Membrane dataset was collected by the Lichtman Lab at Harvard University and was used in the ISB 2013 segmentation challenge [34]. The sample preparation and imaging methods are detailed in Cell (2015) [35]. This dataset contains a total of 20 image planes with 1024x1024 pixel resolution, of which 25% was used for testing. The dataset is available at http://brainiac2.mit.edu/SNEMI3D/.

Table 2.1. Summary of datasets used for this research. Includes total number of images, image size in pixels (pixel resolution), and number of images used for testing.

Dataset name	Total # Images	Image Size	Test Set Size
Mitochondria	80	[1024,1024]	12 images
Mitochondria2	20	[500,500]	5 images
Membrane	20	[1024,1024]	5 images
Vesicles-24nm	11	[319,270]	3 images
Vesicles-20nm	20	[552,322]	4 images

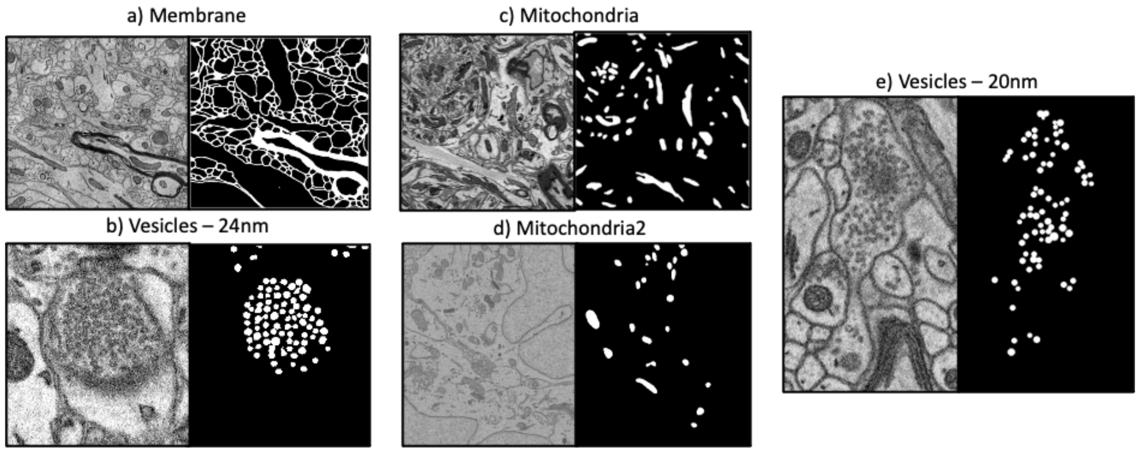


Figure 2.1. Sample images of each dataset. Each panel shows the raw image on the left and corresponding ground truth mask on the right.

2.1.3 Synaptic Vesicles Datasets

There are two vesicles datasets obtained using different cutting intervals: Vesicles-24nm and Vesicles-20nm.

Vesicles-24nm consists of a total 11 image planes of 270x319 pixel resolution. The 24nm here indicates the cutting interval used when imaging. Within this dataset, three images were reserved to be used as a test set, and the remaining made up the train and validation sets. It is to be noted that this is the smallest dataset used in this study.

Vesicles-20nm was collected from a rat cerebellum and imaging was done using SBEM with Gemini 3View, 20nm cutting interval, 2.5keV, 4nm pixels and 12kx9k raster. From large area of the image volume, a 270x319 pixel area with distinct and well-separated vesicles was chosen and 20 image planes of this area was extracted. From the 20 images, 4 images were reserved for testing and the remaining were used in training and validation.

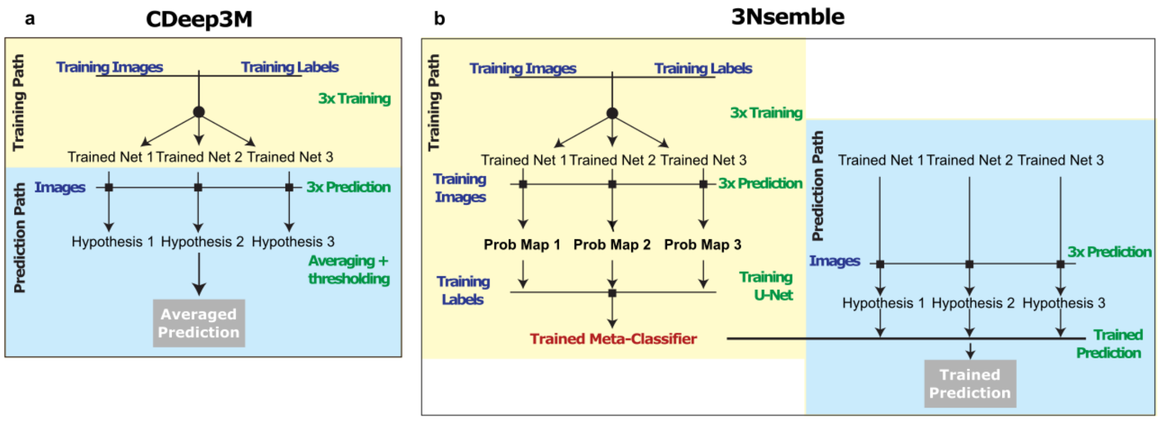


Figure 2.2. Comparison of training and prediction paths for Cdeep3M and 3Nsemble. a) Original workflow showing deterministic combination rule resulting in final averaged prediction b) 3Nsemble workflow showing stacked meta-classifier trained on ensemble level 0 probability maps to make final trained prediction

2.2 Methods

2.2.1 3Nsemble Workflow

3Nsemble employs a U-Net that is stacked onto the level 0 base classifiers to improve the final prediction performance. The training path involves feeding the training images and labels into the 1fm, 3fm and 5fm networks which produce 3 independent probability maps. These probability maps are each network's hypothesis of the prediction. They are then fed into a U-Net for training along with the ground truth labels (see Figure 2.3). For the prediction path, the training images are input into the base classifiers to get 3 prediction hypotheses that are stacked and fed into the trained meta-classifier to output the final segmentation. The complete workflows for the original CDeep3M and improved 3Nsemble systems are detailed in Figure 2.2.

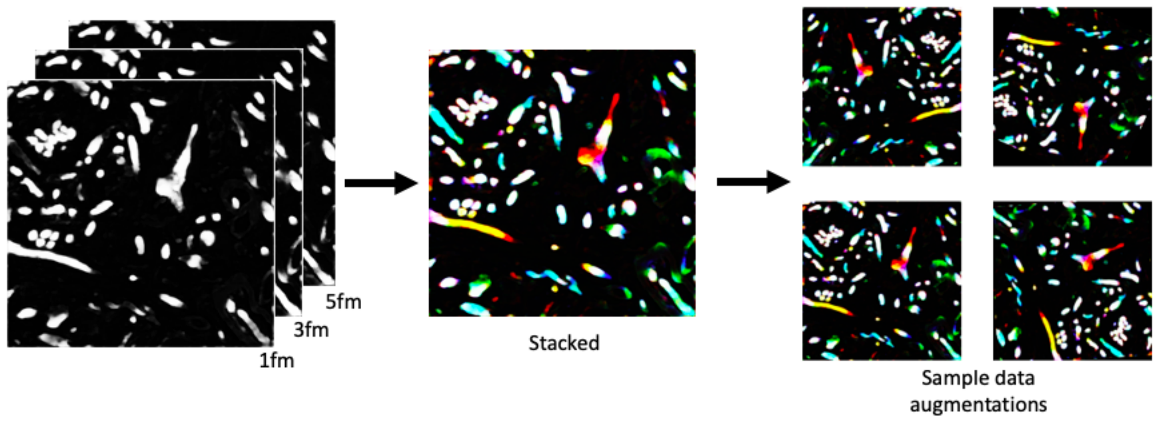


Figure 2.3. Mitochondria example of stacked probability maps from level 0 classifiers being transformed into a single input which undergoes augmentations before being fed into the U-Net for training.

2.2.2 Averaged Prediction

The averaging strategy employed in CDeep3M involves computing the mean of the 1fm, 3fm, 5fm probability maps and thresholding the result. Exhaustive thresholding was used for each dataset with a sweep from 50 to 250 and the optimal threshold was determined by the intersection of F1, precision and recall values (as seen in Figure 2.4). Threshold values for each dataset are listed in Table 2.2.

Table 2.2. Optimal thresholds for each dataset calculated using exhaustive thresholding.

	Mitochondria	Mitochondria2	Membrane	Vesicles-24nm	Vesicles-20nm
Threshold	203	224	164	131	228

2.2.3 Training the Meta-classifier

Using fastai library [36], a python library that simplifies the implementation of fast machine learning algorithms, we were able to implement the trained meta-classifier in Google Colab for comparison with the averaged prediction. Colab was chosen for it's free access to GPU resources, which allows researchers to reduce training time without investing in expensive hardware or paid cloud computing. Fastai has an inbuilt U-Net architecture with pre-trained

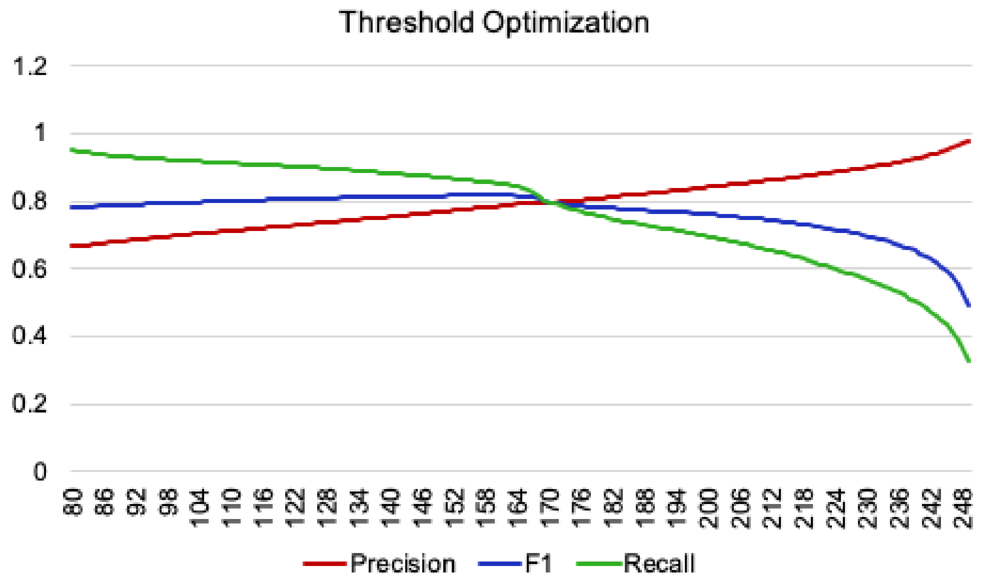


Figure 2.4. Exhaustive thresholding scans through possible intensities in [0,255] and calculates corresponding metrics to find optimum threshold

weights ready for use which we were able to make use of for transfer learning. We used a resnet50 backbone for all datasets in order to balance learning with model complexity. The network hyper-parameters were tuned using suggestions from Leslie Smith's paper [26]. We performed training using the one cycle policy described in Smith's paper which speeds up training significantly. With this policy, the learning rate is steadily incremented for a portion of the training and then decremented for the remainder, leading to superconvergence (See Figure 2.5a).

We specified the optimum learning rate for training each dataset according to the inbuilt Fastai learning rate finder, which plots the loss for a range of learning rates. The maximum rate of change in the resulting plot can be used to determine an optimal learning rate (See Figure 2.5b). We chose the learning rate to be the rounded up number from the point.

We did not implement any weight decay since we found that it over-regularized the model; however, all datasets were trained with early stopping based on the validation loss with specified patience of *n* epochs and a minimum delta of *d* (See Appendix for detailed hyper-parameter

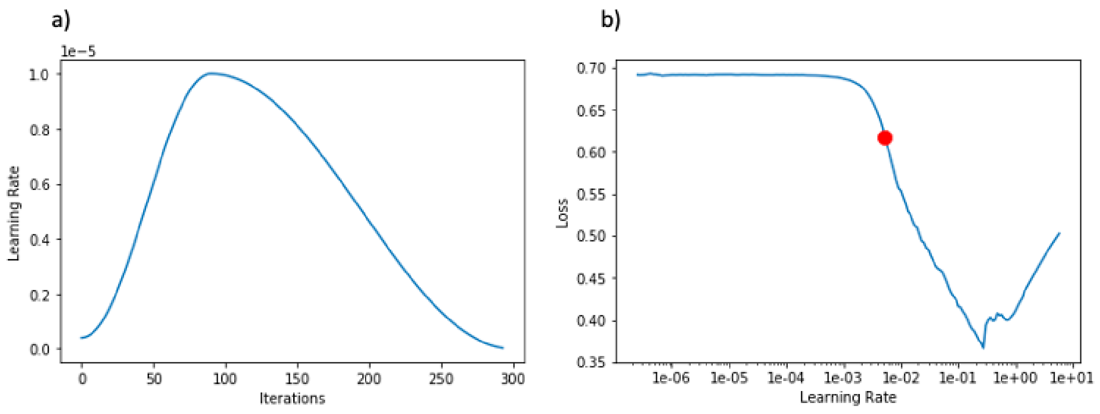


Figure 2.5. a) One Cycle Policy: Example of learning rate configurations during training. b) Example of learning rate plot from Fastai documentation. The red dot indicates point of greatest gradient change [36].

tuning). Another method of regularization used was data augmentations on the training set in the form of -180 to 90 degree rotations and left/right flips. The augmentations were performed randomly with a probability of 0.5. In addition, the size of the larger 1024x1024 images were reduced to 512x512 in order to run the larger resnet50 network on the available GPUs of Google Colab.

We set the maximum number of epochs M for the initial training and $F < M$ for fine tuning. For the fine tuning, we used a smaller learning rate than for the training since the pretrained weights do not need drastic adjustments. Cross entropy loss was used to monitor training progress.

2.2.4 Testing and Evaluation

Testing images were used for the performance evaluation of all the trained models. Metrics used to evaluate both the averaged and trained ensemble segmentations were accuracy, precision, recall, Interscetion Over Union (IOU, also referred to as Jaccard index) and F1 score (also known as Dice score)[1]. These can be calculated using the confusion matrix

[1]For better comparison, evaluation of the experiments were made using the same metrics as in the CDeep3M paper [4].

$[[TN, FP][FN, TP]]$ detailing the percentage of pixels that were True Positive (TP), True Negative (TN), False Positive (FP) and False Negative (FN). From this the metrics are calculated with the following formulas:

$$Accuracy = \frac{(TN + TP) * 100}{TP + TN + FP + FN} \tag{2.1}$$

$$Precision = \frac{TP}{TP + FP} \tag{2.2}$$

$$Recall = \frac{TP}{TP + FN} \tag{2.3}$$

$$IOU = \frac{TP}{TP + FP + FN} \tag{2.4}$$

$$F1Score = \frac{2 * (precision * recall)}{(precision + recall)} \tag{2.5}$$

2.2.5 Validation Experiments

In order to validate whether the stacking approach makes the predictions more robust, two experiments were performed:

The first validation experiment aimed to compare how each combination strategy performs when the image stack contains misaligned planes. To do so, image planes 10 and 15 in the mitochondria dataset were translated by [15,10] and [12,12] pixels respectively. The complete stack containing the misaligned data was fed into the ensemble of classifiers and the resulting 1fm, 3fm and 5fm probability maps were used for each strategy (averaging and trained meta-classifier). The averaging strategy followed the same way as described in Section 2.2.2. For our validation we required that the stacking approach both trained on and tested on a misaligned image. Thus, Image 15 was designated for training while Image 10 was kept separately for testing. All other details of the experiment followed the methods described in Section 2.2.3, and extended details of the trained model and hyper-parameters are listed in Appendix A. This experiment aimed to confirm that the trained meta-classifier would be able to detect that one of the ensemble classifiers predicts better than the others. In this the 2D classifier is expected to perform better than the other 3D classifiers, since the misalignment throws them off track [6].

The second experiment dealt with the injection of noise to the Mitochondria dataset. 20 Mitochondria images were injected with white gaussian noise of zero mean and 0.01 variance. A set of 5 images was used for testing. The performance of CDeep3M and 3Nsemble were compared on these noisy inputs. The meta-classifier was trained using the same conditions that were applied for the non-noise control case to make comparison easier and are detailed in Appendix A.

2.2.6 Software Availability

The CDeep3M data and pretrained models are available here. It can be run via Docker, Singularity, Amazon Web Services (AWS) or Google Colab. 3Nsemble scripts will be made available here via Github and Google Colab. The following is a breakdown of the main scripts available on 3Nsemble[2]:

- **AveragedEnsemble** is a notebook that runs through the averaging, threshold optimization and metrics calculations of the 1fm,3fm and 5fm predictions.
- **TrainedEnsemble** runs through the entire training process of the trained ensemble model, from loading the data, initializing the network, training and fine tuning the network and producing the final metrics on the test set.
- **ShowPredictions** takes a trained prediction network and produces an overlay of False Positive, True Positive and False Negative predictions on the specified image.
- **Stacked** takes the 1fm, 3fm and 5fm predictions and stacks them into a 3D tensor for use in training.

The 3Nsemble code currently has dependencies with Fastai v1, PyTorch and ScikitLearn metrics.

[2]This is list may undergo changes as software implementation is optimized and streamlined for public use

Chapter 3

Results

3.1 Key Results

The results of this study show significant increase in overall performance with relatively little training time (See Table 3.1). In particular, Intersection Over Union (IOU) increased by more than 12% for all datasets. This improvement was followed by precision, recall and F1 score. Accuracy also improved but to a lesser extent. It is important to note that increase in performance depends highly on the CDeep3M prediction. Additionally, datasets with poor baseline predictions to start with have more room to improve and thus we can see larger percentages of improvements. A more detailed summary of the training for each dataset is found in Appendix A.

The large increase in performance is at a small cost of extra training time. The metrics improved for all organelles with less than an hour of total training time, and for the majority was less than half an hour. Note that train time is proportional to dataset size and total epochs. Mitochondria took the most training time and was the biggest dataset (as seen previously in Table 2.1). Table 3.2 contains a summary of the training time and epochs relative to dataset size.

Table 3.1. Comparison of CDeep3M (averaged predictions) and 3Nsemble (trained predictions) on all data sets. Includes epochs total, total train time in minutes, precision, recall, Intersection Over Union (IOU), F1 score and accuracy.

	Precision	**Recall**	**IOU**	**F1**	**Accuracy**
Mitochondria					
CDeep3M	0.8775	0.8891	0.7909	0.8832	0.9824
3Nsemble	0.9428	0.9427	0.8917	0.9427	0.9921
Improvement (%):	7.4	6.0	12.7	6.7	1.0
Mitochondria2					
CDeep3M	0.8808	0.9313	0.8271	0.9054	0.9928
3Nsemble	0.9601	0.9625	0.9262	0.9617	0.9970
Improvement (%):	9.1	3.4	12	6.2	0.4
Membrane					
CDeep3M	0.8131	0.8834	0.7343	0.8468	0.9132
3Nsemble	0.9377	0.9372	0.8823	0.9375	0.9664
Improvement (%):	15.3	6.1	20.2	10.7	5.8
Vesicles-24nm					
CDeep3M	0.6405	0.6020	0.4500	0.6207	0.9093
3Nsemble	0.8872	0.8596	0.7749	0.8732	0.9660
Improvement (%):	38.5	42.8	72.2	40.7	6.2
Vesicles-20nm					
CDeep3M	0.7362	0.7767	0.6076	0.7559	0.9757
3Nsemble	0.8800	0.8589	0.7688	0.8693	0.9876
Improvement (%):	19.5	10.6	26.5	15	1.2

Table 3.2. Comparison of dataset size and train time. Includes training set size, number of epochs and total train time in minutes.

	#Train Images	**Epochs**	**Train Time (min)**
Mitochondria	68	85	54.7
Mitochondria2	15	75	8.6
Membrane	15	97	12.5
Vesicles-24nm	8	144	13
Vesicles-20nm	16	61	6.3

Below are some sample predictions from each dataset (Figures 3.1 to 3.4). Majority of images were brightened for presentation purposes in order to clearly visualize the prediction overlays. Legends for True Positive (TP), False Positive (FP) and False Negative (FN) pixel predictions are included and captions state IOU and F1 scores. Images of 3Nsemble predictions are from the test set.

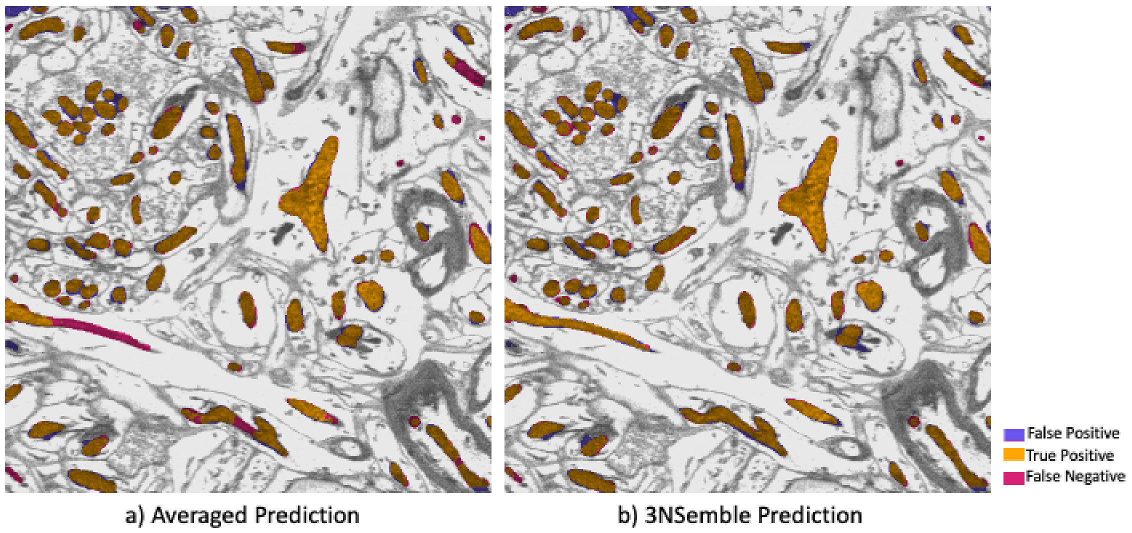


Figure 3.1. Comparison of mitochondria predictions via averaging and stacked generalization. a) Averaged prediction (IOU: 0.76, F1: 0.87) b) 3Nsemble prediction (IOU: 0.81, F1: 0.89), showing visible decrease in FP and FN pixels.

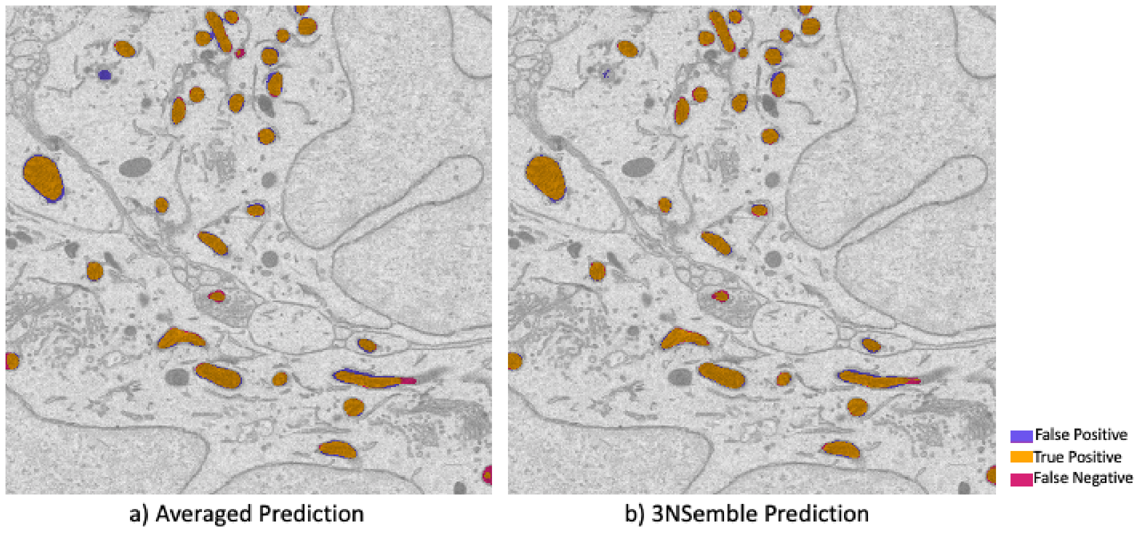


Figure 3.2. Comparison of mitochondria2 predictions via averaging and stacked generalization. a) Averaged prediction (IOU: 0.81, F1: 0.9) b) 3Nsemble prediction (IOU: 0.86, F1: 0.93), showing visible decrease in FP and FN pixels.

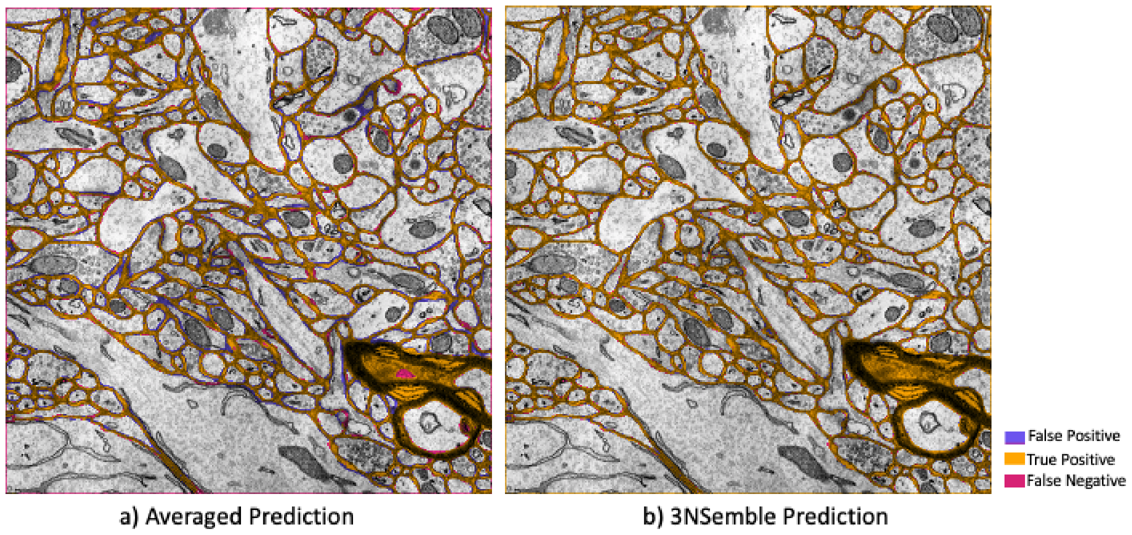


Figure 3.3. Comparison of membrane predictions via averaging and stacked generalization. a) Averaged prediction (IOU: 0.74, F1: 0.85) b) 3Nsemble prediction (IOU: 0.89, F1: 0.94) showing visible increase in TP and decrease in falsely detected pixels

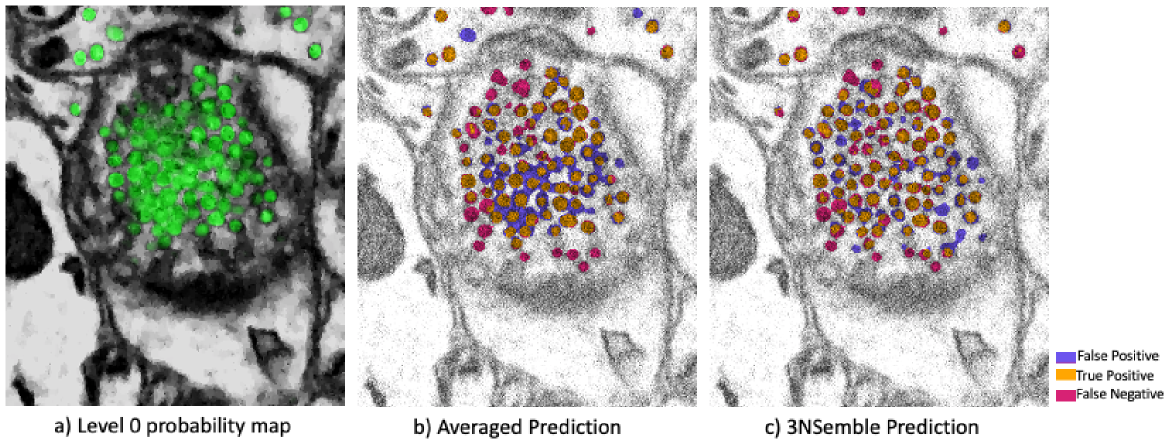


Figure 3.4. Comparison of vesicle-24nm predictions via averaging and stacked generalization. a) Combined probability maps from level 0 base classifiers b) Averaged prediction (IOU: 0.47, F1: 0.64) c) 3Nsemble prediction (IOU: 0.49, F1: 0.66) showing visible decrease in FP pixels

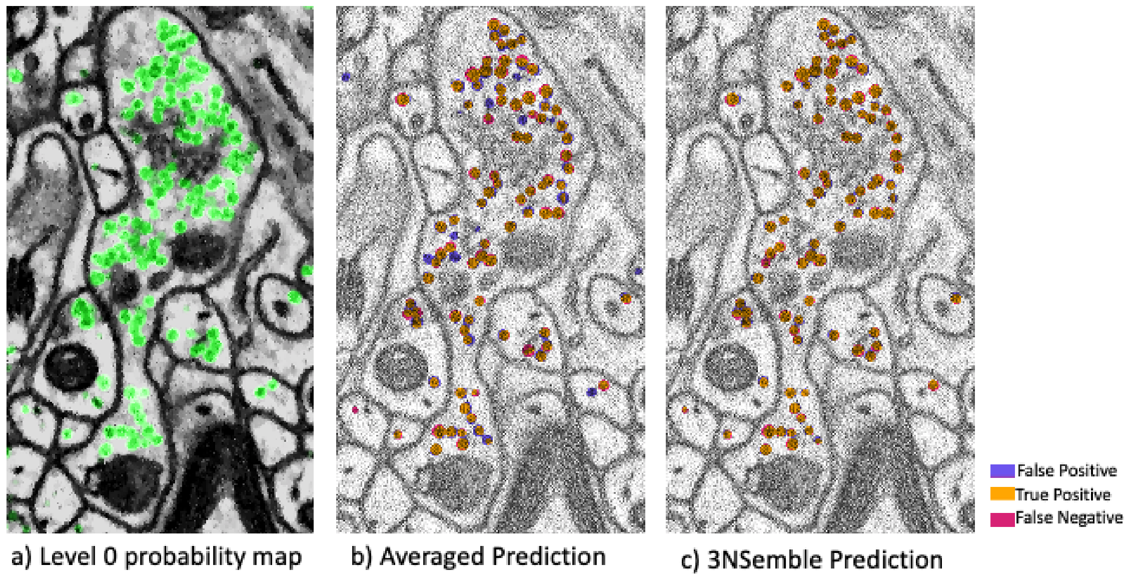


Figure 3.5. Comparison of vesicle-20nm predictions via averaging and stacked generalization. a) Combined probability maps from level 0 base classifiers b) Averaged prediction (IOU: 0.66, F1: 0.80) c) 3Nsemble prediction (IOU: 0.80, F1: 0.89)

3.2 Validation Results

The initial validation experiment as described previously, focused on showing robustness to misaligned image stacks. A summary of the results are shown in Table 3.3. Image 10 and 15 were misaligned in the train and test set respectively and their results are shown underneath each strategy. The implementation of both strategies is described in Section 2.2.5.

The second validation experiment involved injection of noise to the training images. All images were injected with the same type of noise as detailed in Section 2.2.5. The preliminary results of this are shown in Table 3.4, where a comparison of noise-less and noisy mitochondria images are used to test the robustness of 3Nsemble.

Table 3.3. Results of misalignment experiment on the mitochondria dataset. Shows improvement in metrics by using the stacked generalization approach over a simple averaging.

	Precision	Recall	IOU	F1 Score	Accuracy
Averaging	**0.8585**	**0.8280**	**0.7285**	**0.8430**	**0.9736**
Image 10	0.706	0.226	0.206	0.342	0.915
Image 15	0.751	0.169	0.160	0.276	0.933
Stacked Generalization	**0.9213**	**0.9248**	**0.8570**	**0.9230**	**0.9856**
Image 10	0.901	0.500	0.474	0.643	0.946
Image 15	0.930	0.923	0.863	0.926	0.988

Table 3.4. Noise validation experiment on Mitochondria dataset. Images were injected with white gaussian noise with zero mean and 0.01 variance. Comparison of CDeep3M and 3Nsemble performance for noise and no-noise experiments are displayed.

	Precision	**Recall**	**IOU**	**F1**	**Accuracy**
CDeep3M	0.815	0.944	0.778	0.875	0.974
CDeep3M with noise	0.783	0.886	0.712	0.832	0.965
Drop (%)	**4.0**	**6.1**	**8.5**	**4.9**	**0.9**
3Nsemble	0.909	0.931	0.852	0.920	0.985
3Nsemble with noise	0.900	0.910	0.826	0.905	0.982
Drop (%)	**0.9**	**2.3**	**3.1**	**1.6**	**0.3**

Chapter 4

Discussion

3Nsemble segmentations show significant improvement in the accuracy, precision, recall, Intersection Over Union (IOU) and F1 score compared to CDeep3M. The performance improvement outweighs the additional training time. Specifically, IOU saw the biggest increase of 12-70% indicating a greater overlap of the prediction mask and ground truth (Table 3.1). Higher precision and recall scores lead to increased IOU because they are proportional to the percentage of correctly identified positive pixels (Equation 2.2 and 2.3). They also affect the F1 score, which is calculated using the harmonic mean of both (Equation 2.5).

The improvement in metrics is relative to the baseline CDeep3M prediction. Datasets with high performance using averaging see relatively small improvements with the trained meta-classifier. For example, CDeep3M produced a 0.993 accuracy for Mitochondria2 and 3Nsemble improved this by 0.4%. In comparison, Vesicles-24nm saw the lowest CDeep3M accuracy of 0.91 which allowed room for 3Nsemble to improve by 6.2%. This trend can be seen on a larger scale as well: Vesicles-24nm had the lowest CDeep3M metrics but also saw the largest increases with 3Nsemble. This indicates that even when the base classifiers produce poor hypotheses, stacking a trained meta-classifier combines the strengths of each to greatly improve the final prediction without even seeing the original training images.

Data augmentation and early stopping provides ample regularization for the meta-classifier to avoid overfitting. The train and test image metrics are very similar indicating the generalizability of the model to the entire dataset. In addition, the results of the validation experiment also show the robustness of the model given noisy or misaligned images. With more testing in these areas we can further catalogue the extent to which stacking has increased to robustness and generalization of the predictions.

A crucial challenge in deep learning is having enough data to train and test on. Using our methods we are able to show significant improvements in datasets as small as 11 images. This is can be attributed to the ensemble learning approach which alleviates the issue of data size. Other techniques that aided in this were the U-Net architecture and data augmentations. It is always beneficial to have more training data, however labeling ground truths can be time consuming and difficult, therefore it is very significant to see such a small dataset improve using 3Nsemble.

Additionally, it is important to note that manually annotating ground truth masks is prone to small errors or inconsistencies. These can be due to human error or the occasional subjective nature of deciding whether a section is to be labeled or not (for example, non ordinary structures resulting from imaging or tissue anomalies). This can lead to under or over masking of images and can affect the prediction metrics. Therefore, although the segmentation can be done automatically and with high performance, expert analysis is still required to qualitatively validate the segmentations. This is true for any machine learning system. Depending on application, false positives may be preferred to false negatives and depending on the organelle and further analysis tasks, region contours might be more valuable than individual pixel classifications. For example, with membrane segmentation often scientists prioritize segmenting closed and accurate contours of each cell as it is more useful for cell count and 3D analysis.

The validation experiments show promising results on the robustness of 3Nsemble. Further work needs to be done to quantitatively measure robustness against varying misalignment and noise to fully evaluate the extent of the model's robustness.

Another point of exploration is using k-fold cross validation for training the meta-classifer. This is the original approach to stacked generalization as described by Wolpert [18]. We did not apply such a thorough training process but regardless saw success in our strategy. The implementation of k-fold cross validation would be a worthwhile future endeavor.

In summary, stacking a trained meta-classifier significantly increases the performance of the ensemble systems. The additional costs of training and applying the meta-classifier is outweighed by this increased performance and improves the robustness to noise and misalignment. This method can especially aid when encountering poor baseline probability maps. Further testing should be done to quantify the increase in robustness and generalizability to broader data.

Chapter 5

Conclusion

We have presented the benefits of stacked generalization in Electron Microscopy (EM) segmentation. 3Nsemble uses a trained meta-classifier that significantly improves prediction performance compared to CDeep3M's averaging strategy. Even with poor baseline predictions 3Nsemble is able to improve the segmentations greatly. This ensemble technique also increases the robustness of the model to noise and misalignment and can be applied to various organelles and microscopy image data. Highly accurate automated segmentations of organelles greatly benefits the study of connectomics and moves us closer to understanding the brain and brain disorders.

Future work involves training a generalized meta-classifier to work for a range of common organelles and datasets. This can then be made available for predictions and would eliminate extra training time while still enhancing the prediction performance. Early experiments in this show promising results. Other work consists of quantifying 3Nsemble's robustness and generalizability and implementing k-fold cross validation.

Appendix A

Extended Results

Table A.1 summarizes the CDeep3M prediction performance over all datasets. These results form the baseline for which all comparison of 3Nsemble performance is done. Sample predictions for each dataset is shown in the Results section (Figures 3.1 to 3.5). The specific metrics for each sample shown is logged in Table A.2.

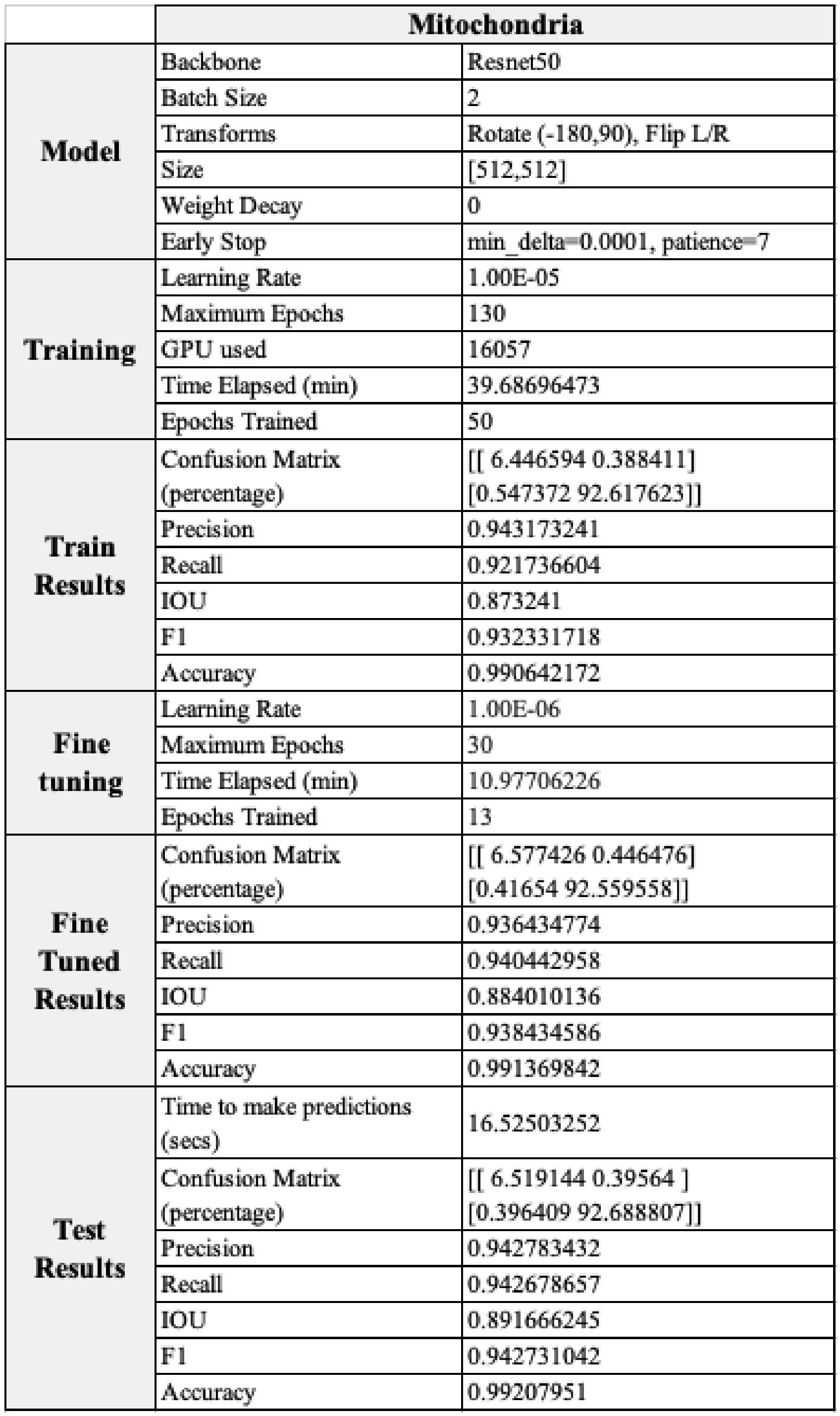
Table A.1. Summary of CDeep3M averaged prediction performance over all datasets

	Threshold	Confusion	Precision	Recall	IOU	F1 Score	Accuracy
Mitochondria	203	[[91.6, 0.93] [0.83, 6.64]]	0.8775	0.8891	0.7909	0.8832	0.9824
Mitochondria2	224	[[95.8, 0.47] [0.26, 3.46]]	0.8808	0.9313	0.8271	0.9055	0.9928
Membrane	164	[[67.4, 5.51] [4.91, 7.42]]	0.8131	0.8834	0.7343	0.8468	0.9132
Vesicles-24nm	131	[[83.5, 4.17] [4.91, 7.42]]	0.6405	0.6020	0.4500	0.6207	0.9093
Vesicles-20nm	228	[[93.8, 1.35] [1.08, 3.76]]	0.7362	0.7767	0.6076	0.7559	0.9757

Table A.2. Performance metrics of CDeep3M sample predictions shown in the Results section

	Precision	Recall	IOU	F1 Score	Accuracy
Mitochondria	0.87	0.86	0.76	0.87	0.97
Mitochondria2	0.86	0.94	0.81	0.90	0.99
Membrane	0.80	0.91	0.74	0.85	0.92
Vesicles-24nm	0.60	0.69	0.47	0.64	0.94
Vesicles-20nm	0.76	0.84	0.66	0.80	0.98

The following figures (A.1 to A.7) show extended details on each dataset's model, training and testing specifications in table form. Although many hyper-parameters were uniform over all experiments, they are listed in these summaries to show that they are available to tweak for future training.

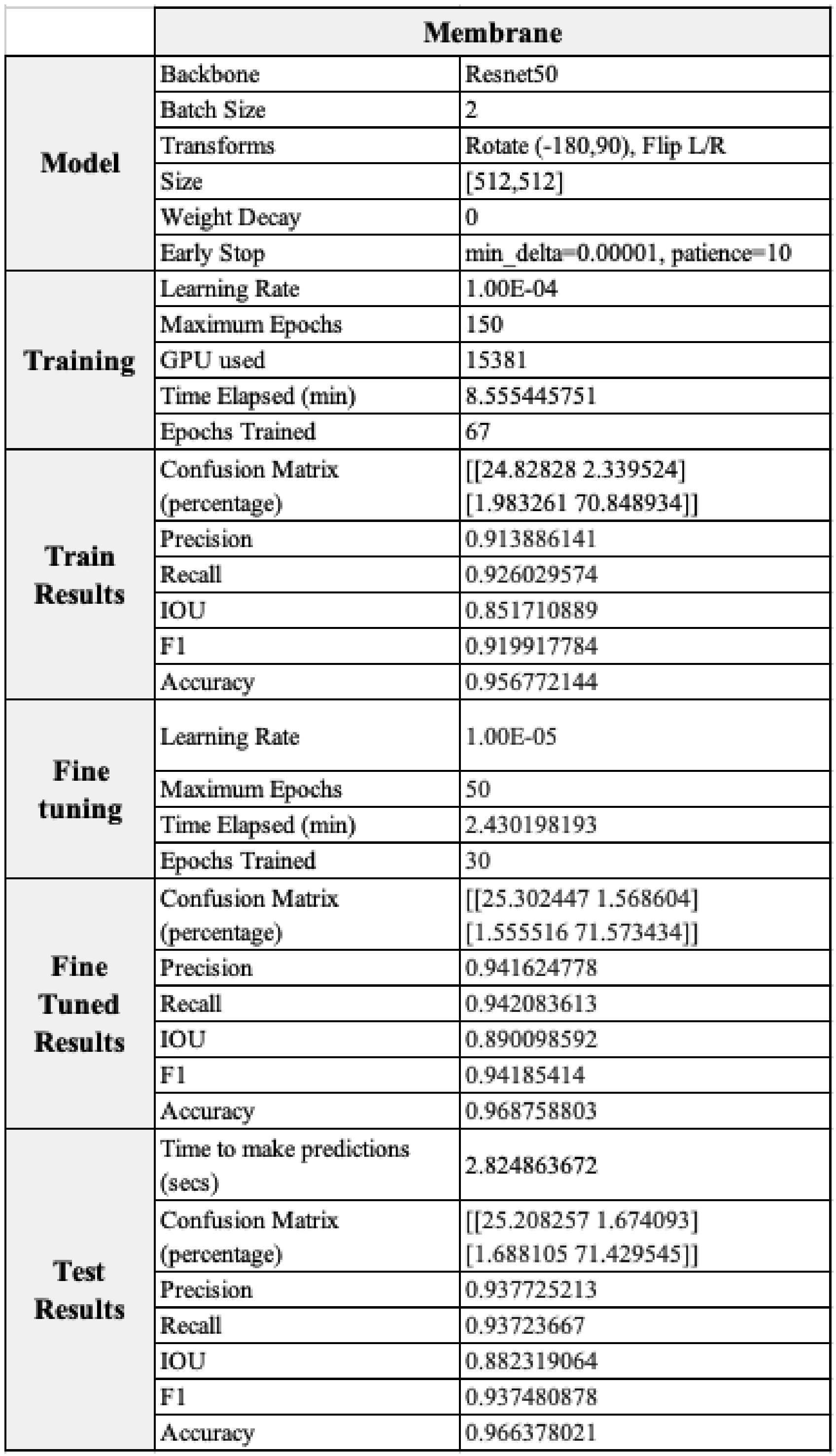


	Mitochondria	
Model	Backbone	Resnet50
	Batch Size	2
	Transforms	Rotate (-180,90), Flip L/R
	Size	[512,512]
	Weight Decay	0
	Early Stop	min_delta=0.0001, patience=7
Training	Learning Rate	1.00E-05
	Maximum Epochs	130
	GPU used	16057
	Time Elapsed (min)	39.68696473
	Epochs Trained	50
Train Results	Confusion Matrix (percentage)	[[6.446594 0.388411] [0.547372 92.617623]]
	Precision	0.943173241
	Recall	0.921736604
	IOU	0.873241
	F1	0.932331718
	Accuracy	0.990642172
Fine tuning	Learning Rate	1.00E-06
	Maximum Epochs	30
	Time Elapsed (min)	10.97706226
	Epochs Trained	13
Fine Tuned Results	Confusion Matrix (percentage)	[[6.577426 0.446476] [0.41654 92.559558]]
	Precision	0.936434774
	Recall	0.940442958
	IOU	0.884010136
	F1	0.938434586
	Accuracy	0.991369842
Test Results	Time to make predictions (secs)	16.52503252
	Confusion Matrix (percentage)	[[6.519144 0.39564] [0.396409 92.688807]]
	Precision	0.942783432
	Recall	0.942678657
	IOU	0.891666245
	F1	0.942731042
	Accuracy	0.99207951

Figure A.1. Details of Mitochondria model parameters and performance

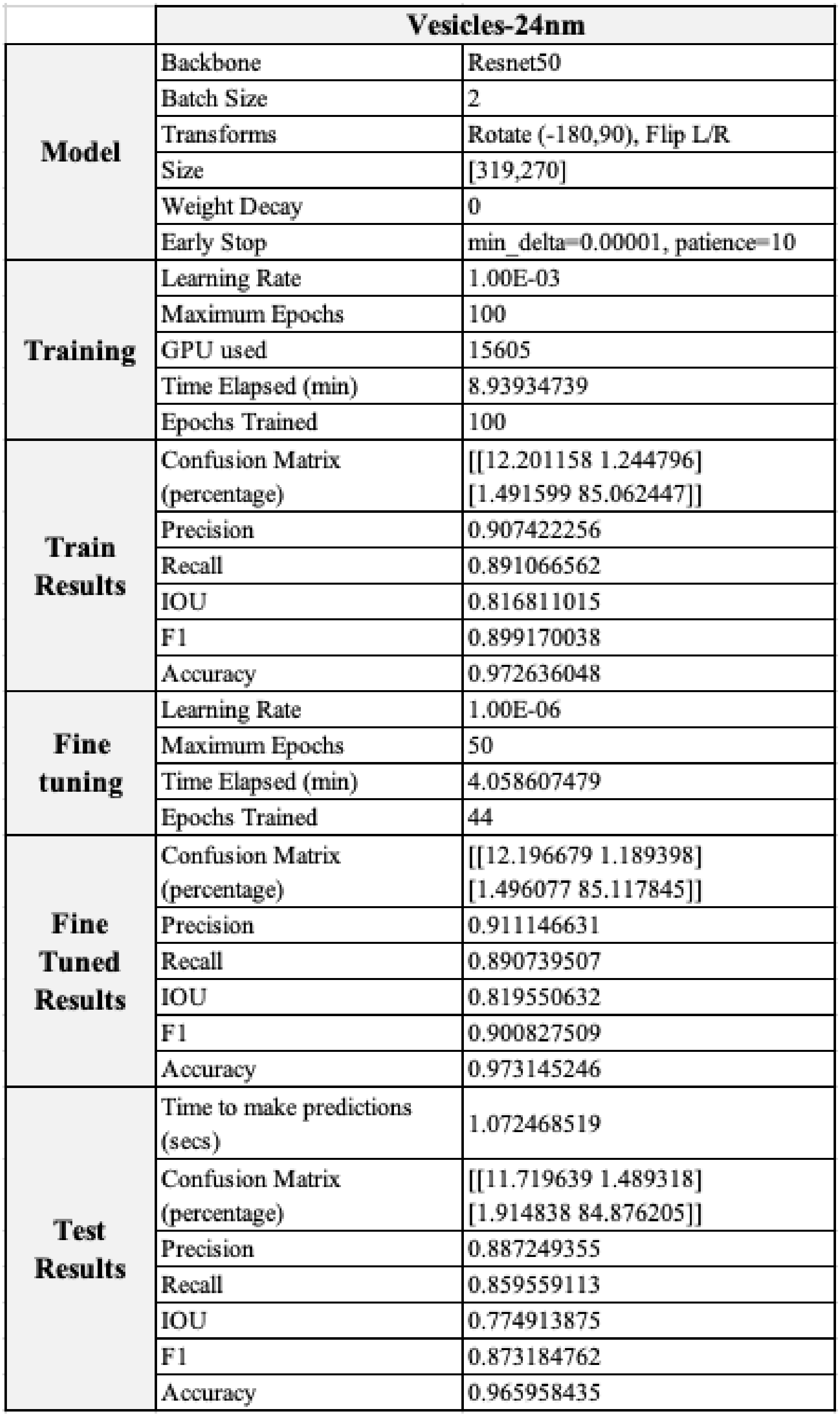
	Mitochondria2	
Model	Backbone	Resnet50
	Batch Size	2
	Transforms	Rotate (-180,90), Flip L/R
	Size	[500,500]
	Weight Decay	0
	Early Stop	min_delta=0.0001, patience=7
Training	Learning Rate	1.00E-04
	Maximum Epochs	100
	GPU used	15381
	Time Elapsed (min)	5.344558966
	Epochs Trained	47
Train Results	Confusion Matrix (percentage)	[[3.798308 0.217723] [0.134523 95.849446]]
	Precision	0.945786502
	Recall	0.965794847
	IOU	0.915132735
	F1	0.955685962
	Accuracy	0.996477538
Fine tuning	Learning Rate	1.00E-06
	Maximum Epochs	50
	Time Elapsed (min)	3.263662024
	Epochs Trained	28
Fine Tuned Results	Confusion Matrix (percentage)	[[3.811969 0.145908] [0.120862 95.921262]]
	Precision	0.963134859
	Recall	0.963134859
	IOU	0.934595161
	F1	0.966191977
	Accuracy	0.997332308
Test Results	Time to make predictions (secs)	2.495912313
	Confusion Matrix (percentage)	[[3.77176 0.153627] [0.146907 95.927707]]
	Precision	0.960863303
	Recall	0.962511058
	IOU	0.92620047
	F1	0.961686475
	Accuracy	0.996994667

Figure A.2. Details of Mitochondria2 model parameters and performance



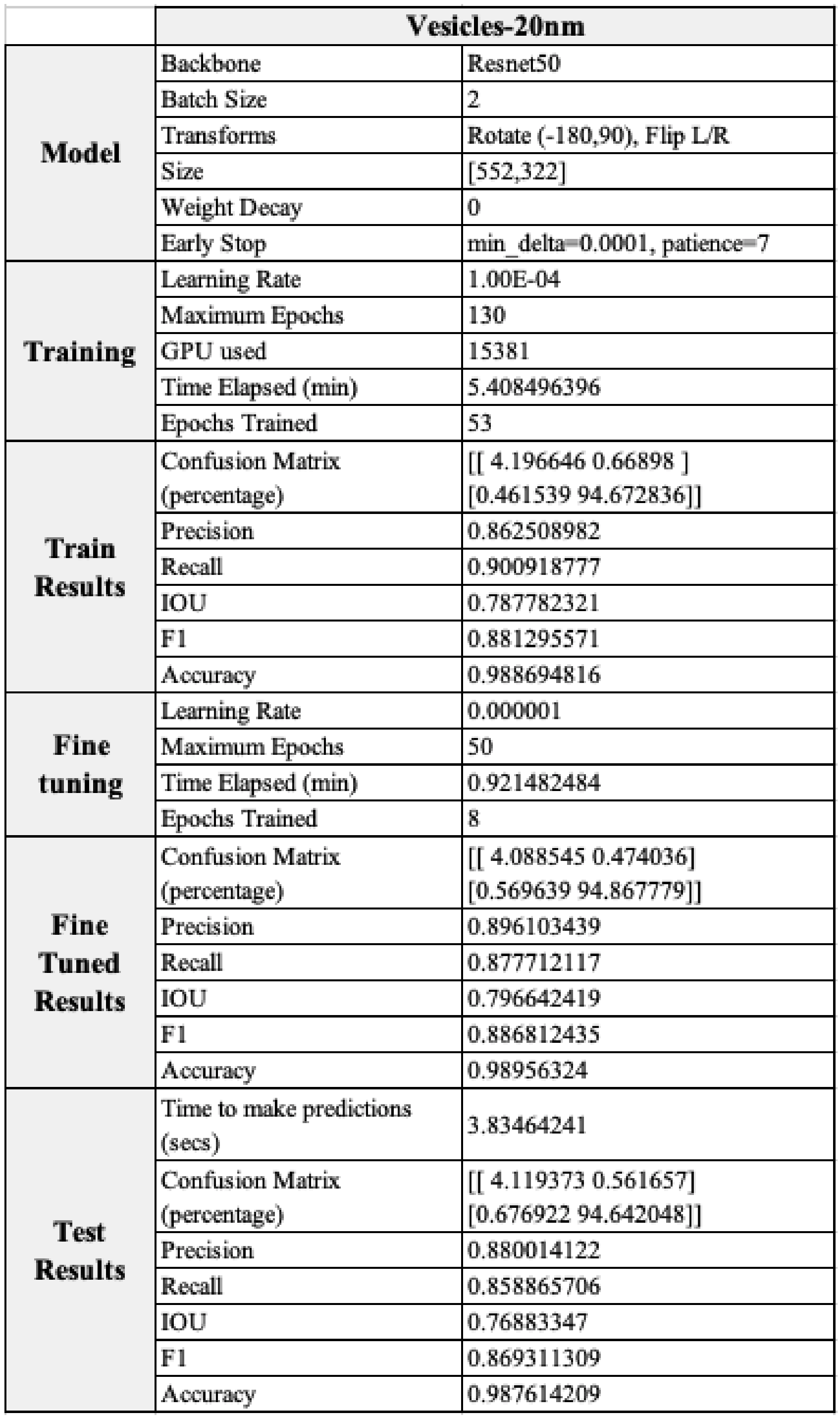
	Membrane	
Model	Backbone	Resnet50
	Batch Size	2
	Transforms	Rotate (-180,90), Flip L/R
	Size	[512,512]
	Weight Decay	0
	Early Stop	min_delta=0.00001, patience=10
Training	Learning Rate	1.00E-04
	Maximum Epochs	150
	GPU used	15381
	Time Elapsed (min)	8.555445751
	Epochs Trained	67
Train Results	Confusion Matrix (percentage)	[[24.82828 2.339524] [1.983261 70.848934]]
	Precision	0.913886141
	Recall	0.926029574
	IOU	0.851710889
	F1	0.919917784
	Accuracy	0.956772144
Fine tuning	Learning Rate	1.00E-05
	Maximum Epochs	50
	Time Elapsed (min)	2.430198193
	Epochs Trained	30
Fine Tuned Results	Confusion Matrix (percentage)	[[25.302447 1.568604] [1.555516 71.573434]]
	Precision	0.941624778
	Recall	0.942083613
	IOU	0.890098592
	F1	0.94185414
	Accuracy	0.968758803
Test Results	Time to make predictions (secs)	2.824863672
	Confusion Matrix (percentage)	[[25.208257 1.674093] [1.688105 71.429545]]
	Precision	0.937725213
	Recall	0.93723667
	IOU	0.882319064
	F1	0.937480878
	Accuracy	0.966378021

Figure A.3. Details of Membrane model parameters and performance



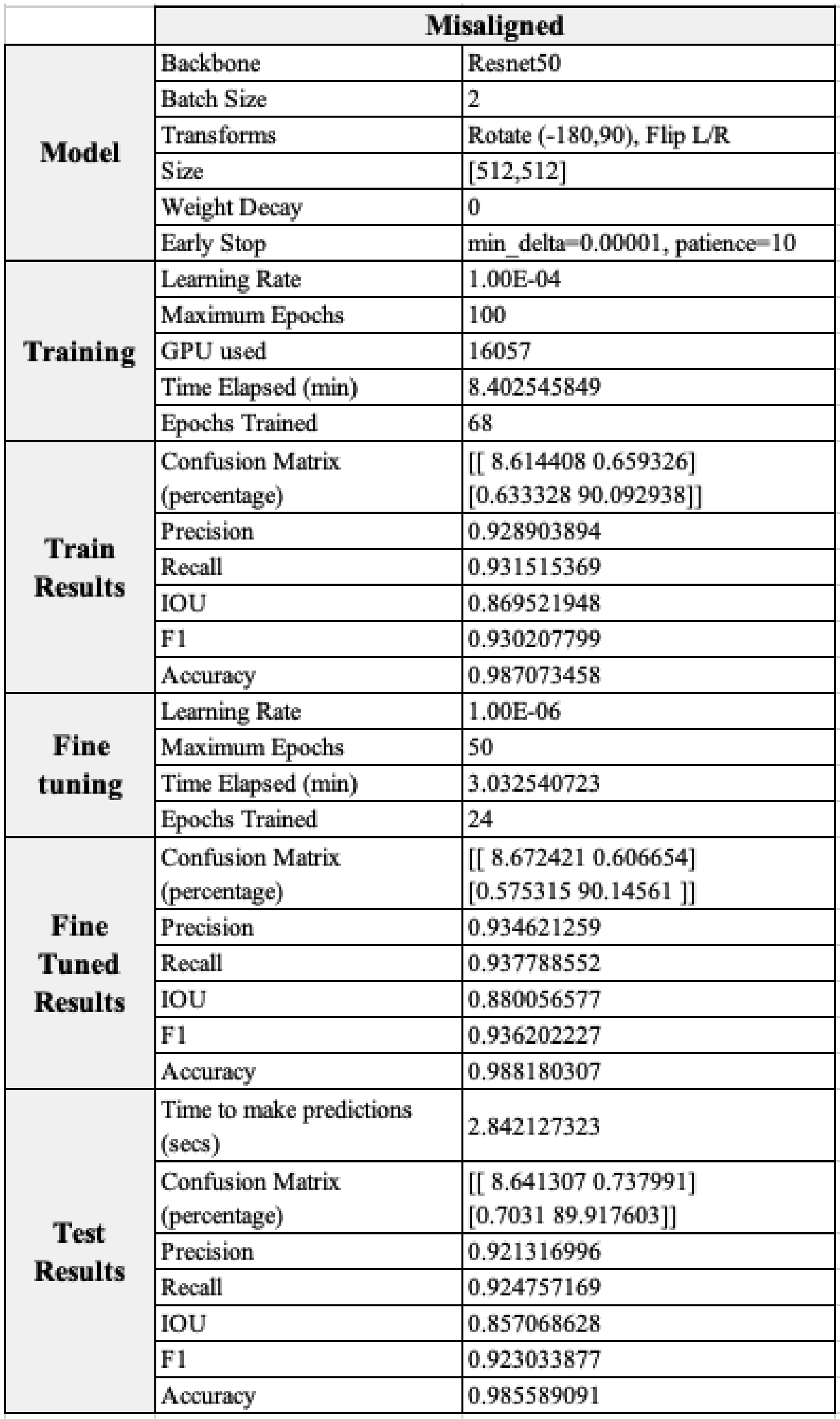
	Vesicles-24nm	
Model	Backbone	Resnet50
	Batch Size	2
	Transforms	Rotate (-180,90), Flip L/R
	Size	[319,270]
	Weight Decay	0
	Early Stop	min_delta=0.00001, patience=10
Training	Learning Rate	1.00E-03
	Maximum Epochs	100
	GPU used	15605
	Time Elapsed (min)	8.93934739
	Epochs Trained	100
Train Results	Confusion Matrix (percentage)	[[12.201158 1.244796] [1.491599 85.062447]]
	Precision	0.907422256
	Recall	0.891066562
	IOU	0.816811015
	F1	0.899170038
	Accuracy	0.972636048
Fine tuning	Learning Rate	1.00E-06
	Maximum Epochs	50
	Time Elapsed (min)	4.058607479
	Epochs Trained	44
Fine Tuned Results	Confusion Matrix (percentage)	[[12.196679 1.189398] [1.496077 85.117845]]
	Precision	0.911146631
	Recall	0.890739507
	IOU	0.819550632
	F1	0.900827509
	Accuracy	0.973145246
Test Results	Time to make predictions (secs)	1.072468519
	Confusion Matrix (percentage)	[[11.719639 1.489318] [1.914838 84.876205]]
	Precision	0.887249355
	Recall	0.859559113
	IOU	0.774913875
	F1	0.873184762
	Accuracy	0.965958435

Figure A.4. Details of Vesicles-24nm model parameters and performance



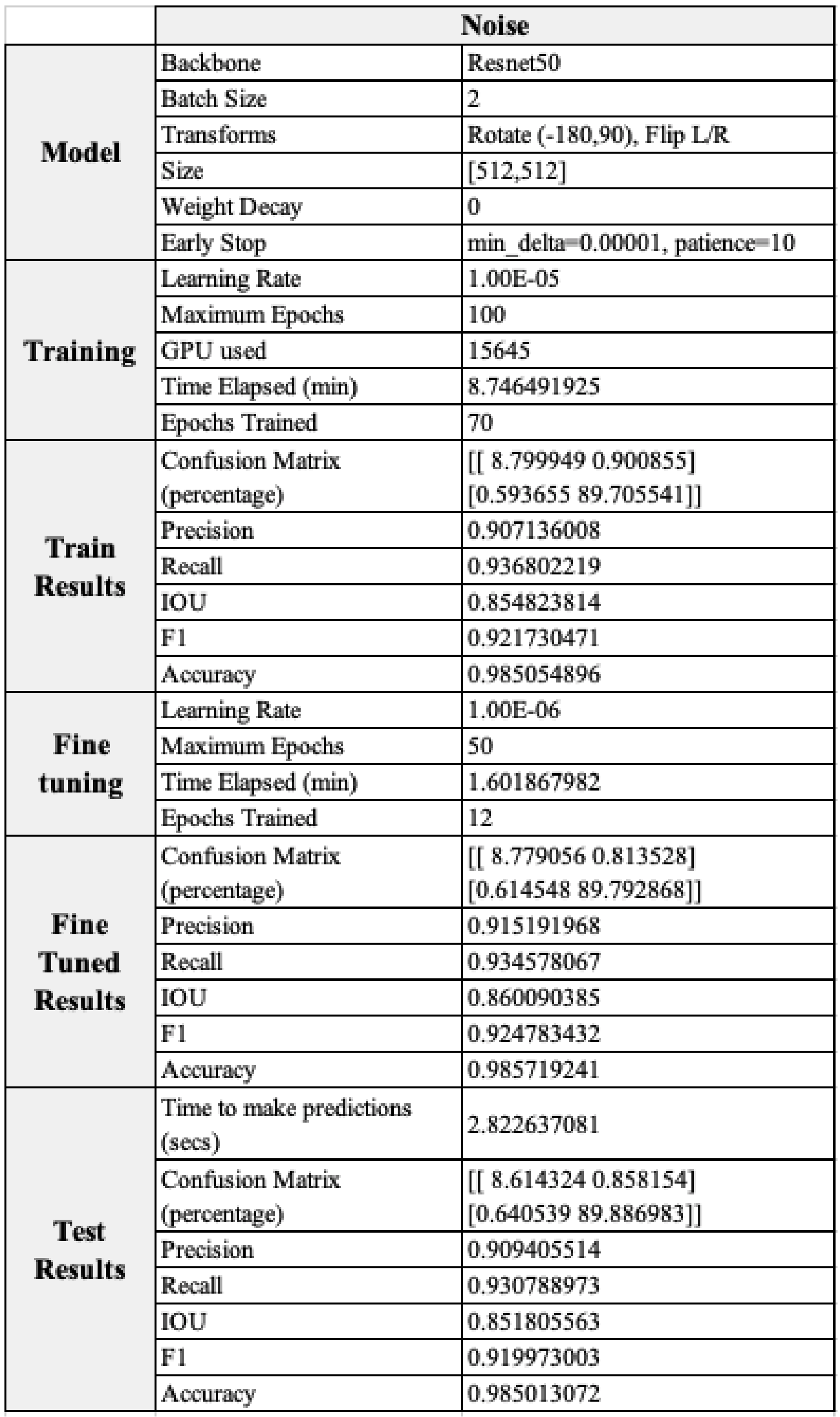
	Vesicles-20nm	
Model	Backbone	Resnet50
	Batch Size	2
	Transforms	Rotate (-180,90), Flip L/R
	Size	[552,322]
	Weight Decay	0
	Early Stop	min_delta=0.0001, patience=7
Training	Learning Rate	1.00E-04
	Maximum Epochs	130
	GPU used	15381
	Time Elapsed (min)	5.408496396
	Epochs Trained	53
Train Results	Confusion Matrix (percentage)	[[4.196646 0.66898] [0.461539 94.672836]]
	Precision	0.862508982
	Recall	0.900918777
	IOU	0.787782321
	F1	0.881295571
	Accuracy	0.988694816
Fine tuning	Learning Rate	0.000001
	Maximum Epochs	50
	Time Elapsed (min)	0.921482484
	Epochs Trained	8
Fine Tuned Results	Confusion Matrix (percentage)	[[4.088545 0.474036] [0.569639 94.867779]]
	Precision	0.896103439
	Recall	0.877712117
	IOU	0.796642419
	F1	0.886812435
	Accuracy	0.98956324
Test Results	Time to make predictions (secs)	3.83464241
	Confusion Matrix (percentage)	[[4.119373 0.561657] [0.676922 94.642048]]
	Precision	0.880014122
	Recall	0.858865706
	IOU	0.76883347
	F1	0.869311309
	Accuracy	0.987614209

Figure A.5. Details of Vesicles-20nm model parameters and performance



	Misaligned	
Model	Backbone	Resnet50
	Batch Size	2
	Transforms	Rotate (-180,90), Flip L/R
	Size	[512,512]
	Weight Decay	0
	Early Stop	min_delta=0.00001, patience=10
Training	Learning Rate	1.00E-04
	Maximum Epochs	100
	GPU used	16057
	Time Elapsed (min)	8.402545849
	Epochs Trained	68
Train Results	Confusion Matrix (percentage)	[[8.614408 0.659326] [0.633328 90.092938]]
	Precision	0.928903894
	Recall	0.931515369
	IOU	0.869521948
	F1	0.930207799
	Accuracy	0.987073458
Fine tuning	Learning Rate	1.00E-06
	Maximum Epochs	50
	Time Elapsed (min)	3.032540723
	Epochs Trained	24
Fine Tuned Results	Confusion Matrix (percentage)	[[8.672421 0.606654] [0.575315 90.14561]]
	Precision	0.934621259
	Recall	0.937788552
	IOU	0.880056577
	F1	0.936202227
	Accuracy	0.988180307
Test Results	Time to make predictions (secs)	2.842127323
	Confusion Matrix (percentage)	[[8.641307 0.737991] [0.7031 89.917603]]
	Precision	0.921316996
	Recall	0.924757169
	IOU	0.857068628
	F1	0.923033877
	Accuracy	0.985589091

Figure A.6. Details of model parameters and performance for misaligned validation experiment. The corresponding CDeep3M predictions were made using a 183 thresholding value.



	Noise	
Model	Backbone	Resnet50
	Batch Size	2
	Transforms	Rotate (-180,90), Flip L/R
	Size	[512,512]
	Weight Decay	0
	Early Stop	min_delta=0.00001, patience=10
Training	Learning Rate	1.00E-05
	Maximum Epochs	100
	GPU used	15645
	Time Elapsed (min)	8.746491925
	Epochs Trained	70
Train Results	Confusion Matrix (percentage)	[[8.799949 0.900855] [0.593655 89.705541]]
	Precision	0.907136008
	Recall	0.936802219
	IOU	0.854823814
	F1	0.921730471
	Accuracy	0.985054896
Fine tuning	Learning Rate	1.00E-06
	Maximum Epochs	50
	Time Elapsed (min)	1.601867982
	Epochs Trained	12
Fine Tuned Results	Confusion Matrix (percentage)	[[8.779056 0.813528] [0.614548 89.792868]]
	Precision	0.915191968
	Recall	0.934578067
	IOU	0.860090385
	F1	0.924783432
	Accuracy	0.985719241
Test Results	Time to make predictions (secs)	2.822637081
	Confusion Matrix (percentage)	[[8.614324 0.858154] [0.640539 89.886983]]
	Precision	0.909405514
	Recall	0.930788973
	IOU	0.851805563
	F1	0.919973003
	Accuracy	0.985013072

Figure A.7. Details of model parameters and performance for noise validation experiment. The corresponding CDeep3M predictions were made using a 170 thresholding value.

Bibliography

[1] Thomas R Insel, Story C Landis, and Francis S Collins. The nih brain initiative. *Science*, 340(6133):687–688, 2013.

[2] Zachary Sweger. Researcher leads 3.8 million dollar project to map developing mice brains, 2020.

[3] Jeff W. Lichtman and Winfried Denk. The big and the small: Challenges of imaging the brain's circuits. *Science*, 334(6056):618–623, 2011.

[4] Matthias G Haberl, Christopher Churas, Lucas Tindall, Daniela Boassa, Sebastien Phan, Eric A Bushong, Matthew Madany, Raffi Akay, Thomas J Deerinck, Steven T Peltier, and Mark H Ellisman. Cdeep3m - plug-and-play cloud based deep learning for image segmentation of light, electron and x-ray microscopy. *bioRxiv*, 2018.

[5] Olaf Sporns, Giulio Tononi, and Rolf Kötter. The human connectome: A structural description of the human brain. *PLOS Computational Biology*, 1, 09 2005.

[6] Tao Zeng, Bian Wu, and Shuiwang Ji. Deepem3d: Approaching human-level performance on 3d anisotropic em image segmentation. *Bioinformatics (Oxford, England)*, 33, 03 2017.

[7] Kevin L Briggman and Davi D Bock. Volume electron microscopy for neuronal circuit reconstruction. *Current opinion in neurobiology*, 22(1):154–161, 2012.

[8] Dan Ciresan, Alessandro Giusti, Luca Gambardella, and Jürgen Schmidhuber. Deep neural networks segment neuronal membranes in electron microscopy images. *Advances in neural information processing systems*, 25:2843–2851, 2012.

[9] Elizabeth Jurrus, Antonio RC Paiva, Shigeki Watanabe, James R Anderson, Bryan W Jones, Ross T Whitaker, Erik M Jorgensen, Robert E Marc, and Tolga Tasdizen. Detection of neuron membranes in electron microscopy images using a serial neural network architecture. *Medical image analysis*, 14(6):770–783, 2010.

[10] Carlos Becker, Karim Ali, Graham Knott, and Pascal Fua. Learning context cues for synapse segmentation. *IEEE transactions on medical imaging*, 32(10):1864–1877, 2013.

[11] Jörgen Kornfeld and Winfried Denk. Progress and remaining challenges in high-throughput volume electron microscopy. *Current opinion in neurobiology*, 50:261–267, 2018.

[12] Zheng Lin, Jesse Jin, and Hugues Talbot. Unseeded region growing for 3d image segmentation., 2000.

[13] Michał Januszewski, Jörgen Kornfeld, Peter H Li, Art Pope, Tim Blakely, Larry Lindsey, Jeremy Maitin-Shepard, Mike Tyka, Winfried Denk, and Viren Jain. High-precision automated reconstruction of neurons with flood-filling networks. *Nature methods*, 15(8):605–610, 2018.

[14] Kisuk Lee, Jonathan Zung, Peter Li, Viren Jain, and H Sebastian Seung. Superhuman accuracy on the snemi3d connectomics challenge. *arXiv preprint arXiv:1706.00120*, 2017.

[15] Siqi Liu, Daguang Xu, S. Kevin Zhou, Thomas Mertelmeier, Julia Wicklein, Anna Jerebko, Sasa Grbic, Olivier Pauly, Weidong Cai, and Dorin Comaniciu. 3d anisotropic hybrid network: Transferring convolutional features from 2d images to 3d anisotropic volumes, 2017.

[16] Robi Polikar. Polikar, r.: Ensemble based systems in decision making. ieee circuit syst. mag. 6, 21-45. *Circuits and Systems Magazine, IEEE*, 6:21 – 45, 10 2006.

[17] Thomas G Dietterich. Ensemble methods in machine learning. In *International workshop on multiple classifier systems*, pages 1–15. Springer, 2000.

[18] David H Wolpert. Stacked generalization. *Neural networks*, 5(2):241–259, 1992.

[19] Shuang-Quan Wang, Jie Yang, and Kuo-Chen Chou. Using stacked generalization to predict membrane protein types based on pseudo-amino acid composition. *Journal of theoretical biology*, 242(4):941–946, 2006.

[20] Kai Ming Ting and Ian H Witten. Issues in stacked generalization. *Journal of artificial intelligence research*, 10:271–289, 1999.

[21] Ashley I Naimi and Laura B Balzer. Stacked generalization: an introduction to super learning. *European journal of epidemiology*, 33(5):459–464, 2018.

[22] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017.

[23] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. *CoRR*, abs/1505.04597, 2015.

[24] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross B. Girshick. Mask R-CNN. *CoRR*, abs/1703.06870, 2017.

[25] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks, 2014.

[26] Leslie N. Smith. A disciplined approach to neural network hyper-parameters: Part 1 – learning rate, batch size, momentum, and weight decay, 2018.

[27] Shuangling Wang, Yilong Yin, Guibao Cao, Benzheng Wei, Yuanjie Zheng, and Gongping Yang. Hierarchical retinal blood vessel segmentation based on feature and ensemble learning. *Neurocomputing*, 149:708–717, 2015.

[28] Julian Zilly, Joachim M Buhmann, and Dwarikanath Mahapatra. Glaucoma detection using entropy sampling and ensemble learning for automatic optic cup and disc segmentation. *Computerized Medical Imaging and Graphics*, 55:28–41, 2017.

[29] Muhammad Moazam Fraz, Paolo Remagnino, Andreas Hoppe, Bunyarit Uyyanonvara, Alicja R Rudnicka, Christopher G Owen, and Sarah A Barman. An ensemble classification-based approach applied to retinal blood vessel segmentation. *IEEE Transactions on Biomedical Engineering*, 59(9):2538–2548, 2012.

[30] Ling Liu, Wenqi Wei, Ka-Ho Chow, Margaret Loper, Emre Gursoy, Stacey Truex, and Yanzhao Wu. Deep neural network ensembles against deception: Ensemble diversity, accuracy and robustness, 2019.

[31] Aurélien Lucchi, Kevin Smith, Radhakrishna Achanta, Graham Knott, and Pascal Fua. Supervoxel-based segmentation of mitochondria in em image stacks with learned shape features. *IEEE transactions on medical imaging*, 31(2):474–486, 2011.

[32] Anna Kreshuk, Christoph N Straehle, Christoph Sommer, Ullrich Koethe, Marco Cantoni, Graham Knott, and Fred A Hamprecht. Automated detection and segmentation of synaptic contacts in nearly isotropic serial electron microscopy images. *PloS one*, 6(10):e24899, 2011.

[33] Kristin Verena Kaltdorf, Maria Theiss, Sebastian Matthias Markert, Mei Zhen, Thomas Dandekar, Christian Stigloher, and Philip Kollmannsberger. Automated classification of synaptic vesicles in electron tomograms of c. elegans using machine learning. *PloS one*, 13(10):e0205348, 2018.

[34] Ignacio Arganda-Carreras, H. Sebastian Seung, Ashwin Vishwanathan, and Daniel R. Berger. Snemi3d: 3d segmentation of neurites in em images, 2013.

[35] Narayanan Kasthuri, Kenneth Jeffrey Hayworth, Daniel Raimund Berger, Richard Lee Schalek, Jos e Angel Conchello, Seymour Knowles-Barley, Dongil Lee, Amelio Vãzquez Reina, Verena Kaynig, Thouis Raymond Jones, et al. Saturated reconstruction of a volume of neocortex. *Cell*, 162(3):648–661, 2015.

[36] Jeremy Howard et al. fastai, 2018.