AxonDeep: Automated Optic Nerve Axon Segmentation in Mice With Deep Learning

Wenxiang Deng¹,², *, Adam Hedberg-Buenz²,³, *, Dana A. Soukup²,³, Sima Taghizadeh¹, Kai Wang⁴, Michael G. Anderson²,³,⁵, ‡, and Mona K. Garvin¹,², * ¹ Department of Electrical and Computer Engineering, The University of Iowa, Iowa City, IA, USA ² Iowa City VA Center for the Prevention and Treatment of Visual Loss, Iowa City VA Health Care System, Iowa City, IA, USA ³ Department of Molecular Physiology and Biophysics, The University of Iowa, Iowa City, IA, USA ⁴ Department of Biostatistics, The University of Iowa, Iowa City, IA, USA ⁵ Department of Ophthalmology and Visual Sciences, The University of Iowa, Iowa City, IA, USA  

Correspondence: Mona K. Garvin, 4318 Seamsan Center for the Engineering Arts and Sciences, The University of Iowa, Iowa City, IA 52242, USA. e-mail: mona-garvin@uiowa.edu

Received: March 31, 2021
Accepted: October 25, 2021
Published: December 21, 2021

Keywords: retinal ganglion cell; axon segmentation; deep learning; semisupervised learning; mice


Introduction

Retinal ganglion cell (RGC) loss is the primary feature of glaucoma, a leading cause of irreversible vision loss.¹,² RGCs are also damaged as a part of several other diseases, including forms of traumatic brain injury (TBI),³,⁴ diabetes,⁵ multiple sclerosis,⁶,⁷ and Alzheimer’s disease,⁸ among others.⁹ To evaluate RGC damage, the two fundamental options are to quantify RGC soma in the retina or RGC axons in the optic nerve, which in health typically have a 1:1 relationship. Because axon damage can occur earlier than somal loss,¹⁰,¹¹ there are many situations in which it is useful to quantify both. Although the identification of RGC-specific markers has helped advance techniques for quantification of RGC soma,¹²,¹³ techniques for quantification of RGC axons, which are smaller and more difficult to label, have lagged. Manual counting of axons has traditionally been adopted by human experts.¹⁴–¹⁸ However, manually counting is not only time-consuming but also prone to vary among...
different experts. Grading of optic nerve appearance based on pathological features has been useful for making some advances, but it is even more subjective and prone to variation.

First-generation automated approaches for quantifying axons, such as AxonJ, AxonMaster, AxoNet, and use of QuPath have several known limitations. For example, AxonJ, which members of our group were involved in developing, was designed to recognize axons only in healthy, not diseased, optic nerves. AxonJ has been demonstrated to not perform well with damaged nerves. It is also relevant that few of the existing approaches were developed using mice (AxonMaster, nonhuman primates; AxoNet, rats; QuPath, rats). It can be expected that deep-learning approaches, especially given their success in numerous medical application domains, ultimately would be more robust. In fact, recently, the deep-learning approach AxoNet has been proposed for providing axon counts; however, this approach still does not provide direct segmentation of the axons, and thus the ability to compute additional potential quantitative measures (such as area distributions) using this approach is limited.

In the present work, we propose a deep-learning approach, named AxonDeep, for the segmentation of optic nerve axons in murine tissue. This work was motivated by the need to overcome some of the known limitations of first-generation approaches and a desire to obtain complete segmentations (allowing quantitative measures of multiple axon measurements, beyond just counts) that are not possible with the current tools. Our deep-learning architecture is based on recent work on asymmetric network structures whereby a deep encoder combined with a lightweight decoder can provide an improved performance over symmetric deep-learning architectures for image segmentation problems with more complex scenes. In developing AxonDeep, two deep-learning approaches were evaluated: (1) a traditional supervised approach using a fully convolutional network (FCN) trained with only labeled data and (2) a semisupervised approach trained with both labeled and unlabeled data using a generative-adversarial-network framework. Using the semisupervised approach to be able also to take advantage of unlabeled data, in addition to labeled data, during training was motivated by a need to help address the challenges associated with the manual effort required to generate training sets. Both deep-learning approaches were compared to the AxonJ approach, with the semisupervised approach found to have the better overall performance. Thus we introduce AxonDeep as a deep-learning-based axon segmentation tool trained based on a semisupervised approach.

## Methods

### Procurement and Preparation of Mouse Optic Nerve Specimens for Image Analysis

#### Mice

Optic nerves (n = 56 nerves, one nerve from each of 56 mice) were collected from three different mouse strains modeling various presentations of healthy and diseased optic nerves: DBA/2J with various degrees of an inherited age-related form of glaucoma (n = 11 nerves); D2.B6-Lystbg-J/Andm (abbreviated hereafter as D2.Lyst) with healthy optic nerves (n = 5 nerves); C57BL/6J that had been subjected to either blast-induced TBI within an enclosed chamber or sham treatment (n = 27 nerves); and Diversity Outbred (J:DO) with healthy optic nerves but predicted to exhibit genetic background-dependent natural variability in optic nerve features (n = 13 nerves). Tissues in the current study were collected from mice contributing to prior publications; however, all data reported herein arise from new analyses performed uniquely for this study. All mice used in this study were originally purchased from The Jackson Laboratory (Bar Harbor, ME, USA). All animals were treated in accordance with the ARVO Statement for the Use of Animals in Ophthalmic and Vision Research. All experimental protocols were approved by the Institutional Animal Care and Use Committee of the University of Iowa.

#### Nerve Processing

Nerves were processed for histology as previously described. In brief, mice were euthanized by carbon dioxide inhalation with subsequent decapitation. Heads were collected and the skulls opened before fixation in half-strength Karnovsky’s fixative (2% paraformaldehyde, 2.5% glutaraldehyde in 0.1 M sodium cacodylate) at 4°C for 16 hours. Optic nerves were dissected from brains and drop fixed in the same fixative for an additional 16 hours at 4°C. Nerves were stained with 1% osmium tetroxide, dehydrated in graded acetone (30%–100%), infiltrated and embedded in resin (Eponate-12; Ted Pella, Redding, CA, USA), and polymerized in a 65°C oven. Semithin (1-μm) cross-sections were cut, transferred to glass slides, stained with 1% paraphenylenediamine, and mounted.

### Procurement of Images for Training, Validation, and Testing of the Network

Light micrographs (physical dimensions: 90.2 × 67.5 μm; resolution: 4140 × 3096 px) were collected
Figure 1. Flowchart of datasets, experimental design, and progression of tool development. The total dataset was composed of a diverse collection of optic nerve specimens from multiple genotypes and strains with natural phenotypic variability (i.e., J:DO), normal health (i.e., D2.Lyst) and forms of damage resulting from naturally occurring disease (i.e., DBA/2J) or inducible injury (i.e., TBI blast). Nerves were qualitatively graded (grade 1: mild/no damage [green]; grade 2: moderate damage [yellow]; grade 3: severe damage [red]) and divided into cohorts with 28 nerves for training, 10 nerves for validation, and 18 nerves for final testing. Across each set, the composition of nerves by damage grade remained consistent for the validation and training sets (ratio of 2:1 grade 1/grade 2 nerve); note that the tool was not designed to quantitate grade 3 nerves with severe damage and that only unlabelled images as part of the training set included grade 3 nerves. Based on annotations by expert 1, a total of 6762 axon centers were marked in the training set, 2668 axon centers were marked in the validation set, 3317 axon centers were marked, and 1103 axons were traced in the final testing set. From the unannotated set of images (n = 50) used in the training set, there were in excess of 150,000 axons.

from stained optic nerve cross-sections at a total magnification ×1000 using identical camera settings, as previously described.17,25,42 In brief, light micrographs were acquired from representative (i.e., a field not atypical from the rest of the nerve) and nonoverlapping fields from one cross-section of each optic nerve using an upright light microscope (BX52; Olympus, Tokyo, Japan) equipped with a CCD camera (DP-72; Olympus).

Separation of Images for Purposes of Training/Validation and Testing of the Deep-Learning Networks

Optic nerve images were divided into two major subdivisions used for different purposes, a “training/validation” set and a “testing” set (Fig. 1). To ensure an equal distribution of nerves with comparable levels of damage between the two sets, a qualitative damage grade was first assigned to each nerve by consensus among a panel of three independent graders (1 = mild or no damage, 2 = moderate damage, and 3 = severe damage), as previously described.14 Representative examples of each damage grade are shown in Supplementary Figure S1. Within grade-1 nerves, two subgroups were used: (i) grade-1 no apparent damage, defined as nerves from strains of healthy mice and histologically free of damage, and (ii) grade-1 mild damage from disease models (i.e., DBA/J43 or blast-induced TBI4) at early stages in which no damage was yet apparent but could be present subclinically. Among the 56 nerves, images from 38 nerves (one to two images per nerve from 38 mice) were assigned to the training/validation set (24 grade-1 nerves: 12 with grade-1 no apparent damage and 12 with grade-1 mild damage; 12 grade-2 nerves; and two grade-3 nerves) and images from the remaining 18 nerves (one image per nerve from 18 mice) were assigned to the test set (12 grade-1 nerves: six with grade-1 no apparent damage and six with grade-1 mild damage; six grade-2 nerves). Axon number in mice is highly dependent on...
AxonDeep: Optic Nerve Axon Segmentation With DL

Obtaining Reference Axon Segmentations to be Used for Training and Validation

As our deep-learning networks provide pixel-based marking of axons, our reference standard to train the network and optimize hyperparameters needed to include complete segmentations of the axons. In other words, we needed to obtain binary axon masks (white = axon pixels; black = non-axon pixels) alongside the original images to train the networks. The supervised network required all input images for training to have complete segmentation information and the semisupervised network still required complete segmentation information for a subset of images. Because obtaining completely traced axons from scratch is labor intensive (even for obtaining the subset of complete segmentations needed for the semisupervised approach), for training purposes, our strategy to obtain complete segmentations involved manually marking axon centers in combination with manually correcting the boundaries of an automated segmentation. (Note that, as discussed later, our strategy for obtaining complete tracings for evaluation purposes on the test set did not involve an automated step, as was used in the training stage, to avoid any bias associated with involving an automated segmentation.)

More specifically, obtaining the reference tracings to be used for training involved the following. From the 4140 × 3096 image(s) available per nerve for training purposes, we first randomly selected one of the images and cropped a subfield of size 1024 × 1024 (location selected at random) for purposes of obtaining complete segmentation tracings. On each of these 1024 × 1024 subfields, we ran the baseline AxonJ approach followed by additional morphological smoothing processes to obtain a smooth starting segmentation for purposes of manual editing. Figure 2B shows an example segmentation of the axon image in Figure 2A. In addition, we independently obtained manual center-point markings by placing a center point at an approximate center of each axon (dead/dying axons are marked separately, but not used in this work), as shown in Figure 2C.

Next, the border of each segmented axon was detected, and the contours were converted into representative keypoints. Here we used a simple heuristic approach to generate the points. More specifically, the Ramer Douglas Peucker algorithm was used to generate a keypoint representation. This greedy algorithm iteratively found a polyline that is close to the contour we segmented, with the maximum distance from the interpolated line between keypoints and the true edge was less than a predetermined distance ε. Here we used an ε of 5.0 pixels. These keypoints were then visualized with a GUI interface, as shown in Figure 2D.

We next combined the results from the manually marked centers (Fig. 2C) and smoothed AxonJ contours as follows: (1) False positives (i.e., axons segmented by AxonJ, but not marked by the human expert) were excluded from the mappings; (2) false negatives (i.e., axons not segmented by AxonJ) were annotated with a small starting circle. An example pruned map can be seen in Figure 2E. Finally, the contours were manually corrected by changing the locations (marked in red) of the interactive keypoints shown in Figures 2E and 2F.

Some relate to the fibrotic and gliotic changes that drastically alter gross nerve appearance. The current study was designed to emphasize training AxonDeep to recognize axons from nerves with more of a continuum in gross appearances (normal, mild damage, moderate damage) and limited testing to only grade-1 and grade-2 nerves. The remaining 36 mild and moderate nerves in the combined training/validation set were further randomly divided into 26 nerves to be used for training and 10 nerves to be used for validation. Note that, as is standard practice with deep-learning techniques, the training process was used for automatically determining the trainable network weights, whereas the validation process was used for deciding hyperparameters and tuning the network. The random division of the 36 mild/moderate nerves into training and validation sets resulted in 17 grade-1 nerves and nine grade-2 nerves in the training set, and seven grade-1 nerves and three grade-2 nerves with moderate damage in the validation set. A reference segmentation was manually obtained on a 1024 × 1024 subfield as described in the next section on each of the 36 mild/moderate nerves allocated to the training/validation sets (6762 total axons on the 26 training images and 2668 total axons on the 10 validation images); however, all 50 available 4140 × 3096 full-sized images (in the training set) were also used as unlabeled images (with more than 150,000 axons total) for helping to train the semisupervised approach (see semisupervised approach).

Obtaining Reference Axon Segmentations to be Used for Training and Validation

As our deep-learning networks provide pixel-based marking of axons, our reference standard to train the network and optimize hyperparameters needed to include complete segmentations of the axons. In other words, we needed to obtain binary axon masks (white = axon pixels; black = non-axon pixels) alongside the original images to train the networks. The supervised network required all input images for training to have complete segmentation information and the semisupervised network still required complete segmentation information for a subset of

Obtaining Reference Axon Segmentations to be Used for Training and Validation

As our deep-learning networks provide pixel-based marking of axons, our reference standard to train the network and optimize hyperparameters needed to include complete segmentations of the axons. In other words, we needed to obtain binary axon masks (white = axon pixels; black = non-axon pixels) alongside the original images to train the networks. The supervised network required all input images for training to have complete segmentation information and the semisupervised network still required complete segmentation information for a subset of
Figure 2. The coupling of manually corrected automated segmentations with manual tracing of center marks of axons was used to construct the reference segmentations for training. (A) Light micrograph of a paraphenylenediamine–stained optic nerve cross section in enhanced format (histogram equalized for better visualization) used as an input image for training. (B) Binary AxonJ result. (C) Manually marked axon centers. (D) AxonJ result in (B) displayed using interactive GUI (clicking on an axon would cause interactive keypoints to appear). (E) Pruned AxonJ result with red interactive keypoints for a sample axon contour indicated. (F) Result of editing sample axon contour. By combining automated segmentation (B), manually traced center marks (C), and manual corrections (E, F), references for training data can be obtained. Scale bar: 5 μm.

Obtaining Reference Axon Counts and Segmentations to be Used for Final Evaluation (Testing)

For the final one-time evaluation of each trained approach on the test set, we also obtained reference manual axon counts and complete segmentations. However, unlike in training, in obtaining complete segmentations we traced the boundaries of the axons from scratch (i.e. not editing an automated result) to avoid any bias associated with use of an automated approach. More specifically, for each of the test images, after obtaining a random $1024 \times 1024$ crop from a raw full-sized image, as shown in Figure 3(A-B), approximated axon centers within the bounding box were manually marked to evaluate the axon count predictions (Fig. 3C). The axons close (less than 120 pixels) to image borders were ignored in the evaluation because it is hard to accurately assess the axons when part of them are outside the border. For the pixel-based evaluation of the axon segmentations, a randomly chosen $400 \times 400$ sized cropped subfield was used for complete tracing by the same expert (rather than editing an existing segmentation as was done for training). Figure 3D shows an example selection of the traced area in the bounding box. Note that a description of the metrics used for comparing the reference standard to the automated approaches for final evaluation appears in the Evaluation subsection.

Deep-learning Approach 1: FCN for Segmentation

Figure 4 provides an illustration of the architecture of our FCN used to provide a pixel-level segmentation of the axons in the image. Overall, it uses an
encoder-decoder framework with use of an established encoder (a deep ResNeXt-50\textsuperscript{49} in our case) in combination with a more lightweight, asymmetric (compared to the encoder) decoder. This type of architecture is an example of a feature pyramid network and has been shown to be successful in image-segmentation and detection tasks.\textsuperscript{23,50–52} For the encoder of the network, we used a deep ResNeXt-50\textsuperscript{49} architecture, which uses a very similar structure as ResNet\textsuperscript{53} while offering better performance. Use of a deep encoder was motivated by the need to consider the context of a relatively large surrounding region in determining whether a pixel belongs to an axon. Note that the asymmetry of the encoder-decoder is different from the symmetric structure used in the popular U-Net framework popular in many pixel-based medical segmentation tasks.\textsuperscript{23}

For the training data, in order to minimize the chance of adjacent axons being segmented as a single object, we separately predicted the axons and the borders between adjacent axons with two channels as output (as shown in Fig. 4 and Figs. 5B and 5C). To generate the borders for training, we performed a morphological dilation on each manually corrected binary axon segmentation and found its overlap with the rest of the morphologically dilated axons. This overlap was defined as the border between adjacent axons (shown in Fig. 5C). Note that at test time, these segmented borders can be used to better separate attached axon segmentation masks. As a result, a single channel of axon image with resolution divisible by 32 was used as input, and the network output consisted of two channels: an axon mask and the borders between nearby axons.

![Figure 3](image-url)
Figure 4. An illustration of the network for our FCN approach. The backbone network is paired with a light-weight feature pyramid-like decoder. The network takes a single channel axon image as input and outputs an axon probability map and borders between adjacent axons.

Figure 5. Examples of training data from analyses of optic nerve micrographs. An example of a (A) raw optic nerve image input, (B) generation of an axon mask (axons appear as white, and non-axons as black), and (C) borders between adjacent axons. Scale bar: 5 μm.

In order to train the network, similar to an approach used previously, at each training step, the total loss was a function combining a soft Dice loss ($L_{\text{Dice}}$) and binary cross entropy loss ($L_{\text{BCE}}$):

$$L_{\text{FCN}} = L_{\text{BCE}} - \log L_{\text{Dice}},$$

where

$$L_{\text{BCE}} = -\sum_i (y_i \cdot \log \hat{y}_i + (1 - y_i) \cdot \log (1 - \hat{y}_i))$$

and

$$L_{\text{Dice}} = \frac{\text{Intersect}}{\text{Union}} = \frac{2 \sum_i y_i \cdot \hat{y}_i + \alpha}{\sum_i y_i + \sum_i \hat{y}_i + \alpha},$$

where $y_i$ and $\hat{y}_i$ are corresponding truth and prediction at each pixel location $i$. Here $\alpha = 1$ provides numerical stability and prevents cases of zeros.

At inference time, to generate the final output of the axon segmentation, a watershed algorithm was applied with axon segmentations minus adjacent borders as basins and axon segmentations as masks.

Deep-learning Approach 2: Semisupervised Learning

To be able to take advantage of both labeled (time-consuming to acquire) and unlabeled data (i.e., just the input images without any manual annotations) during training, we also used a semisupervised learning approach based on incorporating the FCN architecture described above into a generative-adversarial-network (GAN) setup. In a traditional GAN, a generator subnetwork $G$ (designed to generate realistic images from noise) is simultaneously trained with a discriminator network $D$ (designed to differentiate between real images and those generated by the generator). Because the subnetworks are trained together (in an alternating fashion), the networks “compete” in a minimax game.
so that ultimately the generator subnetwork can generate realistic images on its own (to fool the discriminator to try to win the competition). In other words, the addition of the discriminator subnetwork is used to help in the training of the generator subnetwork, but the discriminator subnetwork is not needed once training is complete. (Note that in other applications, one can actually keep the discriminator rather than the generator for purposes of having a starting point for a semisupervised classifier as in Salimans et al.\textsuperscript{58} and take advantage of both labeled and unlabeled data for image-level classification tasks.) However, in our case, rather than just generating realistic-looking images, we wish to take input images and produce a resulting segmentation. Thus we follow a similar approach as previously used\textsuperscript{59,60} and a GAN-like framework to help train a segmentation network with both labeled and unlabeled data. An illustration of our approach can be seen in Figure 6. Intuitively, instead of having a generator $G$ to take noise and generate realistic-looking images, we use a subnetwork $G$ to take an input image and produce a resulting segmentation and a subnetwork $D$ to differentiate (at a pixel level) between segmentations produced by the network $G$ and reference segmentations (i.e., being able to tell whether the segmentation, at each pixel, was from the generator subnetwork or used as the reference ground-truth). Here both $G$ and $D$ for our semisupervised learning approach use the same network structure following the FCN approach in the previous section. Note that after training is complete, as with a traditional GAN, we can just use the trained $G$ subnetwork (and not use the discriminator subnetwork) in practice (e.g., during testing) so that the inputs/outputs are just like they would be with the FCN described in the prior section.

Just as with a traditional GAN, each training iteration alternates between training the $G$ network and training the $D$ network. When training the $G$ network, the weights in the $D$ network are frozen and will not be updated (and vice versa). Both labeled and unlabeled images can be used to update the weights (in each epoch, all 26 labeled images in the training set are used and a random subset of 26 of the 50 unlabeled images, to match the number of labeled images, is used). In training $G$ (i.e., the segmentation network), when the input is a labeled image (i.e., the input $X$ has a corresponding reference segmentation $Y$), the current version of the generator is used to create the generated segmentation map, $G(X)$. A “mixed” image $Mix(G(X), Y)$ is then created by randomly selecting each pixel to have a value either from the generated segmentation map $G(X)$ or the reference image $Y$. A mask image $M$ is created with values of 1 at locations where the pixels came from $Y$ and 0 where pixels came from $G(X)$. The loss function used is the binary cross-entropy loss between the discriminator output (using the original image and mixed image as input) and the mask image of 1s and 0s:

$$L_D = L_{bce}(D(Mix(G(X), Y), X), M).$$

As before, this loss function will encourage the discriminator to have an output of 0 at locations where the input was from the generator network and an output of 1 at locations where the input was from the actual reference segmentation.

In training $D$, when the input is a labeled image (i.e., the input $X$ has a corresponding reference segmentation $Y$), the current version of the generator is used to create the generated segmentation map, $G(X)$. A “mixed” image $Mix(G(X), Y)$ is then created by randomly selecting each pixel to have a value either from the generated segmentation map $G(X)$ or the reference image $Y$. A mask image $M$ is created with values of 1 at locations where the pixels came from $Y$ and 0 where pixels came from $G(X)$. The loss function used is the binary cross-entropy loss between the discriminator output (using the original image and mixed image as input) and the mask image of 1s and 0s:

$$L_D = L_{bce}(D(Mix(G(X), Y), X), M).$$

where $\beta = 10$ to balance supervised loss and GAN loss and $L_{bce}$ is $L_{BCE}$ with a plain Sigmoid function:

$$L_{bce} = -\sum_i \left( y_i \cdot \log \frac{1}{1 + e^{-\hat{y}_i}} + (1 - y_i) \cdot \log \left( 1 - \frac{1}{1 + e^{-\hat{y}_i}} \right) \right).$$

Note that in this case, the generator wants the discriminator to output a value of 1 at each pixel to indicate that the discriminator is fooled into thinking the generator’s output is a true manual tracing. When the input is an unlabeled image $X$, a reference labeled image is not available, so the loss function is solely based on the binary cross-entropy loss indicated in the second part of the equation above:

$$L_G = L_{bce}(D(G(X), X), 1).$$

Similarly, in training $D$ (the discriminator that tries to differentiate, on a pixel-by-pixel level, using both the segmentation map and original input image as input, whether the resulting segmentation came from the generator/segmentation network or a manual reference), when the input is an unlabeled image $X$, the loss function is also based on a binary cross-entropy loss, but this time to encourage the generator’s output to result in a value of 0 at each pixel location from the discriminator (i.e., the discriminator wants to correctly predict that the generator’s output is not a reference segmentation):

$$L_D = L_{bce}(D(G(X), X), 0).$$
Figure 6. Training the semisupervised approach alternates between updating the generator network weights and updating the discriminator network weights. Both the generator and discriminator have the same basic underlying architecture as the FCN approach. (After training only the generator network is retained as the final segmentation network.) The input to the generator is a raw axon image and the input to the discriminator is the raw axon image plus a segmentation output. The network uses both labeled data (with both axon images \(X\) and reference segmentations \(Y\) available) and unlabeled data \(X\) in updating the weights. (A) In updating the generator weights with labeled data, the loss function encourages the generator output to match the reference segmentation and to “fool” the discriminator into thinking that the generated output is a reference output. (B) In updating the generator weights with unlabeled data, the loss function encourages the generator to “fool” the discriminator in thinking that the generated output is a reference output. (C) In updating the discriminator weights with labeled data, the loss function encourages the discriminator to correctly output a 1 for pixels coming from a reference
segmentation and a 0 for pixels coming from a generated segmentation. Note that a randomly mixed segmentation image (between the generated segmentation and reference segmentation) is provided as part of the input to the discriminator. (D) In updating the discriminator weights with unlabeled data, the loss function encourages the discriminator to correctly output a 0 for pixels not coming from a reference segmentation.

Training Strategies

To implement our approaches, the PyTorch framework was used for all of the experiments. A single Nvidia GeForce 1080Ti GPU with 12GB memory was used for training and testing. To train the networks, we used mini-batch stochastic gradient descent (Adam) as the optimizer.51,62 The input image size for training was set to be $512 \times 512$, with a batch size of four, randomly cropped from training images. Before each training mini-batch, data augmentation was applied to the input images with random resize, cropping, flip, Gaussian blur, and contrast changes.

For the learning rate, the FCN approach used an initial value of 1e-3 and was divided by a factor of 10 when the training loss plateaued. For the semisupervised approach, a learning rate of 1e-4 for the $D$ network and 1e-3 for the $G$ network was used.

To determine when to stop training, the networks were evaluated on the validation set every 10 epochs. To balance the quality of axon segmentation and the count of axons generated from post-processing, we used an equal combination of accuracy and the absolute axon number difference between prediction and reference as the metric for the segmentation. More specifically, the receiver operating characteristic curves with respect to prediction thresholds were measured and area under the receiver operating characteristic curves was calculated as a measure of pixel accuracy. The network parameters providing the highest performance on the validation set was used as our final approach for evaluations.

Evaluation

The AxonJ approach and our trained FCN and semisupervised approach were used to provide both axon counts and complete segmentations for each of the 18 test images (corresponding to 3317 marked axon centers and 1103 fully traced axons from the first expert). The axon counts and complete tracings (obtained as discussed previously) from a single expert were used as the reference for comparisons with the automated approaches. The absolute percent axon count difference (as a percentage of the reference count, i.e., the absolute difference in counts between the approach and the first expert divided by the number of counts measured by the first expert and then multiplied by 100) and the Dice similarity coefficient (a pixel-based measure of similarity) between each automated result and the reference were computed for each image and averaged across all images. In addition, the Pearson correlation coefficient ($R$) between the counts provided by each automated approach and the manual reference counts were computed. The axon counts and complete tracings were also obtained from a second expert and compared to those from the first expert (again, measuring the average absolute count difference as a percentage, the Pearson correlation coefficient of the counts, and the Dice similarity metric of the complete tracings).

In addition, to further evaluate the semisupervised approach against a similar structured FCN, we compared these two approaches with different sizes of training data size. Both approaches were trained on different sizes of randomly chosen labeled images: 100% (26 images), 50% (13 images), 25% (6 images), 10% (2 images). Meanwhile, for each training data size, the semisupervised learning approach was also trained on all the unlabeled/unmarked data (50 images). Metrics of absolute count difference, correlations, and Dice coefficients were then compared between the two models trained on each data size.

Results

Overall quantitative results are summarized in Table 1 (with results separated by damage grade available in Supplementary Table S1), with the semisupervised approach having the best overall performance. Comparing the counts resulting from the semisupervised learning approach with the reference counts resulted in a mean absolute percent difference in counts of 4.4%, with a Pearson’s correlation coefficient of $R = 0.97$. The Dice coefficient (a measure of the relative overlap of the pixel-based segmentations) was 0.81 for the semisupervised approach. In contrast, the mean absolute percent difference in counts for the AxonJ approach was significantly ($P < 0.05$ using paired $t$-test) higher (9.0%), the Pearson’s correlation coefficient was significantly ($P < 0.05$ using Williams’ test) lower ($R = 0.86$), and the Dice coefficient was significantly ($P < 0.05$ using paired $t$-test) lower (0.70). The
Table 1. Comparing the Performance of Each Approach on the Final Testing Set

<table>
<thead>
<tr>
<th>Metric</th>
<th>Expert 2</th>
<th>AxonJ</th>
<th>FCN</th>
<th>Semisupervised</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Relative to Expert 1):</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Abs. % diff.</td>
<td>6.6% ± 5.5%</td>
<td>9.0% ± 8.1%</td>
<td>6.2% ± 3.6%</td>
<td>4.4% ± 3.3%</td>
</tr>
<tr>
<td>$R$</td>
<td>0.93</td>
<td>0.86</td>
<td>0.93</td>
<td>0.97</td>
</tr>
<tr>
<td>Dice coefficient</td>
<td>0.79 ± 0.04</td>
<td>0.70 ± 0.10</td>
<td>0.80 ± 0.05</td>
<td>0.81 ± 0.04</td>
</tr>
</tbody>
</table>

Abs. % diff., absolute percent difference; $R$, Pearson’s correlation coefficient.
The absolute percent difference and Dice coefficient values are reported as the mean ± the standard deviation across the corresponding test images.

FCN approach and second expert had similar results that were just slightly worse than the semisupervised approach. Visually, the semisupervised approach also performed best, yielding qualitatively more confident and clean predictions than the FCN approach (Fig. 7). Compared with the AxonJ results, fewer false-positive regions were segmented.

When visually comparing the performance of the semisupervised approach versus FCN with different sizes of training data (Fig. 8), at each training size, the semisupervised approach showed qualitatively cleaner segmentations and less noise, with borders between axons more clearly seen compared to the FCN approach. Quantitatively, with the exception of the absolute percent difference in counts when using only 10% (i.e., two images) of the labeled images for training, the semisupervised approach had a better performance (Table 2). The discrepancy between Dice metric and the absolute percent error of the counts in the 10% case may have been in part due to the counts being more affected by the postprocessing stage in its removal of smaller objects.
Figure 8. Comparing the resultant axon segmentations generated by the fully convolutional network and semisupervised learning approaches with incremental decreases in the amount of training data. Axon segmentations (in blue) generated under direct training by (A–D) the FCN and (E–H) semisupervised learning approach from the same microscopic field. Non-axons are represented in black and red markings highlight instances in which the algorithm of each approach detected borders between adjacent axons to prevent neighboring axons from being segmented as a single axon (A–H). Scale bar: 5 μm.

Table 2. Comparison of FCN and Semisupervised Approach When Training on Different Training-Set Sizes

<table>
<thead>
<tr>
<th></th>
<th>FCN 100%</th>
<th>FCN 50%</th>
<th>FCN 25%</th>
<th>FCN 10%</th>
<th>Semi 100%</th>
<th>Semi 50%</th>
<th>Semi 25%</th>
<th>Semi 10%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abs. % diff.</td>
<td>6.2%</td>
<td>5.6%</td>
<td>8.1%</td>
<td>9.0%</td>
<td>4.4%</td>
<td>4.3%</td>
<td>6.2%</td>
<td>9.4%</td>
</tr>
<tr>
<td>R</td>
<td>0.93</td>
<td>0.93</td>
<td>0.87</td>
<td>0.82</td>
<td>0.97</td>
<td>0.96</td>
<td>0.90</td>
<td>0.83</td>
</tr>
<tr>
<td>Dice coeff.</td>
<td>0.80</td>
<td>0.79</td>
<td>0.78</td>
<td>0.57</td>
<td>0.81</td>
<td>0.80</td>
<td>0.78</td>
<td>0.71</td>
</tr>
</tbody>
</table>

Abs. % diff., absolute percent difference; R, Pearson’s correlation coefficient; Semi, semisupervised approach; Dice coeff., Dice coefficient.

The absolute percent difference and Dice coefficient values are reported as the mean across the corresponding test images.

Discussion and Conclusion

Overall, in this work, we proposed a deep fully convolutional network named AxonDeep for segmenting axons in paraphenylenediamine-stained optic nerve cross sections from mice and presented results of training this network using a supervised approach (using only labeled data) and a semisupervised approach (extending the network in a generative-adversarial-network framework for purposes of training with both labeled and unlabeled data). Our results show a significant improvement over AxonJ using the semisupervised learning framework both visually and based on quantitative metrics comparing axon counts and axon segmentations with an expert reference. In the final test set, we did not include cases of severely damaged nerves as it was already known that AxonJ would perform poorly in such cases. Although severe nerves were not included as part of supervised training and evaluation, here we include a few qualitative examples to indicate the increased ability of the semisupervised approach (AxonDeep) in segmenting axons of damaged nerves versus AxonJ. For example, Figures 9B to 9E show example results of AxonJ in cases of severely damaged nerves. In these cases, there are around 500 axons counted with AxonJ, compared to less than 100 axons annotated by an expert. The output from the semisupervised learning approach can be seen in Figures 9C to 9F. Thus the performance of this iteration of AxonDeep to recognize axons from some severely damaged nerves appears...
promising, but it is important to reiterate the caveat that AxonDeep has only been validated for use on nerve images with normal to moderate degrees of damage.

Different from existing deep-learning approaches for providing axon counts, such as AxoNet,\textsuperscript{28} we directly segment axons. Use of the semisupervised training approach to be able to still use untraced images was one of our strategies for more effectively dealing with the challenge and time-consuming nature of obtaining complete reference segmentations, as is needed for training. Another strategy that we used (for training purposes only) to help address the difficulty in obtaining complete reference segmentations, as previously described in more detail in the methods, was to develop an approach for editing existing segmentations rather than tracing completely from scratch. Before editing, to help avoid any bias in the number of axons marked by the existing segmentation, we only provided the starting automated result for axons whose center points were independently marked by an expert. Thus, although the boundaries of axons themselves for purposes of training may have had a bias toward the starting automated approach, the number of axons in the complete reference segmentations matched the counts provided manually and thus were not biased by the automated approach. (In a prior iteration on a different dataset, we considered editing the automated result directly without this step, but decided it had the potential for biasing the results towards the automated result too much.) As previously mentioned, although we believed this was an accept-
quantifying successive slices of the same optic nerve (which should have near identical axon numbers), or performing the analysis twice on each section, with the microscope slide rotated by 90° in the second iteration, such that averaging can be used to better control for sampling variabilities.

Implementation of AxonDeep could assist in the execution of a broad range of experiments. The mouse optic nerve contains \( \sim 50,000 \) axons, although that number can vary widely according to genetic background.\(^4\) Quantifications of axon number is a gold-standard for measuring disease severity,\(^1\) but the labor-intensive nature of manual axon counting often results in studies instead using qualitative grading scales.\(^2\)\(^,\)\(^3\) As with all of the tools that the field has put forward to count RGC somas\(^4\)(\(^-\)\(^7\)) or axons\(^9\)(\(^-\)\(^7\)) in various animal models, the automated counts performed by AxonDeep greatly reduce the labor of manual counts and eliminate the possibility of user-to-user, lab-to-lab, or model-to-model variability inherent to subjective grading scales. An advantage of AxonDeep is that it performs axon segmentations, as well as counts. Thus it will also be possible to study whether axon size and shape vary during disease state. Given the high interest that has existed for many years in studying differential susceptibility of RGCs with comparatively large versus small soma,\(^7\) and the interesting energetic differences in large versus small axons,\(^8\) AxonDeep could be used to study many quantitative aspects of axon morphology that were previously not practical.

**Acknowledgments**

The authors acknowledge use of the University of Iowa Central Microscopy Research Facility (a core resource supported by the Vice President for Research & Economic Development, the Holden Comprehensive Cancer Center, and the Roy J. and Lucille A. Carver College of Medicine) and the University of Iowa Multidisciplinary Investigations in Visual Science service cores (a resource supported by a P30 grant from the NIH/NEI to the University of Iowa). The contents do not represent the views of the U.S. Department of Veterans Affairs or the U.S. Government.

Supported by Grants I50 RX003002 (WD, AHB, MGA, MKG); T32DK112751 (AHB); P30 EY025580, I01 RX001481, and R21 EY029991 (MGA).

Disclosure: **W. Deng**, None; **A. Hedberg-Buenz**, None; **D.A. Soukup**, None; **S. Taghizadeh**, None; **K. Wang**, None; **M.G. Anderson**, None, **M.K. Garvin**, None

* WD and AHB are joint first authors.
‡ MGA and MKG are joint senior authors.

**References**


51. Seferbekov SS, Iglovikov V, Buslaev A, Shvets A. Feature Pyramid Network for Multi-Class Land


