nature methods

ARTICLES
https://doi.org/10.1038/s41592-018-0049-4

High-precision automated reconstruction of neurons with flood-filling networks

Michał Januszewski[1], Jörgen Kornfeld[2], Peter H. Li[3], Art Pope[3], Tim Blakely[4], Larry Lindsey[4], Jeremy Maitin-Shepard[3], Mike Tyka[4], Winfried Denk[2] and Viren Jain[3]*

Reconstruction of neural circuits from volume electron microscopy data requires the tracing of cells in their entirety, including all their neurites. Automated approaches have been developed for tracing, but their error rates are too high to generate reliable circuit diagrams without extensive human proofreading. We present flood-filling networks, a method for automated segmentation that, similar to most previous efforts, uses convolutional neural networks, but contains in addition a recurrent pathway that allows the iterative optimization and extension of individual neuronal processes. We used flood-filling networks to trace neurons in a dataset obtained by serial block-face electron microscopy of a zebra finch brain. Using our method, we achieved a mean error-free neurite path length of 1.1 mm, and we observed only four mergers in a test set with a path length of 97 mm. The performance of flood-filling networks was an order of magnitude better than that of previous approaches applied to this dataset, although with substantially increased computational costs.

Computers have been used to reconstruct neural 'wires' since the 1970s[1], mainly to capture and display the annotation decisions made by human tracers[2–4]. The use of computers to make those decisions began in earnest after improved volume electron microscopy (EM) methods[5] started yielding datasets for which complete analysis by human annotation would have been prohibitively laborious[6], and the process soon came to rely heavily on machine learning (ML). Although ML-based segmentation algorithms determined most cell boundaries correctly, detection and correction of the remaining errors still required human proofreading involving inspection of fragments (supervoxels)[7,8] or manual 'skeletonization'[9–11].

A recent estimate put the amount of human labor needed to reconstruct a 100^3-μm^3 volume at more than 100,000 h, even with an optimized pipeline[12]. Current manual annotation workflows could be made more efficient still, but are ultimately limited by the need to view all the data. A reduction of the proofreading time by multiple orders of magnitude will require algorithms that not only are substantially more accurate but also 'know their own limits' and provide a list of potentially erroneous segmentation decisions[13].

State-of-the-art automated neurite reconstruction typically starts with a convolutional network that infers the likelihood of each image location belonging to a neurite, using the intensities of the voxels at and near that location[14–18]. Then a different algorithm clusters the 'in' voxels into distinct segments[12,19–22]. Our approach provides supervoxels directly by adding to the classifier a recurrent loop that maintains the current prediction for the object shape. Why is this helpful? Although the cost function we use to train the network is still based purely on a voxel-wise comparison with the training set, the network can now learn to make use of voxels already classified with high reliability. Classification decisions can be based on, for example, whether they result in a more or less plausible shape for the neural process. This is conceptually different from using a segmentation-performance-dependent cost function[23,24]. Here we present a realization of this concept, which we call flood-filling networks (FFNs).

In the following, we (1) describe FFNs and a procedure for combining multiple FFN segmentations to fix errors, (2) compare the accuracy of our method to that of previously published alternatives, using a large-scale songbird dataset, (3) analyze FFN reconstruction errors in detail, and (4) demonstrate generalization of the technique to mouse and *Drosophila* brain tissue through the use of benchmark datasets.

Results

Flood-filling network training, inference, and architecture. We acquired a 96 × 98 × 114 μm region of zebra finch brain imaged with serial block-face EM[25] at a resolution of 9 × 9 × 20 nm. For classifier training, a small fraction (0.02%; 131 million voxels contained in 33 subvolumes of varying size that were distributed throughout the volume) of the dataset was segmented by human annotators with KNOSSOS (408 h). We then used these annotations as a ground truth to train the FFN.

An FFN has two input channels: one for the 3D image data, and one for the current prediction of the object shape, a data structure we call the predicted object map (POM). For each voxel, the POM encodes (using values between 0 and 1) the algorithm's estimate of whether a voxel belongs to the object currently being segmented. During training, we initialized the POM by seeding a single voxel in the center of each 49 × 49 × 25 voxel training example. We automatically generated single-voxel seeds at locations distant from putative cell boundaries to avoid mergers (errors in which two or more processes are erroneously connected to one another). After each iteration of the network inference, the POM's values are used to adjust the network weights via stochastic gradient descent, using a per-voxel cross-entropy (logistic) loss[26] (Fig. 1a,b and Methods).

The outcome of each inference step affects whether and where the field of view (FOV) moves, which voxel classifications are frozen, and when neurite expansion is completed (Methods). Specifically, between inference iterations, the FOV is automatically moved by ±8 voxels (72 nm) laterally and ±4 voxels (80 nm) in the *Z* direction. To optimize the training procedure and remain consistent with the

[1]Google AI, Zürich, Switzerland. [2]Max Planck Institute of Neurobiology, Planegg, Martinsried, Germany. [3]Google AI, Mountain View, CA, USA. [4]Google AI, Seattle, WA, USA. *e-mail: viren@google.com

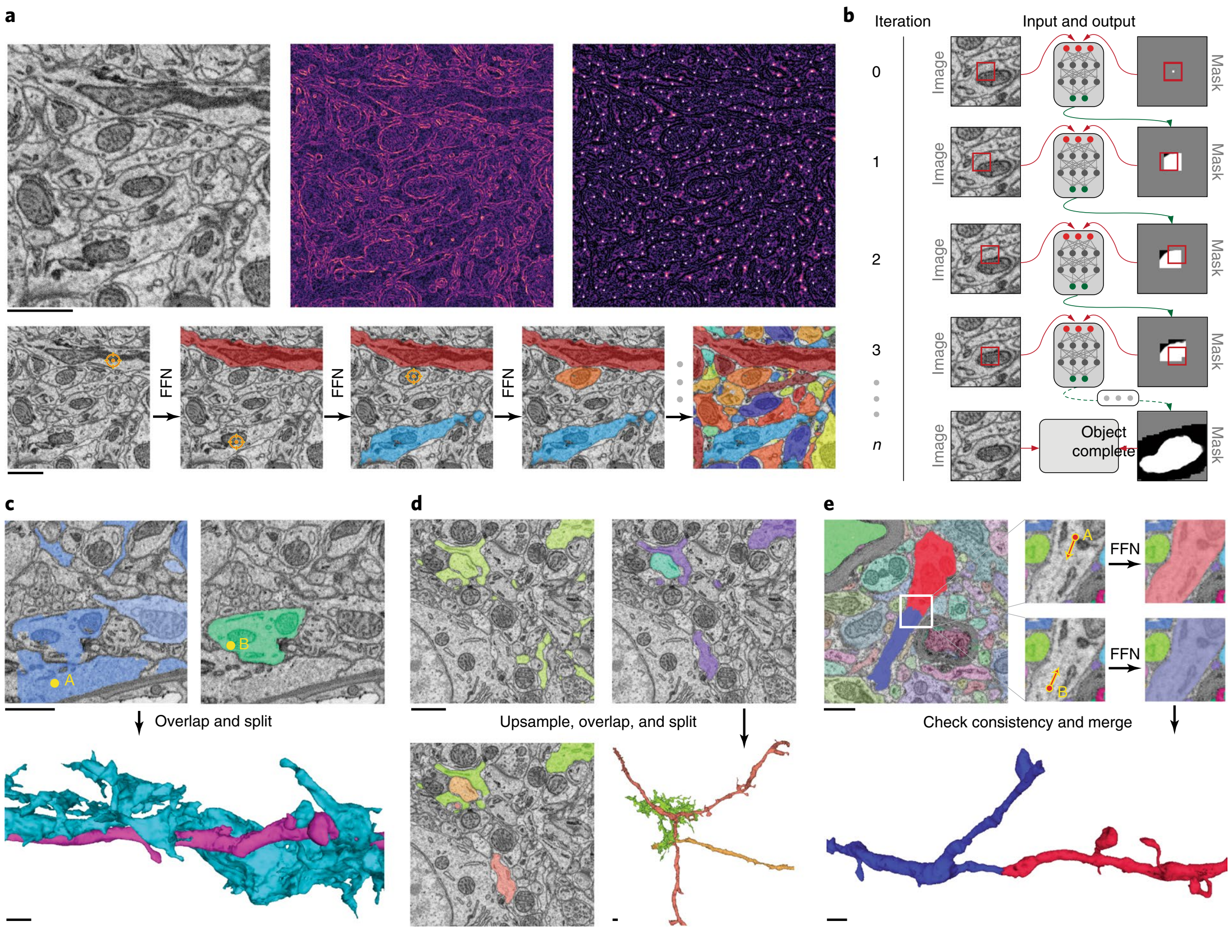

Fig. 1 | The segmentation pipeline, including consensus and agglomeration procedures. a, Segmentation of a subvolume with an FFN. Top row (left to right): EM image data, local intensity-gradient magnitude estimated with the Sobel–Feldman operator (yellow indicates larger magnitudes), and Euclidean distance transform of the gradient magnitude (peaks highlighted by white dots). Bottom row: segmentation progress demonstrated in a slice through a subvolume; the cross-hairs indicate the starting seed point. **b**, The flood-filling inference process for a single object. Red squares indicate the location of the FOV in the EM data (left column) and the POM. Red and green arrows indicate the flow of input and output, respectively. Each iteration (row) consists of one pass through the convolutional network. The first row shows the initialization of the POM with a single pixel (white square) set to a high value. Successive rows show the movement of the FOV and changes to the POM with each iteration. **c**, Multi-seed over-segmentation consensus procedure. Top row: cross-section through the data with the FFN segment seeded by A (left) and by B (right). Note the merger between a glial fragment and a dendritic branch in the left-hand image. Bottom row: surface rendering of the over-segmentation consensus. **d**, Multi-scale over-segmentation consensus. Top row: full-resolution segmentation (left) with an axon–glia merger, and the 2× in-plane-downsampled segmentation (right) with a merger between the same glial fragment and a dendritic branch. Bottom row: multi-scale over-segmentation consensus with both mergers fixed. **e**, Flood-filling agglomeration. Top row: left, split in a dendritic branch; right, zoomed-in views of the region outlined in white on the left, showing segmentations that result when starting from points A (top) and B (bottom). Bottom row: red and blue objects that contain points A and B, respectively, and satisfy the mutual consistency criteria used for FFN agglomeration. Scale bars, 1 µm.

inference procedure, we used a new FOV position only if the POM value of its center voxel exceeded 0.9.

The core architecture of the FFN is a multi-layer convolutional neural network (CNN), which updates the POM values during each iteration on the basis of input data and prior POM values. We chose to use a single 3D FOV size (33 × 33 × 17 voxels, 297 × 297 × 340 nm) for EM-data input, inference output, and recurrent feedback in our FFN implementation.

Irregularity detection and automated tissue classification. Many inference errors occur at data irregularities, such as cutting artifacts or alignment mistakes. Irregularities in the songbird volume EM dataset were too frequent to be ignored but too rare to be effectively learned (affecting at most 1% of the volume). Instead of enriching them in the training set, we detected them via cross-correlation (Methods) and prevented supervoxels to span any of the irregularities.

Segmentation quality often decreased when the neuropil was interrupted by tissue structures such as somata or blood vessels, which are orders of magnitude larger than typical axons, dendrites, and the FOV. To prevent the FFN from venturing into such structures, we trained a separate CNN, which we called the tissue-classification CNN, and used it to delineate such structures.

Hysteresis and approximate scale invariance. The neurite shape reconstructed by an FFN can depend on the placement of the initial seed within the neurite and can change dramatically when the order in which the neurites are reconstructed or the placement of seeds is changed. This variability can, in fact, be used to detect and eliminate mergers, which are hard to fix during proofreading, at the cost of the generation of a few extra splits (errors in which two processes are erroneously disconnected from one another), which are much easier to fix. Specifically, we compared a forward segmentation of the data with its backward segmentation, for which the order of seeds was reversed, and accepted all splits as real (the 'over-segmentation consensus'), which meant that only those mergers that occurred in both segmentations remained. One can generate additional consensus segmentations by varying other segmentation parameters; in addition to reseeding, we explored resampling of the dataset at different resolutions, and found the greatest reduction (82-fold) in the number of mergers (with only a twofold increase in the split rate) for an over-segmentation consensus among five segmentations: standard and reversed seed order at both 9 × 9 × 20 nm and 18 × 18 × 20 nm per voxel, and standard order at 36 × 36 × 40 nm per voxel (Fig. 1c,d and Methods). Combining segmentations is conceptually similar to the process of 'ensembling' in ML, that is, combining the decisions of several different classifiers. Others have applied the averaged predictions of the same classifier to modified versions of the raw data[22,27].

Flood-filling-network agglomeration. To heal splits, we agglomerated segments throughout the volume. Unlike most automated agglomeration approaches, which use separately trained classifiers to find split errors[28,29], our method uses the FFN model itself to perform agglomeration. To determine whether a pair of nearby segments was part of the same neurite, we extracted an ~1-µm^3 subvolume around the point of the segment pair's closest approach, placed seeds in the two objects, and then performed two independent FFN inference runs (i.e., order was not important) while keeping the remaining objects and their voxels fixed (Fig. 1e). If the resulting POMs overlapped to a high degree, the objects were combined (Methods).

Segmentation pipeline. We combined data alignment, tissue classification, FFN inference, over-segmentation consensus, FFN-scored agglomeration, and biological plausibility testing into a pipeline and used it to segment the entire zebra finch volume. The pipeline consisted of the following steps:

1. Fine alignment: the sections within the volumetric dataset were precisely registered by elastic alignment[30]. Compared with a rigid, translation-only alignment, this reduced the number of times the irregularity-detection procedure was triggered 16-fold.
2. Cell-body segmentation: we segmented the parts of the volume corresponding to cell bodies by running an FFN restricted to voxels that had been classified as cell body by the tissue-classification CNN, using seeds manually placed into all 454 somas in the volume (1 human hour was required for manual annotation, but this task could potentially also be automated with 0.97 precision and 0.96 recall[31]). Explicit handling of cell bodies (average diameter, 9 µm) was necessary because their spatial scale was dramatically different from that of neuropil, which comprised the majority of training data for the FFN.
3. Neuropil segmentation: we restricted FFN inference to voxels labeled as neuropil by the tissue classifier and generated five separate segmentations, with standard and reversed seed order at both 9 × 9 × 20 nm and 18 × 18 × 20 nm per voxel, and with standard order at 36 × 36 × 40 nm per voxel.
4. Over-segmentation consensus: we resampled the five partial segmentations to the full resolution of the dataset and reconciled them with over-segmentation consensus to create a base segmentation with a low merge error rate.
5. Agglomeration: we used FFN agglomeration at full resolution to produce a list of merge decisions to combine base segmentation objects. This reduced the split rate by about half (44%) and increased the run length more than tenfold (1,149%).
6. Biological plausibility testing: we filtered the merge decision list so that no agglomerated object contained more than one cell body; 0.006% of the agglomeration decisions were reversed in this process.

Large-scale segmentation accuracy. To measure the accuracy of segmentation results over length scales similar to the path length of neurons in the complete volume, we skeletonized individual neurons[9]. Human annotators used KNOSSOS software (https://knossostool.org/) to manually annotate an individual neuron's structure as a set of nodes and edges. We created a tuning set and a test set, which contained, respectively, 12 and 50 neurons with medians of 0.8 and 1.9 mm, and total path lengths of 13.5 mm and 97 mm (27% and 34% axonal). We used these sets exclusively to optimize the hyperparameters of the segmentation pipeline and to evaluate performance, respectively. While comparing the automatic segmentation to the manually generated skeletons, we found in the skeletons eight mergers between neurites and 66 splits (mostly missed dendritic spines), even after human consensus generation based on at least two independent tracings. Two human experts (M.J. and J.K.) determined the source of each merge discrepancy by examining the raw data and tracings, and fixed the skeletons as needed.

On the basis of the observed overlap with the automated segmentation, we classified each edge of the ground truth skeletons as correctly reconstructed in the segmentation, omitted (one or both end nodes not in any segment), split (nodes in different objects), or part of a merged segment[12,32]. About 1.4% of the total path length in the imaged volume was manually skeletonized. This allowed us to automatically detect all splits that occurred in the skeletonized neurites, but the observed number of skeleton-to-skeleton mergers was a severe underestimation of the true number, because for each cell, only a small fraction of the cells it could merge with were skeletonized. To correctly estimate the merger rate, we also classified a segment as containing a merger if it met both of the following two criteria: it contained at least two skeleton nodes, and any part of the segment extended >2.2 µm from the skeleton to which those nodes belonged.

Finally, we calculated an expected run length (ERL) that measured the average neurite length contained in the segment belonging to a randomly placed starting point. Any starting point falling into a segment that contained a merge error was assigned a zero length. Note that the ERL penalizes merge errors much more severely than most, if not all, other metrics[12,32] currently in use. This is consistent with the fact that manual identification and correction of mergers rather than splits has traditionally taken much more human proofreading time.

Our final reconstruction (FFN-c, which resulted from application of the full pipeline including over-segmentation consensus and agglomeration) reached an ERL of 1.1 mm and contained four mergers on the 97-mm-neurite-length skeleton test set (see Fig. 2 for qualitative analysis and Fig. 3 for quantitative analysis including split counts). None of these mergers were between the ground truth skeletons, and we detected them with the heuristic procedure described above.

To put the performance of FFN-c in perspective, we applied two state-of-the-art alternative approaches to our zebra finch dataset and quantified the segmentation performance. The first ('baseline') approach combined a 3D convolutional neural network trained to produce long-range affinity graphs[16], affinity-graph watershed segmentation[33], and random forest object agglomeration by GALA[21,28]. We optimized the parameters for the affinity-graph watershed procedure by grid search and carried out GALA agglomeration with a random forest classifier trained on the subvolumes of labeled data. The second approach we evaluated was SegEM[12], in which 3D convolutional neural network boundary predictions are over-segmented with watershed. We used the same 3D convolutional network as the baseline, and we optimized the parameters by grid search over full volume segmentations. Among these alternative approaches, the

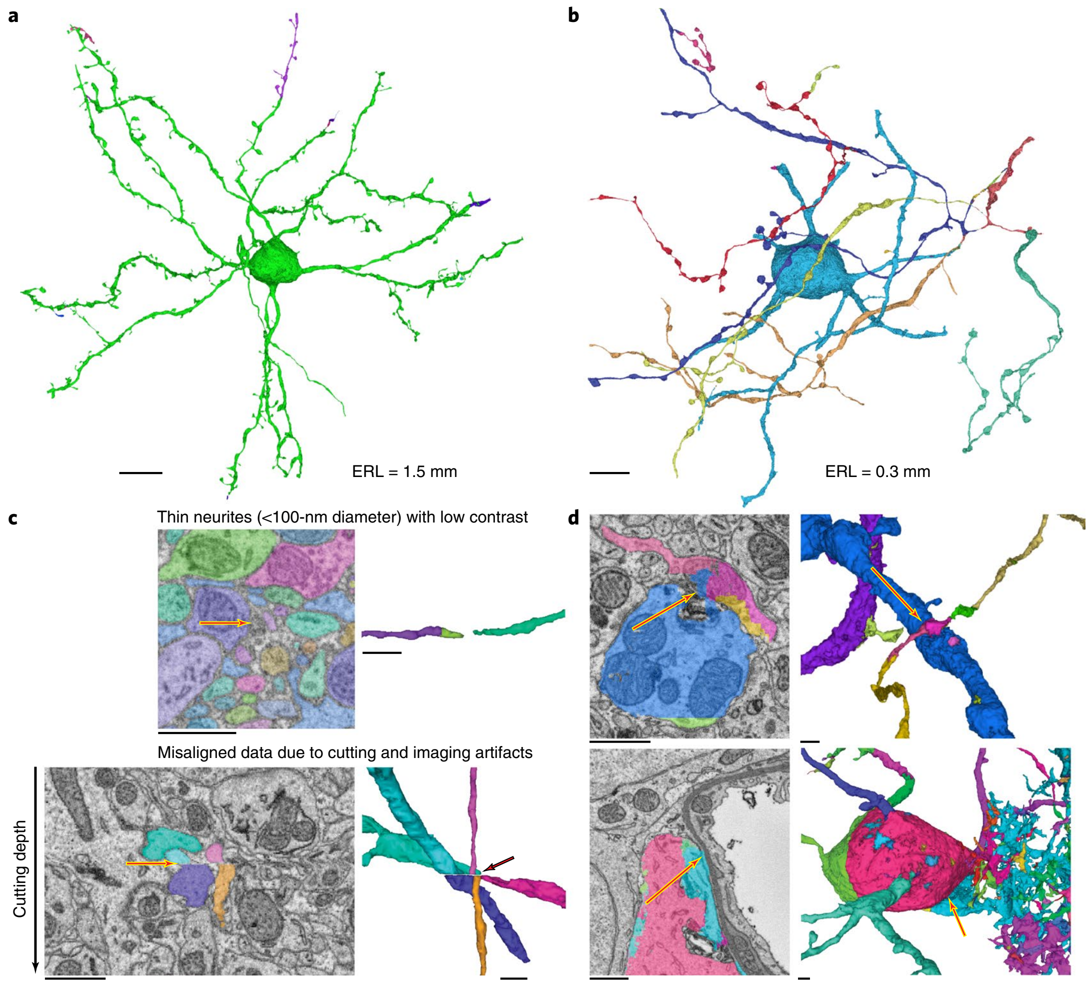
a
ERL = 1.5 mm
b
ERL = 0.3 mm
c
Thin neurites (<100-nm diameter) with low contrast
Misaligned data due to cutting and imaging artifacts
Cutting depth
d

Fig. 2 | Inspection-based analysis of segmentation accuracy. a,**b**, Neuron reconstructions (FFN-c) with largest (**a**) and shortest (**b**) expected run lengths of 1.5 mm and 0.3 mm, respectively. **c**, Zoomed-in views of splits (indicated by arrows) caused by low contrast (top) and slice misalignment (bottom). **d**, Supervoxel mergers (indicated by arrows) between a dendrite and an axon (top) and between a neuronal cell body and a glial process (bottom). Scale bars, 10 µm (**a**,**b**) or 1 µm (**c**,**d**).

baseline method achieved the highest ERL (112 µm; Fig. 3), which was an order of magnitude worse than the FFN results.

Errors by neurite type. We manually classified fragments of neurites in ground truth skeletons as axons or dendrites, and annotated the locations of the base and the head of 182 dendritic spines. We then used these data to measure error rates of the FFN-c segmentation for the different neurite categories (Supplementary Note). We observed that the automated reconstruction was better than human annotation with respect to identification of dendritic spines (95% and 91% recall, respectively). Although precision remained close to 100% in both sets and was slightly higher for the automated results (99.7% and 100% for automated reconstruction versus 98% and 99% for human-generated reconstruction for dendrites and axons, respectively), recall for both dendrites and axons in the automated-reconstruction set was still inferior to that obtained by human annotation (68% and 48% for the automated process versus 89% and 85% for human-generated data for dendrites and axons, respectively).

Other species and imaging methods. FIB-25 is a public dataset of *Drosophila* optic lobe imaged by focused ion beam scanning EM at 8 × 8 × 8 nm that has been used to benchmark segmentation methods[8]. FFNs applied to the test set (23 × 19 × 24 µm, ~100× smaller in volume than the songbird dataset) outperformed all other methods applied to this dataset, including a recently introduced U-Net structured learning technique[17] and a combinatorial energy-based method[29]. Error metrics and visual inspection confirmed that the FFN reconstruction was accurate with respect to the ground truth (see Supplementary Note for details on the experiment and Supplementary Software for FFN code and a demonstration using FIB-25).

Similarly used as a public segmentation benchmark, SNEMI3d is a dataset of mouse somatosensory cortex imaged with serial-section automated tape-collecting ultramicrotome scanning EM at 6 × 6 × 29 nm[34]. FFNs applied to the held-out test set (6 × 6 × 3 µm, ~10,000× smaller than the songbird dataset) achieved 'superhuman' performance according to the competition's leaderboard (http://brainiac2.mit.edu/SNEMI3D/leaders-board as of 10 May 2018) and surpassed 17 of 18 other methods; the remaining one used many dataset-specific training augmentation strategies and 16× test-time augmentation averaging[18]. However, manual inspection of the results suggested that the absolute number of remaining errors was probably too low for accurate comparison of the two top-performing methods (Supplementary Note).

Discussion

FFNs differ from other ML-based segmentation approaches[12,16,20,35,36] in several ways: a recurrent network architecture, the

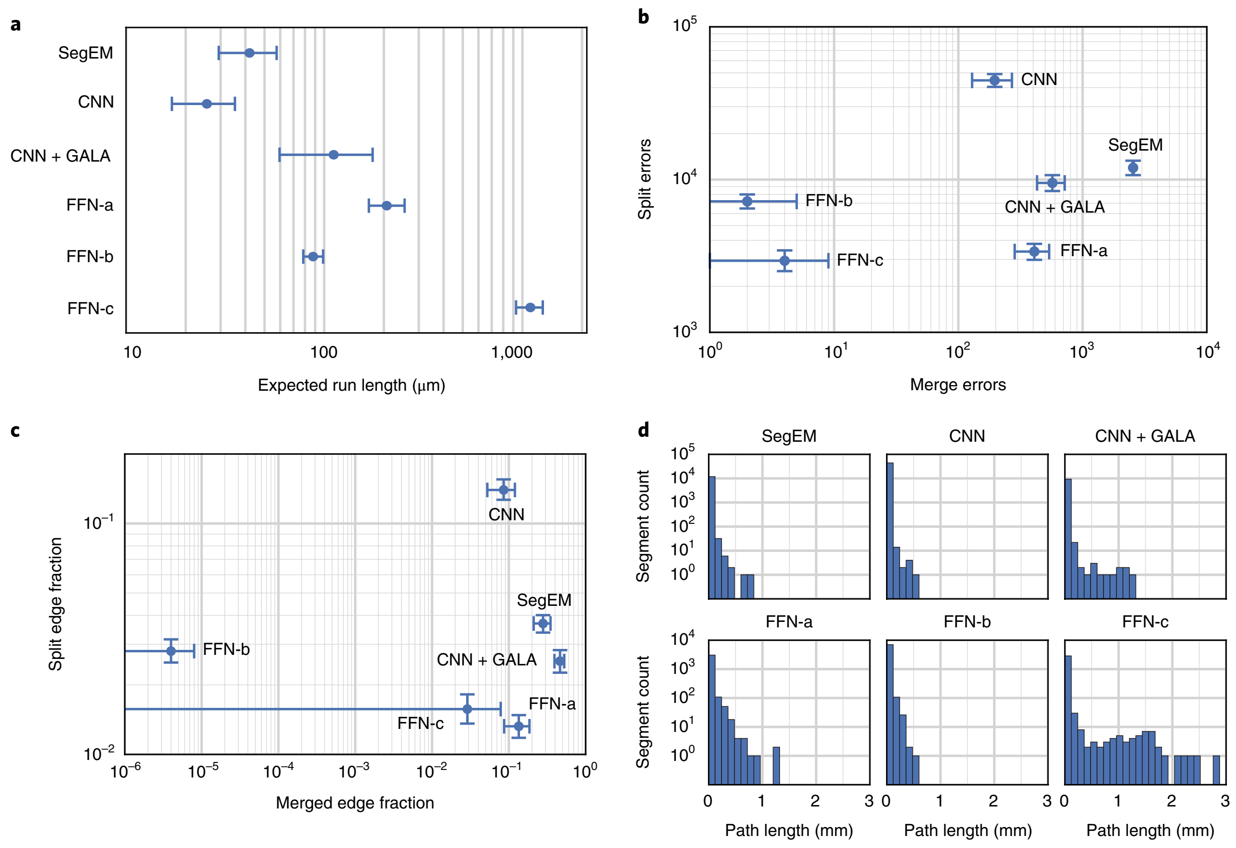

Fig. 3 | Quantitative analysis of segmentation accuracy. a–d, Evaluation and comparison of accuracy based on 50 manually traced and verified skeletons showing (**a**) the expected run length, (**b**) the total number of splits and mergers across all cells, (**c**) the fraction of skeleton edges split versus the fraction contained in merged segments, and (**d**) path length histograms for merger-free segments. FNN-a, single-pass FFN segmentation; FFN-b, FFN segmentation after multi-seed and multi-resolution consensus; FFN-c, result of the entire FFN pipeline, including FFN agglomeration. Error bars represent 95% confidence intervals and were calculated via the bootstrap method with 10,000 resamples.

direct generation of segments (as opposed to reliance on a separate clustering step), and an inference procedure that segments objects one at a time (Supplementary Video). Related work has merged boundary prediction and region filling as a learned watershed algorithm applied to 2D segmentation[37,38], as well as a recurrent network applied to extend the shape of segmented neurites along a single axis[39]. There is also related work in the context of natural image segmentation[40–42]. Unlike the method of Romera-Paredes and Torr[40], FFNs use multiple iterations of the network to delineate a single object. Therefore, objects can grow or shrink across iterations, and the maximal size of a segmented object is not constrained by the network architecture. And unlike the method described by Pinheiro et al.[41], FFNs are trained end-to-end and use an explicit mechanism to specify the object being segmented, rather than treating the central pixel in the FOV as special.

We exploited several additional capabilities of FFNs, such as the ability to reduce mergers by ensembling multiple segmentations generated through variation of the seed point location. Application of these techniques to a roughly 1,000,000-μm^3 volume of songbird brain tissue yielded automatically segmented neurons with an average error-free run length of 1.1 mm.

The main disadvantage of FFNs is the high computational cost. For example, a single pass of the fully convolutional FFN over a full volume is an order of magnitude more computationally expensive than the more traditional 3D convolution-pooling architecture in the baseline approach we used for comparison. This is because multiple and partially overlapping inference computations are required to segment a single object and to implement the sequential nature of multiple-object segmentation. However, we found that the basic building blocks of FFN inference can be used to build pipelines with similar performance but considerably reduced computation costs. For example, we tested a coarse-to-fine approach in which FFN segmentation at a specific resolution of the data (e.g., 1×) was applied only in areas that were not already segmented at a coarser resolution and, on FIB-25, found little performance degradation compared with that of the full-resolution FFN approach, but with computational costs that were an order of magnitude lower (Supplementary Note).

Moreover, the benefits of FFN segmentation are likely to outweigh even substantial additional computational costs, given the savings in human proofreading time that follow from the improvements in reconstruction accuracy. The very low rate of mergers in the FFN reconstruction would greatly reduce the need for manual splitting of undersegmented 3D objects, one of the most laborious parts of proofreading. The large size of the automatically generated segments should make the shape-based prediction of potential splits more reliable and thus make a replacement of the laborious manual segmentation step by focused annotation feasible. As the error-free path length increases, it becomes more and more likely that any segment that contains a segmentation error will also violate one of the known topological properties common to all neurites, such as the expectation that all neurites either connect to a cell body or reach the border of (and thus go beyond) the imaged volume, and such violations could become an efficient way to guide the proofreading process and provide a measure for the residual error rate in the segmentation.

Received: 16 March 2018; Accepted: 3 May 2018;
Published online: 16 July 2018

References

1. Macagno, E. R., Levinthal, C. & Sobel, I. Three-dimensional computer reconstruction of neurons and neuronal assemblies. *Annu. Rev. Biophys. Bioeng.* **8**, 323–351 (1979).
2. Harris, K. M., Jensen, F. E. & Tsao, B. Three-dimensional structure of dendritic spines and synapses in rat hippocampus (CA1) at postnatal day 15 and adult ages: implications for the maturation of synaptic physiology and long-term potentiation. *J. Neurosci.* **12**, 2685–2705 (1992).
3. Ventura, R. & Harris, K. M. Three-dimensional relationships between hippocampal synapses and astrocytes. *J. Neurosci.* **19**, 6897–6906 (1999).
4. Fiala, J. C. Reconstruct: a free editor for serial section microscopy. *J. Microsc.* **218**, 52–61 (2005).
5. Briggman, K. L. & Bock, D. D. Volume electron microscopy for neuronal circuit reconstruction. *Curr. Opin. Neurobiol.* **22**, 154–161 (2012).
6. Jain, V., Seung, H. S. & Turaga, S. C. Machines that learn to segment images: a crucial technology for connectomics. *Curr. Opin. Neurobiol.* **20**, 653–666 (2010).
7. Kim, J. S. et al. Space-time wiring specificity supports direction selectivity in the retina. *Nature* **509**, 331–336 (2014).
8. Takemura, S.-Y. et al. Synaptic circuits and their variations within different columns in the visual system of *Drosophila*. *Proc. Natl. Acad. Sci. USA* **112**, 13711–13716 (2015).
9. Helmstaedter, M., Briggman, K. L. & Denk, W. High-accuracy neurite reconstruction for high-throughput neuroanatomy. *Nat. Neurosci.* **14**, 1081–1088 (2011).
10. Helmstaedter, M. et al. Connectomic reconstruction of the inner plexiform layer in the mouse retina. *Nature* **500**, 168–174 (2013).
11. Cardona, A. TrakEM2: an ImageJ-based program for morphological data mining and 3D modeling. in *Proc. ImageJ User and Developer Conference* 18–19 (Centre de Recherche Public Henri Tudor: Luxembourg, 2006).
12. Berning, M., Boergens, K. M. & Helmstaedter, M. SegEM: efficient image analysis for high-resolution connectomics. *Neuron* **87**, 1193–1206 (2015).
13. Plaza, S. M. Focused proofreading to reconstruct neural connectomes from EM images at scale. in *Deep Learning and Data Labeling for Medical Applications* (eds. Carneiro, G. et al.) 249–258 (Springer, Cham, 2016).
14. Jain, V. et al. Supervised learning of image restoration with convolutional networks. in *Proc. IEEE 11th International Conference on Computer Vision* 636–643 (IEEE, New York, 2007).
15. Ciresan, D., Giusti, A., Gambardella, L. M. & Schmidhuber, J. Deep neural networks segment neuronal membranes in electron microscopy images. in *Advances in Neural Information Processing Systems 25* (eds. Pereira, F. et al.) 2852–2860 (Neural Information Processing Systems Foundation, La Jolla, CA, 2012).
16. Turaga, S. C. et al. Convolutional networks can learn to generate affinity graphs for image segmentation. *Neural Comput.* **22**, 511–538 (2010).
17. Funke, J. et al. A deep structured learning approach towards automating connectome reconstruction from 3D electron micrographs. *arXiv* Preprint at https://arxiv.org/abs/1709.02974 (2017).
18. Lee, K., Zung, J., Li, P., Jain, V. & Seung, H. S. Superhuman accuracy on the SNEMI3D Connectomics Challenge. *arXiv* Preprint at https://arxiv.org/abs/1706.00120 (2017).
19. Andres, B., Koethe, U., Helmstaedter, M., Denk, W. & Hamprecht, F. A. Segmentation of SBFSEM volume data of neural tissue by hierarchical classification. in *Pattern Recognition: Proceedings of the 30th DAGM Symposium* (ed. Rigoll, G.)142–152 (Springer, Berlin, 2008).
20. Kaynig, V. et al. Large-scale automatic reconstruction of neuronal processes from electron microscopy images. *Med. Image Anal.* **22**, 77–88 (2015).
21. Knowles-Barley, S. et al. RhoanaNet pipeline: dense automatic neural annotation. *arXiv* Preprint at https://arxiv.org/abs/1611.06973 (2016).
22. Beier, T. et al. Multicut brings automated neurite segmentation closer to human performance. *Nat. Methods* **14**, 101–102 (2017).
23. Turaga, S. C., Briggman, K. L., Helmstaedter, M., Denk, W. & Seung, H. S. Maximin affinity learning of image segmentation. in *Advances in Neural Information Processing Systems 22* (eds. Bengio, Y. et al.) 1865–1873 (Neural Information Processing Systems, La Jolla, CA, 2009).
24. Jain, V. et al. Boundary learning by optimization with topological constraints. in *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition* 2488–2495 (IEEE, New York, 2010).
25. Denk, W. & Horstmann, H. Serial block-face scanning electron microscopy to reconstruct three-dimensional tissue nanostructure. *PLoS Biol.* **2**, e329 (2004).
26. Goodfellow, I., Bengio, Y. & Courville, A. *Deep Learning* (MIT Press, Cambridge, MA, 2016).
27. Ronneberger, O., Fischer, P. & Brox, T. U-Net: Convolutional networks for biomedical image segmentation. in *Medical Image Computing and Computer-Assisted Intervention—MICCAI 2015* (eds. Navab, N. et al.) 234–241 (Springer, Cham, 2015).
28. Nunez-Iglesias, J., Kennedy, R., Plaza, S. M., Chakraborty, A. & Katz, W. T. Graph-based active learning of agglomeration (GALA): a Python library to segment 2D and 3D neuroimages. *Front. Neuroinform.* **8**, 34 (2014).
29. Maitin-Shepard, J., Jain, V., Januszewski, M., Li, P. & Abbeel, P. Combinatorial energy learning for image segmentation. *arXiv* Preprint at https://arxiv.org/abs/1506.04304 (2015).
30. Saalfeld, S., Fetter, R., Cardona, A. & Tomancak, P. Elastic volume reconstruction from series of ultra-thin microscopy sections. *Nat. Methods* **9**, 717–720 (2012).
31. Dorkenwald, S. et al. Automated synaptic connectivity inference for volume electron microscopy. *Nat. Methods* **14**, 435–442 (2017).
32. Pallotto, M., Watkins, P. V., Fubara, B., Singer, J. H. & Briggman, K. L. Extracellular space preservation aids the connectomic analysis of neural circuits. *eLife* **4**, e08206 (2015).
33. Zlateski, A. & Seung, H. S. Image segmentation by size-dependent single linkage clustering of a watershed basin graph. *arXiv* Preprint at https://arxiv.org/abs/1505.00249 (2015).
34. Kasthuri, N. et al. Saturated reconstruction of a volume of neocortex. *Cell* **162**, 648–661 (2015).
35. Martin, D. R., Fowlkes, C. C. & Malik, J. Learning to detect natural image boundaries using local brightness, color, and texture cues. *IEEE Trans. Pattern Anal. Mach. Intell.* **26**, 530–549 (2004).
36. Funke, J., Andres, B., Hamprecht, F. A., Cardona, A. & Cook, M. Efficient automatic 3D-reconstruction of branching neurons from EM data. in *Proc. IEEE Conference on Computer Vision and Pattern Recognition* 1004–1011 (IEEE, New York, 2012).
37. Wolf, S., Schott, L., Kothe, U. & Hamprecht, F. Learned watershed: end-to-end learning of seeded segmentation. in *Proc. IEEE International Conference on Computer Vision (ICCV)* 2030–2038 (IEEE, New York, 2017).
38. Bai, M. & Urtasun, R. Deep watershed transform for instance segmentation. in *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* 2858–2866 (IEEE, New York, 2017).
39. Meirovitch, Y. et al. A multi-pass approach to large-scale connectomics. *arXiv* Preprint at https://arxiv.org/abs/1612.02120 (2016).
40. Romera-Paredes, B. & Torr, P. H. S. Recurrent instance segmentation. in *Computer Vision—ECCV 2016* (eds. Leibe, B. et al.) 312–329 (Springer, Cham, 2016).
41. Pinheiro, P. O , Lin, T.-Y., Collobert, R., & Dollár, P. Learning to refine object segments. in *Computer Vision–ECCV 2016* (eds. Leibe, B. et al.) 75–91 (Springer, Cham, 2016).
42. Ren, M. & Zemel, R. S. End-to-end instance segmentation with recurrent attention. in *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* 293–301 (IEEE, New York, 2017).

Acknowledgements

We thank T.L. Dean and B. Agüera y Arcas for useful discussions and support. We also thank M.S. Fee and J. Shlens for comments on the manuscript.

Author contributions

M.J. and V.J. conceived FFNs. M.J. developed pipelines and performed experiments. J.K. and W.D. acquired EM data and ground truth annotations. A.P. aligned the dataset. M.J. and J.K. analyzed results. V.J., M.J., W.D., P.H.L., J.M.-S., and J.K. developed skeleton metrics. M.J. and V.J. developed tissue-classification models. T.B., L.L., M.T., and J.M. developed software for annotation and visualization. V.J., M.J., W.D., and J.K. wrote the manuscript. V.J. supervised the project.

Competing interests

J.K. holds shares of Ariadne service GmbH. M.J., P.H.L., A.P., T.B., L.L., J.M.-S., M.T., and V.J. are employees of Google LLC, which sells cloud computing services.

Additional information

Supplementary information is available for this paper at https://doi.org/10.1038/s41592-018-0049-4.

Reprints and permissions information is available at www.nature.com/reprints.

Correspondence and requests for materials should be addressed to V.J.

Methods

Detection of tissue irregularity. We used cross-correlation to detect irregularities in the input EM data caused by artifacts in the image-acquisition process or by imprecise alignment of the images. For every pair of neighboring sections, we extracted patches of 160×160 pixels, centered at every node of a 2D grid with a 40-pixel step size; computed the normalized cross-correlation of the two patches corresponding to every grid node using fast Fourier transform convolution in FULL mode (i.e., convolution results were computed for every point of overlap, even if partial, by padding with zeros); and identified the peak in the correlation image. The peak offsets from the image centers estimate the local section-to-section lateral motion and form a sparsely sampled vector field over the whole volume, with vectors confined to the x–y plane. On the basis of results from experiments with FFN inference in areas affected by data irregularities, we set the threshold for the detection of an irregularity at 4 pixels in either direction.

Tissue-type classifier. We trained a convolutional network to predict whether a voxel belonged to one of six categories that represented general structural features of the image volume. First, we manually labeled 26.7 million voxels (0.016% of the volume) at $2\times$ reduced lateral resolution as blood vessel (4.4 million voxels), cell body (11.5 million voxels), myelin (1.5 million voxels), neuropil (7.4 million voxels), or 'out-of-tissue' (1.8 million voxels). Manual annotations (5 h of human time) were sparsely created on every 500th slice by two authors (V.J. and M.J.) with a custom web-based tool that enabled them to manually paint voxels with a modifiable brush size (Supplementary Fig. 1).

We then used TensorFlow[43] to train a 3D convolutional network to classify a $65\times65\times65$ patch centered on each manually labeled voxel. The network contained three 'convolution-pooling' modules[44] consisting of convolution ($3\times3\times3$ kernel size, 64 feature maps, VALID mode where convolution results are computed only where the image and filter overlap completely) and max pooling ($2\times2\times2$ kernel size, $2\times2\times2$ stride, VALID mode), followed by one additional convolution ($3\times3\times3$ kernel size, 16 feature maps, VALID), a fully connected layer (512 nodes, expressed as a point-wise convolution), and a six-class softmax output layer[26]. We trained the network by stochastic gradient descent with a minibatch size of 16 and four asynchronous replicas[45]. During training, each of the six classes was sampled equally often. Training was terminated after 1 million updates.

Inference with the trained network was applied to all voxels in the image volume by dilated convolution, which is several orders of magnitude more efficient than a naive sliding-window inference strategy[46]. Finally, the analog [0,1]-valued network predictions were thresholded and used to prevent certain image regions from being segmented, as detailed in the section "Segmentation pipeline."

Flood-filling networks (architecture, training, inference). *Architecture.* The FFN comprised a stack of 3D convolutions in SAME mode (input and output of every layer of equal size, with input implicitly padded with zeros to achieve this) with skip connections, rectified linear (ReLU) nonlinearities[26], $3\times3\times3$ kernel sizes, and 32 feature maps in every layer but the last. The network had 19 convolutional layers with a total of 472,353 trainable weights (Supplementary Fig. 2). The input module contained a ReLU nonlinearity sandwiched between two 3D convolutions. This was followed by eight residual modules, which together performed a ReLU nonlinearity, 3D convolution, ReLU nonlinearity, and 3D convolution. The last layer performed a voxel-wise convolution that combined input from all feature maps ($1\times1\times1$ kernel size with a single output feature map). The input and output of the network were equal in spatial size—$33\times33\times17$ voxels. The input was formed by a two-channel image, with channel 1 containing EM data (to normalize, we subtracted 128 from the original intensities and divided the result by 33) and channel 2 the current state of the POM in logit form. The output of the network was the updated POM, again in logit form.

We chose an architecture with 'full pre-activation residual modules'[47] because it showed better convergence than alternatives (no skip connections or other proposed variants of skip connections[48]). Note that our network contained as many as 74 times[22] fewer weights than other CNN-based segmentation approaches.

The FFN was implemented in TensorFlow[43] and trained with voxelwise cross-entropy loss:

$$\sum_{i=0}^{33\cdot33\cdot17} -\log(p_i)g_i - \log(1-p_i)(1-g_i)$$

where p_i is the predicted voxel value and g_i is the ground truth label after smoothing. Training proceeded for 7 d with asynchronous stochastic gradient descent at a learning rate of 0.001, in a distributed setting with 32 NVIDIA Tesla K40 GPUs and batches of four examples.

Training example sampling. We formed the initial set of training examples by extracting all subvolumes $49\times49\times25$ voxels in size and fully contained within one of the 33 regions densely segmented by human annotators. The size of the subvolume was chosen to allow FOV movement by one step in every direction (8 voxels in XY, and 4 voxels in Z).

We binarized the ground-truth segmentation within every subvolume by setting voxels belonging to the same object as the central voxel of the subvolume to 0.95, and the rest of the voxels to 0.05. These soft labels[26] provided the object mask probability map that the FFN was trained to predict.

For every initial training example, the fraction (f_a) of voxels set to 0.95 was calculated. The training examples were then partitioned into 17 'probability' classes, such that an example was assigned to class i if $t_{i-1} \le f_a < t_i$ and t = (0.0, 0.01, 0.02, 0.03, 0.04, 0.05, 0.06, 0.075, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1). For example, a training example with $f_a = 0.5$ would be assigned to class 12. During training, each of the 17 classes was sampled equally often.

Seed list generation. We applied the 3D Sobel filter to the EM data and set all pixels for which the difference between the raw filter output and Gaussian-smoothed ($\sigma = 49/6$) filter output was positive to 1, and all others to 0. We then computed the Euclidean distance transform of the resulting binary image and selected local maxima (peaks) as seeds.

These seeds were then consumed serially in raster order. At the completion of an object, we discarded all seeds that were inside the object or less than 3 voxels from its surface.

Field-of-view movement. The FOV of the FFN was moved via the following procedure. To start segmentation of a new object, we initialized a list of positions to be visited (Q) with a location obtained from the seed list. In the segmentation loop, a location (x, y, z) was extracted from the head of Q, and the FOV was moved to that position, which was then marked as visited. Visited locations were stored to ensure that every location was visited at most once during segmentation of an object; locations were stored at $8\times$ reduced resolution in the XY directions and $4\times$ reduced resolution in the Z direction. This reduced resolution effectively determined the minimum step size by which the FOV could be moved, and was used to control the efficiency of inference.

After an inference call at the new position, all POM voxels within the FOV were updated, except those that had been previously updated by the FFN and had a prior value < 0.5 and a new value greater than the prior value (this biased the network toward splits in areas where predictions of background/foreground were not consistent between iterations). A cuboid of POM values $(x - \Delta x \le x \le x + \Delta x) \times (y - \Delta y \le y \le y + \Delta y) \times (z - \Delta z \le z \le z + \Delta z)$ was then extracted, and the maximum value was identified on every one of its faces. Whenever this value matched or exceeded the movement threshold of 0.9, the corresponding location was appended to Q unless it had been visited before.

The inference loop was terminated when Q was empty. At that point, if the number of voxels with POM values ≥ 0.6 was $\ge 1{,}000$, a new segment was created consisting of those voxels; otherwise the POM values were reset to 0.05 without a segment being created. Segmentation was terminated when no more seeds were available to start new inference runs. We used $\Delta x = \Delta y = 8$ and $\Delta z = 4$, which were the largest values that did not result in an increased number of errors in our tests while remaining computationally tractable.

Criterion for model evaluation and selection. In addition to the densely labeled ground truth data, we had a $560\times560\times250$ voxel subvolume of the J0126 dataset exhaustively skeletonized by human annotators using Knossos and used the resulting set of 221 skeleton fragments to optimize the FFN performance.

During training, a snapshot of the network weights ('checkpoint') was saved every hour. After training was completed, we ran FFN inference over the densely skeletonized subvolume with every available checkpoint, and we evaluated the resulting segmentation with skeleton metrics. We selected the checkpoint that had the greatest expected run length among the set of checkpoints with the fewest mergers (in our case this corresponded to the set of checkpoints with zero skeleton-to-skeleton mergers).

Distributed inference. To carry out FFN inference efficiently over the whole 663-GB dataset, we split the dataset into overlapping $500\times500\times500$ voxel subvolumes with 64-pixel overlap in every direction. We ran FFN inference as described above for every subvolume independently, distributing the computational load over a cluster of machines that contained GPUs.

The global segmentation was built with these partial segmentations. After discarding a 32-voxel-wide envelope (half of the overlap area), we calculated 'core' segmentations for every subvolume by computing the connected components of the remaining segmentation. For each overlap area, we established the correspondence between the segments of the two contributing subvolumes by first calculating for each segment the center-plane (parallel to the cube surface) cross-sectional overlap. We used this conservative merging procedure, in which only between-cube segment pairs with mutually maximal overlap were merged, to avoid spurious mergers (−84%) during the creation of the global segmentation, at a cost of increased splits (+28%).

Multi-resolution over-segmentation consensus. The over-segmentation consensus procedure relies on objects intersecting in voxel space. In the case of consensus between segmentations at different resolutions, we upsampled the lower-resolution segmentation with nearest-neighbor interpolation. We then applied seeded watershed segmentation to the Euclidean distance transform of the higher-resolution segmentation as the height map, using the upsampled segmentation as seeds. This did not change the topology of the upsampled segmentation, but it prevented voxel-level differences between the two segmentations from causing the generation of new segments during the over-segmentation-consensus procedure.

To reduce the number of splits in the multi-resolution over-segmentation consensus procedure, we removed all objects containing fewer than 100,000 voxels from the lower-resolution segmentation before upsampling it and all objects with fewer than 1,000 voxels from the consensus.

Cell-body segmentation. We created a separate segmentation containing only cell bodies, and used it as the initial state of the segmentation for all subsequent FFN inference. To do so, we performed three FFN inference runs at resolutions of $9\times9\times20$ nm (original), $18\times18\times20$ nm and $36\times36\times40$ nm, with areas of the volume not classified as cell body by the tissue-type classifier masked out. We then created a multi-resolution over-segmentation consensus, removed all objects with fewer than 10 million voxels, resampled the segmentation at an isotropic resolution of 160 nm, computed the Euclidean distance transform within the cell-body segments, and used seeded watershed with manually placed seeds (cell-body-center annotations) to separate adjacent cell bodies. The corrected segmentation was upsampled back to the original resolution of the dataset, and the separated cell bodies were used as seeds for a watershed transform. Background voxels of the full-resolution uncorrected cell-body segmentation were masked out so that no new voxels were labeled by watershed.

Flood-filling network agglomeration. *Candidate object-pair generation.* We considered only those supervoxel pairs for which a cuboid with a radius of $5\times5\times5$ voxels existed that overlapped with both supervoxels. For each of those pairs we computed agglomeration scores (details below) and a decision-point location, defined as the midpoint of the shortest line connecting the supervoxels.

We selected pairs where each of the objects (A, B) contained at least 10,000 voxels, and we extracted a $101\times101\times51$ voxel subvolume of EM data and segmentation, centered at the decision point. We then removed A and B from the segmentation and generated two new segments, A* and B*, by running FFN inference within the subvolume. First, to generate A*, we placed the seed at the peak of the Euclidean distance transform of A. If the segment generated failed to cover at least 60% of A, we started a new attempt, again using the maximum of the distance transform, but this time excluding a cuboid area of radius (8, 8, 4) voxels around the original seed point. We repeated this procedure up to 16 times before using an analogous procedure to generate B*.

Agglomeration scoring. After binarizing the POMs with a threshold of 0.5, we computed recovered voxel fractions (f_{AA}, f_{AB}, f_{BA}, and f_{BB}, where f_{AB} is the fraction of B found in A*, and so on), the Jaccard index J_{AB} between the two segments (defined as the size of the intersection between two voxel sets divided by the size of their union), and the number of voxels contained in A* or B* that had been 'deleted' (i.e., during inference their value in the POM fell from >0.8 to <0.5) during one of the runs (d_A, d_B).

Agglomeration steps. After scoring all decision points, we re-examined all those for which the tissue-irregularity detection criterion was triggered for the associated subvolume. For those points we computed the cross-correlation between neighboring ($z+1$) and next-neighboring ($z+2$) sections within the subvolume used for scoring and computed local shift vectors ($\mathbf{m}_z$ and $\mathbf{m}_{z+1}$) as described in the section "Detection of tissue irregularity." If $\mathbf{m}_z$ satisfied the tissue-irregularity criteria but $\mathbf{m}_{z+1}$ did not, we assumed that the artifact was limited to a single slice and replaced section $z+1$ with image data from section $z+2$, realigned the subvolume using only translations that used the $\mathbf{m}_z$ slice-shift vectors (updated to reflect the substituted section), and performed FFN inference without tissue-irregularity detection. Finally, we reran the scoring procedure with a larger subvolume of $201\times201\times101$ voxels for all decision points for which any of the recovered voxel fractions were less than 0.4.

We then calculated connected components, considering two segments as being connected if (1) none of the recovered voxel fractions were less than 0.6, (2) the fraction of deleted points was less than 2% in one of the scoring runs, and (3) the objects from the two scoring runs were very similar in voxel space in that the Jaccard index was at least 0.8. All parameters mentioned above were optimized with the 'validation' set of 12 ground-truth skeletons. We then checked whether any of the components contained more than one segment associated with a cell body indicating a connection between different cells. When such a case occurred, we eliminated the connection by removing the weakest supervoxel-to-supervoxel connection in the shortest path between two such cell-body segments. This procedure was repeated until no connections between cells remained.

Manual annotation. All manual annotations were generated with KNOSSOS software (http://www.knossostool.org). For the skeleton ground-truth data (tuning and test set), we used the advanced tracing mode to create spatial graphs of the neurons of interest. The dense annotations used as neural network training data were generated in the segmentation mode of KNOSSOS at randomly chosen locations over the dataset.

Songbird dataset acquisition and animal research ethical compliance. The J0126 dataset was acquired from area X of an adult male zebra finch (for details on extracellular-space-preserving perfusion and heavy-metal staining, see ref. [49]) using serial block-face EM on a Zeiss UltraPlus scanning electron microscope with 1.6-kV beam voltage, 1-nA beam current, and a scan rate of 3.3 MHz. The image pixel size was 9 nm and the cutting thickness of the microtome was 20 nm. All animal experiments were approved by the Regierungspräsidium Karlsruhe and were in accordance with the laws of the German federal government.

Statistics. The confidence intervals in Fig. 3 (confidence level, 95%) were estimated via the bootstrap method with 10,000 resamples.

Reporting Summary. Further information on experimental design is available in the Nature Research Reporting Summary linked to this article.

Code availability. The code has been released under the Apache 2.0 open-source license and is available as Supplementary Software and at https://github.com/google/ffn.

Data availability. The data that support the findings of this study are available from the corresponding author upon reasonable request.

References

43. Abadi, M. et al. TensorFlow: large-scale machine learning on heterogeneous distributed systems. *arXiv* Preprint at https://arxiv.org/abs/1603.04467 (2016).
44. LeCun, Y. et al. Backpropagation applied to handwritten zip code recognition. *Neural Comput.* **1**, 541–551 (1989).
45. LeCun, Y. A , Bottou, L , Orr, G. B., & Müller, K.-R . Efficient BackProp. in *Neural Networks: Tricks of the Trade* (eds. Montavon, G., Orr, G. & Müller, K.-R.) 9–48 (Springer, Berlin, 2012).
46. Tschopp, F. Efficient convolutional neural networks for pixelwise classification on heterogeneous hardware systems. *arXiv* Preprint at https://arxiv.org/abs/1509.03371 (2015).
47. He, K., Zhang, X., Ren, S. & Sun, J. Identity mappings in deep residual networks. *arXiv* Preprint at https://arxiv.org/abs/1603.05027 (2016).
48. He, K., Zhang, X., Ren, S. & Sun, J. Deep residual learning for image recognition. *arXiv* Preprint at https://arxiv.org/abs/1512.03385 (2015).
49. Kornfeld, J. et al. EM connectomics reveals axonal target variation in a sequence-generating network. *eLife* **6**, e24364 (2017).

nature research

Corresponding author(s): Viren Jain

Reporting Summary

Statistical parameters

When statistical analyses are reported, confirm that the following items are present in the relevant location (e.g. figure legend, table legend, main text, or Methods section).

n/a	Confirmed	
☐	☒	The exact sample size (*n*) for each experimental group/condition, given as a discrete number and unit of measurement
☐	☒	An indication of whether measurements were taken from distinct samples or whether the same sample was measured repeatedly
☐	☒	The statistical test(s) used AND whether they are one- or two-sided *Only common tests should be described solely by name; describe more complex techniques in the Methods section.*
☒	☐	A description of all covariates tested
☐	☒	A description of any assumptions or corrections, such as tests of normality and adjustment for multiple comparisons
☐	☒	A full description of the statistics including central tendency (e.g. means) or other basic estimates (e.g. regression coefficient) AND variation (e.g. standard deviation) or associated estimates of uncertainty (e.g. confidence intervals)
☒	☐	For null hypothesis testing, the test statistic (e.g. *F*, *t*, *r*) with confidence intervals, effect sizes, degrees of freedom and *P* value noted *Give P values as exact values whenever suitable.*
☒	☐	For Bayesian analysis, information on the choice of priors and Markov chain Monte Carlo settings
☒	☐	For hierarchical and complex designs, identification of the appropriate level for tests and full reporting of outcomes
☒	☐	Estimates of effect sizes (e.g. Cohen's *d*, Pearson's *r*), indicating how they were calculated
☐	☒	Clearly defined error bars *State explicitly what error bars represent (e.g. SD, SE, CI)*

Our web collection on statistics for biologists may be useful.

Software and code

Policy information about availability of computer code

Data collection	No software was used.
Data analysis	The code has been released under the Apache 2.0 open-source license and is available at https://github.com/google/ffn. KNOSSOS 5.0 was used for annotation of skeletons and is available at https://knossostool.org/ under the GNU GPL v2.0 open-source license.

For manuscripts utilizing custom algorithms or software that are central to the research but not yet described in published literature, software must be made available to editors/reviewers upon request. We strongly encourage code deposition in a community repository (e.g. GitHub). See the Nature Research guidelines for submitting code & software for further information.

Data

Policy information about availability of data

All manuscripts must include a data availability statement. This statement should provide the following information, where applicable:

- Accession codes, unique identifiers, or web links for publicly available datasets
- A list of figures that have associated raw data
- A description of any restrictions on data availability

The data that support the findings of this study are available from the corresponding author upon reasonable request.

Field-specific reporting

Please select the best fit for your research. If you are not sure, read the appropriate sections before making your selection.

☒ Life sciences ☐ Behavioural & social sciences ☐ Ecological, evolutionary & environmental sciences

For a reference copy of the document with all sections, see nature.com/authors/policies/ReportingSummary-flat.pdf

Life sciences study design

All studies must disclose on these points even when the disclosure is negative.

Sample size	n=1 zebra finch brain was imaged, due to the investment required to obtain such a dataset. n=50 ground truth neurons were manually annotated (skeletonized) as ground truth for evaluating the automated methods; this number was chosen a compromise between investment required to obtain such annotations and a representative sampling of cells in the volume (n=50 constitutes roughly 10% of those cells with somas in the volume).
Data exclusions	No data were excluded.
Replication	Computational reconstructions of the volume were successfully reproduced using the described pipeline and methods. Several experiments were performed on publicly available datasets (SNEMI3d and FIB-25) and experimental parameters used for those datasets are described in detail in Supplementary Table 12.
Randomization	The n=50 neurons using in the test set evaluation were randomly selected among the somas in the image volume.
Blinding	Investigators chose the ground truth neurons randomly but were not explicitly blinded during the selection process. Since there was no investigator input into the selection process (i.e., the test set was randomly chosen), there is no further notion of blinding that would be applicable in this situation.

Reporting for specific materials, systems and methods

Materials & experimental systems

n/a	Involved in the study	
☒	☐	Unique biological materials
☒	☐	Antibodies
☒	☐	Eukaryotic cell lines
☒	☐	Palaeontology
☐	☒	Animals and other organisms
☒	☐	Human research participants

Methods

n/a	Involved in the study	
☒	☐	ChIP-seq
☒	☐	Flow cytometry
☒	☐	MRI-based neuroimaging

Animals and other organisms

Policy information about studies involving animals; ARRIVE guidelines recommended for reporting animal research

Laboratory animals	Adult male zebra finch (wild type, >120 dph, days post hatching).
Wild animals	*Provide details on animals observed in or captured in the field; report species, sex and age where possible. Describe how animals*

Wild animals	*were caught and transported and what happened to captive animals after the study (if killed, explain why and describe method; if released, say where and when) OR state that the study did not involve wild animals.*
Field-collected samples	*For laboratory work with field-collected samples, describe all relevant parameters such as housing, maintenance, temperature, photoperiod and end-of-experiment protocol OR state that the study did not involve samples collected from the field.*