Cross-Classification Clustering: An Efficient Multi-Object Tracking Technique for 3-D Instance Segmentation in Connectomics

Yaron Meirovitch[1,*,‡], Lu Mi[1,*], Hayk Saribekyan[1],
Alexander Matveev[1], David Rolnick[1], Casimir Wierzynski[2], and Nir Shavit[1,3]

[1]Computer Science and Artificial Intelligence Laboratory, MIT
[2]Intel AI Research [3]Neural Magic

[*]These authors equally contributed to this work [‡]yaron.mr@gmail.com

Abstract

Pixel-accurate tracking of objects is a key element in many computer vision applications, often solved by iterated individual object tracking or instance segmentation followed by object matching. Here we introduce cross-classification clustering (3C), a new technique that simultaneously tracks all objects in an image stack. The key idea in cross-classification is to efficiently turn a clustering problem into a classification problem by running a logarithmic number of independent classifications, letting the cross-labeling of these classifications uniquely classify each pixel to the object labels. We apply the 3C mechanism to achieve state-of-the-art accuracy in connectomics – nanoscale mapping of the brain from electron microscopy volumes. Our reconstruction system introduces an order of magnitude scalability improvement over the best current methods for neuronal reconstruction, and can be seamlessly integrated within existing single-object tracking methods like Google's flood-filling networks to improve their performance. This scalability is crucial for the real-world deployment of connectomics pipelines, as the best performing existing techniques require computing infrastructures that are beyond the reach of most labs. We believe 3C has valuable scalability implications in other domains that require pixel-accurate tracking of multiple objects in image stacks or video.

1. Introduction

Object tracking is an important component in many computer vision applications and has been extensively studied [1, 8, 9, 11, 15, 39]. It occurs both in video segmentation and 3-D object reconstruction based on 2-D images.

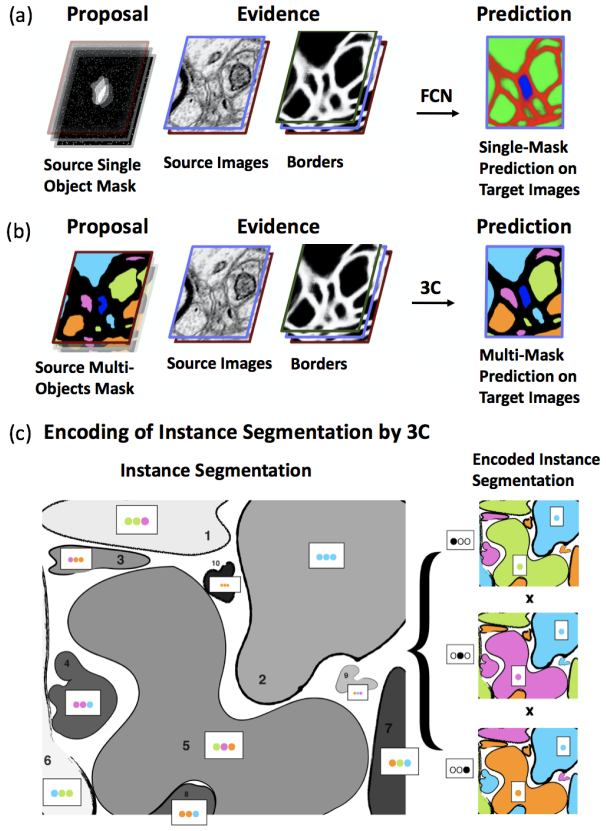


Figure 1. Transferring instance segmentation from source images to target images, within an image stack, with fully convolutional networks (FCNs). (a) Single-object tracking, as in flood-filling networks [14] versus (b) multiple-object tracking with our cross-classification clustering (3C) algorithm. (c) The encoding of an instance segmentation by 3C: One 10-objects segmented image is encoded into a product of three 4-objects segmented images.

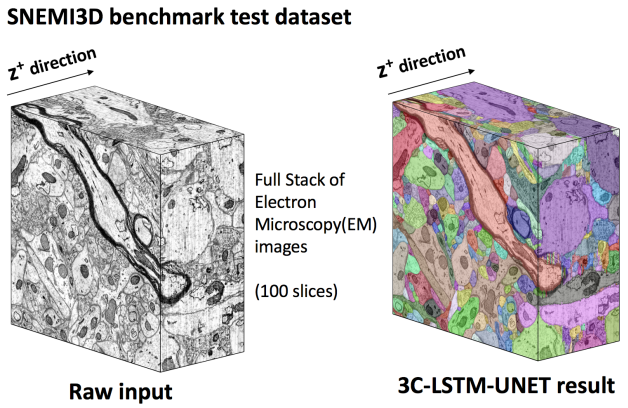


Figure 2. The raw input electron microscopy (EM) full image stack and our 3C-LSTM-UNET results in the SNEMI3D benchmark.

Though tracking objects through multiple images remains a highly researched problem in computer vision, less attention has been given to efficient algorithms performing simultaneous (one-shot) tracking of multiple interrelated objects [9], so as to reduce the redundancy of repeated single-object tracking.

The field of connectomics, mapping the brain at the level of individual neurons and the synapses between them, offers one of the most challenging settings for testing multiple object-tracking algorithms. Such a map can be made only from extremely high-resolution images taken by electron microscopes, where the sheer volume of data that needs to be processed (petabyte-size image stacks), the desired accuracy and speed (terabytes per hour [21]), and the complexity of the neurons' morphology, present a daunting computational task. By analogy to traditional object tracking, imagine that instead of tracking a single sheep through multiple video frames, one must track an entire flock of sheep that intermingle as they move, change shape, disappear and reappear, and obscure each other [20]. As a consequence of this complexity, several highly successful tracking approaches from other domains, such as the "detect and track" approach [9], are less immediately applicable to connectomics.

Certain salient aspects are unique to the connectomics domain: **a)** All objects are of the same type (biological cells) and sub-categorizing them has little relevance to the reconstruction problem. Object textures are invariant and background is generally also uninformative. **b)** Most of the image is foreground, with tens to hundreds of objects in a single megapixel image. **c)** Objects have variable, finely branched shapes. **d)** Alignment of images is imperfect, and distance between images (z-resolution) is greater than between pixels of the same image (xy-resolution), sometimes breaking the objects' continuity. **e)** Some 3-D objects reside parallel to the image stack, and are thus short-lived in the z direction, going back and forth in that limited space with extremely large extensions in some image planes.

In this work, we propose 3C, a new technique which achieves volumetric instance segmentation by transferring segmentation knowledge from one image to another, simultaneously classifying the pixels of the target image(s) with the labels of the matching objects on the source image(s). This algorithm is optimized for the setting of connectomics, in which objects frequently branch and come together, but is suitable for a wide range of applications. The main advantage of our solution is its ability, unlike prior single-object tracking methods for connectomics [14, 24], to simultaneously and jointly segment osculating intermingled objects while saving the redundant feedforward inference computations of co-related objects. Moreover, instead of extending single masks, our detectors perform clustering by taking into account information on all visible objects. The efficiency and accuracy of 3C are demonstrated on four connectomics datasets: the public SNEMI3D benchmark dataset, shown in Figure 2, the widely studied mouse somatosensory cortex dataset [16] (*S1*), a Lichtman Lab Rat V1 dataset (*ECS*), and a newly aligned mouse peripheral nervous system dataset (*PNS*), where possible, comparing to other competitive results in the field of connectomics.

1.1. Related work

A variety of techniques from the past decade addressed the task of neuron segmentation from electron microscopy volumes. An increasing effort was dedicated to the narrower instance segmentation problem of densely segmenting all pixels of a volume according to foreground object instances (nerve and support cells), known as *saturated reconstruction*. Note that unlike a typical daily-life image, a $1k \times 1k$ electron microscope image section with nanometer resolution of the brain contains hundreds to thousands of object instances (cross sections), with very little background ($< 10\%$). Below, we briefly survey the saturated reconstruction pipelines that seem to us most influential and highly related to the approach undertaken here.

Algorithms for saturated reconstruction of connectomics data have proved most accurate when they combined many different machine learning techniques [5, 19]. The basic concept shared in many of the successful techniques is the hierarchical connectomics processing of Andres et al. [2], taking advantage of the well-known hierarchical image segmentation framework [3, 10, 25, 32]. This is still the most common approach in connectomics segmentation pipelines: firstly detecting object borders in 2-D/3-D and gradually agglomerating information to form the final objects [5, 7, 13, 18, 19, 36]. The elevation maps obtained from the border detectors are treated as estimators of the true border probabilities [7], which are used to define an over-segmentation of the image, foreground connected components on top of a background canvas. The assumption is that each of the connected components straddles at most a single true ob-

ject. Therefore it may need to be agglomerated with other connected components (heuristically [12, 34] or based on learned weights of hand-crafted features [2, 26, 27]), but should not be broken down into smaller segments. Numerous 3-D reconstruction systems follow this bottom-up design [5, 6, 18, 19, 22, 26, 27, 29]. A heavily engineered implementation of hierarchical segmentation [19] still occupies the leading entry in the still-running classical SNEMI3D connectomics contest of 2013 [4], evaluated in terms of the uniform instance segmentation correctness metrics of the normalized Rand-Error [37] and the Variation of Information [23] metrics.

A promising new approach was recently taken with the introduction of flood-filling networks (FFN; [14]) by Januszewski et al. and concurrently and independently MaskExtend [24] by Meirovitch et al. As seen in Figure 1(a), these algorithms take a mask defining the object prediction on a source image(s), and then use a fully convolutional network (FCN) to classify which pixels in the target image(s) belong to the singly masked object of the source image(s). This process is repeated throughout the image stack in different directions, segmenting and tracking each time a single object, while gradually filling the 3-D shape of complex objects. This provides accuracy improvements on several benchmarks and longer object run-length [14] compared to previous hierarchical segmentation algorithms (e.g., [5]). However, these single-object trackers are not readily deployable for large-scale applications, especially when objects are tightly packed and intermingled with each other, because then individual tracking becomes highly redundant, forcing the algorithm to revisit pixels of related image contexts many times[1]. Furthermore, the existing single-object trackers do not take advantage of the multi-object scene to better understand the spatial correlation between different 3-D objects. The approach taken here generalizes the single-object approach to connectomics to achieve simpler and more effective volumetric instance segmentation of the neuronal space.

1.2. Contribution

Cross-classification clustering (3C) provides a scalable framework for 3-D instance segmentation problems and multiple objects tracking applications, with the following main contributions:

- We propose a simple FCN approach, tackling the less studied problem of mapping an instance segmentation between two related images. Compared to single object tracking, our algorithm predicts the shape of each object by taking into account all other objects visible in the input.
- We propose a novel technique that turns a clustering problem into a classification problem by running a logarithmic number of independent classifications on the pixels of an image with N objects (for possibly large N, bounded only by the number of pixels).
- We show empirically that the simultaneous tracking ability of 3C is more efficient than independently tracking all objects.
- We conduct extensive experimentation with four connectomics datasets under different evaluation criteria and a performance analysis, to show the efficacy and efficiency of the 3C technique on the problem of neuronal reconstruction in connectomics.

2. Methodology

We present *cross-classification clustering* (henceforth *3C*), a technique that extends the single object classification approach into a mechanism that simultaneously and efficiently classifies all objects in a given image based on a proposed instance segmentation of a context-related image. One can think of the context-related image and its segmentation as a collection of labeled masks to be simultaneously remapped together to the new target image, as in Figure 1(b). The immediate difficulty of such simultaneous settings is that this generalization is a clustering problem: unlike FFNs and MaskExtend (shown in Figure 1(a)), that produce a binary output ("YES" for extending the object and otherwise "NO"), in any volume, we really do not know how many classification labels we might need to capture all the objects, or more importantly how to represent those instances in ways usable for supervised learning. Overcoming this difficulty is a key contribution of our new 3C algorithmic approach.

Cross-Classification Clustering: We begin by explaining the main idea behind our new *cross-classification clustering* (3C) technique and differentiate it from object-centered flood-filling/extend methods. Our 3C technique extends the single-object classification approach into a mechanism that simultaneously classifies pixels for an *a priori* unknown set of object labels. More formally, our goal is to approximate a classification function f. The function f takes as input a voxel v and a segmentation s (a 2-D or 3-D matrix over the integers, with each integer a label representing an object) and outputs a decision label. If s has no split error, then f outputs l if and only if v belongs to the object labeled l in s. More generally, s is allowed to be an over-segmentation (i.e., several labels representing the same object) and then the output of f is determined probabilistically. As a result, a repeated application of f can correct a split error. Merge errors can only simplify the decision task and are also acceptable (but are not corrected).

We proceed by encoding the input label set before applying the classification operations. Assume that the input seg-

[1] Such approaches thus take time linear in the number of objects and in the number of pixels, with a large constant that depends on object density.

mentation s is an integer matrix over some set N of labels, each of the labels representing an object. For simplicity, and slightly abusing notation, let us number them $\{1, \cdots, N\}$. We define a new space of labels, the k-length strings A^k over a predetermined alphabet A where $n = |A|^k \geq N$ is an upper bound on the number of objects we expect in a classification. In the example in Figure 1(c), A is represented by 4 colors, and the encoder, χ, defines $k = 3$ maps, so we have a total of $4^3 = 64$ possible strings of length 3 to which the $L = 10$ objects can be mapped. Each of the objects is colored by χ to a random string that corresponds to the combination of colors. Thus, for example, object 5 is mapped to the string (Green, Purple, Orange) and object 1 is (Green, Green, Purple). We can define the classification function f on string labels as the product of k traditional classifications, each with an input segmentation of labels in A, and an output of labels in A. Slightly abusing notation, let the direct product of images $\chi(s) = \chi_1(s) \times \cdots \times \chi_k(s) \in M^k$ be the relabeling of the segmentation s where each image (or tensor) $\chi_i(s)$ is the projection of $\chi(s)$ in the i-th location (a single coloring of the segmentation). Then we can re-define f on $\chi(s)$ as

$$f(v, \chi(s)) = f'(v, \chi_1(s)) \times \cdots \times f'(v, \chi_k(s)), \qquad (1)$$

where each $f'(\chi_k(s))$ is a traditional classification function and $\times$ is the concatenation operation on labels in A. The key idea is that f' is a classification of v based on an instance segmentation with fewer labels (a projection). In the example in Figure 1(c), even though in the map representing the most significant digit of the original objects 5 and 1, they are both Green, when we perform the classification and take the cross labeling of all three maps, the two objects are classified into distinct labels.

3-D Reconstruction System: Our 3-D reconstruction system based on the 3C algorithm consists of the following steps (Step 2 through 4 are shown in Figure 3):

1) Seeding and labeling the volume with an initial imperfect 2-D instance segmentation that overlaps all objects except for their boundaries (2-D over-segmentation).

2) Encoding the labeled seeds into a new space using the 3C rule χ.

3) Applying a fully convolutional network $\log(N)$ times to transfer the projected labeled seeds from the source image to the target images, and then take their cross labeling.

4) Decoding the set of network outputs to the original label space using the inverse 3C rule χ^{-1}.

5) Agglomerating labels into 3-D consistent structures based on the overlap of the original seeding and the segments predicted from other sections.

In order to generate the initial 2-D instance segmentation of the images, we define and label 2-D masks that oversegment all objects. For this we followed common practice in connectomics, computing object borders with FCNs and searching for local minima in 2-D on the border elevation maps. Subsequently, and by the example of Figure 3, we encode the seeds of the section Z with the 4 colors of the alphabet, k=5 times, according to the 3C procedure.

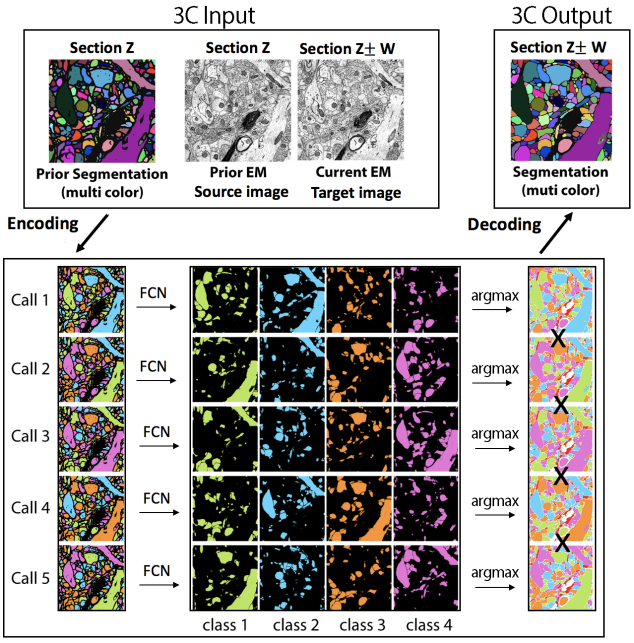


Figure 3. The instance segmentation transfer mechanism of 3C: Encoding the seed labels into a new space using the 3C rule (here represented by 4 colors). Applying a fully convolutional network k=$\log(N)$ times to transfer the projected instance segmentation images from the source image to the target image (here applied 5 times). Decoding the set of network outputs to the original space using the inverse 3C rule.

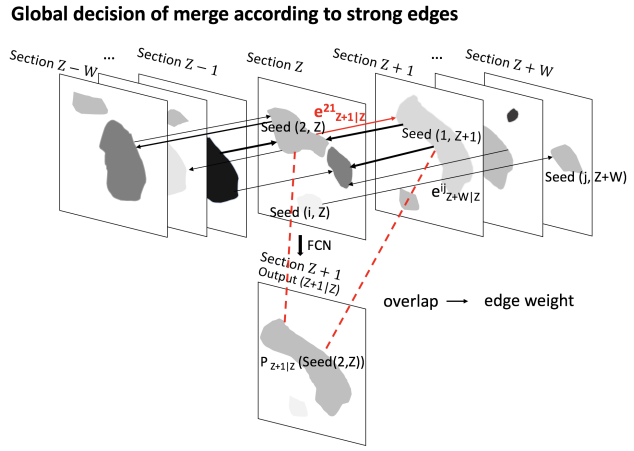


Figure 4. A schematic view of a global merge decision. The edge weight between seed i and seed j at sections Z and Z+W, receptively, $e^{ij}_{Z+W|Z}$, is calculated by the ratio of the overlapping areas of seed j and the 3C prediction of seed i from its image Z to image Z+W. Seeds that over-segment a common object tend to get merged due to a path of strong edges.

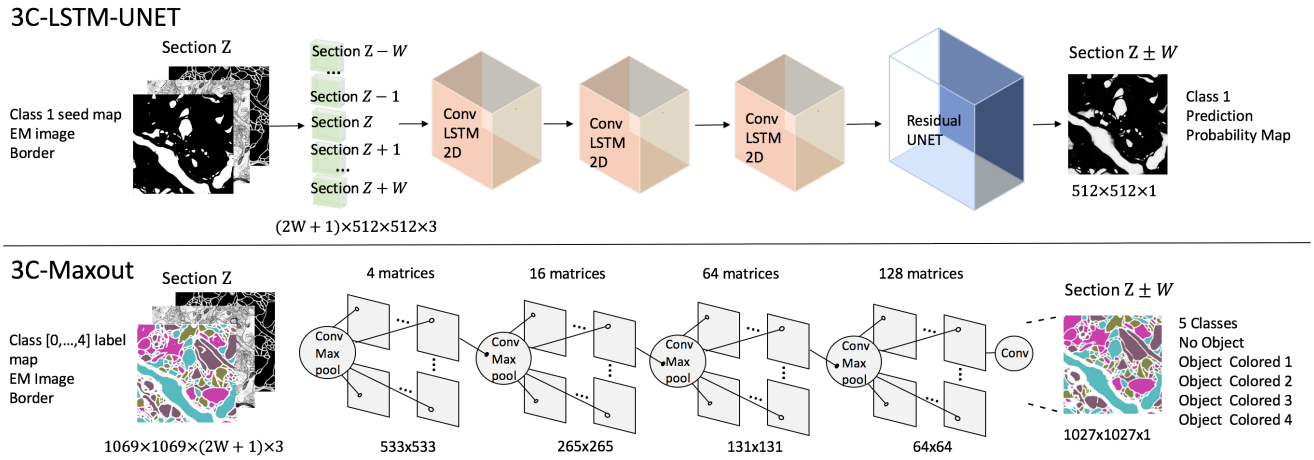


Figure 5. A schematic view of the 3C networks. The input layers have 3 channels of the raw image, seed mask and border probability, for $2W+1$ consecutive sections (images). The output is a feature map of seed predictions in section $Z \pm W$ (binary or labeled). Top: 3C-LSTM-UNET. Network architecture was implemented for the SNEMI3D dataset to optimize for accuracy. The inputs are processed with three consecutive Conv-LSTM modules, followed by a symmetric Residual U-Net structure. Bottom: A 3C-Maxout. Network architecture was implemented for the Harvard Rodent Cortex and PNS datasets to optimize for speed.

A fully convolutional neural network is then trained to learn the correct mapping relationship between interrelated images of sections Z and $Z \pm W$, which determines which pixels in target image $Z \pm W$ belong to which seeds in source image Z. For training, we use saturated 3-D ground truth of the 3-D consistent objects. This approach allows us to formalize the reconstruction problem as a tracking problem between independent images, and to deal with the tendency of objects to disappear/appear in different portions of an enormous 3-D dataset.

We first note that 3C is a direct generalization of those single-object trackers which allows tracking several objects simultaneously. However, while single-object trackers are easy to implement in a Depth-First-Search mode, multi-object tracking is more suitably worked out in a Breadth-First Search mode, gradually expanding the horizon of a set of objects. We also note that the 3C mechanism does not attempt to correct any error in the initial imperfect instance segmentation of the source images, be it an incorrect merge or two separated 2-D objects, or an over-segmentation of a border-free region (2-D split error). The latter are addressed by the agglomeration rule (merge errors in s are assumed to rarely occur).

One component that is still not described in detail is how the 3C segmentation transfers are utilized to agglomerate the 2-D masks (as in Figure 4). For agglomeration, 3C-FCN is applied from all source images to their target images, which are at most W image sections apart from each other across the image stack (the z dimension). We collect overlaps between all co-existing segmentations, namely, those occurring by the original 2-D seeding, and those by the 3C segmentation transfer from source to target images. This leaves $2W + 1$ instance segmentation cases for each image (including the initial seeding), which directly links seeds of different sections. Formally, the overlaps of different labels define a graph whose nodes are the 2-D seed mask labels and the directed weighted edges are their overlap ratio from the source to the target. Instead of optimizing this structure (as in *Fusion* of [17]), we found that agglomerating all masks of sufficient overlap delivers adequate accuracy even for a small W. We do however make forced linking on lower probability edges to avoid "orphan" objects that are too small, which is biologically impossible. We provide further detail in Supplementary Materials.

3. Experiments

The SNEMI3D challenge is a widely used benchmark for connectomic segmentation algorithms dealing with anisotropic EM image stacks [4]. Although the competition ended in 2014, several leading labs recently submitted new results on this dataset, improving the state-of-the-art. Recently Plaza et al. suggested that benchmarking connectomics accuracy on small datasets as SNEMI3D is misleading as large-scale "catastrophic errors" are hard to assess [30]. Moreover, the clustering metrics such as Variation of Information [23] and Rand Error [37] are inappropriate since they are not centered around the connectomics goal of unraveling neuron shape and inter-neuron connectivity. We therefore conduct experiments on three additional datasets and show the Rand-Error results only on the canonical SNEMI3D dataset (hundreds of submissions by dozens of labs). To further assess the quality of 3C at large scale, we demonstrate results on the widely studied large-scale dataset by Kasthuri et al. [16] (*S1 Dataset*). To assess our

results in terms of the end-goal of connectomics, neuronal connectivity, we further assess the synaptic connectivity of the 3C objects using the NRI metric [31] (*ECS Dataset*). In the final experiment we focus on the tracking ability of 3C on a newly acquired large scale dataset (*PNS Dataset*).

3.1. SNEMI3D Dataset

In order to implement 3C on the SNEMI3D dataset, we first created an initial set of 2-D labeled seed in the entire volume. These were generated based on the regional 2-D minima of the border probability map. The border probability map was generated by Residual U-Net. This network is known for its excellent average pixel accuracy for border detection [33, 19]. Second, the 3C algorithm was used to transfer 2-D labeled seeds through the volume, as shown in Figure 5. Finally, the original 2-D labeled seeds and transferred labeled seeds were agglomerated if their overlap ratio exceeded 0.1. All orphans were greedily agglomerated to their best match. In order to achieve better accuracy, we tested 3C with various network architectures, and evaluated their accuracy. To date, convolutional LSTMs (ConvLSTM) have been showing good performance for sequence data [38]. In order to adapt these with the high pixel-accuracy required for connectomics, here we combined both ConvLSTM and U-Net as the basic network structure based on the 3C algorithm. The network is trained to learn an instance segmentation of one image based on the proposed instance segmentation of a nearby image with similar context. According to both qualitative and quantitative analysis, we found that the 3C-LSTM-UNET architecture outperforms others (see supplementary materials). It is built from three ConvLSTM modules and a symmetric Residual U-Net. A schematic view of our architecture is given in Figure 5; the results compared with other tested network architectures are presented in Table 1. Here we empirically found that $W{=}2$ delivers adequate accuracy. Our implementation for the SNEMI3D test used Keras with Tensorflow backend. We ran all experiments on a server with a single Tesla V100-PCIE GPU. We trained our model by minimizing binary-entropy through back-propagation, using Adam with a learning rate of 10^{-4}.

In order to illustrate the near state-of-the-art accuracy of 3C, we submitted our result to the public SNEMI3D challenge website and reproduced the results using two common baseline models, the 3-D watershed transform (a region-growing technique) and Neuroproof agglomeration [27]. Our watershed code was adopted from Matveev et al. [22]. Neuroproof, as other traditional agglomerating-techniques, trains a random forest on merge decisions of neighboring objects, a critical module in several past leading connectomics pipelines [26, 27, 28]. These baseline methods were fed with the same high-quality border maps fed to our 3C reconstruction system. The comparisons of 2-D results with ground truth (section $Z{=}30$) are shown in Figure 6. Our 3C result has fewer merge- and split-errors, and outperforms the two baselines by a large margin. Furthermore, we favorably compared our algorithm to other state of art works recently published in Nature Methods [5, 14]. In the SNEMI3D challenge leader board the Rand-Error of our 3C result was 0.041, compared with the 0.06 achieved by a human annotator. Our accuracy (ranked 3rd) outperforms most of the classical pipelines frequently updating their latest results to this table, many of which by a large margin, and is slightly behind the slower neuron-by-neuron Google's FFN segmentation for this volume. The results are summarized in Table 2.

Network Structure	Number of Parameters	Training/Validation Accuracy
3C-LSTM-UNET	**4M**	**0.963/0.961**
3C-UNET-LSTM	4M	0.861/0.864
3C-UNET	31M	0.881/0.880
3C-Residual UNET	17M	0.915/0.863

Table 1. Comparison of 3C-LSTM-UNET, 3C-UNET-LSTM,3C-UNET, 3C-Residual UNET on SNEMI3D data for number of parameters, training accuracy and validation accuracy.

Model	Rand	VI	VI_{split}	VI_{merge}	Complexity
Watershed	0.113	0.67	0.55	0.12	-
Neuroproof	0.104	0.55	0.42	0.13	-
Multicuts	0.068	0.41	0.34	0.07	-
3C	**0.041**	**0.31**	**0.19**	**0.12**	**O**($V \log N$)
FFN	0.029	-	-	-	O(VN)
Human val.	0.060	-	-	-	-

Table 2. Comparison of Watershed, Neuroproof [27], Multicut [5], human values, 3C and FFN [14] on the SNEMI3D dataset for Rand-Error, Variation of Information *VI*, VI split, VI merge. Time Complexity: N is the number of objects and V is number of pixels. For empirical comparison see the performance section. We do not have access to FFN and the human values outputs and hence their VI metric is missing.

3.2. Harvard Rodent Cortex Datasets (*ECS*, *S1*)

We describe two additional tests: (1) 3C on datasets with known synaptic connectivity (subsets of *ECS*, *S1*), and (2) a light-weight agglomeration-free reconstruction applied to large-scale dataset (*S1*).

Connectivity-based Test: Following [30], which recently advocated connectivity-based evaluation of connectomics, the accuracy of the pipeline was evaluated using the NRI metric [31]. In a nutshell, the NRI ranges between 0 and 1, measuring how well a given neuronal segmentation preserves the object connectivity between neural synapses (1 being optimal). Note, that the NRI used in the experiment

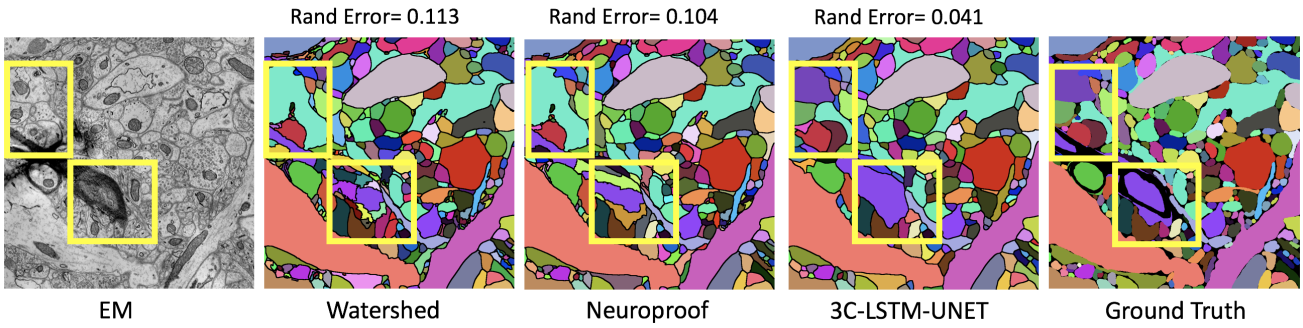


Figure 6. SNEMI3D: The 3C-LSTM-UNET Results compared with baseline techniques: Watershed, Neuroproof, and ground truth.

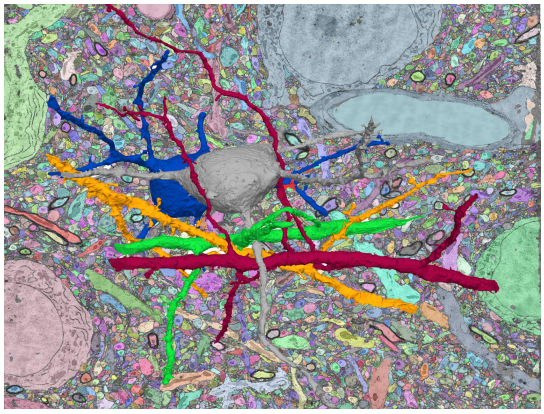


Figure 7. Results on Kasthuri et al. [16] *S1*. Fast light-weight 3C-Max-Out operating on 3-D seeds, without agglomeration. Background: Segmented section. Foreground: five 3-D reconstructed objects.

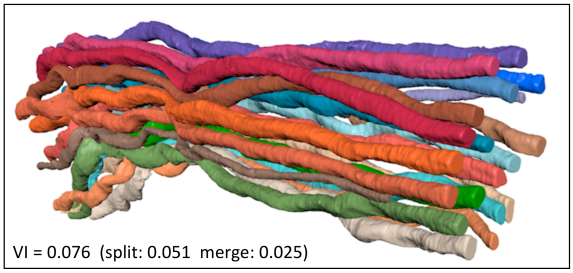


Figure 8. 3C-Maxout-Out Results of recursively tracking all objects (axons) directly from the PNS raw images (no post-processing).

requires synapse ground truth and hence is not evaluated in our first experiment.

For the first test, we used the lightest known successful design of FCNs for connectomics of [22] (*Max-Out*) (for border and 3C computations), reconstructing the test set of [16] (*S1*) (3C with FOV of 109 pixels). The NRI score of this 3C-MaxOut segmentation was 0.54, compared to 0.41 of a traditional agglomeration pipeline [22]. For the second test, we were granted permission to reconstruct a recently collected rat cortex dataset of the Lichtman group at Harvard (*ECS*). This test allowed to compare 3C to the excellent agglomerative approach of [28] (4th on SNEMI3D), while using exactly their U-Net [33] border as inputs to our 3C network. On the test set our NRI score was 0.86, compared to 0.73 of the agglomeration pipeline.

Large-Scale Reconstruction (*S1*): Finally, we also ran a fast version of 3C on the entire S1 dataset (90 GigaVoxel: 1840 slices, 6 nm x 6 nm x 30 nm per voxel). In this experiment, we omitted the bipartite step of the reconstruction algorithm to achieve better scalability and letting 3C run on 3-D masks computed by local minima of the border probability maps. This implementation is highly scalable since it has no agglomeration step, while the 3C masks are updated on-the-fly in a streaming fashion every 100 slices. The Max-Out implementation is attractive for large-scale systems because it is efficiently parallelized on multi-core systems with excellent cache behavior on CPUs (nominated best paper in PPoPP for these qualities). Figure 7 shows five objects that span the whole volume of S1 (Video will become available on the paper's website).

3.3. Peripheral Nervous System (*PNS*) Dataset

We were curious to see how well the 3C framework can be utilized to track objects completely recursively and based on raw images, in a pure streaming mode. This is the ultimate test of the tracking ability of the 3C algorithm, eliminating dependencies on agglomerating and post-processing procedures (as done by our full reconstruction framework and by others, e.g., FFN [14]).

We chose a previously unpublished motor nerve bundle from a (newborn) mouse contributed by the Lichtman lab at Harvard for this purpose because it is a closed system in which all objects are visible in the first and last image sections of the 915-images dataset. This dataset is important to neurobiologists since it contains the entire neural input (21 axons) of a complete tiny muscle near the neck.

We applied the 3C algorithm using the lightweight FCN Max-Out architecture of Matveev et al. [22], currently the fastest border detector in connectomics, which had already

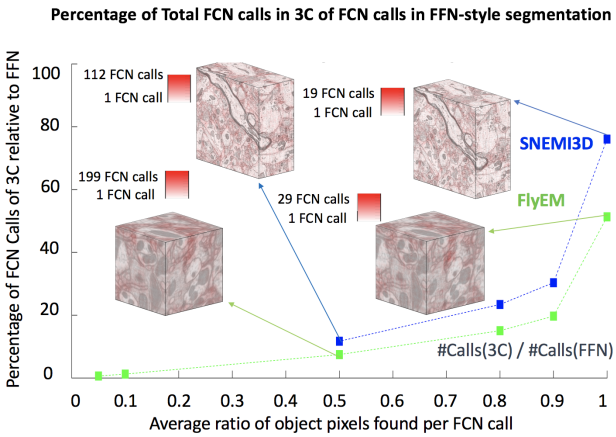


Figure 9. Compute cost per pixel using FFN-Style segmentation. We computed the number of times a pixel is participating in object detection (red) for two public datasets (SNEMI3D, FlyEM), and compared to the number of classification calls in 3C.

been used for single-object tracking [24]. Details of architecture and training are presented in the Supplementary Materials.

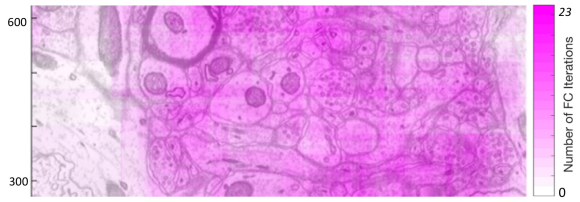
The 3C algorithm was able to track all objects without painting within any wrong object (i.e., zero merge errors); results are shown in Figure 8. Out of the 21 axons, 20 were recursively reconstructed to their full extent (i.e., split errors in only one object). One extremely thin axon disappeared from the image and reappeared after 7 sections and was not reconstructed. The axon run-length for all reconstructed axons was above 70 microns (and > 900 sections) until all of these exited the volume on the last slice in the image stack.

This experiment demonstrates two capabilities: **a)** 3C-Maxout performs well in a tracking task directly from raw images, even in the difficult connectomics regime. **b)** Our training procedures show satisfying generalization abilities, learning from a relatively small number of examples.

4. Scalability Comparisons

In this section, we compare the relative scalability of 3C to FFN and MaskExtend, as far as possible without having access to the FFN code. 3C is a direct generalization of the FFN and MaskExtend techniques [14, 24], which augment the (pixel) support set of each object, one at a time. The 3C technique simultaneously augments all the objects from its input image(s) after merely a logarithmic number of iterations (see Figure 3). This allows us directly to compare the two types of approaches based on the number of iterations required, ignoring details of implementation.

FFN: We compare the number of FCNs calls in FFN with those in 3C assuming both reconstruct all objects flawlessly. We fix the FCN model for the two approaches. We note that although 3C and FFN invoke their FCNs a logarithmic and a linear number of times respectively, FFN can



	Instructions	Cycles	L1 Cache Pressure
MaskExtend	$101,991 * 10^9$	$64,244 * 10^9$	$1,242.953MB/sec$
3C	$17,656 * 10^9$	$22,506 * 10^9$	$601,961MB/sec$

Figure 10. Compute cost per pixel with single-object tracking methods [24]. The number of calls per pixel is color-coded in purple. For the highly dense areas 23 calls of the object detector are required. Table. Performance counter statistics for the execution of [24] and the 3C-Max-Out FCN on a stack of 100 images.

run on very small inputs, centered around a small region of interest, and in this way saves overlaps with future iterations on nearby objects. At each iteration, FFN will output an entire 3-D mask of the object around the center pixel. We assume that a fraction of those pixels will require revisiting (zero for best-case scenario). Figure 9 depicts the number of FCN calls and their ratio for FlyEM [35] and SNEMI3D [4] for FFN and 3C. Each pixel in 3C participates in a logarithmic function of the number of objects visible in its field of view (FoV), color-coded red in the data cubes of Figure 9. The y-axis depicts the ratio between the FCN calls by 3C to that of FFN, for several ratios of object pixels found per FCN call. A zero ratio means that no pixels are found for the object in a single FCN call, while 1 means that all object pixels are found and require no further revisiting. We can see from the graph that 3C is much more efficient than FFN when there is a fraction of object pixels that require revisit after a single call of the FCN. The revisiting of some pixels is also reported by [14], as the 3-D output has greater uncertainty far from the initial pixels. For a revisit ratio of 0.5, 3C is more than 10x faster than FFN on FlyEM, whereas if predictions are perfect (ratio=1), it is only about 1.3x faster.

MaskExtend: Figure 10 repeats the above procedure with MaskExtend [24], comparing its number of FCN calls (color-coded in purple) with those in 3C. As before, MaskExtend is more wasteful than 3C, propagating some pixels into its FCN model 23 times. In addition, the instruction and cycle counts as well as the L1 Cache pressure are much larger for MaskExtend (for equal multi-core infrastructure and inference system [22]).

5. Conclusion

The domain of connectomics presents an especially difficult task for image segmentation. Manually annotating the neurons of a cubic millimeter of mouse brain would require thousands of human-years, and due to the yet-unsatisfying accuracy of automatic segmentation, proofreading remains out of scope. New computer vision techniques are hence in

great demand, to unravel even a minute mammalian circuit at its entirety. Foremost, here we presented a simpler, more "end-to-end" and "less moving parts" approach, improving on the accuracy of many leading connectomics systems. Our solution is computationally cheap, can be achieved with lightweight FCNs, and is at least an order of magnitude faster than its relative, flood-filling networks. Although the main theme of this paper was tackling neuronal reconstruction, we believe that the mechanism to achieve it, a 3C-based FCN mapper of instance segmentation from one image to another, can inspire 2-D/3-D instance segmentation or video tracking applications.

References

[1] A. Adam, E. Rivlin, and I. Shimshoni. Robust fragments-based tracking using the integral histogram. In *Computer vision and pattern recognition, 2006 IEEE Computer Society Conference on*, volume 1, pages 798–805. IEEE, 2006.

[2] B. Andres, U. Köthe, M. Helmstaedter, W. Denk, and F. A. Hamprecht. Segmentation of SBFSEM volume data of neural tissue by hierarchical classification. In *Joint Pattern Recognition Symposium*, pages 142–152. Springer, 2008.

[3] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik. Contour detection and hierarchical image segmentation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 33(5):898–916, 2011.

[4] I. Arganda-Carreras, S. C. Turaga, D. R. Berger, D. Cireşan, A. Giusti, L. M. Gambardella, J. Schmidhuber, D. Laptev, S. Dwivedi, J. M. Buhmann, et al. Crowdsourcing the creation of image segmentation algorithms for connectomics. *Frontiers in neuroanatomy*, 9:142, 2015.

[5] T. Beier, C. Pape, N. Rahaman, T. Prange, S. Berg, D. D. Bock, A. Cardona, G. W. Knott, S. M. Plaza, L. K. Scheffer, et al. Multicut brings automated neurite segmentation closer to human performance. *Nature Methods*, 14(2):101–102, 2017.

[6] M. Berning, K. M. Boergens, and M. Helmstaedter. SegEM: efficient image analysis for high-resolution connectomics. *Neuron*, 87(6):1193–1206, 2015.

[7] D. Ciresan, A. Giusti, L. M. Gambardella, and J. Schmidhuber. Deep neural networks segment neuronal membranes in electron microscopy images. In *Advances in neural information processing systems*, pages 2843–2851, 2012.

[8] J. Fan, W. Xu, Y. Wu, and Y. Gong. Human tracking using convolutional neural networks. *IEEE Transactions on Neural Networks*, 21(10):1610–1623, 2010.

[9] R. Girdhar, G. Gkioxari, L. Torresani, M. Paluri, and D. Tran. Detect-and-track: Efficient pose estimation in videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 350–359, 2018.

[10] K. Haris, S. N. Efstratiadis, N. Maglaveras, and A. K. Katsaggelos. Hybrid image segmentation using watersheds and fast region merging. *IEEE Transactions on image processing*, 7(12):1684–1699, 1998.

[11] D. Held, S. Thrun, and S. Savarese. Learning to track at 100 fps with deep regression networks. In *European Conference on Computer Vision*, pages 749–765. Springer, 2016.

[12] M. Helmstaedter. Cellular-resolution connectomics: challenges of dense neural circuit reconstruction. *Nature methods*, 10(6):501–507, 2013.

[13] V. Jain, J. F. Murray, F. Roth, S. Turaga, V. Zhigulin, K. L. Briggman, M. N. Helmstaedter, W. Denk, and H. S. Seung. Supervised learning of image restoration with convolutional networks. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8. IEEE, 2007.

[14] M. Januszewski, J. Kornfeld, P. H. Li, A. Pope, T. Blakely, L. Lindsey, J. Maitin-Shepard, M. Tyka, W. Denk, and V. Jain. High-precision automated reconstruction of neurons with flood-filling networks. *Nature methods*, 15(8):605, 2018.

[15] K. Kang, W. Ouyang, H. Li, and X. Wang. Object detection from video tubelets with convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 817–825, 2016.

[16] N. Kasthuri, K. J. Hayworth, D. R. Berger, R. L. Schalek, J. A. Conchello, S. Knowles-Barley, D. Lee, A. Vázquez-Reina, V. Kaynig, T. R. Jones, et al. Saturated reconstruction of a volume of neocortex. *Cell*, 162(3):648–661, 2015.

[17] V. Kaynig, A. Vazquez-Reina, S. Knowles-Barley, M. Roberts, T. R. Jones, N. Kasthuri, E. Miller, J. Lichtman, and H. Pfister. Large-scale automatic reconstruction of neuronal processes from electron microscopy images. *Medical image analysis*, 22(1):77–88, 2015.

[18] S. Knowles-Barley, V. Kaynig, T. R. Jones, A. Wilson, J. Morgan, D. Lee, D. Berger, N. Kasthuri, J. W. Lichtman, and H. Pfister. RhoanaNet pipeline: Dense automatic neural annotation. *arXiv preprint arXiv:1611.06973*, 2016.

[19] K. Lee, J. Zung, P. Li, V. Jain, and H. S. Seung. Superhuman accuracy on the SNEMI3D connectomics challenge. *arXiv preprint arXiv:1706.00120*, 2017.

[20] J. W. Lichtman and W. Denk. The big and the small: challenges of imaging the brains circuits. *Science*, 334(6056):618–623, 2011.

[21] J. W. Lichtman, H. Pfister, and N. Shavit. The big data challenges of connectomics. *Nature neuroscience*, 17(11):1448–1454, 2014.

[22] A. Matveev, Y. Meirovitch, H. Saribekyan, W. Jakubiuk, T. Kaler, G. Odor, D. Budden, A. Zlateski, and N. Shavit. A multicore path to connectomics-on-demand. In *Proceedings of the 22Nd ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming*, PPoPP '17, pages 267–281, New York, NY, USA, 2017. ACM.

[23] M. Meilă. Comparing clusterings. *Journal of multivariate analysis*, 98(5):873–895, 2007.

[24] Y. Meirovitch, A. Matveev, H. Saribekyan, D. Budden, D. Rolnick, G. Odor, S. K.-B. T. R. Jones, H. Pfister, J. W. Lichtman, and N. Shavit. A multi-pass approach to large-scale connectomics. *arXiv preprint arXiv:1612.02120*, 2016.

[25] L. Najman and M. Schmitt. Geodesic saliency of watershed contours and hierarchical segmentation. *IEEE Transactions on pattern analysis and machine intelligence*, 18(12):1163–1173, 1996.

[26] J. Nunez-Iglesias, R. Kennedy, T. Parag, J. Shi, and D. B. Chklovskii. Machine learning of hierarchical clustering to segment 2D and 3D images. *PloS one*, 8(8):e71715, 2013.

[27] T. Parag, A. Chakraborty, S. Plaza, and L. Scheffer. A context-aware delayed agglomeration framework for electron microscopy segmentation. *PloS one*, 10(5):e0125825, 2015.

[28] T. Parag, F. Tschopp, W. Grisaitis, S. C. Turaga, X. Zhang, B. Matejek, L. Kamentsky, J. W. Lichtman, and H. Pfister. Anisotropic EM segmentation by 3D affinity learning and agglomeration. *arXiv preprint arXiv:1707.08935*, 2017.

[29] S. M. Plaza and S. E. Berg. Large-scale electron microscopy image segmentation in Spark. *arXiv preprint arXiv:1604.00385*, 2016.

[30] S. M. Plaza and J. Funke. Analyzing image segmentation for connectomics. *Frontiers in Neural Circuits*, 12:102, 2018.

[31] E. P. Reilly, J. S. Garretson, W. R. Gray Roncal, D. M. Kleissas, B. A. Wester, M. A. Chevillet, and M. J. Roos. Neural reconstruction integrity: A metric for assessing the connectivity accuracy of reconstructed neural networks. *Frontiers in Neuroinformatics*, 12:74, 2018.

[32] X. Ren and J. Malik. Learning a classification model for segmentation. In *null*, page 10. IEEE, 2003.

[33] O. Ronneberger, P. Fischer, and T. Brox. U-Net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015*, pages 234–241. Springer, 2015.

[34] S. Seung. *Connectome: How the brain's wiring makes us who we are*. Houghton Mifflin Harcourt, 2012.

[35] S.-y. Takemura, C. S. Xu, Z. Lu, P. K. Rivlin, T. Parag, D. J. Olbris, S. Plaza, T. Zhao, W. T. Katz, L. Umayam, et al. Synaptic circuits and their variations within different columns in the visual system of drosophila. *Proceedings of the National Academy of Sciences*, 112(44):13711–13716, 2015.

[36] S. C. Turaga, J. F. Murray, V. Jain, F. Roth, M. Helmstaedter, K. Briggman, W. Denk, and H. S. Seung. Convolutional networks can learn to generate affinity graphs for image segmentation. *Neural computation*, 22(2):511–538, 2010.

[37] R. Unnikrishnan, C. Pantofaru, and M. Hebert. Toward objective evaluation of image segmentation algorithms. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (6):929–944, 2007.

[38] S. Xingjian, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-c. Woo. Convolutional LSTM network: A machine learning approach for precipitation nowcasting. In *Advances in neural information processing systems*, pages 802–810, 2015.

[39] S. Yoo, K. Yun, J. Y. Choi, K. Yun, and J. Choi. Action-decision networks for visual tracking with deep reinforcement learning. CVPR, 2017.