OPEN ACCESS Freely available online

PLOS ONE

Machine Learning of Hierarchical Clustering to Segment 2D and 3D Images

Juan Nunez-Iglesias[1]*, Ryan Kennedy[2], Toufiq Parag[1], Jianbo Shi[2], Dmitri B. Chklovskii[1]

1 Janelia Farm Research Campus, Howard Hughes Medical Institute, Ashburn, Virginia, United States of America, **2** Department of Computer and Information Science, University of Pennsylvania, Philadelphia, Pennsylvania, United States of America

Abstract

We aim to improve segmentation through the use of machine learning tools during region agglomeration. We propose an active learning approach for performing hierarchical agglomerative segmentation from superpixels. Our method combines multiple features at all scales of the agglomerative process, works for data with an arbitrary number of dimensions, and scales to very large datasets. We advocate the use of variation of information to measure segmentation accuracy, particularly in 3D electron microscopy (EM) images of neural tissue, and using this metric demonstrate an improvement over competing algorithms in EM and natural images.

Citation: Nunez-Iglesias J, Kennedy R, Parag T, Shi J, Chklovskii DB (2013) Machine Learning of Hierarchical Clustering to Segment 2D and 3D Images. PLoS ONE 8(8): e71715. doi:10.1371/journal.pone.0071715

Editor: Xi-Nian Zuo, Institute of Psychology, Chinese Academy of Sciences, China

Received February 28, 2013; **Accepted** July 2, 2013; **Published** August 20, 2013

Funding: Funding was provided by the Howard Hughes Medical Institute. The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Competing Interests: The authors have declared that no competing interests exist.

* E-mail: jni@janelia.hhmi.org

Introduction

Image segmentation, a fundamental problem in computer vision, concerns the division of an image into meaningful constituent regions, or segments.

In addition to having applications in computer vision and object recognition (Figure 1), it is becoming increasingly essential for the analysis of biological image data. Our primary motivation is to understand the function of neuronal circuits by elucidating neuronal connectivity [1,2]. In order to distinguish synapses and follow small neuronal processes, resolutions of 10 nm are necessary in 3D and provided only by electron microscopy (EM). On the other hand, individual neurons often extend over millimeter ranges. This disparity of scales results in huge image volumes and makes automated segmentation an essential part of neuronal circuit reconstruction.

Additionally, automated segmentation of EM images presents significant challenges compared to that of natural images (Figure 2), including identical textures within adjacent neurons, mitochondria and vesicles within cells that look (to a classifier) similar to the boundaries between cells, and elongated, intertwined shapes where small errors in boundary detection result in large errors in neuron network topology. The methods we introduce here, however, are generally applicable and extend to images of arbitrary dimension, which we demonstrate by segmenting both EM data and natural image data.

A common approach in the field is to perform oversegmentation into small segments called *superpixels*, and then to merge these into larger regions [3,4]. A merging algorithm consists of a merging criterion, or policy, that determines which merges are most likely, and a merging strategy, that determines how to merge segments (for example, through simulated annealing [3], probabilistic graphical models [5], or hierarchical clustering [6]). Often, much effort is devoted to the generation of a pixel-level boundary probability map by training a classifier that predicts boundaries between objects from pixel-level features [4,7–11]. Meanwhile, oversegmentation and agglomeration are performed in a straightforward fashion, for example using watershed [12] to generate superpixels, and the mean boundary probability over the contour separating adjacent superpixels [4] as the merge criterion. Boundary mean has been a relatively effective merge priority function for hierarchical agglomeration because every merge results in longer boundaries along adjacent regions. Therefore, as the agglomeration proceeds, the mean becomes an increasingly reliable estimate of the merge probability.

We hypothesized that agglomeration could be improved by using more information than just the boundary mean, despite the latter's desirable characteristics. A priority function could draw from many additional features, such as boundary variance and region texture. Using training data in which pairs of superpixels have been labeled as "merge" or "don't merge", we could then apply machine learning techniques to predict from those features whether two superpixels should be merged. With that simple approach, however, we found that the guaranteed effectiveness of the mean could easily disappear. Similarly to the case with the boundary mean, the region sizes progressively increase and so does the amount of evidence for or against a merge. However, we could encounter a combination of features for which we had no training data.

To get around this problem, we developed an active learning paradigm that generates training examples across every level of the agglomeration hierarchy and thus across very different segment scales. In active learning, the algorithm determines what example it wants to learn from next, based on the previous training data. For agglomerative segmentation, we ask the classifier which two regions it believes should be merged, and compare those against

Figure 1. Illustration of the advantages of our approach. *Top left*: Input image. *Top right*: segmentation using only a boundary map [4]. *Bottom left*: using multiple cues with a single level of learning. *Bottom right*: using multiple cues with our agglomerative learning method.
doi:10.1371/journal.pone.0071715.g001

the ground truth to obtain the next training example. By doing this at all levels of the agglomeration hierarchy, we ensure that we have samples from all parts of the feature space that the classifier is likely to encounter.

Past learning methods either used a manual combination of a small number of features [4,13], or they used more complex feature sets but operated only on the scale of the original superpixels [14,15]. (We discuss two notable exceptions [16,17] in the Discussion section.) We instead learn by performing a hierarchical agglomeration while comparing to a gold standard segmentation. This allows us to obtain samples from region pairs at all scales of the segmentation, corresponding to levels in the hierarchy. Although Jain *et al.* independently presented a similar approach called LASH [6], there are some differences in our approach that yield some further improvements in segmentation quality, as we explain later.

We describe below our method for collecting training data for agglomerative segmentation. Throughout a training agglomeration, we consult a human-generated gold standard segmentation to determine whether each merge is correct. This allows us to learn a merge function at the many scales of agglomeration. We show that our learned agglomeration outperforms state of the art agglomeration algorithms in natural image segmentation (Figure 1).

To evaluate segmentations, we advocate the use of variation of information (VI) as a metric and show that it can be used to improve the interpretability of segmentation results and aid in their analysis.

The ideas in this work are implemented in an open-source Python library called Gala that performs agglomeration learning and segmentation in arbitrary dimensions.

Methods

1 Active Learning of Agglomeration

The method described below is illustrated and summarized in Figure 3.

Let $I \in \mathbb{R}^n$ be an input image of dimension d having n pixels. (Throughout the text, we will use "pixel" and "voxel" interchangeably.) We assume an initial oversegmentation S of I into $m << n$ "superpixels", $S = \{S_1, \ldots, S_m\}$, defined as disjoint sets of connected pixels that do not substantially cross true segment boundaries. An agglomerative segmentation of the image is defined by a grouping $A = \{A_1, \ldots, A_p\}$ of disjoint sets of superpixels from S. It is a testament to the power of abstraction of agglomerative methods that we will no longer use d, or n in what follows.

There are many methods to obtain A from I and S. We chose the framework of hierarchical agglomeration for its inherent scalability: each merge decision is based only on two regions. For this method we require two definitions: a region adjacency graph (RAG) and a merge priority function (MPF) or policy.

The RAG is defined as follows. Each node v_i corresponds to a grouping A_i of superpixels, where we initialize $A_i \equiv \{S_i\}$, for $i = 1, \ldots, m$. An edge $e_{i,j}$ is placed between v_i and v_j if and only if a pixel in A_i is adjacent to a pixel in A_j.

We then define the merge priority function (MPF) or policy $\pi : \{\mathcal{G}, V \times V\} \mapsto \mathcal{D} \subseteq \mathbb{R}$, where $\mathcal{G}$ is the set of RAGs and V is the set of nodes beloging to a RAG. $\mathcal{D}$, the range of the policy, is typically [0,1], but could be any totally ordered set. Hierarchical agglomeration is the process of progressively merging nodes in the graph in the order specified by π. When two nodes are merged, the

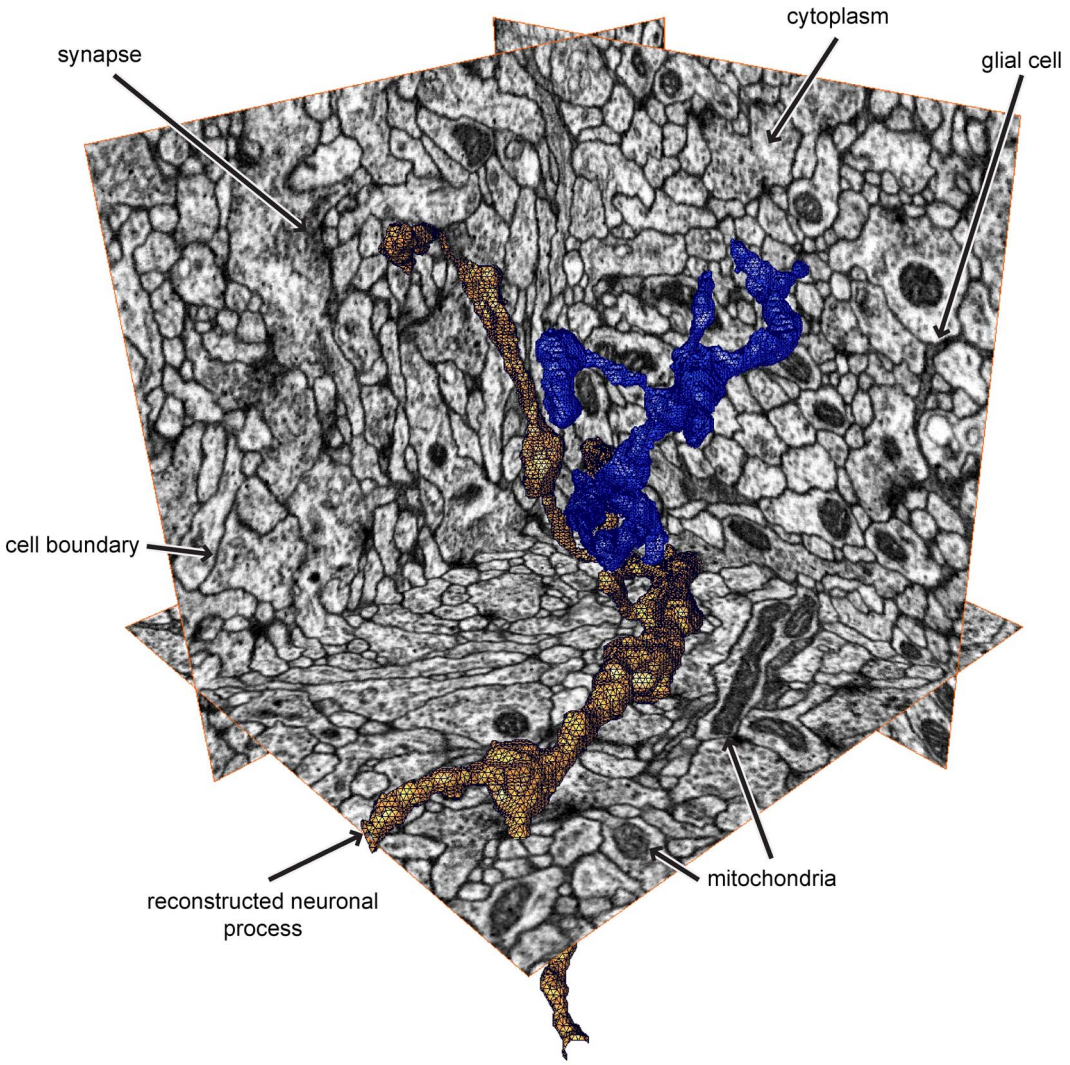

Figure 2. Representative 3D EM data and sample reconstructions. Note that the data is isotropic, meaning it has the same resolution along every axis. The goal of segmentation here is to partition the volume into individual neurons, two of which are shown in orange and blue. The volume is densely packed by these thin neuronal processes taking long, tortuous paths.
doi:10.1371/journal.pone.0071715.g002

set of edges incident on the new node is the union of their incident edges, and the MPF value for those edges is recomputed. (A general policy might need to be recomputed for *all* edges after a merge, but here we consider only local policies: the MPF is only recomputed for edges for which one of the incident nodes has changed).

The mean probability of boundary along the edge is but one example of a merge priority function. In this work, we propose finding an optimal π using a machine learning paradigm. To do this, we decompose π into a feature map $f : \{\mathcal{G}, V \times V\} \mapsto \mathbb{R}^q$ and a classifier $c : \mathbb{R}^q \mapsto [0,1]$. Then take $\pi = c \circ f$, and the problem of learning π reduces to three steps: finding a good training set, finding a good feature set, and training a classifier. In this work, we focus on the first question. The method we describe in the following paragraphs is summarized in Figure 3.

We first define the optimal agglomeration A^* given the superpixels S and a gold standard segmentation U by assigning each superpixel to the ground truth segment with which it shares the most overlap:

$$A^*(S,U) = \{A_i^*\}_{i=1}^{|U|} \tag{1}$$

$$\text{where } A_i^* = \left\{ S_j : i = \arg\max_{k=1,\ldots,|U|} |S_j \cap U_k| \right\}_{j=1}^{|S|}. \tag{2}$$

From this, we can work out a label between two regions: -1 or "should merge" if both regions are subsets of the same gold standard region, 1 or "don't merge" if each region is a subset of a different gold standard region, and 0 or "don't know" if either region is not a subset of any gold standard region:

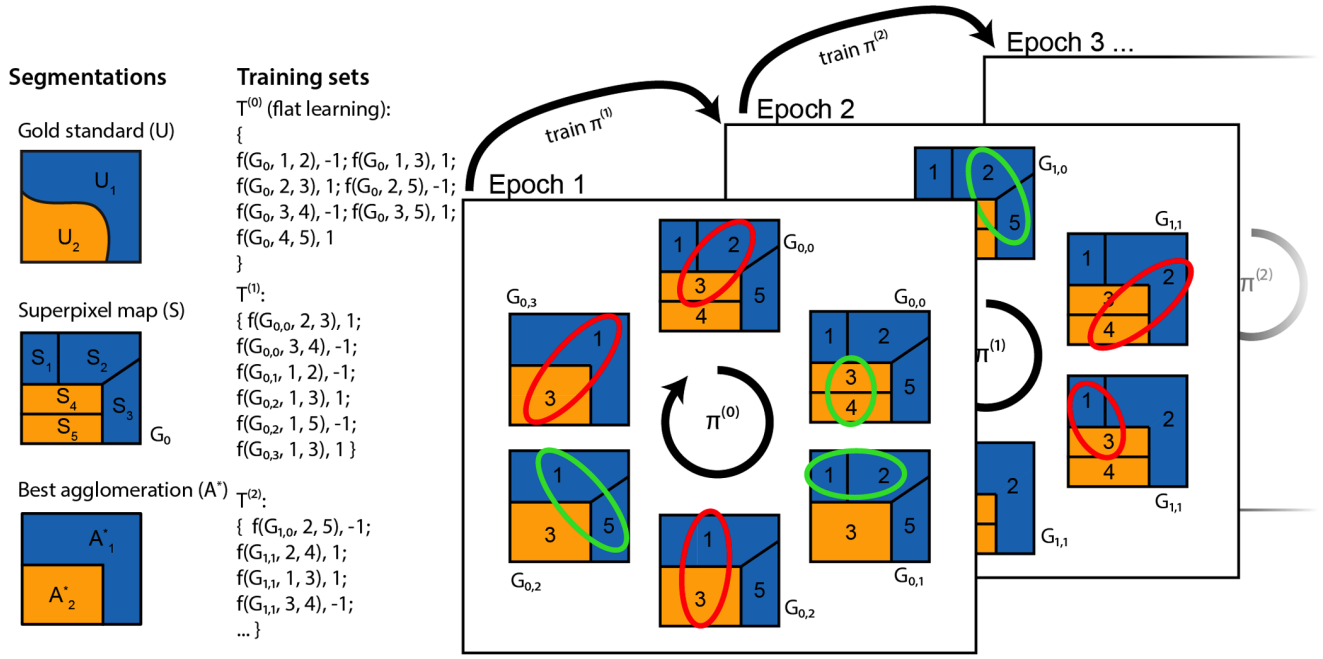

Figure 3. Schematic of our approach. *First column:* A 2D image has a given gold standard segmentation U, a superpixel map S (which induces an initial region adjacency graph, G_0), and a "best" agglomeration given that superpixel map A^*. *Second column:* Our procedure gives training sets at all scales. "f" denotes a feature map. $G_{i,j}$ denotes graph agglomerated by policy $\pi^{(i)}$ after j merges. Note that j only increases when we encounter an edge labeled -1. *Third column:* We learn by simultaneously agglomerating and comparing against the best agglomeration, terminating when our agglomeration matches it. The highlighted region pair is the one that the policy, $\pi^{(k)}$, determines should be merged next, and the color indicates the label obtained by comparing to A^*. After each training epoch, we train a new policy and undergo the same learning procedure. For clarity, in the second and third columns, we abbreviate A_i with just the index i in the second and third arguments to the feature map. For example, $f(G_{0,0},2,3)$ indicates the feature map from graph $G_{0,0}$ and edge (v_2,v_3), corresponding to regions A_2 and A_3.
doi:10.1371/journal.pone.0071715.g003

$$\ell(A^*,A_i,A_j)=\begin{cases}-1, \text{ if } A_i\subseteq A^*_u, A_j\subseteq A^*_u \text{ for some } u\\ 1, \text{ if } A_i\subseteq A^*_u, A_j\subseteq A^*_v \text{ for some } u\neq v\\ 0, \text{ otherwise}\end{cases} \quad (3)$$

Now, given an initial policy $\pi^{(0)}$ and a feature map f, we can obtain an initial agglomeration training set as follows: Start with an initially empty training set T. For every edge (u,v) suggested by $\pi^{(0)}$, compute its label $\ell_{u,v}$. If it is -1, add the training example $\{f(G,u,v),\ell_{u,v}\}$ to T and merge nodes u and v. Otherwise, add the training example but do not merge the two nodes. Repeat this until the agglomeration induced by the RAG G matches A^*, and use T to train a classifier c. We call this loop a training epoch.

After epoch $k=1,\ldots,K$, we obtain a classifier $c^{(k)}$ that induces a policy $\pi^{(k)}=c^{(k)}\circ f$.

There remains the issue of choosing a suitable initial policy. We found that the mean boundary probability or even random numbers work well, but, to obtain the fastest convergence, we generate the training set consisting of every labeled edge in the initial graph (with no agglomeration), $T^{(0)}=\{(f(G,e),\ell_e)\}_{e\in E}$, and an initial policy is given by the classifier trained on this "flat learning" set.

2 Cues and Features

In this section, we describe the feature maps used in our work. We call primitive features "cues", from which we compute the actual features used in the learning. We did not focus on these maps extensively, and expect that these are not the last word with respect to useful features for agglomerative segmentation learning.

For natural images, we use the gPb oriented boundary map [4] and a texton map [18]. For any feature calculated from gPb, the probability associated with an edge pixel was taken from the oriented boundary map corresponding to the orientation of the edge pixel. We calculated each edge pixel's orientation by fitting line segments to the boundary map and calculating the orientation of each line segment. By fitting line segments we are able to accurately calculate the orientation of each edge pixel, even near junctions where the gradient orientation is ambiguous [4]. In addition, we use a texton cue that includes L*a*b* color channels as well as filter responses to the MR8 filter bank [19,20]. The textons were discretized into 100 bins using the k-means algorithm.

For EM data, we use four separate cues: a probability map of cell boundaries, cytoplasm, mitochondria, and glia. Mitochondria were labeled by hand using the active contours function in the ITK-SNAP software package [21]. Boundaries and glia were labeled using the manually proofread segmentation in Raveler [21], with cytoplasm being defined as anything not falling into the prior three categories. Our initial $500\times500\times500$ voxel volume was divided into 8 $250\times250\times250$ voxel subvolumes. To obtain the pixel-level probability map for each subvolume, we trained using the fully labeled 7 other subvolumes using Ilastik [22] and applied the obtained classifier. Rather than using all the labels, we used all the boundary labels ($\sim$10 M total) and smaller random samples of the cytoplasm, mitochondria, and glia labels ($\sim$1 M each). We found that this resulted in stronger boundaries and much reduced computational load.

Let u and v be adjacent nodes of the current segmentation, and let $b_{u,v}$ be the boundary separating them. From each cue described above, we calculated the following features, which we concatenated into a single feature vector.

2.1 Pixel-level features. For u, v, and $b_{u,v}$, we created a histogram of 10 or 25 bins, and computed 3 or 9 approximate quantiles by linear interpolation of the histogram bins. We also included the number of pixels, the mean value and 3 central moments. Additionally, we used the differences between the central moments of u and v, and the Jensen-Shannon divergence between their histograms.

2.2 Mid-level features. For natural image segmentation, we added several mid-level features based on region orientation and convex hulls. For orientation features, the orientation of each region is estimated from the region's second moment matrix. We use the angle between the two regions, as well and the angles between each region and a line segment connecting their centroids, as features. For convex hull features, we calculated the volume of the convex hull of each region, as well as for their union, and used the ratios between these convex hulls volumes and the volumes of the regions themselves as a measure of the convexity of regions.

Results

1 Evaluation

Before we describe the main results of our paper, a discussion of evaluation methods is warranted, since even the question of the "correct" evaluation method is the subject of active research.

The most commonly used method is boundary precision-recall [4,7]. A test segmentation and a gold standard can be compared by finding a one-to-one match between the pixels constituting their segment boundaries. Then, matched pixels are defined as true positives (TP), unmatched pixels in the automated segmentation are false positives (FP), and unmatched pixels in the gold standard are false negatives (FN). A measure of closeness to the gold standard is then given by the precision and recall values, defined as $P=TP/(TP+FP)$ and $R=TP/(TP+FN)$. The precision and recall can be combined into a single score by the F-measure, $F=2PR/(P+R)$. A perfect segmentation has $P=R=F=1$.

The use of boundary precision-recall has deficiencies as a segmentation metric, since small changes in boundary detection can result in large topological differences between segmentations. This is particularly problematic in neuronal EM images, where the goal of segmentation is to elucidate the connectivity of extremely long, thin segments that have tiny (and error-prone) branch points. For such images, the number of mislabeled boundary pixels is irrelevant compared to the precise location and topological impact of the errors [9,10]. In what follows, we shall therefore focus on region-based metrics, though we will show boundary PR results in the context of natural images to compare to previous work.

The region evaluation measure of choice in the segmentation literature has been the Rand index (RI) [23], which evaluates pairs of points in a segmentation. For each pair of pixels, the automatic and gold standard segmentations agree or disagree on whether the pixels are in the same segment. RI is defined as the proportion of point pairs for which the two segmentations agree. Small differences along the boundary have little effect on RI, whereas differences in topology have a large effect.

However, RI has several disadvantages, such as being sensitive to rescaling and having a limited useful range [24]. An alternative segmentation distance is the variation of information (VI) metric [25], which is defined as a sum of the conditional entropies between two segmentations:

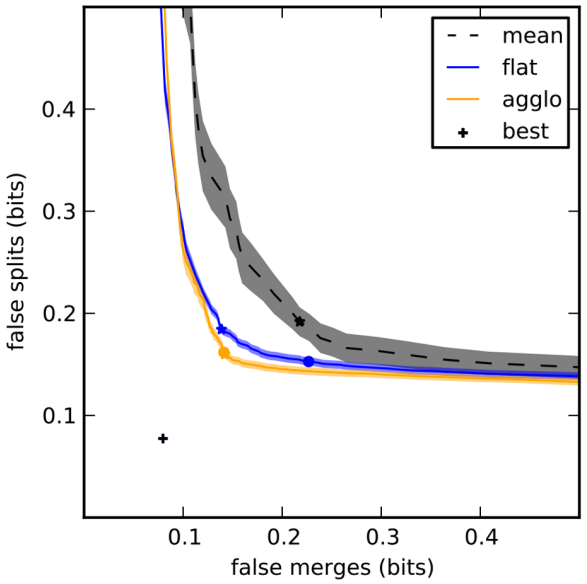

Figure 4. Split VI plot for different learning or agglomeration methods. Shaded areas correspond to mean $\pm$ standard error of the mean. "Best" segmentation is given by optimal agglomeration of superpixels by comparing to the gold standard segmentation. This point is not (0,0) because the superpixel boundaries do not exactly correspond to those used to generate the gold standard. The standard deviation of this point ($n=8$) is smaller than the marker denoting it. Stars mark minimum VI (sum of false splits and false merges), circles mark VI at threshold 0.5.
doi:10.1371/journal.pone.0071715.g004

$$VI(S,U)=H(S|U)+H(U|S), \tag{4}$$

where S is our candidate segmentation and U is our ground truth. $H(S|U)$ can be intuitively understood as the answer to the question: "given the ground truth (U) label of a random voxel, how much more information do we need to determine its label in the candidate segmentation (S)?"

VI overcomes all of the disadvantages of the Rand index and has several other advantages, such as being a formal metric [25]. Although VI has been used for evaluating natural image segmentations [4], its use in EM has been limited. In what follows, we explore further the properties of VI as a measure of segmentation quality and conclude that it is superior to the Rand index for this task, especially in the context of neuronal images.

Like the Rand index, VI is sensitive to topological changes but not to small variations in boundary changes, which is critical in EM segmentation. Unlike RI, however, errors in VI scale linearly in the size of the error whereas the RI scales quadratically. This makes VI more directly comparable between volumes. In addition, because RI is based on point pairs, and because the vast majority of pairs are in disjoint regions, RI has a limited useful range very near 1, and that range is different for each dataset. In contrast, VI ranges between 0 and $\log(K)$, where K is the number of objects in the image. Furthermore, due to its basis in information theory, it is measured in bits, which makes it easily interpretable. For example, a VI value of 1 means that on average, each neuron is split in 2 equally-sized fragments in the automatic segmentation (or vice-versa). No such mapping exists between RI and a physical intuition. Finally, because VI is a metric, differences in VI correspond to our intuition about distances in Euclidean space,

which allows easy comparison of VI distances between many candidate segmentations.

VI is by its definition (Equation 4) broken down into an oversegmentation/false-split term $H(S|U)$ and an undersegmentation/false-merge term $H(U|S)$. To make this explicit, we introduce in this work the split-VI plot of $H(S|U)$ on the y-axis against $H(U|S)$ on the x-axis, which shows the tradeoff between oversegmentation and undersegmentation in a manner similar to boundary PR curves (see Figures 4 and 5). Since VI is the sum of those two terms, isoclines in this plot are diagonal lines sloping down. A slope of -1 corresponds to equal weighting of under- and oversegmentation, while slopes of $-a$ correspond to a weighting of a of undersegmentation relative to oversegmentation. Finding an optimal segmentation VI is thus as easy as finding a tangent for a given curve. The split-VI plot is particularly suited to agglomerative segmentation strategies: the merging of two segments can only result in an arc towards the bottom-right of the plot; false merges result in mostly rightward moves, while true merges result in mostly downward moves.

In addition, each of the under- and oversegmentation terms can be further broken down into its constituent errors. The oversegmentation term of a VI distance is defined as $H(S|U) = -\sum_u P(u)H(S|U=u)$. From this definition, we introduce the VI breakdown plot, of $H(S|U=u)$ against $P(U=u)$ for every value of u, and vice-versa. In Figure S1, we show how this breakdown can be used to gain insight into the errors found in automatic segmentations by identifying those segments that contribute most to the VI.

In light of the utility of VI, our evaluation is based on VI, particularly for EM data. For natural images, we also present boundary precision-recall and other measures, to facilitate comparison to past work. In addition to boundary PR values, RI, and VI, we show values for the covering, a measure of overlap between segments [4]. For each of these measures, we show results for the optimal dataset scale (ODS), the optimal image scale (OIS), and for the covering measure we also show the result of the best value using any threshold of the segmentation (Best). For boundary evaluation, we also report the average precision (AP), which is the area under the PR curve.

2 Algorithms

We present in this paper the segmentation performance of several agglomerative algorithms, defined below. As a baseline we show results from agglomeration using only the mean boundary probability between segments ("mean").

For natural images, we also show the results when oriented boundary maps are used ("mean-orient"), which is the algorithm presented by Arbeláez *et al.* [4] and was shown in their work to outfoperform previous agglomerative methods. (Our results vary slightly from those of Arbeláez, due to implementation differences).

Our proposed method, using an actively-trained classifier and agglomeration, is denoted as "agglo". For details, see Section 1 of the Methods and Figure 3. Briefly, using a volume for which the true segmentation is known, we start with an initial oversegmentation, followed by an agglomeration step in which every merge is checked against the true segmentation. True merges proceed and are labeled as such, while false merges do not proceed, but are labeled as false. This accumulates a training dataset until the agglomeration matches the true segmentation. At this point, a new agglomeration order is determined by training, and the procedure is repeated a few times to obtain a large training dataset, the statistics of which will match those encountered during a test agglomeration.

A similar method, described by Jain *et al.* [6] is denoted as "lash" in Figures S2 and S3. In that work, merges proceed regardless of whether they are true or false according to the ground truth, and each merge is labeled by taking the sign of the change in Rand index resulting from the merge. We used our own implementation of LASH, using our own feature maps, to compare only the performance of the learning strategies.

In order to show the effect of our agglomerative learning, we also compare using a classifier trained on only the initial graph before agglomeration ("flat").

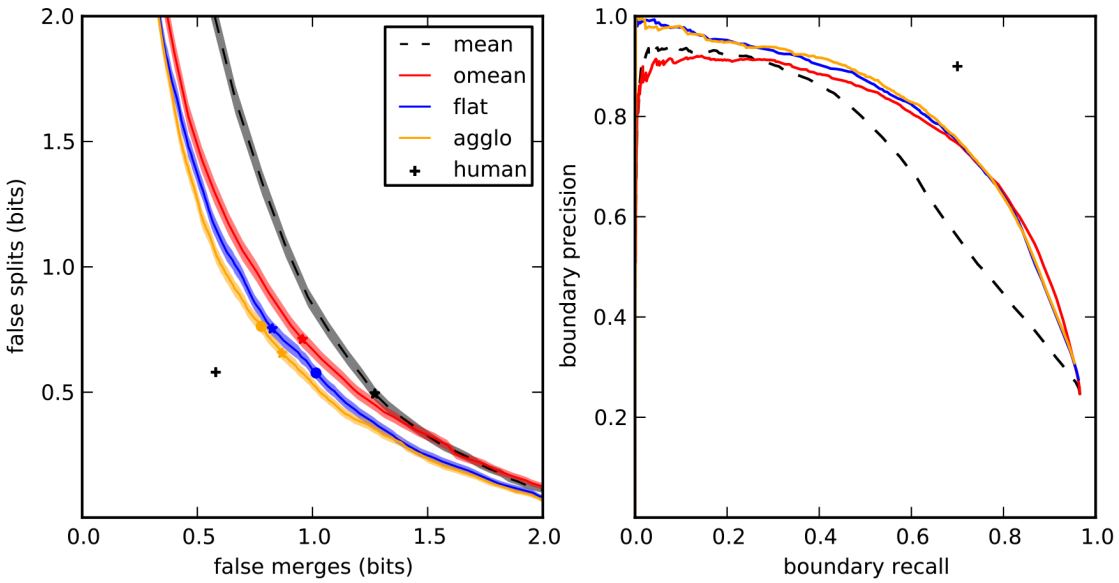

Figure 5. Evaluation of segmentation algorithms on BSDS500. Left: split-VI plot. Stars represent optimal VI (minimum sum of x and y axis), circles represent VI at threshold $p=0.5$. Right: boundary precision-recall plot.
doi:10.1371/journal.pone.0071715.g005

3 Segmentation of FIBSEM Data

Our starting dataset was a $500 \times 500 \times 500$ voxel isotropic volume generated by focused ion beam milling of *Drosophila melanogaster* larval neuropil, combined with scanning electron microscope imaging of the milled surface [26]. This results in a volume with 10 nm resolution in the x, y and z axes, in which cell boundaries, mitochondria, and various other cellular components appear dark (Figure 2). Relative to other EM modalities, such as serial block face scanning EM (SBFSEM) [27] or serial section transmission EM (ssTEM) [28,29], FIBSEM has a smaller field of view, but yields isotropic resolution and can be used to reconstruct important circuits. Recently published work has demonstrated a $28 \times 28 \times 56$ volume imaged at $7 \times 7 \times 7$ resolution [30], and the latest volumes being imaged exceed $65 \times 65 \times 65$ with 8 nm isotropic voxels (C. Shan Xu and Harald Hess, pers. commun). These dimensions are sufficient to capture biologically interesting circuits in the *Drosophila* brain, such as multiple columns in the medulla (part of the visual system) [31] or the entire antennal lobe (involved in olfaction) [32].

To generate a gold standard segmentation, an initial segmentation based on pixel intensity alone was manually proofread using software specifically designed for this purpose (called Raveler) [2]. We then used the 8 probability maps described in Section 2 of the Methods in a cross-validation scheme, training on one of the 8 volumes and testing on the remaining 7, for a total of 56 evaluations per training protocol (but only 8 for mean agglomeration, which requires no training).

Compared with mean agglomeration or with a flat learning strategy, our active agglomerative learning algorithm improved segmentation performance modestly but significantly (Figure 4).

In addition, the agglomerative training appears to dramatically improve the probability estimates from the classifier. If the probability estimates from a classifier are accurate, then, under reasonable assumptions, we expect the minimum VI to occur at or near $p = 0.5$. However, this is not what occurs after learning on the flat graph: the minimum occurs much earlier, at $p = 0.28$, after which the VI starts climbing. In contrast, after agglomerative learning, the minimum does indeed occur at $p = 0.51$ (Figure 6a).

This suggests that agglomerative learning improves the classifier probability estimates. Indeed, the minimum VI and the VI at $p = 0.5$ converge after 4 agglomerative learning epochs and stay close for 19 epochs or more (Figure 6b). This accuracy can be critical for downstream applications, such as estimating proofreading effort [33].

4 Segmentation of the SNEMI3D Challenge Data

Although we implemented our algorithm to work specifically on isotropic data, we attempted to segment the publicly available SNEMI3D challenge dataset (available at http://brainiac2.mit.edu/SNEMI3D), a $6 \times 6 \times 30$ resolution serial section scanning EM (ssSEM) volume. For this, we used the provided boundary probability maps of Ciresan *et al.* [34]. A fully 3D workflow, including 3D watershed supervoxels, predictably did not impress (adjusted Rand error 0.335, placed 3rd of 4 groups, 15th of 21 attempts). However, with just one modification (generating watershed superpixels in each plane separately), running GALA out of the box in 3D placed us in 1st place (as of this submission), with an adjusted Rand error of 0.125. (Note: our group name in the challenge is "FlyEM". To see individual submissions in addition to group standings, it is necessary to register and log in.) This demonstrates that the GALA framework is general enough to learn simultaneous 2D segmentation and 3D linkage, despite its focus on fully isotropic segmentation. We expect that the addition of linkage-specific features would further improve GALA's performance in this regime.

5 Berkeley Segmentation Dataset

We also show the results of our algorithm on the Berkeley Segmentation Dataset (BSDS500) [4], a standard natural image segmentation dataset, and show a significant improvement over the state of the art in agglomerative methods.

Our algorithm improves segmentation as measured by all the above evaluation metrics (Table 1). At the optimal dataset scale (ODS), our algorithm reduced the remaining error between oriented mean agglomeration [4] and human-level segmentation by at least 20% for all region metrics, including a reduction of 28% for VI. The improvement obtained by agglomerative learning over flat learning is smaller than in EM data; we believe this is due to the smaller range of scales found between superpixels and segments in our natural images. Nevertheless, this slight improvement demonstrates the advantage of our learning method: by learning at all scales, the classifier achieves a better segmentation since it can dynamically adjust how features are interpreted based on the region size.

Figure 5a shows the split VI plot while Figure 5b shows the boundary precision-recall curves. The results are similar in both cases, with agglomerative learning outperforming all other algorithms.

In Figure 7, we show the performance of our algorithm on each test image compared to the algorithm in [4]. The majority of test images show a better (i.e. lower) VI score.

Several example segmentations are shown in Figure 8. By learning to combine multiple cues that have support on larger, well-defined regions, we are able to successfully segment difficult images even when the boundary maps are far from ideal.

Discussion and Conclusions

We have presented a method for learning agglomerative segmentation. By performing agglomeration while comparing with a ground truth, we learn to merge segments at all scales of agglomeration. And, by guiding the agglomeration with the previous best policy, we guarantee that the examples we learn match those that will be encountered during a test agglomeration. Indeed, the difference in behavior between agglomerative learning and flat learning is immediately apparent and striking when watching the agglomerations occur side by side (see Video S4).

LASH [6] is a similar approach to ours that has nonetheless important conceptual differences. We use our gold standard segmentation to guide agglomeration during learning – preventing false merges – while they follow their current policy to completion, and use the sign of the change in Rand index as the learning label. A case can be made for either approach: in our case, we can train merges and non-merges from correct segments of arbitrary size, while LASH might diverge from the correct segmentation early on and then essentially train on noisy segments. We have anecdotally observed this advantage in play when we successfully used training data from a 250^3 voxel volume to segment a 500^3 voxel test volume. On the other hand, our own classifier might not get suitable training data for the times it diverges from a correct segmentation. Mixed training datasets from both strategies could turn out to be the best approach, and we will explore this possibility in future work.

Another difference is that Jain *et al.* only keep the training data from the last training epoch, while we concatenate the data from all epochs. In our experiments, we saw a significant improvement, relative to LASH, in segmentation accuracy in natural image data

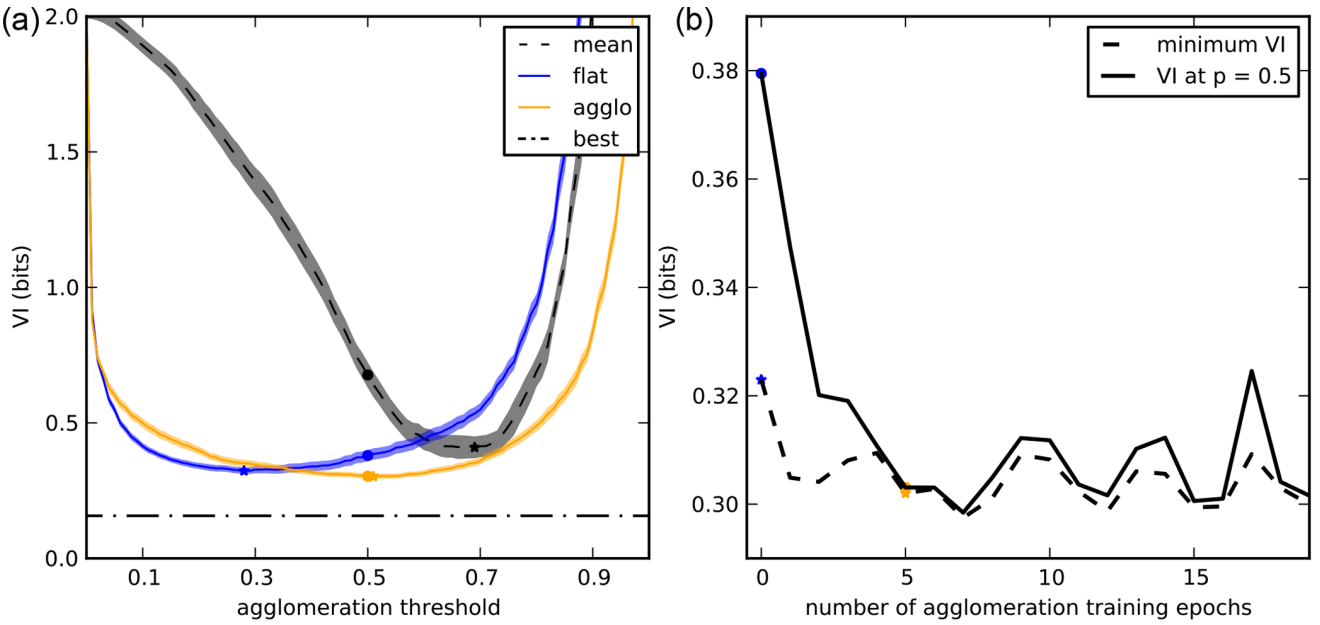

Figure 6. Agglomerative learning improves merge probability estimates during agglomeration. (Flat learning is equivalent to 0 agglomerative training epochs.) (a) VI as a function of threshold for mean, flat learning, and agglomerative learning (5 epochs). Stars indicate minimum VI, circles indicate VI at $p=0.5$. (b) VI as a function of the number of training epochs. The improvement in minimum VI afforded by agglomerative learning is minor (though significant), but the improvement at $p=0.5$ is much greater, and the minimum VI and VI at $p=0.5$ are very close for 4 or more epochs.
doi:10.1371/journal.pone.0071715.g006

(Figure S2). In EM data, the improvement was still present but only at higher undersegmentation values (over-merging), with LASH displaying a smaller advantage earlier in the agglomeration (Figure S3).

Recent work also attempts to use machine learning to classify on a merge hierarchy starting from watershed superpixels [17]. Liu *et al.*'s method cleverly chooses the right watershed threshold locally by learning directly on the merge tree nodes. However, the algorithm uses a single hierarchy of watershed superpixels obtained with different merge thresholds. This means that errors in the original hierarchy cannot be corrected by the machine learning approach, and watershed thresholding has been previously shown to give poor segmentation results [6]. Our method, in contrast, updates the merge hierarchy after each training epoch, potentially rectifying any prior errors. Liu *et al.*'s novel use of merge potentials to dynamically find the optimal threshold in each branch of the hierarchy, however, could be useful in the case of GALA.

Bjoern Andres, Fred Hamprecht and colleagues have devoted much effort to the use of graphical models to perform a one-shot agglomeration of supervoxels [5,35–37]. Although they only learn region merge probabilities at the base level of supervoxels, their use of conditional random fields (CRFs) to find the most consistent merge configuration is an advantage that our greedy, hierarchical approach lacks. On the other hand, their approach has two distinct disadvantages, in scalability and proofreadability.

First, the theoretical scalability of a global optimization is limited, which could become a problem as volumes exceed the teravoxel range. In contrast, GALA and other hierarchical methods could theoretically be implemented in a Pregel-like massively parallel graph framework [38], allowing the segmentation of extremely large volumes in time proportional to the number of supervoxels.

Second, despite the significant progress of the last decade, the accuracy of all currently available segmentation methods is orders of magnitude too small for their output to be used directly without human proofreading [2,39]. GALA operates locally, which makes

Table 1. Evaluation on BSDS500. Higher is better for all measures except VI, for which lower is better.

Algorithm	Covering			RI		VI		F-measure		
	ODS	OIS	Best	ODS	OIS	ODS	OIS	ODS	OIS	AP
human	0.72	0.72	–	0.88	0.88	1.17	1.17	0.80	0.80	–
agglo	**0.612**	**0.669**	**0.767**	**0.836**	**0.862**	**1.56**	**1.36**	**0.728**	**0.760**	**0.777**
flat	0.608	0.658	0.753	0.830	0.859	1.63	1.42	0.726	**0.760**	**0.776**
oriented mean [4]	0.584	0.643	0.741	0.824	0.854	1.71	1.49	0.725	0.759	0.758
mean	0.540	0.597	0.694	0.791	0.834	1.80	1.63	0.643	0.666	0.689

ODS uses the optimal scale for the entire dataset while OIS uses the optimal scale for each image.
doi:10.1371/journal.pone.0071715.t001

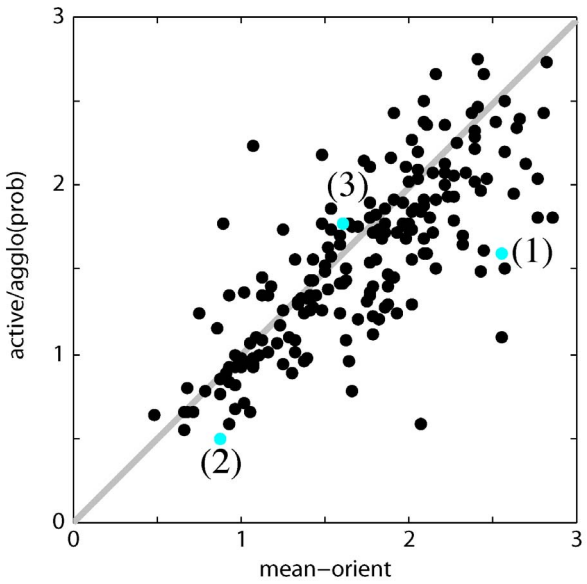

Figure 7. Comparison of oriented mean and actively learned agglomeration. as measured by VI at the optimal dataset scale (ODS). Each point represents one image. Numbered and colored points correspond to the example images in Figure 8.
doi:10.1371/journal.pone.0071715.g007

proofreading possible because manually adding a cut or merge only affects a few nearby predictions. Furthermore, proofreading can occur on any of the scales represented by the hierarchy. In contrast, because of the global optimization associated with the CRF approach, adding human-determined constraints to the supervoxel graph affects merge probabilities everywhere, resulting in expensive re-computation and the possibility that already-proofread areas need to be revisited.

A lot of the effort in connectomics focuses on the segmentation of anisotropic serial-section EM volumes [16,40,41]. Much like Liu *et al.*, Vazquez-Reina *et al.* use watershed segmentations of boundary probability maps at multiple thresholds on each different plane of the serial-section stack. They then use a CRF to link segments from consecutive sections at potentially different watershed thresholds. Funke *et al.*, in contrast, use a superpixel-less approach to obtain simultaneous segmentation within planes and linkage between planes [16]. Their within-plane segmentation optimizes a segmentation energy term with smoothness constraints, which eliminates many of the weaknesses of watersheds. Although the separation of segmentation and linkage between sections is not necessary in isotropic datasets, these approaches could inspire extensions of GALA specifically aimed at anisotropic segmentation.

The feature space for agglomeration is also worthy of additional exploration. For EM data, we included pixel probabilities of boundary, cytoplasm, mitochondria, and glia. Classifier predictions for synapses and vesicles might give further improvements [42]. Additionally, we found that most errors in our EM data are "pinch" errors, in which a neuronal process is split at a very thin channel. In these cases, features based on sums over voxels tend to be weakly predictive, because the number of voxels between the two segments is small. We are therefore actively exploring features based on segment shape and geometry, which have indeed been very useful in the work of Andres *et al.* discussed above [5,35–37]. Furthermore, we note that community-standard implementation of features will aid in the comparison of different learning and

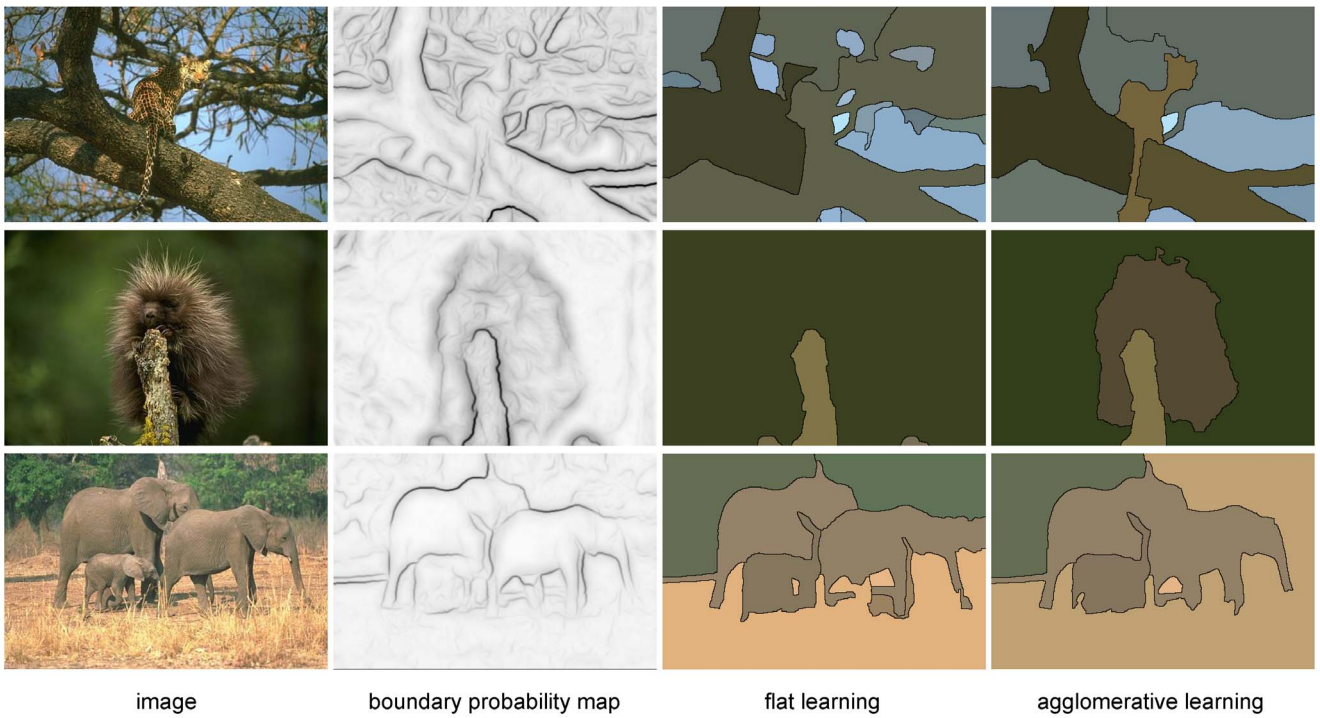

Figure 8. Example segmentations on natural images. *Top row*: Despite having a very noisy boundary map, using additional cues allows us to segment the objects successfully. *Middle row*: Although there are many weak edges, region-based texture information helps give a correct segmentation. *Bottom row*: A failure case, where the similar texture of elephants causes them to be merged even though a faint boundary exists between them. For all rows, the VI ODS threshold was used. The rows correspond top to bottom to the points identified in Figure 7.
doi:10.1371/journal.pone.0071715.g008

agglomeration algorithms, which are at present difficult to evaluate because they are conflated with the feature computation. A direct comparison of the segmentation performance of CRFs and agglomerative methods, disentangled from feature maps, would serve to advance the field.

A weakness of our method is its requirement for a full gold standard segmentation for training. This data might not be easily obtained, and indeed this has been a bottleneck in moving the method "from benchside to bedside", so to speak. We are therefore in the process of modifying the method to a semi-supervised approach that would require far less training data to achieve similar performance.

Finally, the field of neuronal reconstruction will depend on segmentation algorithms that not only segment well, but point to the probable location of errors. Although it requires further improvements in speed, scalability, and usability, our method is a first step in that direction.

Data and Code Availability

The source code for the Gala Python library can be found at: https://github.com/janelia-flyem/gala.

The EM dataset presented here in this work can be found at:

https://s3.amazonaws.com/janelia-free-data/Janelia-Drome-larva-FIBSEM-segmentation-data.zip.

Supporting Information

Figure S1 Illustration of the VI breakdown plot. (A) The VI breakdown plot shows the conditional entropy of each segment against the segment size (as a proportion of image size). Since the VI is the sum of the products of these two numbers, the hyperbolae are isoclines of contribution to the final VI. It is instantly obvious that the VI is dominated by oversegmentation errors. (B)–(C) Looking at the points with the highest contribution, we can examine the kinds of errors we are making, to improve our algorithm in the future. In this case, segment 18 from the gold standard is split at a "pinch" in the neuronal process, and this is a typical error among the others examined. (It is necessary to look at several planes to determine that the two segments should be connected).
(PDF)

Figure S2 LASH vs GALA performance on natural image data. Note that this plot shows the performance of *our own implementation* of the LASH learning protocol, using our own features.
(TIF)

Figure S3 LASH vs GALA performance on our EM dataset. Note that this plot shows the performance of *our own implementation* of the LASH learning protocol, using our own features.
(TIF)

Video S1 Agglomeration by classifiers trained using GALA (left) or a flat learning protocol (right). It rapidly becomes obvious that the GALA-trained classifier can confidently merge regions of arbitrary size, while the flat-trained classifier hesitates to continue merging once it encounters moderately-sized regions.
(MP4)

Acknowledgments

We thank Bill Katz for critical reading of the manuscript, C. Shan Xu and Harald Hess for the generation of the image data, Mat Saunders for generation of the ground truth data, Shaul Druckmann for help editing figures, and Viren Jain, Louis Scheffer, Steve Plaza, Phil Winston, Don Olbris and Nathan Clack for useful discussions.

Author Contributions

Conceived and designed the experiments: JNI DBC. Performed the experiments: JNI RK TP. Analyzed the data: JNI RK. Wrote the paper: JNI RK JS DBC. Helped with writing: TP JS DBC.

References

1. Anderson JR, Jones BW, Yang JH, Shaw MV, Watt CB, et al. (2009) A computational framework for ultrastructural mapping of neural circuitry. PLoS biology 7: e1000074.1.
2. Chklovskii DB, Vitaladevuni S, Scheffer LK (2010) Semi-automated reconstruction of neural circuits using electron microscopy. Current opinion in neurobiology 20: 667–675. 1, 5, 9, 12.
3. Ren Malik (2003) Learning a classification model for segmentation. In: ICCV 2003: 9th International Conference on Computer Vision. IEEE, 10–17 vol.1. 1, 2.
4. Arbeláez P, Maire M, Fowlkes C, Malik J (2010) Contour detection and hierarchical image segmentation. PAMI 33: 898–916. 1, 2, 5, 6, 7, 8, 10, 17, 19.
5. Andres B, Kappes JH, Beier T, Kothe U, Hamprecht FA (2011) Probabilistic image segmentation with closedness constraints. In: 2011 IEEE International Conference on Computer Vision (ICCV). IEEE, pp. 2611–2618. 2, 11, 12.
6. Jain V, Turaga S, Briggman K, Helmstaedter M, Denk W, et al. (2011) Learning to agglomerate superpixel hierarchies. Advances in Neural Information Processing Systems 24. 2, 8, 11.
7. Martin DR, Fowlkes CC, Malik J (2004) Learning to detect natural image boundaries using local brightness, color, and texture cues. IEEE Transactions on Pattern Analysis and Machine Intelligence 26: 530–549. 2, 6.
8. Dollar P, Tu Z, Belongie S (2006) Supervised learning of edges and object boundaries. In: Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on. IEEE, volume 2, pp. 1964–1971. 2.
9. Turaga S, Briggman K, Helmstaedter M, Denk W, Seung H (2009) Maximin affinity learning of image segmentation, Adv. Neural Info Proc Syst 22. 2, 6.
10. Jain V, Bollmann B, Richardson M, Berger D, Helmstaedter M, et al. (2010) Boundary learning by optimization with topological constraints. Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on: 2488–2495. 2, 6.
11. Jurrus E, Paiva ARC, Watanabe S, Anderson JR, Jones BW, et al. (2010) Detection of neuron membranes in electron microscopy images using a serial neural network architecture. Medical Image Analysis 14: 770–783. 2.
12. Vincent L, Soille P (1991) Watersheds in digital spaces: an efficient algorithm based on immersion simulations. PAMI 13: 583–598. 2.
13. Grundmann M, Kwatra V, Han M, Essa I (2010) Efficient hierarchical graph-based video segmentation. In: Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on. IEEE, pp. 2141–2148. 2.
14. Andres B, Köthe U, Helmstaedter M, Denk W, Hamprecht F (2008) Segmentation of SBFSEM Volume Data of Neural Tissue by Hierarchical Classification. Pattern Recognition 5096: 142–152. 2.
15. Cheng B, Liu G, Wang J, Huang Z, Yan S (2011) Multi-task low-rank affinity pursuit for image segmentation. ICCV. 2.
16. Funke J, Andres B, Hamprecht FA, Cardona A, Cook M (2012) Efficient automatic 3Dreconstruction of branching neurons from EM data. Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on: 1004–1011. 2,12.
17. Liu T, Jurrus E, Seyedhosseini M, Ellisman M, Tasdizen T (2012) Watershed merge tree classification for electron microscopy image segmentation. Pattern Recognition, ICPR 2012: 133–137. 2, 11.
18. Brendel W, Todorovic S (2010) Segmentation as maximumweight independent set. Neural Information Processing Systems 4. 5.
19. Varma M, Zisserman A (2005) A statistical approach to texture classification from single images. International Journal of Computer Vision 62: 61–81. 5.
20. Brendel W, Todorovic S (2010) Segmentation as maximum weight independent set. In: Neural Information Processing Systems. volume 4. 5.
21. Yushkevich PA, Piven J, Cody Hazlett H, Gimpel Smith R, Ho S, et al. (2006) Userguided 3D active contour segmentation of anatomical structures: Significantly improved efficiency and reliability. Neuroimage 31: 1116–1128.5.
22. Sommer C, Straehle C, Koethe U, Hamprecht FA (2011) ilastik: Interactive learning and segmentation toolkit. In: 8th IEEE International Symposium on Biomedical Imaging (ISBI 2011). 5.
23. Rand WM (1971) Objective criteria for the evaluation of clustering methods. Journal of the American Statistical Association 66: 846–850. 6.

24. Vinh NX, Epps J, Bailey J (2010) Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance. The Journal of Machine Learning Research 9999: 28372854. 6.
25. Meila M (2003) Comparing clusterings. In: Proceedings of the Sixteenth Annual Conference on Computational Learning Theory (COLT). Springer. 6, 7.
26. Knott G, Marchman H, Wall D, Lich B (2008) Serial section scanning electron microscopy of adult brain tissue using focused ion beam milling. The Journal of neuroscience: the official journal of the Society for Neuroscience 28: 2959–2964. 9.
27. Denk W, Horstmann H (2004) Serial block-face scanning electron microscopy to reconstruct three-dimensional tissue nanostructure. PLoS biology 2: e329. 9.
28. Hayworth KJ, Kasthuri N, Schalek R (2006) Automating the collection of ultrathin serial sections for large volume TEM reconstructions. Microscopy and Microanalysis 12: 86–87. 9.
29. Harris KM, Perry E, Bourne J, Feinberg M, Ostroff L, et al. (2006) Uniform serial sectioning for transmission electron microscopy. The Journal of neuroscience: the official journal of the Society for Neuroscience 26: 12101–12103. 9.
30. Lichtman JW, Denk W (2011) The big and the small: challenges of imaging the brain's circuits. Science (New York, NY) 334: 618–623. 9.
31. Takemura SY, Lu Z, Meinertzhagen IA (2008) Synaptic circuits of the Drosophila optic lobe: the input terminals to the medulla. The Journal of comparative neurology 509: 493–513. 9.
32. Laissue PP, Reiter C, Hiesinger PR, Halter S, Fischbach KF, et al. (1999) Threedimensional reconstruction of the antennal lobe in Drosophila melanogaster. The Journal of comparative neurology 405: 543–552. 9.
33. Plaza SM, Scheffer LK, Saunders M (2012) Minimizing manual image segmentation turnaround time for neuronal reconstruction by embracing uncertainty. PLoS ONE: In press. 9.
34. Ciresan D, Giusti A, Gambardella LM, Schmidhuber J (2012) Deep neural networks segment neuronal membranes in electron microscopy images. In: Proceedings of Neural Information Processing Systems. pp. 2852–2860. 9.
35. Andres B, Kroeger T, Briggman KL, Denk W, Korogod N, et al. (2012) Globally optimal closed-surface segmentation for connectomics. ECCV: 778–791. 11, 12.
36. Andres B, Koethe U, Kroeger T, Helmstaedter M, Briggman KL, et al. (2012) 3D segmentation of SBFSEM images of neuropil by a graphical model over supervoxel boundaries. Medical Image Analysis 16: 796–805. 11, 12.
37. Andres B, Köthe U, Helmstaedter M, Denk W, Hamprecht F (2008) Segmentation of SBFSEM volume data of neural tissue by hierarchical classification. Pattern recognition: 142–152. 11, 12.
38. Malewicz G, Austern MH, Bik AJC, Dehnert JC, Horn I, et al. (2010) Pregel: A System for Large-Scale Graph Processing. In: SIGMOD. New York, New York, USA: ACM Press, p. 135. 11.
39. Jurrus E, Watanabe S, Giuly RJ, Paiva ARC, Ellisman MH, et al. (2013) Semi-automated neuron boundary detection and nonbranching process segmentation in electron microscopy images. Neuroinformatics 11: 5–29. 12.
40. Vazquez-Reina A, Gelbart M, Huang D, Lichtman J, Miller E, et al. (2011) Segmentation fusion for connectomics. ICCV. 12.
41. Laptev D, Vezhnevets A, Dwivedi S, Buhmann JM (2012) Anisotropic ssTEM Image Segmentation Using Dense Correspondence across Sections. Berlin, Heidelberg: MICCAI. pp. 323–330. 12.
42. Kreshuk A, Straehle CN, Sommer C, Koethe U, Cantoni M, et al. (2011) Automated Detection and Segmentation of Synaptic Contacts in Nearly Isotropic Serial Electron Microscopy Images. PLoS ONE 6: e24899. 12.