Boundary Learning by Optimization with Topological Constraints

Viren Jain[4,*], Benjamin Bollmann[4,*], Mark Richardson[4], Daniel R. Berger[4,5], Moritz N. Helmstaedter[3], Kevin L. Briggman[3], Winfried Denk[3], Jared B. Bowden[2], John M. Mendenhall[2], Wickliffe C. Abraham[6], Kristen M. Harris[2,†], Narayanan Kasthuri[1], Ken J. Hayworth[1], Richard Schalek[1], Juan Carlos Tapia[1], Jeff W. Lichtman[1] and H. Sebastian Seung[4,5]

[1]Department of Molecular and Cellular Biology, Center for Brain Science, Harvard University, Cambridge, MA, USA

[2]Center for Learning and Memory, Department of Neurobiology, University of Texas at Austin, TX, USA

[3]Department of Biomedical Optics, Max Planck Institute for Medical Research, Heidelberg, Germany

[4]Brain & Cognitive Sciences, Massachusetts Institute of Technology, Cambridge, MA, USA

[5]Howard Hughes Medical Institute, Cambridge, MA, USA

[6]Dept. of Psychology and Brain Health and Repair Research Center, Univ. of Otago, Dunedin, New Zealand

Abstract

Recent studies have shown that machine learning can improve the accuracy of detecting object boundaries in images. In the standard approach, a boundary detector is trained by minimizing its pixel-level disagreement with human boundary tracings. This naive metric is problematic because it is overly sensitive to boundary locations. This problem is solved by metrics provided with the Berkeley Segmentation Dataset, but these can be insensitive to topological differences, such as gaps in boundaries. Furthermore, the Berkeley metrics have not been useful as cost functions for supervised learning. Using concepts from digital topology, we propose a new metric called the warping error that tolerates disagreements over boundary location, penalizes topological disagreements, and can be used directly as a cost function for learning boundary detection, in a method that we call Boundary Learning by Optimization with Topological Constraints (BLOTC). We trained boundary detectors on electron microscopic images of neurons, using both BLOTC and standard training. BLOTC produced substantially better performance on a 1.2 million pixel test set, as measured by both the warping error and the Rand index evaluated on segmentations generated from the boundary labelings. We also find our approach yields significantly better segmentation performance than either gPb-*OWT-UCM or multiscale normalized cut, as well as Boosted Edge Learning trained directly on our data.*

The accurate detection of object boundaries in images has been a long-standing challenge for computer vision. Rigorous research on this problem requires some method for quantifying machine performance. One method is to compare machine-generated boundaries with human tracings of boundaries on a set of images. The ideal metric for machine-human disagreement should:

1. tolerate minor differences in boundary location,
2. penalize topological disagreements,
3. and serve as a convenient cost function for supervised learning.

The first two properties are critical if the goal of boundary detection is to generate image segmentations, i.e., divide an image into regions corresponding to objects. A topological error, like a gap in a boundary, can cause two regions to become incorrectly merged, a drastic segmentation error. On the other hand, minor differences in boundary location have little effect on the shapes of regions in a segmentation. The third property is important because the supervised learning approach has become increasingly accepted as a means of achieving more accurate boundary detection (see the top algorithms in the Berkeley Boundary Detection Benchmark).

If the boundary detection algorithm is designed by hand, the performance metric can be created as an afterthought. But supervised learning starts from the metric, minimizing it to find a boundary detection algorithm. Therefore the proper choice of a metric is even more crucial in supervised learning than in conventional hand-designed approaches. If the ultimate goal of learning is segmentation, it is especially important to insist on a metric with properties (1) and (2).

Previous metrics have generally lacked one or more of the above properties. For example, the naive metric is the *pixel error*, the number of image pixels on which machine and human boundary labelings disagree. This metric unduly penalizes minor disagreements over boundary location. These ought to be considered inconsequential, since they are ubiquitous when comparing the boundary labelings provided by different humans.

*These authors made comparable contributions to this paper.

†Supported by NIH/NINDS R37 NS021184 and NIH/NIBIB EB002170.

More sophisticated metrics are provided with the Berkeley Segmentation Dataset (BSDS), a collection of natural images along with human boundary tracings [1]. The Berkeley metrics are based on a correspondence between machine and human boundary pixels obtained by solving a minimum cost bipartite assignment problem. Since matches are allowed between pixels that are within some distance cutoff, small disagreements over boundary locations are not penalized. But the Berkeley metrics have a major deficiency: they may fail to penalize a gap in the boundary between two regions, and hence do not satisfy property (2) above.

The pixel error and the Berkeley metrics also differ in their convenience for supervised learning. Because pixel error is so simple, it can easily serve as a cost function to be minimized. In BSDS research, pixel error has been used to train simple classifiers that combine hand-designed features [1, 2], as well as boosted classifiers that combine large numbers of simple features [3]. Pixel error has also been used to train convolutional networks to perform boundary detection directly on raw images from biological microscopy, with no use of hand-designed features [4].

In contrast, no one has succeeded in using the Berkeley metrics to *learn* boundary detection, probably because they are so complex. They have only been applied to *evaluate* the performance of boundary detection algorithms after learning. To compensate during supervised learning for the pixel error's undue emphasis on boundary locations, researchers have blurred the human boundaries [2] or averaged multiple human labelings to yield an estimate of boundary probabilities [3]. Neither of these simple fixes deals satisfactorily with both issues (1) and (2) in the list above.

In summary, the pixel error is a convenient cost function for supervised learning, but is too sensitive to boundary locations. The Berkeley metrics tolerate disagreements in boundary location, but are too insensitive to topological disagreements, and have not been convenient for use in supervised learning.

In this paper, we use concepts from digital topology to define the *warping error*, a new metric for boundary detection that possesses all three properties in the list above. Learning by minimizing warping error is called Boundary Learning by Optimization with Topological Constraints (BLOTC). We apply BLOTC to train a boundary detector on human tracings of electron microscopic (EM) images of neural tissue. The warping error of the BLOTC detector is significantly better than that of a "standard" detector, one trained by minimizing pixel error. Image segmentations can be generated by finding connected components of the thresholded boundary detector output. The BLOTC segmentation is quantifiably superior to the standard segmentation, confirming our intuition that properties (1) and (2) are critical for a boundary detection metric if the ultimate goal is segmentation.

We trained Boosted Edge Learning [3] on our dataset and show that segmentations generated from this approach are significantly inferior to those produced by either pixel error or BLOTC training of a convolutional network. We also show that *gPb*-OWT-UCM [5] and multiscale normalized cuts [6] produce far worse segmentations. This is not altogether surprising since these algorithms were intended for natural images rather than EM images, but it nevertheless provides a useful comparison.

The Rand error has also been proposed as a metric of segmentation performance that satisfies the desired properties listed previously [7, 8]. It will be defined below, and compared with the warping error. We will use both error metrics to quantify performance on the test set, although only the warping error will be used for training. The relation of BLOTC to a recently proposed method of supervised learning based on minimization of Rand error [9] will be explained later.

1. Existing metrics

Let $\mathcal{B}$ be a space of binary or black-and-white images of a given size, and $\mathcal{A}$ a space of analog or gray scale images of the same size. A binary image $L \in \mathcal{B}$ is termed a *boundary labeling* of an image $I \in \mathcal{A}$, if a "0" pixel in L indicates the presence of a boundary between two objects in the image, and a "1" pixel indicates the presence of an object. This is opposite the usual convention, but it will be convenient later when we use digital topology, as in that field the "1" pixels are the *foreground* objects of the image. The disagreement between two boundary labelings can be quantified using the pixel error and the Berkeley metrics. The boundary labelings can also be converted into segmentations, which are compared using the Rand error. All of these metrics are explained below (see also Figure 1).

1.1. Pixel error

Let l_i denote the value of the boundary labeling L at image location i. The pixel error of L with respect to another binary labeling L^* is the number of pixel locations at which the two labelings disagree. This can also be written as the squared Euclidean distance $\|L - L^*\|^2$, which is equivalent to the Hamming distance since the labels are binary-valued. The pixel error is appealing because of its simplicity, but suffers from a serious defect. It is *overly* sensitive to minor displacements in the location of a boundary that are ubiquitous even when comparing one human boundary labeling to another. These disagreements cause no qualitative differences in the interpretation of the image, but can lead to large quantitative differences in pixel error. Figure 1 illustrates that two boundary labelings with identical pixel error relative to the true boundary labeling may yield segmentations of very different quality.

1.2. Berkeley metrics

To remove the sensitivity to minor differences in boundary location, Martin et al. introduced another set of metrics, which are provided with the Berkeley Segmentation

Dataset [1]. They computed a correspondence between a machine boundary labeling and a human boundary labeling by solving a minimum cost bipartite assignment problem. Boundary pixels in one labeling can be matched to boundary pixels in the other, provided that the matching pixels are within some distance cutoff of each other. Metrics based on this correspondence tolerate small differences in boundary localization, but still have limitations. First, they may be insensitive to gaps in boundaries, which can lead to mergers between two regions in a segmentation [8]. Second, it is not clear how to efficiently optimize them for supervised learning.

1.3. Rand error

The Rand index is a well-known measure of the similarity between two data clusterings [10]. Recently, the Rand index has been proposed as a measure of segmentation performance, since a segmentation can be regarded as a clustering of pixels [7]. The Rand index is defined as a measure of agreement. Here we instead define the closely related Rand error, which is a measure of disagreement.

More formally, define a segmentation as an integer-valued labeling of an image. Each object in a segmentation consists of a set of pixels sharing a common label. The Rand error is the frequency with which the two segmentations disagree over whether a pair of pixels belongs to same or different objects:

$$R(S,T) = \binom{N}{2}^{-1} \sum_{i \neq j} \left| \delta(S_i, S_j) - \delta(T_i, T_j) \right|,$$

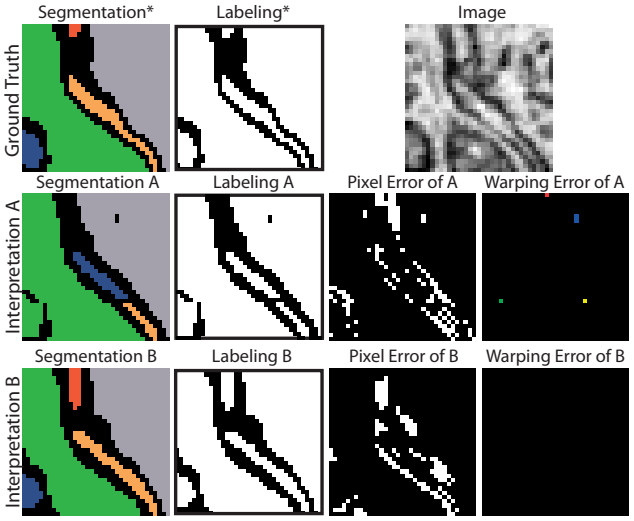
where the sum is over all pairs of distinct pixels. The function $\delta(S_i, S_j) = 1$ if pixels i and j belong to the same object and 0 if they belong to different objects. The Rand error equals 0 when the two segmentations agree completely, and equals 1 when they disagree completely. These numerical values are transposed for the more commonly used Rand index, which is defined as $1 - R(S,T)$.

The Rand error evaluates whether the overall grouping of pixels into separate objects is correct. Small differences in the location of object boundaries will increase the Rand error slightly, but the merging of two objects or the splitting of an object will tend to increase the Rand error by a large amount. but will usually be less numerically significant than, for example, the erroneous merging of two regions.

2. Digital topology and the warping error

This section introduces a novel metric for comparing boundary labelings, based on concepts from the field of digital topology. If L^* can be transformed into L by a sequence of pixel flips that each

1. preserve a set of desired topological properties
2. occur only at locations within a mask M,



	Error of A	Error of B	$\frac{ErrorB}{ErrorA}$
Pixel Error	$115/1225$	$115/1225$	1
Berkeley: 1-(F measure)	0.018	0.066	3.67
Rand error	0.08	0.06	.75
Warping Error	$5/1225$	$0/1225$	0

Figure 1. Comparison of different metrics on a 35×35 pixel image from the training set used in Section 5. Within each segmentation frame, pixels with the same color correspond to a single object. Interpretation A has both topological and geometric differences with the ground truth, whereas Interpretation B has geometric discrepancies but no topological errors. The Rand error and warping error show a large relative difference between interpretations B and A, while the pixel error does not. The Berkeley F-measure *favors* the topologically incorrect interpretation. Warping error of interpretation A shows four topological errors: red pixel denotes object deletion, green pixel denotes an object merger, yellow pixel denotes an object split, and blue pixels denote creation of a hole. The Berkeley F-Measure was converted to an error measure by subtracting from 1, the value of perfect agreement for that measure. For evaluation of the Berkeley F-measure, a boundary labeling was computed by completely thinning a binary flip of the in/out map (in accordance with procedures defined in the benchmark code provided with [1]).

then we will say L is a *warping of* L^*, or $L \lhd L^*$. The first condition constrains L and L^* to be topologically equivalent. The second condition can be used to constrain L to be geometrically similar to L^*. Both conditions will be explained in more detail below.

Now consider the pixel error of T relative to warpings of L^*. The *warping error* between some candidate labeling T and a reference labeling L^* is the Hamming distance (or equivalently squared Euclidean distance) between L^* and the "best warping" of L^* onto T :

$$D(T||L^*) = \min_{L \lhd L^*} \|T - L\|^2 \qquad (1)$$

In Figure 1 we illustrate the warping error.

2.1. Topological constraints

To impose topological constraints on the warping, we use concepts from digital topology, a field that extends the concepts of continuous-space topology to digital images. One of the most fundamental principles of this field is that complementary definitions of adjacency must be used for foreground ("1") and background ("0") pixels, so that a digital analog of the Jordan Curve Theorem holds [11]. We use the 4-adjacency for foreground and the 8-adjacency for background, and calculate connected components based on these adjacencies.

A major practical goal of digital topology is to identify methods of altering the *geometry* of objects within a digital image without altering any topological properties of the image. A *simple point* is defined as a location in a binary image at which the pixel can be flipped to its complementary value without changing any topological properties of the image. These properties include, for example, the number of κ-connected components of the foreground and the number of $\bar{\kappa}$-connected components of the background.

Bertrand has given conditions for a simple point based on his notion of *topological number*. Let $(\kappa, \bar{\kappa})$ be a complementary pair of adjacencies. The topological number $T_\kappa(\mathbf{p}, L)$ is the number of foreground connected components in the neighborhood of p (i.e., the 3×3 patch of L centered at p) under κ-adjacency. The topological number $T_{\overline{\kappa}}(\mathbf{p}, \bar{L})$ is the number of background connected components in the neighborhood of p under $\overline{\kappa}$-adjacency. A point p is defined as κ-simple if and only if $T_\kappa(\mathbf{p}, L) = T_{\overline{\kappa}}(\mathbf{p}, \overline{L}) = 1$. [12]. Note that the computation of the topological numbers is local, based only on the neighborhood of the point. Therefore whether a point is simple can be checked rapidly.

Although flipping a single simple point is guaranteed to preserve topology, it is not true that simultaneously flipping an arbitrary set of *multiple* simple points will preserve topology. Hence, many algorithms that deform digital images by altering simple points instead perform a sequence of flips, where any particular flip is made at a point that is simple relative to the current state of the image. Since all flips preserve topology, such a sequence of flips is a topology-preserving deformation of the original image (sometimes called a homotopic deformation). The converse has also been proven: two images that are topologically equivalent in the sense of sharing isomorphic adjacency trees can always be transformed into each other by a sequence of changes in the values of simple pixels [13].

In short, by definition flipping a "simple" point is a topology-preserving operation and flipping a "non-simple" point is not topology-preserving. Non-simple points can be classified by the nature of the topological change they would cause by being flipped (see the supplementary material for mathematical details of this classification, and Figure 1 for an example of this classification). The possible topological changes are splitting, merging, hole addition/deletion, or object addition/deletion. We may wish to allow some of these types of changes, and therefore flipping of some types of non-simple pixels.

We will write simple(L) to denote the set of simple points of L, and topo(L) to denote the set of simple and non-simple points of L that preserve some desired set of topological properties.

Algorithm 1 Descent algorithm for warping a binary image L^* to an analog image T, under geometric constraints set by the binary image M.

warp($L^* \in \mathcal{B}$,$T \in \mathcal{A}$,$M \in \mathcal{B}$)
$L := L^*$
do
  $S :=$ simple$(L) \cap M$
  $i := \operatorname{argmax}_{j \in S} |t_j - l_j|$, breaking ties randomly
  if $|t_i - l_i| > 0.5$
    $l_i := 1 - l_i$
  else
    return L
  end

2.2. Geometric constraints

There are many ways to define the mask, depending on the exact nature of the desired geometric constraints. In our implementation below, we choose M to be the set of all pixels within Euclidean distance 5 from the background of L^*. This allows the foreground of L to expand arbitrarily, as long as topology is preserved. But the foreground can shrink by only a limited amount. Note that basing M on L^* makes warping an asymmetric relation.

2.3. Descent algorithm for warping

We do not know of an efficient algorithm for finding the global minimum in Eq. (1), and indeed this is likely to be an NP-hard problem. However, there is a very simple descent algorithm for finding local minima. During warping, we are allowed to flip simple points of L that lie inside the mask M, i.e., points in the set simple$(L) \cap M$. Flipping any such pixel j of L satisfying $|t_j - l_j| > 0.5$ produces a new warping with smaller error. The descent algorithm greedily picks the pixel for which this error reduction is the largest, breaking ties randomly.

Since $\|T - L\|^2$ is decreasing, the algorithm is guaranteed to converge to a local minimum of the warping error. How problematic is the lack of an efficient algorithm for finding a global minimum? We do not think that it is a problem in practice. The warpings found by our descent algorithm look reasonable to the eye. In spite of the random choice of flipped pixels, the results are highly reproducible in practice. In the dataset studied later on, the outcomes of dozens of runs of warping on a 65,536 pixel image contained less than ten pixel differences, or less than 0.01% of the value of the warping error. We note that the Berkeley metrics are not deterministic either, due to the use of random nodes and edges in the graph for the correspondence

problem. Furthermore, a linear time approximate algorithm is used to solve the correspondence problem, so that the global minimum is not found in this case either.

We also note that warping in two and higher dimensions is fundamentally more difficult than in one dimension, for which there are efficient algorithms like dynamic time warping or Viterbi alignment based on dynamic programming. Nevertheless, it may be possible to improve upon our descent algorithm. For example, previous work on homotopic thinning algorithms may suggest certain strategies for finding better or more consistent minima, such as using the distance transform to determine flipping order [14].

2.4. Warping error versus Rand error

At first glance, it may seem that the warping error measures only boundary detection performance. But we would like to argue that it is also a good measure of segmentation performance. This is because digital topology tells us how any single pixel affects the global topology of an image. Let L be a best warping of L^* onto T. Any pixel in the symmetric difference set $L \Delta T = L \backslash T \cup T \backslash L$ represents a *topological error* in T because flipping its value in L (which is topologically equivalent to the ground truth L^*) would cause a topological change. Thus the warping error is an upper bound on the number of topologically-relevant boundary labeling errors in T (if a geometric mask is used, then the warping error also includes labeling errors of a geometric nature). Therefore, if segmentations are generated from T and L^* by finding their connected components, then the warping error should be a reasonable measure of the topological disagreements between the segmentations.

The Rand error is becoming more popular as a metric of segmentation performance [7], and can be used as a cost function for gradient learning of image segmentation [9]. The Rand error can be used to compare segmentations in which regions are noncontiguous clusters of pixels. Such segmentations are not equivalent to boundary labelings, so the warping error cannot be applied. In many applications, such as the one studied below, this is not a significant limitation.

The warping error can be distinguished from the Rand error in other respects. The warping error can penalize all kinds of topological errors, including the presence of holes and handles, but the Rand error penalizes only connectivity errors. In certain medical imaging situations, control of such aspects of topology is especially important [15]. The Rand error mildly penalizes shifts in boundary location, while the warping error ignores them altogether. The warping error weights a topological error by the number of pixels involved in the error itself, while the Rand error weights a split or merger by the number of pixels in the objects associated with the errors.

Algorithm 2 Stochastic online gradient learning. The learning rate parameter η is small and positive.

gradient($I \in \mathcal{A}$,$L \in \mathcal{B}$,$\vec{w}$,k)
for iter = 1 to k
  i =random location in random image
  $\vec{w} := \vec{w} - \eta \nabla_{\vec{w}} d(f_i(\vec{w}), l_i)$
end
return $\vec{w}$

3. Supervised learning

3.1. Learning with pixel error

A popular way of applying supervised learning to boundary detection is to find an image patch classifier, a function that maps an image patch to an estimate of the probability that the central pixel is a boundary. The classifier output will be called the boundary map and written as $F_I(\vec{w})$, where I is the image, and $\vec{w}$ specifies the adjustable parameters of the classifier. The analog values in the boundary map can be thresholded at θ to produce a binary boundary labeling $H(F_I(\vec{w}) - \theta)$, where the image-valued quantity $H(M)$ represents the Heaviside step function applied to each pixel location in a map M.

Supervised learning could be formulated as the minimization of the pixel error $\|H(F_I(\vec{w}) - \theta) - L^*\|^2$ with respect to the classifier parameters $\vec{w}$, where L^* is a human boundary labeling of the same image. However, it is often easier to optimize a smooth cost function that depends on the real-valued output of the classifier. Minimizing the squared error $\|F_I(\vec{w}) - L^*\|^2$ serves as an approximation to optimizing the binary classification error. Since the squared error cost function depends smoothly on the analog output of the classifier, gradient descent can be applied to find a local minimum. A "batch" implementation computes the gradient of the cost function for the whole image or set of images. An "online" implementation computes the gradient of the cost function for a single pixel. Since the pixel is chosen at random, the average of the online gradient is equal to the batch gradient, which means that online learning is a form of stochastic gradient descent.

3.2. Learning with warping error

The warping error is superior to the pixel error, if the goal of boundary detection is segmentation. Therefore it would make sense to base supervised learning on the warping error, formulating it as the optimization of $D(H(F_I(\vec{w}) - \theta)||L^*)$ with respect to $\vec{w}$, which is the dual optimization:

$$\min_{\vec{w}} \min_{L \lhd L^*} \|H(F_I(\vec{w}) - \theta) - L\|^2 .$$

We call this method Boundary Learning by Optimization with Topological Constraints, or BLOTC. Note that standard training is the case where no warping of L^* is allowed, i.e., the geometric constraint becomes completely tight. In order to make this cost function easier to optimize, we again

Algorithm 3 Boundary Learning by Optimization with Topological Constraints (BLOTC)

blotc($I \in \mathcal{A}$,$L^* \in \mathcal{B}$,$M \in \mathcal{B}$,k_1,k_2)
$L := L^*$
$\vec{w}$:=random initialization
$\vec{w}$:=**gradient**(L,$\vec{w}$,k_1)
repeat
  L :=**warp**(L,$F_I(\vec{w})$,M)
  $\vec{w}$:=**gradient**(I,L,$\vec{w}$,k_2)
until convergence
return $\vec{w}$

use a smooth approximation of the binary error. One possibility is to use the squared error defined above:

$$\min_{\vec{w}} \min_{L \lhd L^*} ||F_I(\vec{w}) - L||^2,$$

For the experiments described in Section 4, we used the closely related square-square loss (defined in the supplementary material). BLOTC is carried out as described in Algorithm 3, by alternating between gradient descent for $\vec{w}$ (Algorithm 2) and descent for L (Algorithm 1).

4. Segmentation of Electron Microscopic Images of Brain Tissue

Recent advances in electron microscopy (EM) have enabled the automated collection of nanoscale images of brain tissue [16, 17]. The resulting image datasets have renewed interest in automated computer algorithms for analyzing EM images [4, 18, 19]. From the neuroscience perspective, development of such algorithms is important for the goal of finding connectomes, complete connectivity maps of a brain or piece of brain [20, 21, 22]. To find connectomes, two image analysis problems must be solved. First, each synapse must be identified in the images. Second, the "wires" of the brain, its axons and dendrites, must be traced through the images. If both problems are solved, then it would be possible to trace the wires running from every synapse to the cell bodies of its parent neurons, thereby identifying all pairs of neurons connected by synapses.

Serial sections of rat hippocampus were obtained using the Automatic Tape Collecting Lathe Ultra-Microtome (ATLUM) [16]. Sections were 29.8 nm thick, and images of each section were taken at a resolution of roughly 4 nm/pixel using a JEOL scanning electron microscope. The images were downsampled to approximately 16nm/pixel for analysis. Our goal in this paper was to segment the images into regions corresponding to the cross sections of distinct neurons. This is a first step toward full 3d reconstruction of neuron shapes, but in this paper we only address 2d computations. A level set approach has recently been pursued for segmentation of similar imagery [23].

The segmentation task is especially challenging because the neurons contain many intracellular organelles such as mitochondria, and this internal clutter can be distracting.

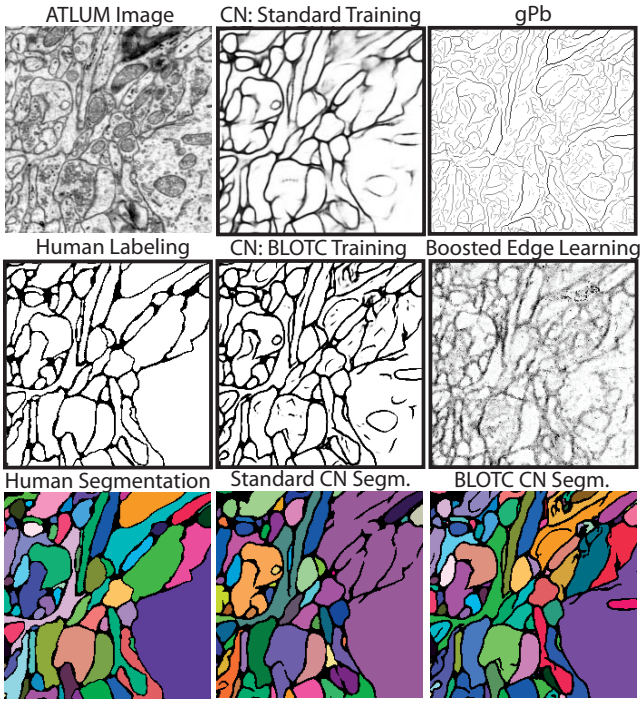


Figure 2. Visual comparison of output from a convolutional nework (CN) trained in the standard way and a convolutional network trained with BLOTC, on an image from the test set. Segmentations of CN output were generated using connected components at a threshold chosen optimally for each method based on quantifications in Figures 3 and 4. The BLOTC CN segmentation has a substantially smaller number of split and merger errors compared to the CN trained in the standard way. Boundary labelings from gPb [8] and Boosted Edge Learning (BEL) [3] are also displayed; additional comparisons are shown in the supplementarymaterial.

Two kinds of boundaries are visible in the images: external boundaries between neurites, and internal boundaries of the intracellular organelles. There is an obvious question concerning how the boundary detector should be trained. We definitely want the external boundaries of neurons to be detected. But do we want the internal boundaries to be detected?

It turns out that this decision can be left up to the computer, if BLOTC training is used. We implemented a version of BLOTC in which some topological changes were allowed in the warping described in Section 2.1. In addition to flipping simple pixels, the warping was allowed to flip certain types of non-simple pixels. For example, the warping was allowed to create holes within objects, thereby not penalizing the computer for detecting internal boundaries even if they were not traced by the human. Additional details are available in the supplementary material.

A 256×256 bounding box from a set of 100 aligned images was cropped from the dataset to form a $256 \times 256 \times 100$ image stack. Each image in the stack was manually traced by a human using the ITK-SNAP software package. All external boundaries of neurons were traced in each image,

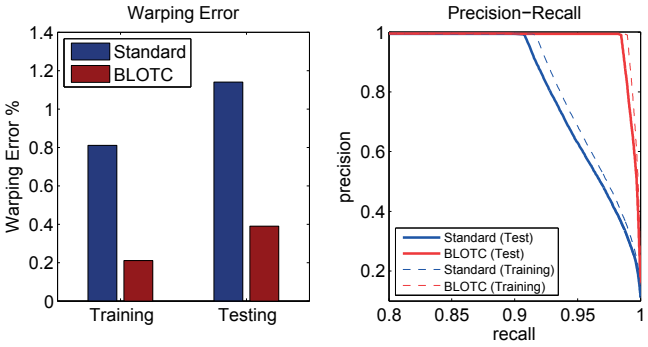


Figure 3. Comparison of standard and BLOTC learning using the warping error metric on a 5.2 megapixel training set and a 1.2 megapixel testing set. The left plot shows warping error when classifier output is thresholded at 0.5, demonstrating a large relative reduction in error from standard to BLOTC training. The right plot shows the precision-recall on outside voxel classification accuracy, which compensates for the class-imbalance. The warping error for all methods is provided in the supplementary material.

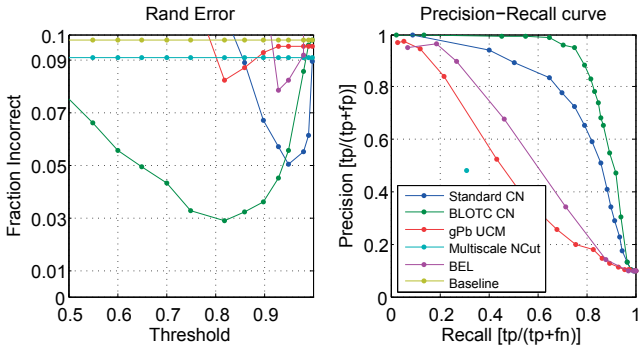


Figure 4. Generalization performance using the Rand error and precision-recall of correctly classified connected pairs of pixels on a 1.2 megapixel testing set. The relative reduction in Rand error between standard and BLOTC results is large (40%). The precision-recall curve compensates for the class imbalance in pixel connectivity (in the ground truth, most pixels are disconnected from one another). *gPb*-OWT-UCM corresponds to segmentations generated by an oriented water transform on *gPb* contour detector output, followed by conversion into a hierarchical region tree [8]. Multiscale Ncut corresponds to multiscale normalized cut [6]. This method requires as input the number of objects in an image; we provided the average number of objects in a training set image. Boosted Edge Learning (BEL) was learned using the same 5.2 megapixel training set as the convolutional networks, with a 30×30 field of view and identical classifier parameters to those used in [3], with code provided by the authors. Several methods for generating segmentations from BEL output were tested; a watershed approach worked best. Baseline corresponds to a segmentation in which all pixels are disconnected from one another.

but the internal boundaries were mostly neglected. Figure 2 shows an example of human tracing. From this dataset, 80 images were used as a 5.2 megapixel training set and 20 images were set aside as a 1.2 megapixel testing set.

For our image patch classifier, we used a convolutional network. It has already been shown that convolutional networks provide state-of-the-art performance at boundary detection in EM images [4], when trained by standard methods; in the supplementary material, we describe details of this classifier.. However, it is important to note that BLOTC is not limited to convolutional networks, but can be used to train any kind of classifier.

We trained two convolutional networks with identical architectures containing 6 hidden layers, 24 feature maps in each hidden layer, and full connectivity between feature maps in adjacent layers. Each individual filter was 5×5 pixels, but the multiple-layers yield an effective field of view of the classifier of 28×28 pixels. The standard network was trained for 1,000,000 updates using the traditional optimization with the labels fixed. The BLOTC network was initialized with the weights from a standard network after 500,000 gradient updates and then further trained for 500,000 additional updates using BLOTC optimization. Each network was trained in roughly 18 hours, using a layer-wise procedure that iteratively adds hidden layers to the network architecture. The training used a fast shared-memory GPU implementation that provides between a 50-100$\times$ increase in training speed as compared to CPU implementations.

The results of training are shown in Figure 2. Both BLOTC and standard networks do a good job of detecting boundaries between neurons, and ignore most intracellular structures such as vesicles. The BLOTC network strongly detects some mitochondrial boundaries that were not in the human tracing, presumably because it was not penalized for doing so. The standard network cannot manage to ignore mitochondrial boundaries, even though it was trained to ignore them. This is presumably because mitochondrial boundaries often resemble external boundaries, at least locally. The BLOTC network produces more binary output, as if it were more confident.

Figure 3 shows that the warping error of the BLOTC network is much lower than that of the standard network. The value of the error is low (1% or less), which is primarily a consequence of the fact that boundary pixels are relatively rare; thus it is helpful to view performance using the precision-recall curves shown in Fig. 3.

Figure 4 demonstrates the superiority of the BLOTC network using the Rand error. The value of the Rand error for all methods is low (less than 10%). As before, this is because the class distribution is skewed. In the ground truth, most pairs of pixels belong to different objects. Therefore the trivial segmentation in which every pixel belongs to a different object has a Rand error of less than 10%. This sets the baseline for performance in Figure 4. We also benchmarked several leading competitors: multiscale normalized cuts [6], *gPb*-OWT-UCM [8], and Boosted Edge Learning (see Supplementary Information for details regarding use of these methods). Their performance was much worse than that of our convolutional networks, and only barely better than the baseline set by the trivial segmentation. Of course,

it is reasonable to assume that one reason *gPb*-OWT-UCM and multiscale normalized cuts yield inferior results is that they were designed for natural images, while the convolutional networks have been optimized for EM images. Yet Boosted Edge Learning was also trained on the EM images, and its performance is still closer to that of *gPb*-OWT-UCM than our networks.

5. Discussion

A number of recent papers have introduced segmentation algorithms that employ topological constraints within complex *inference* optimizations. Many of these methods are fundamentally interactive and require human interaction (such as entering seed points [24, 25]), or make topological assumptions about a test image [26].

In contrast, we use topology to define the warping error metric and introduce a *learning* algorithm, BLOTC, that optimizes this metric. It is worth comparing our approach to Maximin Affinity Learning of Image Segmentation (MALIS), which is based on minimizing the Rand error [9]. The two approaches are similar, in the sense that the warping and Rand errors are both metrics of segmentation performance that satisfy the three properties listed at the beginning of this paper. Section 2.4 explained the differences between the warping and Rand errors, which may determine whether BLOTC or MALIS is preferable in any given application.

Our empirical results provide strong evidence that in the challenging domain of EM images, the accuracy of segmentations produced from a boundary detector learned with BLOTC can dramatically exceed other state of the art methods. Although the results presented here are for 2d segmentation, the method can be applied to fully 3d segmentation using 3d patch classifiers [4] and the 3d digital topology formalism [12][1].

There are several important directions for future work. Convolutional networks and BLOTC are both domain-general methods, and thus it will be interesting to investigate natural image segmentation. The alternating optimization we employ in BLOTC is simple and apparently effective, but a truly joint optimization of some kind may be superior. Preliminary work suggests that BLOTC can be used to learn from sparse labels and perform interactive segmentation. Finally, it should also be possible to extend BLOTC to boundary detectors that classify the edges of an affinity graph rather than pixels [28, 29, 9].

Acknowledgements: we are grateful to T. Yung Kong and Srinivas Turaga for helpful discussions. We also thank the authors of [1, 6, 3, 8] for making code available.

References

[1] D. R. Martin, C. C. Fowlkes, and J. Malik, "Learning to detect natural image boundaries using local brightness, color, and texture cues." *IEEE Trans Pattern Anal Mach Intell*, May 2004.

[2] D. Martin, C. Fowlkes, and J. Malik, "Learning to detect natural image boundaries using brightness and texture," *NIPS*, 2003.

[3] P. Dollar, Z. Tu, and S. Belongie, "Supervised Learning of Edges and Object Boundaries," in *CVPR*, 2006.

[4] V. Jain, J. Murray, F. Roth, S. Turaga, V. Zhigulin, K. Briggman, M. Helmstaedter, W. Denk, and H. Seung, "Supervised Learning of Image Restoration with Convolutional Networks," in *ICCV*, 2007.

[5] M. Maire, P. Arbelaez, C. Fowlkes, and J. Malik, "Using contours to detect and localize junctions in natural images," in *CVPR*, 2008.

[6] T. Cour, F. Benezit, and J. Shi, "Spectral segmentation with multiscale graph decomposition," in *CVPR*, 2005.

[7] R. Unnikrishnan, C. Pantofaru, and M. Hebert, "Toward objective evaluation of image segmentation algorithms," *PAMI*, 2007.

[8] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik, "From contours to regions: An empirical evaluation," in *Proc. CVPR*, 2009.

[9] S. Turaga, K. Briggman, M. Helmstaedter, W. Denk, and H. Seung, "Maximin affinity learning of image segmentation," in *NIPS*, 2009.

[10] W. Rand, "Objective criteria for the evaluation of clustering methods," *Journal of the American Statistical Association*, 1971.

[11] T. Kong and A. Rosenfeld, "Digital Topology: introduction and survey," *Computer Vision Graphics and Image Processing*, 1989.

[12] G. Bertrand and G. Malandain, "A new characterization of three-dimensional simple points," *Pattern Recognition Letters*, 1994.

[13] A. Rosenfeld, T. Kong, and A. Nakamura, "Topology-preserving deformations of two-valued digital pictures," *Graphical Models and Image Processing*, vol. 60, no. 1, pp. 24–34, 1998.

[14] C. Pudney, "Distance-ordered homotopic thinning: a skeletonization algorithm for 3D digital images," *Comp. Vision & Image Und.*, 1998.

[15] X. Han, C. Xu, and J. Prince, "A topology preserving level set method for geometric deformable models," *PAMI*, 2003.

[16] K. Hayworth, N. Kasthuri, R. Schalek, and J. Lichtman, "Automating the collection of ultrathin serial sections for large volume TEM reconstructions," *Microscopy and Microanalysis*, 2006.

[17] W. Denk and H. Horstmann, "Serial block-face scanning electron microscopy to reconstruct three-dimensional tissue nanostructure," *PLoS Biol*, vol. 2, no. 11, p. e329, 2004.

[18] E. Jurrus, M. Hardy, T. Tasdizen, P. Fletcher, P. Koshevoy, C. Chien, W. Denk, and R. Whitaker, "Axon tracking in serial block-face scanning electron microscopy," *Medical Image Analysis*, 2008.

[19] B. Andres, U. Koethe, M. Helmstaedter, W. Denk, and F. Hamprecht, "Segmentation of SBFSEM volume data of neural tissue by hierarchical classification," in *DAGM*. Springer, 2008, pp. 142–152.

[20] J. Lichtman and J. Sanes, "Ome sweet ome: what can the genome tell us about the connectome?" *Curr. Opion. in Neuro.*, 2008.

[21] H. Seung, "Reading the Book of Memory: Sparse Sampling versus Dense Mapping of Connectomes," *Neuron*, 2009.

[22] M. Helmstaedter, K. Briggman, and W. Denk, "3D structural imaging of the brain with photons and electrons," *Curr. Opion. in Neurob.*, 2009.

[23] A. Vazquez-Reina, E. Miller, and H. Pfister, "Multiphase geometric couplings for the segmentation of neural processes," in *CVPR*, 2009.

[24] S. Vicente, V. Kolmogorov, and C. Rother, "Graph cut based image segmentation with connectivity priors," in *CVPR*, 2008.

[25] Y. Zeng, D. Samaras, W. Chen, and Q. Peng, "Topology cuts: A novel min-cut/max-flow algorithm for topology preserving segmentation in N–D images," *Computer Vision & Image Und.*, 2008.

[26] S. Nowozin and C. Lampert, "Global connectivity potentials for random field models." CVPR, 2008.

[27] W. Denk and H. Horstmann, "Serial block-face scanning electron microscopy to reconstruct three-dimensional tissue nanostructure," *PLoS Biol*, vol. 2, no. 11, p. e329, 2004.

[28] C. Fowlkes, D. Martin, and J. Malik, "Learning affinity functions for image segmentation: Combining patch-based and gradient-based approaches," in *CVPR*, 2003.

[29] S. Turaga, J. Murray, V. Jain, F. Roth, M. Helmstaedter, K. Briggman, W. Denk, and H. Seung, "Convolutional networks can learn to generate affinity graphs for image segmentation," *Neural Computation*, In Press.

[1] Early versions of this work were developed using Serial Block-Face Scanning Electron Microscopy data [27] for 3d segmentation.

Boundary Learning by Optimization with Topological Constraints Supplementary Material

Viren Jain[4,*], Benjamin Bollmann[4,*], Mark Richardson[4], Daniel R. Berger[4,5], Moritz N. Helmstaedter[3], Kevin L. Briggman[3], Winfried Denk[3], Jared B. Bowden[2], John M. Mendenhall[2], Wickliffe C. Abraham[6], Kristen M. Harris[2,†], Narayanan Kasthuri[1], Ken J. Hayworth[1], Richard Schalek[1], Juan Carlos Tapia[1], Jeff W. Lichtman[1] and H. Sebastian Seung[4,5]

[1]Department of Molecular and Cellular Biology, Center for Brain Science, Harvard University, Cambridge, MA, USA

[2]Center for Learning and Memory, Department of Neurobiology, University of Texas at Austin, TX, USA

[3]Department of Biomedical Optics, Max Planck Institute for Medical Research, Heidelberg, Germany

[4]Brain & Cognitive Sciences, Massachusetts Institute of Technology, Cambridge, MA, USA

[5]Howard Hughes Medical Institute, Cambridge, MA, USA

[6]Dept. of Psychology and Brain Health and Repair Research Center, Univ. of Otago, Dunedin, New Zealand

1. Details of Experimental Procedures

In this section we provide additional details of the experimental comparisons that were performed in Section 4 of the main text. We also show an extended presentation of the warping error results shown in the main text. In particular Figure 1 shows the warping error on the test set of the convolutional network methods along with BEL and *gPb*-OWT-UCM. For this comparison, a threshold of *gPb*-OWT-UCM and BEL was chosen according to the threshold that achieved lowest Rand error also on the test set (shown in Figure 4 of the main text). These results are consistent with the relative ordering of algorithms that the Rand index produced, but the relative reduction in error between the methods is larger (for example, the *gPb*-OWT-UCM method has almost ten times as much warping error as the highest performer, BLOTC CN).

Figure 2 also shows a visual depiction of the segmentation and boundary maps of all methods that are discussed.

1.1. Multiscale Normalized Cut

Multiscale normalized cut was performed using publicly available code provided by the authors of [1]: `http://www.seas.upenn.edu/~timothee/software/ncut_multiscale/ncut_multiscale.html`

This technique requires that the number of objects in the image be specified by the user. We are interested in completely automated segmentation in which such information would not usually be available. Therefore we provided to the code the average number of objects in a training set image. However, we also tried providing the code with the true number of objects in each test set image. Then the results of "Multiscale Ncut" shown in Figure 4 of the main text improve slightly, but remain worse than all other techniques.

1.2. *gPb*-OWT-UCM

The *gPb*-OWT-UCM algorithm (global probability of boundary followed by the oriented watershed transform and a hierarchical region construction by ultrametric contour maps) was performed using publicly available code provided by the authors of [2, 3]: `http://www.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/gpb/grouping.zip`

The following is a summary of the algorithm implemented by the code, as described in [2]. First, the *gPb* contour detector was applied directly to the raw EM images, producing an 8-channel oriented localized probability of boundary map, as well as a single-channel thinned contour image which is the result shown in Figure 2 of the main text. The 8-channel boundary map was then converted to an oversegmentation using the oriented watershed transform (OWT), and then an ultrametric contour map (UCM) according to the dissimilarity between regions as determined by mean probability of boundary value. Finally, the ultrametric contour map was partitioned using connected components at various thresholds (where the x-axis in Figure 4 for this technique corresponds a segmentation of the UCM thresholded at a value of $255 * (1 - x)$, to compensate for the range of the UCM, and then the binary boundary map is converted to an in/out map prior to connected components by flipping the binary values).

1.3. Boosted Edge Learning

The Boosted Edge Learning algorithm was applied to our training set of EM images using publicly available code provided by the authors of [4]: `http://www.loni.ucla.edu/~ztu/Download.htm`

*These authors made comparable contributions to this paper.

†Supported by NIH/NINDS R37 NS021184 and NIH/NIBIB EB002170.

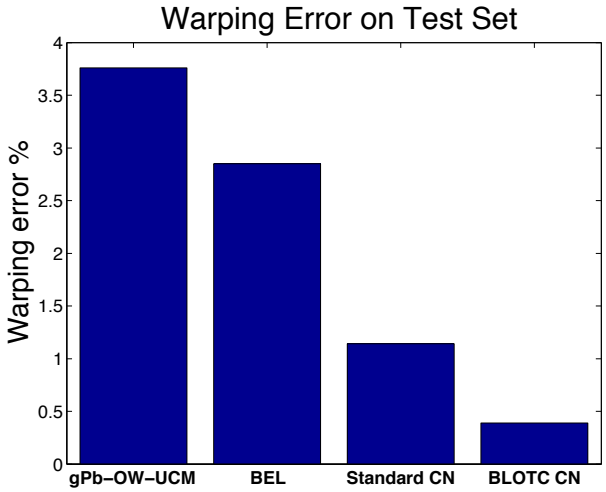


Figure 1. Warping error on test set.

A probabilistic boosting tree of depth 10 was used with 120 weak classifiers in each AdaBoost node. A patch size of 30 was used, which is a slightly larger than the field of view of the convolutional networks that were used. These were the default parameters provided in the code.

After training was complete, we proceeded to quantify segmentation performance on the test set of images. Here we had to choose our own method for generating segmentations from BEL output. The simple procedure of finding connected components of the thresholded BEL output (which was used for the CNs and *gPb*-OWT-UCM) produced a poor Rand error, as did the watershed transform on the negative of the BEL output. Therefore we applied the watershed transform only after using MATLAB's `imimposemin` command to damp local maxima of the BEL output, constraining them to occur where the BEL output was greater than a threshold. The watershed transform was performed with 8 connectivity (4 connectivity gave worse results).

1.4. Convolutional Networks

A convolutional network is an alternating sequence of linear filtering and nonlinear transformation operations. The input and output layers include one or more images, while intermediate layers contain "hidden" units with images called feature maps that are the internal computations of the algorithm. The activity of feature map a in layer k is given by

$$I_{k,a} = f\left(\sum_b w_{k,ab} \otimes I_{k-1,b} - \theta_{k,a}\right) \qquad (1)$$

where $I_{k-1,b}$ are feature maps that provide input to $I_{k,a}$, and $\otimes$ denotes the convolution operation. The function f is the sigmoid $f(x) = 1/\left(1+e^{-x}\right)$ and $\theta_{k,a}$ is a bias parameter. Gradient learning of this architecture can be performed with an version of the backpropagation algorithm [5, 6].

Our experiments were performed on the gray scale EM images and hence the networks contain a single image in the input layer. It is straightforward to extend this approach to color images by assuming an input layer with multiple images (e.g., RGB color channels). For numerical reasons, it is preferable to use input and target values in the range of 0 to 1, and hence the 8-bit integer intensity values of the dataset (values from 0 to 255) were normalized to lie between 0 and 1.

As our loss function during optimization of the convolutional network, we used the square-square loss

$$l(x,\hat{x}) = x\max(0, 1-\hat{x}-m)^2 + (1-x)\max(0,\hat{x}-m)^2$$

with $m = 0.2$, rather than the squared loss $(x-\hat{x})^2$. This loss function is a better approximation of the true binary classification error we seek to optimize. This is particularly important in the scenario in which the classifier has predicted the correct class of most examples (based on thresholding the analog output), and there are relatively few remaining incorrectly classified examples. In this case, the small amount of remaining error in the squared loss related to pushing values to 0 or 1 may dominate the error associated with incorrectly classified examples. The square-square loss allows the training to "give up" when the classifier output is correct by a sufficient margin. This gives the classifier more flexibility to achieve higher binary classification accuracy.

Batch learning is inefficient in this context, as the training set has millions of pixels and it is not practical to compute the gradient with respect to the entire training set for each update, particularly when many hundreds of thousands of updates may be required in order to reach convergence. Therefore we adapted stochastic online learning to this problem. We employed a minibatch implementation in which a randomly chosen 14×14 patch of the network output was used to compute each gradient update. A localized patch shares computation in a convolutional network and is therefore especially efficient to compute.

Segmentations were generated by thresholding the analog output of the convolutional network and then performing connected components with the same $\kappa = 4$ and $\bar{\kappa} = 8$ adjacency using in warping.

2. Classification of non-simple points

As discussed in Section 4 of the main text, in our experiments with BLOTC learning we allowed the warping to flip certain non-simple points in addition to any simple point. In particular, the warping was allowed to create holes within objects, therefore not penalizing the computer for detecting internal boundaries even if they were not traced by the human. The warping was also allowed to create new objects,

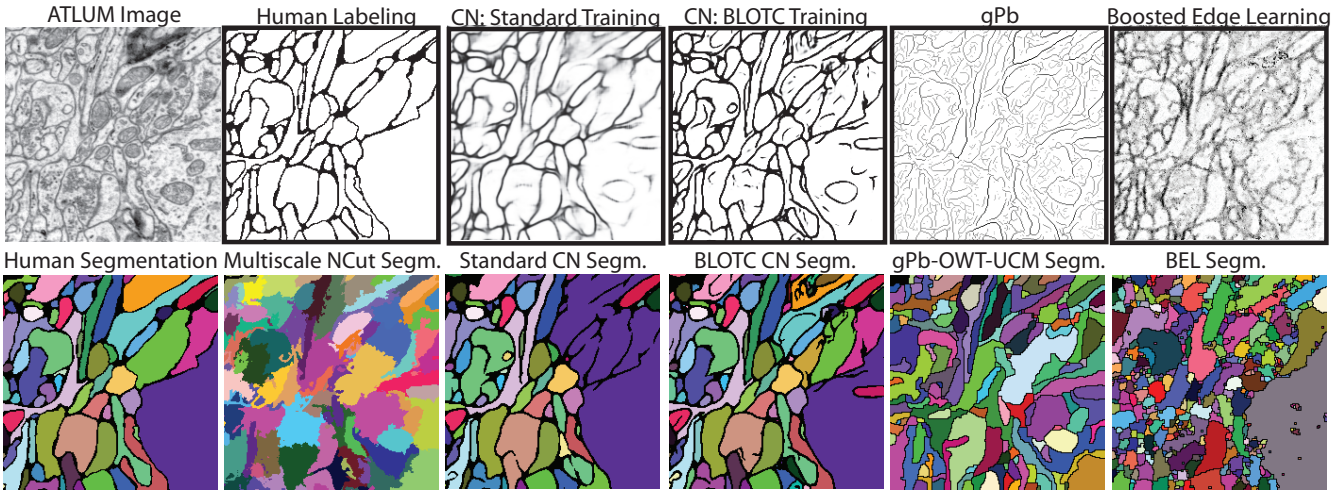


Figure 2. Comparison of boundary detector outputs and corresponding segmentations on an image from the test set. Both Standard and BLOTC CN segmentations were generated by connected components on the respective boundary detector outputs at threshold 0.75. For gPb and BEL, segmentations were generated at the optimal threshold according to the Rand error. Multiscale NCut directly generates a segmentation and thus no threshold was used (in the results shown above, the true number of objects in this specific test image was provided as input to to the multiscale normalized cut routine).

in case certain objects in the image were neglected in the human tracing (in practice this occured extremely rarely).

In order to allow only certain topological changes, it is necessary to be able to classify the type of topological change flipping a particular non-simple point will cause. By "change topology" we mean alter the number of components, or cavities. We would thus like to know which topological quantity is affected by flipping a particular non-simple point. In this supplementary section we discuss how non-simple points can be classified rigorously using additional concepts from the field of digital topology.

2.1. Identifying addition and deletion of objects using topological numbers

Certain cases can be identified based on topological number alone (see Section 2.1 of the main text for a review of topological numbers). Let $(\kappa, \overline{\kappa})$ be some complementary adjacency relation in either a 2d or 3d space.

Theorem 2.1. *Topological number characterization of object deletion. Suppose q is an element of the foreground L. Then flipping q will result in a κ-component deletion if and only if $T_\kappa(q, L) = 0$.*

Proof. The proof (given in [7]) is simple: suppose that q is flipped, then a connected component of L is removed if and only if q is an isolated point, in which case $T_\kappa(q, L) = 0$. Conversely suppose $T_\kappa(q, L) = 0$, then q must be an isolated point with respect to the foreground. Therefore flipping q results in an object deletion.

Note that object deletion is only one way to decrease the number of foreground connected components. The merging of two existing components is the other way. □

A similar statement can be offered for object addition:

Theorem 2.2. *Topological number characterization of object addition. Suppose q is an element of the background $\overline{L}$. Then flipping q will result in a κ-component addition if and only if $T_\kappa(q, L) = 0$.*

Proof. Suppose that q is flipped, then a connected component L is added if and only if q is an isolated point, in which case $T_\kappa(q, L) = 0$. Conversely suppose $T_\kappa(q, L) = 0$, then q must be an isolated point with respect to the foreground. Therefore flipping q results in an object addition. Again, we note that object addition is only one way to add to the number of components. Splitting an existing object into two is the other. □

Topological numbers can also be used to identify the creation and deletion of cavities, using a similar criteria based on the "background" topological number. Recall that the background is the unique infinite $\overline{\kappa}$-connected component of $\overline{L}$, and by definition any other $\overline{\kappa}$-connected component of $\overline{L}$ is called a cavity in L.

Topological number characterization of cavity deletion. Suppose q is an element of the background $\overline{L}$. Then flipping q will result in a $\overline{\kappa}$-component (cavity) deletion if and only if $T_{\overline{\kappa}}(q, \overline{L}) = 0$.

Proof. Suppose that q is flipped, then a connected component of $\overline{L}$ is removed if and only if q is an isolated background point, in which case $T_{\overline{\kappa}}(q, \overline{L})$=0. Conversely suppose $T_{\overline{\kappa}}(q, \overline{L}) = 0$, then q must be an isolated background point. Therefore flipping q results in deletion of a cavity. □

Theorem 2.3. *Topological number characterization of cavity addition. Suppose q is an element of the foreground L. Then flipping q will result in a $\overline{\kappa}$-component (cavity) addition if and only if $T_{\overline{\kappa}}(q, \overline{L}) = 0$.*

Proof. Suppose that q is flipped, then a connected component in $\overline{L}$ is added if and only if q is an isolated background point, in which case $T_{\overline{\kappa}}(q, \overline{L}) = 0$. Conversely suppose $T_{\overline{\kappa}}(q, \overline{L}) = 0$, then q must be an isolated point with respect to the background. Therefore flipping q results in the creation of cavity.

As in the case of foreground components, deletion/addition of cavities by flipping isolated background points is *not* the only way in which the number of background components can be changed. A cavity can be merged (with other cavities or the infinite background component) and split into two cavities by a single flip of some pixel q. However in such cases it will not be true that $T_{\overline{\kappa}}(q, \overline{L}) = 0$. □

We note that if $T_{\kappa}(q, L) = 0$ then it follows $T_{\overline{\kappa}}(q, \overline{L}) = 1$ and vice versa. This is because an isolated foreground (background) point is clearly surrounded by a background (foreground), which under normal adjacency relations for either κ or $\overline{\kappa}$ will be connected as a single component within such a neighborhood itself. Finally we recall that if $T_{\kappa}(q, L) = T_{\overline{\kappa}}(q, \overline{L}) = 1$ the point is simple and thus causes no topological change when altered. Therefore, we have a characterization for all points for which $T_{\kappa}(q, L) \in \{0, 1\}$ and $T_{\overline{\kappa}}(q, \overline{L}) \in \{0, 1\}$.

2.2. Identifying splits, mergers, and hole addition/deletion using extended topological numbers

Topological numbers as originally defined are insufficient to characterize the precise nature of topological changes caused by flipping non-simple points for which $T_{\overline{\kappa}}(q, \overline{L}) > 1$ or $T_{\kappa}(q, L) > 1$. For example, flipping a pixel q from background to foreground when $T_{\kappa}(q, L) > 1$ clearly implies a topological change since q is non-simple; however, the nature of this change depends on *non-local* properties that are not captured by the local neighborhood from which $T_{\kappa}(q, L)$ is computed. We need access to global connectivity information that would enable us to distinguish between, for example, merging two objects versus creating a hole. Global connectivity is well defined over the image, however the binary image itself does not represent such information directly. Hence, such classifications require additional measures beyond topological numbers that take into account this global information [5].

References

[1] T. Cour, F. Benezit, and J. Shi, "Spectral segmentation with multiscale graph decomposition," in *CVPR*, 2005.

[2] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik, "From contours to regions: An empirical evaluation," in *Proc. CVPR*, 2009.

[3] M. Maire, P. Arbelaez, C. Fowlkes, and J. Malik, "Using contours to detect and localize junctions in natural images," in *CVPR*, 2008.

[4] P. Dollar, Z. Tu, and S. Belongie, "Supervised Learning of Edges and Object Boundaries," in *CVPR*, 2006.

[5] V. Jain, "Machine learning of image analysis with convolutional networks and topological constraints," Ph.D. dissertation, Massachusetts Institute of Technology, 2009.

[6] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[7] G. Bertrand and G. Malandain, "A new characterization of three-dimensional simple points," *Pattern Recognition Letters*, 1994.