

000
001
002
003
004
005
006
007
008
009
010
011
012
013
014
015054
055
056
057
058
059
060
061
062
063
064
065
066
067
068
069
070
071
072
073
074
075
076
077
078
079
080
081
082
083
084
085
086
087
088
089
090
091
092
093
094
095
096
097
098
099
100
101
102
103
104
105
106
107

Graph-Based Neural Reconstruction from Skeletonized 3D Networks

Anonymous CVPR submission

Paper ID 0446

Abstract

Advancements in electron microscopy image acquisition have created massive connectomics datasets in the terabyte range that make manual reconstruction of neuronal structures infeasible. Current state-of-the-art automatic methods segment neural membranes at the pixel level followed by agglomeration methods to create full neuron reconstructions. However, these approaches widely neglect global geometric properties that are inherent in the graph structure of neural wiring diagrams. In this work, we follow bottom-up pixel-based reconstruction by a top-down graph-based method to more accurately approximate neural pathways. We first generate skeletons in 3D from the membrane labels of the pixel-based segmentation. We then simplify this skeletonized 3D network into a 3D graph with nodes corresponding to labels from the segmentation and edges identifying potential locations of segmentation errors. We use a CNN classifier trained on ground truth data to generate edge weights on the 3D graph corresponding to error probabilities. We then apply a multicut algorithm to generate a partition on the graph that improves the final segmentation. Because the 3D graph is small and encodes top-down information our method is efficient and globally improves the neural reconstruction. We demonstrate the performance of our approach on multiple real-world connectomics datasets with an average variation of information improvement of $X \times$.

1. Introduction

The field of connectomics is concerned with reconstructing the wiring diagram of the brain at nanometer resolutions to enable new insights into the workings of the brain [8, 18]. Recent advancements in image acquisition using multi-beam serial-section electron microscopy (sSEM) have allowed researchers to produce terabytes of image data every hour [12]. It is not feasible for domain experts to manually reconstruct this vast amount of im-

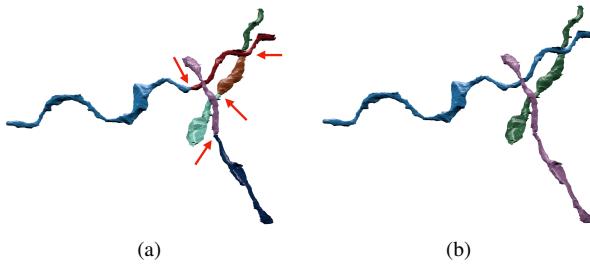


Figure 1: Example improvement of neural reconstruction. (a) We extract 3D skeletons from pixel-based segmentation algorithms to create a 3D graph representation. Edges with high segmentation error probabilities are indicated by the red arrows. (b) We improve the accuracy of segmentation using a graph partitioning algorithm, leveraging both local and global information.

age data [11]. State-of-the-art automatic reconstruction approaches use pixel-based segmentation with convolutional neural networks (CNNs) followed by agglomeration strategies [21, 23, 27, 30, 33, 43]. These *bottom-up pixel-based* methods produce excellent results but still fall short of acceptable error rates for large volumes.

We present a *top-down graph-based* method that builds on the outputs of bottom-up pixel-based segmentation approaches. We first extract 3D skeleton networks from the input segmentation and generate a simplified 3D graph (Fig. 1a). We train a CNN classifier on the agglomerated regions in the segmentation data to detect errors. We run the classifier to populate the graph edge weights with error probabilities. We then use a graph optimization algorithm to partition the graph into the final improved reconstruction by enforcing domain-specific global constraints from biology (Fig. 1b).

Our approach operates at a level of abstraction above existing pixel-based methods. This allows us to leverage both local and global information to produce more accurate reconstructions. Our method is independent of image reso-

108 lution and acquisition parameters, enabling its application
109 to isotropic and anisotropic image data without retraining.
110 Using the 3D graph induced by the segmentation allows us
111 to enforce global biological constraints on the reconstruction.
112 Our dual approach of assessing local decisions in a
113 global context yields accuracy improvements over existing
114 reconstruction methods.

115 This work makes the following contributions: (1) a novel
116 top-down method using graphs from skeletonized 3D net-
117 works for improved neural reconstruction of connectomics
118 data; (2) a region-based CNN classifier to detect errors us-
119 ing the 3D graph as global constraint; (3) an empirical eval-
120 uation of our method on several connectomics datasets; (4)
121 our method yields improved performance over a state-of-
122 the-art pixel-based reconstruction approach on average by
123 $\textcolor{red}{X}$ percent without drastically increasing the running time.

125 2. Related Work

126 We review some of the most successful segmentation
127 methods that have been applied to large-scale EM images
128 in connectomics.

131 **Pixel-based methods.** A large amount of connectomics
132 research considers the problem of extracting segmentation
133 information at the pixel (i.e., voxel) level from the raw EM
134 images. Some early techniques apply computationally ex-
135 pensive graph partitioning algorithms with a single node
136 per pixel [1]. However, these methods do not scale to ter-
137 abyte datasets. More recent methods train classifiers to pre-
138 dict membrane probabilities per image slice either using
139 2D [4, 14, 19, 21, 41] or 3D CNNs [23, 33, 40].

140 Oftentimes these networks produce probabilities for the
141 affinity between two voxels (i.e., the probability that adja-
142 cent voxels belong to the same neuron). The MALIS cost
143 function is specifically designed for generating affinities
144 that produce good segmentations [2]. More recently, flood-
145 filling networks produce segmentations by training an end-
146 to-end neural network that goes from EM images directly
147 to label volumes [16]. These networks produce impressive
148 accuracies but at a high computational cost.

150 **Region-based methods.** Several pixel-based approaches
151 generate probabilities that neighboring pixels belong to the
152 same neuron. Often a watershed algorithm will then cluster
153 pixels into super-pixels [43]. Many methods build on top of
154 these region-based strategies and train random-forest classi-
155 fiers to produce the final segmentations [21, 27, 29, 30, 43].

156 **Error-correction methods.** Some recent research builds
157 on top of these region-based methods to correct errors in
158 the segmentation either using human proofreading [15, 22,
159 10, 9] or fully automatically [32, 44]. However, to our

162 knowledge, our method is the first to extract a 3D graph
163 from pixel-based segmentations for a true top-down error
164 correction approach. This allows us to enforce domain-
165 specific biology constraints and efficient graph partitioning
166 algorithms. Many segmentation and clustering algorithms
167 use graph partitioning techniques [1] or normalized cuts for
168 traditional image segmentation [17, 36, 38]. Even though
169 graph partitioning is an NP-Hard problem [5] there are sev-
170 eral useful multicut heuristics that provide good approxima-
171 tions with reasonable computational costs [13]. We use the
172 method of Keuper et al. [20] to partition the extracted 3D
173 graph into the final neural reconstruction.

175 3. Method

176 There are two types of errors that can occur in connec-
177 toomics segmentation. The first, called a split error, occurs
178 when there are two segments that should have been merged.
179 The second, called a merge error, happens when one seg-
180 ment should be split into two. Generally, it is much more
181 difficult to correct merge errors than to correct split errors,
182 as the space of possible split proposals grow quickly [28].
183 Thus, most reconstruction approaches are tuned towards
184 over-segmentation with many more split than merge errors.
185 Our method takes as input over-segmentations of EM image
186 volumes generated by state-of-the-art connectomics recon-
187 struction pipelines (Sec. 4.2). Our goal is to identify loca-
188 tions of split errors and merge the corresponding segments
189 automatically.

190 From the input segmentation we generate a graph G with
191 nodes N and edges E with non-negative edge weights w_e .
192 The nodes correspond to label segments from the segmenta-
193 tion with edges between segments considered for merging.
194 Ideally, our graph has edges corresponding to all of the seg-
195 ments that were erroneously split. To compute this graph
196 we generate a skeleton for every segment in the pixel-based
197 segmentation. The skeletonized 3D network is a simplified
198 representation of the overall branching structure of the neu-
199 rons. From these skeletons we identify potential merge loca-
200 tions and produce the corresponding edges for the graph.
201 To find actual merges we run a classification CNN to gener-
202 ate edge weights corresponding to merge probabilities. We
203 then use a multicut algorithm to generate a partition on the
204 graph where nodes in the same partition are assigned the
205 same output label in the improved segmentation. We will
206 now discuss the three major components to our framework
207 (graph creation, edge weights assignment, and graph parti-
208 tioning) in more detail.

209 3.1. Node Generation

210 The simplest node generation strategy creates one node
211 for every unique segment label in the input volume. How-
212 ever, some of the millions of labels in the volume corre-
213 spond to very small structures that are likely the result of

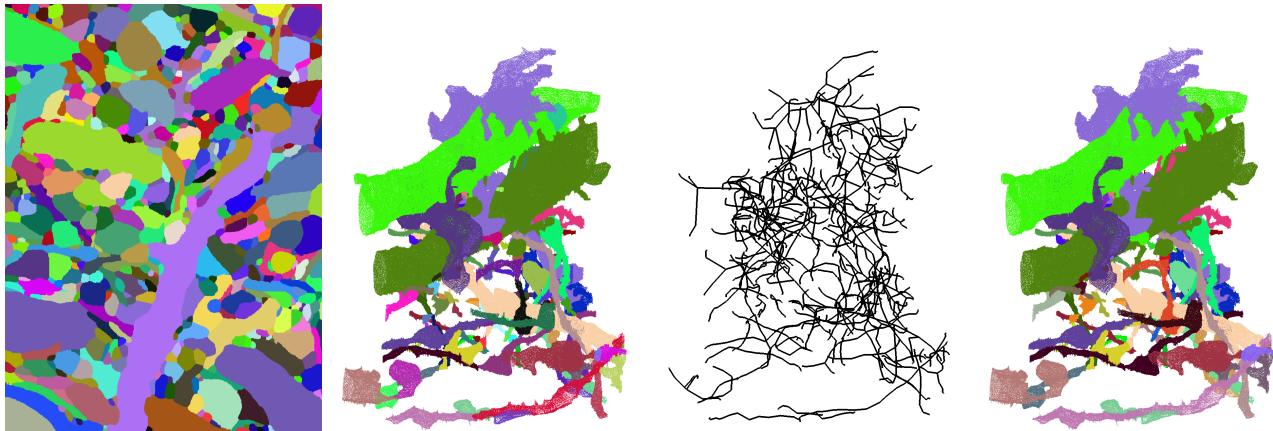


Figure 2: Overview of the proposed methods.

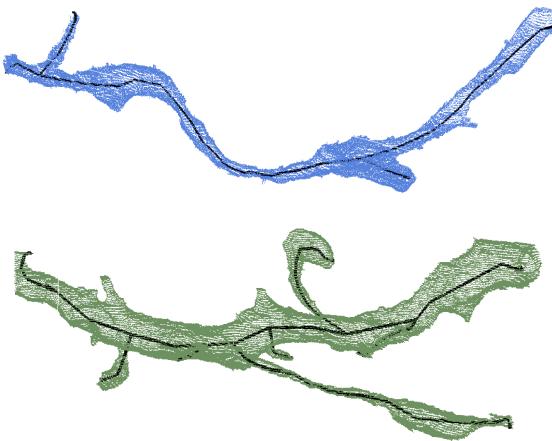


Figure 3: Example skeletons (in black) extracted from segments (blue and green) using the TEASER algorithm.

segmentation errors, typically in regions with noisy raw image data. It is difficult to extract useful shape features from these segments because of their small, often random, shape. We prune these nodes from the graph by removing all segments with fewer than a threshold $t_{seg} = 20,000$ voxels. This removed on average 56% of the segments in our datasets (Sec. 4.1). Despite the large number of segments, these regions only take up 1.6% of the total volume on average.

3.2. Edge Generation

A typical approach to generating edges produces an edge between all adjacent segments. Two segments l_1 and l_2 are considered adjacent if there is a pair of adjacent voxels with one labeled l_1 and the other labeled l_2 . For example, pixel-based agglomeration methods such as Neuro-

Proof [29] and GALA [27] consider all pairs of adjacent segments for merging. However, this method produces too many edges in the graph for graph-based optimization approaches. We identify a smaller number of pairs of segments to consider as graph edges using the following approach.

First, we extract a skeleton from each segment in the label volume using the TEASER algorithm [34, 42]. Fig. 3 shows an example of two extracted skeletons (in black). These skeletons consist of a sequence of *joints*, i.e., locations that are locally a maximum distance from the segment boundary, with line segments connecting successive joints. We prune the joints that are within $t_{jnt} = 50$ voxels of each other to reduce unnecessary branching. We refer to joints that have only one connected neighbor as *endpoints*. Many of the segments that are erroneously split have nearby endpoints (Fig. 4). We make use of this fact to merge segments with the following two-pass pruning algorithm.

In the first pass, we iterate over all endpoints e belonging to a segment S and create a set of segments \mathbb{S}'_e that includes all labels that are within t_{low} voxels from e . Elements of \mathbb{S}'_e are candidates for merging. However, this first pass often leads to too many candidates, requiring an additional pass for further pruning. In the second pass, we consider all of the segments in \mathbb{S}'_e for every endpoint e . If a segment $S' \in \mathbb{S}'_e$ has an endpoint within t_{high} voxels of e , the segment S and S' are considered for merging. We store the midpoints between the two endpoints as the center of the potential merges in the set \mathbb{S}_c . **HP: add how the final graph is constructed**

3.3. Edge Weights Assignment

We assign non-negative edge weights w_e to each edge where the weight corresponds to the probability that two nodes belong to the same neuron. Instead of using hand-

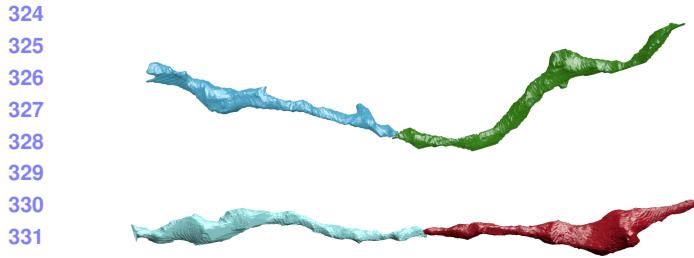


Figure 4: Two erroneously split segments.

324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377

crafted features to compute the similarity between adjacent nodes, we train a 3D CNN classifier to learn from the manually labeled oversegmentation input volume (Sec. 4.1).

3.3.1 Classifier Input

We extract a cubic region of interest (ROI) around each endpoint e in \mathbb{S}_c as input to the CNN. The CNN receives three input channels for every voxel in the ROI around segments l_1 and l_2 . The input in all of the channels is in the range $\{-0.5, 0.5\}$. The first channel is 0.5 only if the corresponding voxel has label l_1 . The second channel is 0.5 only if the corresponding voxel has label l_2 . The third channel is 0.5 if the corresponding voxel is either l_1 or l_2 .

3.3.2 Network Architecture & Training

We use the CNN architecture by Chatfield et al. [3]. It consists of three layers of double convolutions followed by a max pooling step. The first max pooling layer is anisotropic with pooling only in the x and y dimensions. The output of this final pooling step is flattened into a 1D vector that is input into two fully connected layers. The final layer produces probabilities with a sigmoid activation function [6]. All of the other activation functions are LeakyReLU [24].

For training we use a stochastic gradient descent optimizer with Nesterov’s accelerated gradient [26]. We employ dropouts of 0.2 after every pooling layer and the first dense layer, and a dropout of 0.5 after the final dense layer to prevent overfitting. We discuss all other network parameters in Sec. 4.4.

3.4. Graph Partitioning

After constructing the 3D graph we apply graph partitioning using multicut to compute the final segmentation. Using top-down graph partitioning allows us to apply biological constraints on the output. Neuroscientists know that neuronal connectivity graphs in the brain are acyclic (i.e., the graphs have a genus of zero). We enforce this constraint by finding a multicut partition of the graph that generates a *forest* of nodes, i.e., a set of trees where no segment has

378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431

a cycle. To solve this constraining multicut problem we use the method by Keuper et al. [20] that produces a feasible solution by greedy additive edge contraction.

4. Experimental Results

We evaluate our method by comparing it to a state-of-the-art pixel-based reconstruction approach using datasets from two different animals.

4.1. Datasets

Kasthuri. The Kasthuri dataset consists of scanning electron microscope images of the neocortex of a mouse brain [18]. This dataset is $5342 \times 3618 \times 338$ voxels in size. The resolution of the dataset is $3 \times 3 \times 30 \text{ nm}^3$ per voxel. We evaluate our methods using the left cylinder of this 3-cylinder dataset. We downsample the dataset in the x and y dimensions to give a final resolution of $6 \times 6 \times 30 \text{ nm}^3$ per voxel. We divide the dataset into two volumes (Vol. 1 and Vol. 2) along the x dimension, where each volume is $8.0 \times 10.9 \times 10.1 \mu\text{m}^3$ or $1335 \times 1809 \times 338$ voxels.

FlyEM. The FlyEM dataset comes from the mushroom body of a 5-day old adult male Drosophila fly imaged by a focused ion-beam milling scanning electron microscopy [37]. The mushroom body in this species is the major site of associative learning. The original dataset contains a $40 \times 50 \times 120 \mu\text{m}^3$ volume with a resolution of $10 \times 10 \times 10 \text{ nm}^3$ per voxel. We use two cubes (Vol. 1 and Vol. 2) of size $10 \times 10 \times 10 \mu\text{m}^3$ or $1000 \times 1000 \times 1000$ voxels.

4.2. Pixel-Based Segmentations

The segmentation on the Kasthuri dataset was computed by agglomerating 3D supervoxels produced by the z-watershed algorithm from 3D affinity predictions [43]. A recent study by Funke et al. [35] demonstrated superior performance of such methods over existing ones on anisotropic data. We learn 3D affinities using MALIS loss with a U-net [39, 33]. We apply the z-watershed algorithm with suitable parameters to compute a 3D oversegmentation of the volume. The resulting 3D oversegmentation is then agglomerated using the technique of context-aware delayed agglomeration to generate the final segmentation [29].

For the FlyEM data, based on the authors’ suggestion [37], we applied a context-aware delayed agglomeration algorithm [29] that shows improved performance on this dataset over the pipeline used in the original publication. This segmentation framework learns voxel and supervoxel classifiers with an emphasis to minimize under-segmentation error. At the same time this framework produces lower over-segmentation than standard algorithms. The algorithm first computes multi-channel 3-D predictions

432 for membranes, cell interiors, and mitochondria, among
 433 other cell features. The membrane prediction channel is
 434 used to produce an over-segmented volume using 3D wa-
 435 tershed, which is then agglomerated hierarchically up to a
 436 certain confidence threshold. We used exactly the same pa-
 437 rameters as the publicly available code for this algorithm.
 438

439 4.3. Graph Pruning Parameters

440 The two parameters for the graph pruning algorithm
 441 (Sec. 3.1) are t_{low} and t_{high} . Ideally, the merge can-
 442 didates output by this algorithm will contain all possible pos-
 443 itive examples with a very limited number of negative ex-
 444 amples. After considering various thresholds, we find that
 445 $t_{low} = 240 \text{ nm}$ and $t_{high} = 600 \text{ nm}$ produce the best re-
 446 sults considering this objective.
 447

448 In our implementation we use nanometers for these
 449 thresholds and not voxels. Connectomics datasets often
 450 have lower sample resolutions in z . Using nanometers al-
 451 lows us to have uniform units across all of these datasets
 452 and calculate the thresholds in voxels at runtime. For ex-
 453 ample, the thresholds in voxels are $t_{low} = (40, 40, 8)$ and
 454 $t_{high} = (100, 100, 20)$ for the anisotropic Kasthuri dataset
 455 and $t_{low} = (24, 24, 24)$ and $t_{high} = (60, 60, 60)$ for the
 456 isotropic FlyEM dataset.
 457

458 4.4. Classifier Training

459 We use the left cylinder of the Kasthuri dataset for train-
 460 ing and validation. We train on 80% of the potential merge
 461 candidates for this volume. We validate the CNN classifier
 462 on the remaining 20% of candidates. We apply data aug-
 463 mentation to the generated examples to increase the size of
 464 the training datasets. We consider all rotations of 90 degrees
 465 along the xy -plane in addition to mirroring along the x and
 466 z axes. This produces 16 times more training data.
 467

468 We consider networks with varying input sizes, optimiz-
 469 ers, loss functions, filter sizes, learning rates, and activation
 470 functions. The supplemental material includes information
 471 on the experiments that determined these final parameters.
 472 Table 1 provides the parameters of the final network. There
 473 are 7,294,705 learnable parameters in our final architecture.
 474 All the parameters are randomly initialized following the
 475 Xavier uniform distribution [7]. Training concluded after
 476 34 epochs.
 477

478 4.5. Error Metric

479 We evaluate the performance of the different methods
 480 using the split version of variance of information (VI) [25].
 481 Given a ground truth labeling GT and our automatically re-
 482 constructed segmentation SG , over and undersegmentation
 483 are quantified by the conditional entropy $H(GT|SG)$ and
 484 $H(SG|GT)$, respectively. Since we are measuring the en-
 485 tropy between two clusterings, better VI scores are closer to
 the origin.

Parameters	Values	486
Loss Function	Mean Squared Error	487
Optimizer	SGD with Nesterov Momentum	488
Momentum	0.9	489
Initial Learning Rate	0.01	490
Decay Rate	$5 * 10^{-8}$	491
Activation	LeakyReLU ($\alpha = 0.001$)	492
Kernel Sizes	$3 \times 3 \times 3$	493
Filter Sizes	$16 \rightarrow 32 \rightarrow 64$	494
		495

496 Table 1: Training parameters.
 497

498 4.6. Variation of Information Results

500 In Fig. 5, we show the VI results of the pixel-based re-
 501 reconstructions of the Kasthuri and FlyEM data (Sec. 4.2) for
 502 varying thresholds of agglomeration (green). We use one of
 503 these segmentations (green circle) as our input dataset with
 504 an agglomeration threshold of 0.3 for all datasets. The re-
 505 sults from our method are shown in red for varying X HP:
 506 add parameters. We show comparisons to an oracle (blue)
 507 that correctly partitions the graph from our method based
 508 on ground truth.
 509

510 Our algorithm improves the accuracy of the reconstruction
 511 for every dataset, reducing the VI split score on aver-
 512 age by X% and only increasing the VI merge score by X%.
 513 Scores closer to the origin are better for this metric, and in
 514 every instance our results are below the green curve. We
 515 see significant improvements on the Kasthuri datasets (VI
 516 split reduction of X% and X% on the training and testing
 517 datasets respectively) and more modest improvements on
 518 the FlyEM datasets (reduction of X% and X%). This is be-
 519 cause the baseline segmentation algorithm for the isotropic
 520 FlyEM data (Sec. 4.2) performs much better, reducing the
 521 potential for improvements. It is well known that isotropic
 522 datasets are easier to segment using state-of-the-art region-
 523 based methods than anisotropic ones [31].
 524

525 Fig. 6 shows successful merges on the Kasthuri Vol. 2
 526 dataset. Several of these examples combine multiple con-
 527 secutive segments that span the volume. In the third exam-
 528 ple we correct the over-segmentation of a dendrite. Fig. 7
 529 shows some failure cases (red circles). In two of these ex-
 530 amples the algorithm correctly predicted several merges but
 531 made one error. In the third example (blue circle) a merge
 532 error in the initial segmentation propagated to our output.
 533 We now analyze how each major component of our method
 534 contributes to this final result.
 535

536 4.7. Graph Pruning Results

537 Table 2 shows the results of pruning the skeleton graph
 538 using the algorithm discussed in Sec. 3.1. This edge prun-
 539 ing is essential for the graph partitioning algorithm, which
 has a computational complexity dependence on the num-

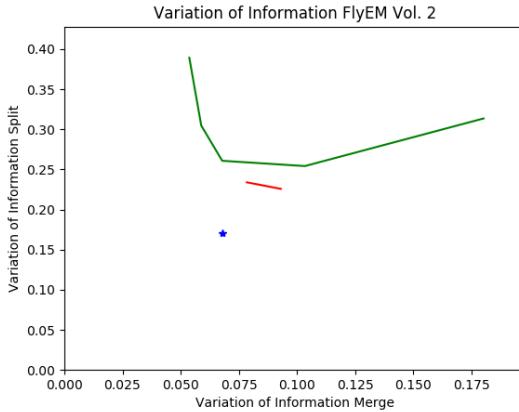
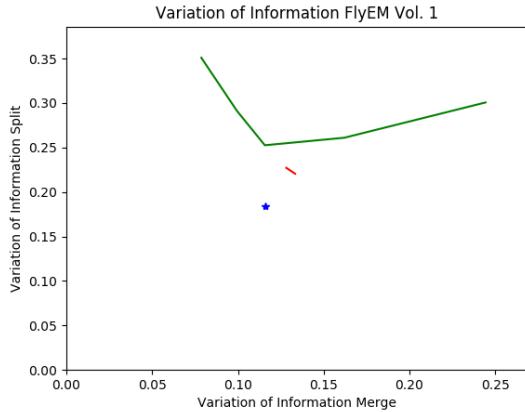
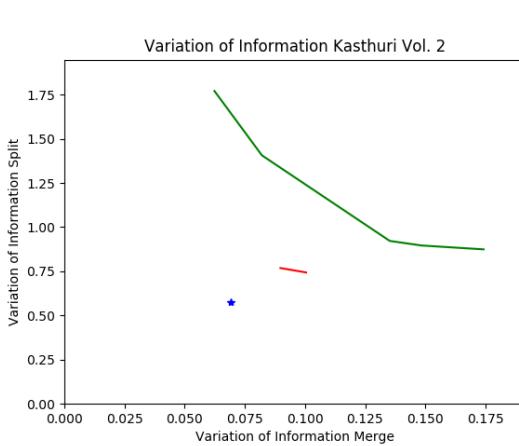
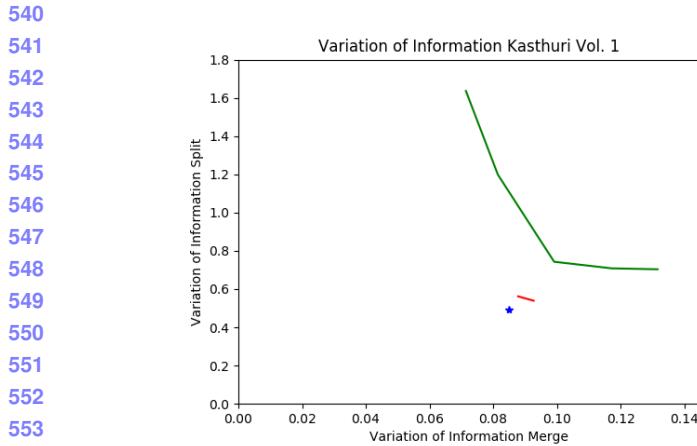


Figure 5: VI scores of our method (red) compared to the baseline segmentation (green) and an oracle (blue) that optimally partitions the graph based on ground truth.

Dataset	Baseline	After Pruning
Kasthuri Vol. 1	763 / 21,242	753 / 3,459
Kasthuri Vol. 2	1,010 / 26,073	904 / 4,327
FlyEM Vol. 1	269 / 14,875	262 / 946
FlyEM Vol. 2	270 / 16,808	285 / 768

Table 2: The results of our graph pruning approach compared to the baseline graph with all adjacent regions. We show the number of true merge locations (e.g., 763) compared to total number of edges in the graph (e.g., 21,242) for each case.

ber of edges. The baseline algorithm considers all adjacent regions for merging. Our method removes a significant portion of these candidates while maintaining a large number of the true merge locations (e.g., 753 compared to 763). Our pruning heuristic removes at least $6\times$ the number of edges on all datasets, achieving a maximum removal rate of $20\times$.

Equally important is the number of split errors that remain after pruning. These are the locations that we want

to merge to create a more accurate reconstructions. For every dataset, the number of true split errors remains constant before and after pruning. **HP: not sure what you mean** However, since our heuristic does not enforce an adjacency constraint of two regions when constructing edges in the graph, the difference does not indicate the number of examples excluded by pruning. Fig. 8 shows an example segment with a split error (green segment) that our algorithm missed. Of the successful examples in Fig. 6, the second and fourth groupings contain pairs of non-adjacent segments that were merged by our method. **HP: this paragraph needs more work to make it clearer**

4.8. CNN Classification Results

Figure 9 shows the receiver operating characteristic (ROC) curve of our CNN classifier for all datasets. Since our CNN only takes as input a region of the label volume we can train on anisotropic data and test on isotropic data. This provides a major benefit given the time-intensive task of manually generating ground truth for each dataset at various resolutions.

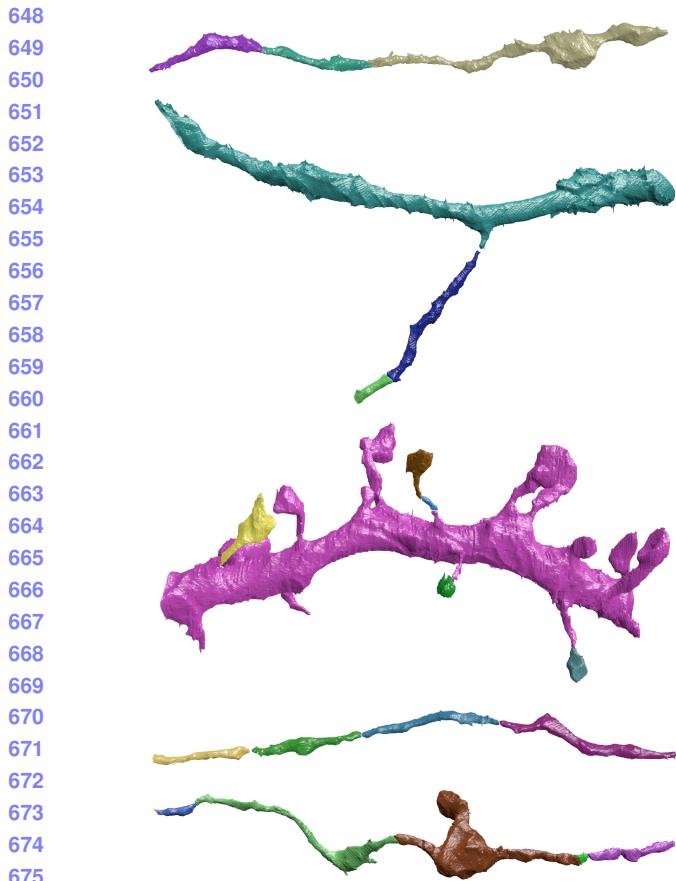


Figure 6: Segments of neurons that were correctly merged by our method.

As shown by the ROC curve, the test results on the Kasthuri data are better than the results for FlyEM. We believe this is in part because of the differences in the datasets (i.e., isotropy and xy resolution). To test this hypothesis, we also evaluate the performance of the FlyEM datasets when the network trains on FlyEM Vol. 1 and infers on FlyEM Vol. 2.¹ The blue dotted curve in the figure shows a slight performance increase in this case. However, the improvement is minor, which led us to use the CNN trained on the anisotropic data for the rest of our experiments.

4.9. Graph Optimization Results

The graph optimization strategy using multicut increases our accuracy over using just the CNN. Table 3 shows the changes in precision, recall, and accuracy for all four datasets compared to the CNN. The precision increases on each dataset, although the recall decreases on all but one of the datasets. Since it is more difficult to correct merge

¹Since the FlyEM datasets have significantly fewer examples, we initialize the network with the weights from the Kasthuri training and have an initial learning rate of 10^{-4} .

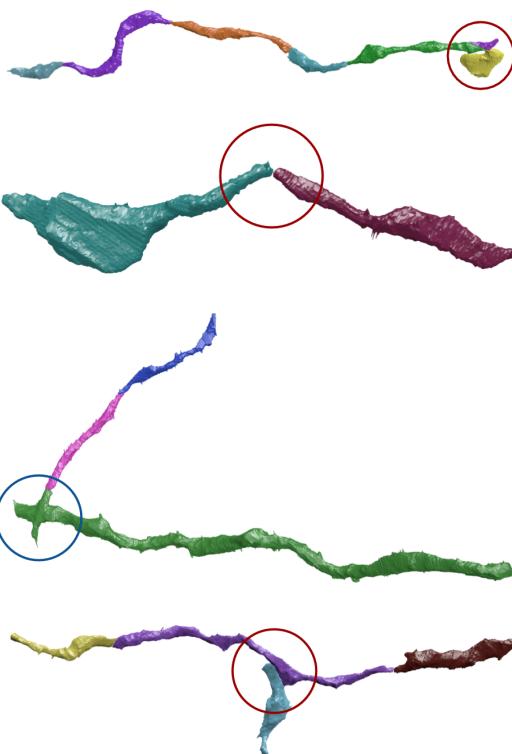


Figure 7: Circles indicate areas of wrong merges by our method (red) or by the initial pixel-based segmentation (blue).



Figure 8: Example merge candidates.

errors than split errors, it is often desirable to sacrifice recall for precision. Over the three testing datasets, applying a graph-based partitioning strategy reduced the number of merge errors by X, Y, and Z, respectively. HP: would be good to show the actual precision / recall numbers, too (at least in supplemental). maybe swap the ROC figure with a precision /recall figure?

5. Conclusions

We present a novel method for improved neuronal reconstruction in connectomics that extends existing pixel-based reconstruction strategies using skeletonized 3D networks.

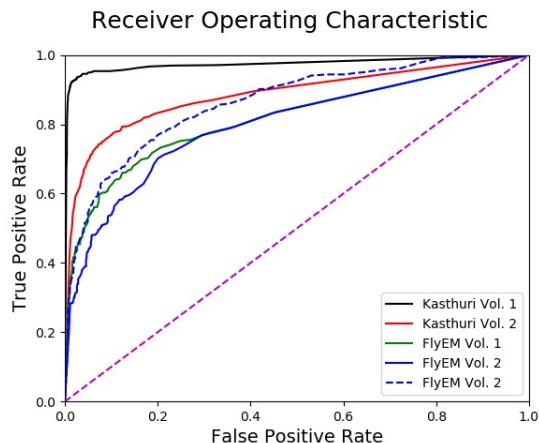


Figure 9: The receiver operating characteristic (ROC) curves of our CNN for all four datasets.

Dataset	Δ Precision	Δ Recall	Δ Accuracy
Kasthuri Training	+3.60%	-0.01%	+0.60%
Kasthuri Testing	+7.59%	-1.77%	+1.38%
FlyEM Vol. 1	+2.68%	+0.76%	+0.66%
FlyEM Vol. 2	+2.22%	-1.05%	+0.29%

Table 3: Precision, recall, and accuracy changes between CNN only and graph-optimized reconstructions for the training and three test datasets.

We show significant accuracy improvements on datasets from two different species. The main benefits of our approach are that it enforces domain-specific constraints at the global graph level while incorporating pixel-based classification information.

There is significant room for additional research and improvements. We can augment the graph with additional information from the image data, such as synaptic locations, cell morphology, locations of mitochondria, etc. This would allow us to enforce additional biological constraints during graph partitioning. For example, we could then enforce the constraint that a given segment only has post- or pre-synaptic connections. An augmented graph would also be helpful for splitting improperly merged segments by adding additional terms to the partitioning cost function. Finally, we believe that the benefits of top-down enhancements from graph optimization can extend beyond connectomics to other domains, such as medical image segmentation.

References

- [1] B. Andres, T. Kroeger, K. L. Briggman, W. Denk, N. Korogod, G. Knott, U. Koethe, and F. A. Hamprecht. Globally optimal closed-surface segmentation for connectomics. In

- European Conference on Computer Vision*, pages 778–791. Springer, 2012. 2
- [2] K. Briggman, W. Denk, S. Seung, M. N. Helmstaedter, and S. C. Turaga. Maximin affinity learning of image segmentation. In *Advances in Neural Information Processing Systems*, pages 1865–1873, 2009. 2
- [3] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman. Return of the devil in the details: Delving deep into convolutional nets. *arXiv preprint arXiv:1405.3531*, 2014. 4
- [4] D. Ciresan, A. Giusti, L. M. Gambardella, and J. Schmidhuber. Deep neural networks segment neuronal membranes in electron microscopy images. In *Advances in neural information processing systems*, pages 2843–2851, 2012. 2
- [5] E. D. Demaine, D. Emanuel, A. Fiat, and N. Immorlica. Correlation clustering in general weighted graphs. *Theoretical Computer Science*, 361(2-3):172–187, 2006. 2
- [6] K.-I. Funahashi. On the approximate realization of continuous mappings by neural networks. *Neural networks*, 2(3):183–192, 1989. 4
- [7] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 249–256, 2010. 5
- [8] D. Haehn, J. Hoffer, B. Matejek, A. Suissa-Peleg, A. K. Al-Awami, L. Kamentsky, F. Gonda, E. Meng, W. Zhang, R. Schalek, et al. Scalable interactive visualization for connectomics. In *Informatics*, volume 4, page 29. Multidisciplinary Digital Publishing Institute, 2017. 1
- [9] D. Haehn, V. Kaynig, J. Tompkin, J. W. Lichtman, and H. Pfister. Guided proofreading of automatic segmentations for connectomics. *arXiv preprint arXiv:1704.00848*, 2017. 2
- [10] D. Haehn, S. Knowles-Barley, M. Roberts, J. Beyer, N. Kasthuri, J. Lichtman, and H. Pfister. Design and evaluation of interactive proofreading tools for connectomics. *IEEE Transactions on Visualization and Computer Graphics (Proc. IEEE SciVis 2014)*, 20(12):2466–2475, 2014. 2
- [11] D. Haehn, S. Knowles-Barley, M. Roberts, J. Beyer, N. Kasthuri, J. W. Lichtman, and H. Pfister. Design and evaluation of interactive proofreading tools for connectomics. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):2466–2475, 2014. 1
- [12] D. G. C. Hildebrand, M. Cicconet, R. M. Torres, W. Choi, T. M. Quan, J. Moon, A. W. Wetzel, A. S. Champion, B. J. Graham, O. Randlett, et al. Whole-brain serial-section electron microscopy in larval zebrafish. *Nature*, 545(7654):345–349, 2017. 1
- [13] A. Horňáková, J.-H. Lange, and B. Andres. Analysis and optimization of graph decompositions by lifted multicut. In *International Conference on Machine Learning*, pages 1539–1548, 2017. 2
- [14] V. Jain, B. Bollmann, M. Richardson, D. Berger, M. Helmstädt, K. Briggman, W. Denk, J. Bowden, J. Mendenhall, W. Abraham, K. Harris, N. Kasthuri, K. Hayworth, R. Schalek, J. Tapia, J. Lichtman, and S. Seung. Boundary learning by optimization with topological constraints. In *Proc. IEEE CVPR 2010*, pages 2488–2495, 2010. 2

- 864 [15] Janelia Farm. Raveler. <https://openwiki.janelia.org/wiki/display/flyem/Raveler>, 2014. Accessed on 11/01/2016. 2 918
- 865 [16] M. Januszewski, J. Maitin-Shepard, P. Li, J. Kornfeld, S. M. Plaza, T. Parag, G. B. Huang, D. J. Olbris, M. A. Saunders, and P. K. Rivlin. Annotating synapses in large em datasets. *arXiv preprint arXiv:1409.1801*, 2014. 5 919
- 866 [17] J. H. Kappes, M. Speth, G. Reinelt, and C. Schnörr. Higher-order segmentation via multicut. *Computer Vision and Image Understanding*, 143:104–119, 2016. 2 920
- 867 [18] N. Kasthuri, K. J. Hayworth, D. R. Berger, R. L. Schalek, D. Rolnick, Y. Meirovitch, T. Parag, H. Pfister, V. Jain, J. W. Lichtman, E. S. Boyden, and N. Shavit. Morphological error detection in 3d segmentations. *arXiv preprint arXiv:1705.10882*, 2017. 2 921
- 868 [19] J. A. Conchello, S. Knowles-Barley, D. Lee, A. Vázquez-Reina, V. Kaynig, T. R. Jones, et al. Saturated reconstruction of a volume of neocortex. *Cell*, 162(3):648–661, 2015. 1, 4 922
- 869 [20] V. Kaynig, A. Vazquez-Reina, S. Knowles-Barley, M. Roberts, T. R. Jones, N. Kasthuri, E. Miller, J. Lichtman, and H. Pfister. Large-scale automatic reconstruction of neuronal processes from electron microscopy images. *Medical image analysis*, 22(1):77–88, 2015. 2 923
- 870 [21] M. Keuper, E. Levinkov, N. Bonneel, G. Lavoué, T. Brox, O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 234–241. Springer, 2015. 1, 2, 4 924
- 871 [22] M. Knowles-Barley, V. Kaynig, T. R. Jones, A. Wilson, M. Sato, I. Bitter, M. A. Bender, A. E. Kaufman, and J. Morgan, D. Lee, D. Berger, N. Kasthuri, J. W. Lichtman, and M. Nakajima. Teasar: Tree-structure extraction algorithm and B. Andres. Efficient decomposition of image and mesh graphs by lifted multicut. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1751–1759, 2015. 2, 4 925
- 872 [23] S. Knowles-Barley, V. Kaynig, T. R. Jones, A. Wilson, P. Schlegel, M. Costa, and G. S. Jefferis. Learning from J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on pattern analysis and machine annotation. arXiv preprint arXiv:1611.06973*, 2016. 1, 2 926
- 873 [24] S. Knowles-Barley, M. Roberts, N. Kasthuri, D. Lee, H. Pfister, and J. W. Lichtman. Mojo 2.0: Connectome annotation tool. *Frontiers in Neuroinformatics*, (60), 2013. 2 927
- 874 [25] K. Lee, A. Zlateski, V. Ashwin, and H. S. Seung. Recursive training of 2d-3d convolutional networks for neuronal boundary prediction. In *Advances in Neural Information Processing Systems*, pages 3573–3581, 2015. 1, 2 928
- 875 [26] A. L. Maas, A. Y. Hannun, and A. Y. Ng. Rectifier nonlinearities improve neural network acoustic models. In *Proc. NeurIPS*, volume 30, 2013. 4 929
- 876 [27] M. Meila. Comparing clusterings by the variation of information. In *COLT*, volume 3, pages 173–187. Springer, 2003. 5 930
- 877 [28] Y. Nesterov. A method of solving a convex programming problem with convergence rate $O(1/k^2)$. 4 931
- 878 [29] J. Nunez-Iglesias, R. Kennedy, S. M. Plaza, A. Chakraborty, S. Turaga, K. Briggman, M. Helmstaedter, W. Denk, and W. T. Katz. Graph-based active learning of agglomeration (gala): a python library to segment 2d and 3d neuroimages. *Frontiers in neuroinformatics*, 8, 2014. 1, 2, 3 932
- 879 [30] T. Parag. What properties are desirable from an electron microscopy segmentation algorithm. *arXiv preprint arXiv:1503.05430*, 2015. 2 933
- 880 [31] T. Parag, A. Chakraborty, S. Plaza, and L. Scheffer. A context-aware delayed agglomeration framework for electron microscopy segmentation. *PLOS ONE*, 10(5):1–19, 05 934
- 881 [32] T. Parag, F. Tschopp, W. Grisaitis, S. C. Turaga, X. Zhang, A. Zlateski and H. S. Seung. Image segmentation by size-dependent single linkage clustering of a watershed basin graph. *arXiv preprint arXiv:1505.00249*, 2015. 1, 2, 4 935
- 882 [33] T. Parag, F. Tschopp, W. Grisaitis, S. C. Turaga, X. Zhang, B. Matejek, L. Kamentsky, J. W. Lichtman, and H. Pfister. Anisotropic em segmentation by 3d affinity learning and agglomeration. *arXiv preprint arXiv:1707.08935*, 2017. 1, 2 936
- 883 [34] T. Parag, F. Tschopp, W. Grisaitis, S. C. Turaga, X. Zhang, B. Matejek, L. Kamentsky, J. W. Lichtman, and H. Pfister. Anisotropic em segmentation by 3d affinity learning and agglomeration. *arXiv preprint arXiv:1707.08935*, 2017. 1, 2 937
- 884 [35] T. Parag, F. Tschopp, W. Grisaitis, S. C. Turaga, X. Zhang, B. Matejek, L. Kamentsky, J. W. Lichtman, and H. Pfister. Anisotropic em segmentation by 3d affinity learning and agglomeration. *arXiv preprint arXiv:1707.08935*, 2017. 1, 2 938
- 885 [36] T. Parag, F. Tschopp, W. Grisaitis, S. C. Turaga, X. Zhang, B. Matejek, L. Kamentsky, J. W. Lichtman, and H. Pfister. Anisotropic em segmentation by 3d affinity learning and agglomeration. *arXiv preprint arXiv:1707.08935*, 2017. 1, 2 939
- 886 [37] T. Parag, F. Tschopp, W. Grisaitis, S. C. Turaga, X. Zhang, B. Matejek, L. Kamentsky, J. W. Lichtman, and H. Pfister. Anisotropic em segmentation by 3d affinity learning and agglomeration. *arXiv preprint arXiv:1707.08935*, 2017. 1, 2 940
- 887 [38] T. Parag, F. Tschopp, W. Grisaitis, S. C. Turaga, X. Zhang, B. Matejek, L. Kamentsky, J. W. Lichtman, and H. Pfister. Anisotropic em segmentation by 3d affinity learning and agglomeration. *arXiv preprint arXiv:1707.08935*, 2017. 1, 2 941
- 888 [39] T. Parag, F. Tschopp, W. Grisaitis, S. C. Turaga, X. Zhang, B. Matejek, L. Kamentsky, J. W. Lichtman, and H. Pfister. Anisotropic em segmentation by 3d affinity learning and agglomeration. *arXiv preprint arXiv:1707.08935*, 2017. 1, 2 942
- 889 [40] T. Parag, F. Tschopp, W. Grisaitis, S. C. Turaga, X. Zhang, B. Matejek, L. Kamentsky, J. W. Lichtman, and H. Pfister. Anisotropic em segmentation by 3d affinity learning and agglomeration. *arXiv preprint arXiv:1707.08935*, 2017. 1, 2 943
- 890 [41] T. Parag, F. Tschopp, W. Grisaitis, S. C. Turaga, X. Zhang, B. Matejek, L. Kamentsky, J. W. Lichtman, and H. Pfister. Anisotropic em segmentation by 3d affinity learning and agglomeration. *arXiv preprint arXiv:1707.08935*, 2017. 1, 2 944
- 891 [42] T. Parag, F. Tschopp, W. Grisaitis, S. C. Turaga, X. Zhang, B. Matejek, L. Kamentsky, J. W. Lichtman, and H. Pfister. Anisotropic em segmentation by 3d affinity learning and agglomeration. *arXiv preprint arXiv:1707.08935*, 2017. 1, 2 945
- 892 [43] T. Parag, F. Tschopp, W. Grisaitis, S. C. Turaga, X. Zhang, B. Matejek, L. Kamentsky, J. W. Lichtman, and H. Pfister. Anisotropic em segmentation by 3d affinity learning and agglomeration. *arXiv preprint arXiv:1707.08935*, 2017. 1, 2 946
- 893 [44] T. Parag, F. Tschopp, W. Grisaitis, S. C. Turaga, X. Zhang, B. Matejek, L. Kamentsky, J. W. Lichtman, and H. Pfister. Anisotropic em segmentation by 3d affinity learning and agglomeration. *arXiv preprint arXiv:1707.08935*, 2017. 1, 2 947
- 894 [45] T. Parag, F. Tschopp, W. Grisaitis, S. C. Turaga, X. Zhang, B. Matejek, L. Kamentsky, J. W. Lichtman, and H. Pfister. Anisotropic em segmentation by 3d affinity learning and agglomeration. *arXiv preprint arXiv:1707.08935*, 2017. 1, 2 948
- 895 [46] T. Parag, F. Tschopp, W. Grisaitis, S. C. Turaga, X. Zhang, B. Matejek, L. Kamentsky, J. W. Lichtman, and H. Pfister. Anisotropic em segmentation by 3d affinity learning and agglomeration. *arXiv preprint arXiv:1707.08935*, 2017. 1, 2 949
- 896 [47] T. Parag, F. Tschopp, W. Grisaitis, S. C. Turaga, X. Zhang, B. Matejek, L. Kamentsky, J. W. Lichtman, and H. Pfister. Anisotropic em segmentation by 3d affinity learning and agglomeration. *arXiv preprint arXiv:1707.08935*, 2017. 1, 2 950
- 897 [48] T. Parag, F. Tschopp, W. Grisaitis, S. C. Turaga, X. Zhang, B. Matejek, L. Kamentsky, J. W. Lichtman, and H. Pfister. Anisotropic em segmentation by 3d affinity learning and agglomeration. *arXiv preprint arXiv:1707.08935*, 2017. 1, 2 951
- 898 [49] T. Parag, F. Tschopp, W. Grisaitis, S. C. Turaga, X. Zhang, B. Matejek, L. Kamentsky, J. W. Lichtman, and H. Pfister. Anisotropic em segmentation by 3d affinity learning and agglomeration. *arXiv preprint arXiv:1707.08935*, 2017. 1, 2 952
- 899 [50] T. Parag, F. Tschopp, W. Grisaitis, S. C. Turaga, X. Zhang, B. Matejek, L. Kamentsky, J. W. Lichtman, and H. Pfister. Anisotropic em segmentation by 3d affinity learning and agglomeration. *arXiv preprint arXiv:1707.08935*, 2017. 1, 2 953
- 900 [51] T. Parag, F. Tschopp, W. Grisaitis, S. C. Turaga, X. Zhang, B. Matejek, L. Kamentsky, J. W. Lichtman, and H. Pfister. Anisotropic em segmentation by 3d affinity learning and agglomeration. *arXiv preprint arXiv:1707.08935*, 2017. 1, 2 954
- 901 [52] T. Parag, F. Tschopp, W. Grisaitis, S. C. Turaga, X. Zhang, B. Matejek, L. Kamentsky, J. W. Lichtman, and H. Pfister. Anisotropic em segmentation by 3d affinity learning and agglomeration. *arXiv preprint arXiv:1707.08935*, 2017. 1, 2 955
- 902 [53] T. Parag, F. Tschopp, W. Grisaitis, S. C. Turaga, X. Zhang, B. Matejek, L. Kamentsky, J. W. Lichtman, and H. Pfister. Anisotropic em segmentation by 3d affinity learning and agglomeration. *arXiv preprint arXiv:1707.08935*, 2017. 1, 2 956
- 903 [54] T. Parag, F. Tschopp, W. Grisaitis, S. C. Turaga, X. Zhang, B. Matejek, L. Kamentsky, J. W. Lichtman, and H. Pfister. Anisotropic em segmentation by 3d affinity learning and agglomeration. *arXiv preprint arXiv:1707.08935*, 2017. 1, 2 957
- 904 [55] T. Parag, F. Tschopp, W. Grisaitis, S. C. Turaga, X. Zhang, B. Matejek, L. Kamentsky, J. W. Lichtman, and H. Pfister. Anisotropic em segmentation by 3d affinity learning and agglomeration. *arXiv preprint arXiv:1707.08935*, 2017. 1, 2 958
- 905 [56] T. Parag, F. Tschopp, W. Grisaitis, S. C. Turaga, X. Zhang, B. Matejek, L. Kamentsky, J. W. Lichtman, and H. Pfister. Anisotropic em segmentation by 3d affinity learning and agglomeration. *arXiv preprint arXiv:1707.08935*, 2017. 1, 2 959
- 906 [57] T. Parag, F. Tschopp, W. Grisaitis, S. C. Turaga, X. Zhang, B. Matejek, L. Kamentsky, J. W. Lichtman, and H. Pfister. Anisotropic em segmentation by 3d affinity learning and agglomeration. *arXiv preprint arXiv:1707.08935*, 2017. 1, 2 960
- 907 [58] T. Parag, F. Tschopp, W. Grisaitis, S. C. Turaga, X. Zhang, B. Matejek, L. Kamentsky, J. W. Lichtman, and H. Pfister. Anisotropic em segmentation by 3d affinity learning and agglomeration. *arXiv preprint arXiv:1707.08935*, 2017. 1, 2 961
- 908 [59] T. Parag, F. Tschopp, W. Grisaitis, S. C. Turaga, X. Zhang, B. Matejek, L. Kamentsky, J. W. Lichtman, and H. Pfister. Anisotropic em segmentation by 3d affinity learning and agglomeration. *arXiv preprint arXiv:1707.08935*, 2017. 1, 2 962
- 909 [60] T. Parag, F. Tschopp, W. Grisaitis, S. C. Turaga, X. Zhang, B. Matejek, L. Kamentsky, J. W. Lichtman, and H. Pfister. Anisotropic em segmentation by 3d affinity learning and agglomeration. *arXiv preprint arXiv:1707.08935*, 2017. 1, 2 963
- 910 [61] T. Parag, F. Tschopp, W. Grisaitis, S. C. Turaga, X. Zhang, B. Matejek, L. Kamentsky, J. W. Lichtman, and H. Pfister. Anisotropic em segmentation by 3d affinity learning and agglomeration. *arXiv preprint arXiv:1707.08935*, 2017. 1, 2 964
- 911 [62] T. Parag, F. Tschopp, W. Grisaitis, S. C. Turaga, X. Zhang, B. Matejek, L. Kamentsky, J. W. Lichtman, and H. Pfister. Anisotropic em segmentation by 3d affinity learning and agglomeration. *arXiv preprint arXiv:1707.08935*, 2017. 1, 2 965
- 912 [63] T. Parag, F. Tschopp, W. Grisaitis, S. C. Turaga, X. Zhang, B. Matejek, L. Kamentsky, J. W. Lichtman, and H. Pfister. Anisotropic em segmentation by 3d affinity learning and agglomeration. *arXiv preprint arXiv:1707.08935*, 2017. 1, 2 966
- 913 [64] T. Parag, F. Tschopp, W. Grisaitis, S. C. Turaga, X. Zhang, B. Matejek, L. Kamentsky, J. W. Lichtman, and H. Pfister. Anisotropic em segmentation by 3d affinity learning and agglomeration. *arXiv preprint arXiv:1707.08935*, 2017. 1, 2 967
- 914 [65] T. Parag, F. Tschopp, W. Grisaitis, S. C. Turaga, X. Zhang, B. Matejek, L. Kamentsky, J. W. Lichtman, and H. Pfister. Anisotropic em segmentation by 3d affinity learning and agglomeration. *arXiv preprint arXiv:1707.08935*, 2017. 1, 2 968
- 915 [66] T. Parag, F. Tschopp, W. Grisaitis, S. C. Turaga, X. Zhang, B. Matejek, L. Kamentsky, J. W. Lichtman, and H. Pfister. Anisotropic em segmentation by 3d affinity learning and agglomeration. *arXiv preprint arXiv:1707.08935*, 2017. 1, 2 969
- 916 [67] T. Parag, F. Tschopp, W. Grisaitis, S. C. Turaga, X. Zhang, B. Matejek, L. Kamentsky, J. W. Lichtman, and H. Pfister. Anisotropic em segmentation by 3d affinity learning and agglomeration. *arXiv preprint arXiv:1707.08935*, 2017. 1, 2 970
- 917 [68] T. Parag, F. Tschopp, W. Grisaitis, S. C. Turaga, X. Zhang, B. Matejek, L. Kamentsky, J. W. Lichtman, and H. Pfister. Anisotropic em segmentation by 3d affinity learning and agglomeration. *arXiv preprint arXiv:1707.08935*, 2017. 1, 2 971