Guided Proofreading of Automatic Segmentations in Connectomics

Anonymous CVPR submission

Paper ID 0947

Abstract

Automatic cell image segmentation methods in connectomics produce merge and split errors, which require correction through proofreading. Previous research has identified the visual search for these errors as the bottleneck in interactive proofreading. To aid error correction, we develop two classifiers to recommend candidate merge and split errors and their corrections to the user. These classifiers are informed by training a convolutional neural network with known errors in automatic segmentations against expert-labeled ground truth. Our classifiers detect potentially-erroneous regions by considering a large context region around a segmentation boundary. Corrections can then be performed as yes/no decisions resulting in faster correction times than previous methods. We evaluate our approach on connectomics datasets of different species and compare correction performance of novice and expert users against different existing systems. We report significant improvements on pure automatic and pure manual proofreading.

1. Introduction

In connectomics, neuroscientists annotate neurons and their connectivity within 3D volumes to gain insight into the functional structure of the brain. Rapid progress in automatic sample preparation and electron microscopy (EM) acquisition techniques has made it possible to image large volumes of brain tissue at $\approx 4\,nm$ per pixel to identify cells, synapses, and vesicles. For $40\,nm$ thick sections, a $1\,mm^3$ volume of brain contains 10^{15} voxels, or 1 petabyte of data. With so much data, manual annotation is infeasible, and automatic annotation methods are needed [12, 22, 25, 17].

Automatic annotation by segmentation and classification of brain tissue is challenging [1]. The state of the art uses supervised learning with convolutional neural networks [8], or potentially even unsupervised learning [6]. Typically, cell membranes are detected in 2D images, and the resulting region segmentation is grouped into geometrically-consistent cells across registered sections. Cells may also be segmented across registered sections in 3D directly. Using dynamic programming techniques [23] and a GPU cluster, these classifiers can segment ≈ 1 terabyte of data per hour [15]. This is sufficient to keep up with the 2D data capture process on state-of-the-art electron microscopes (though 3D registration is still an expensive offline operation).

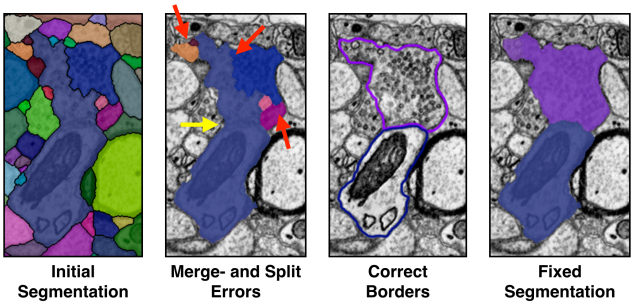

Figure 1. The most common proofreading corrections are fixing split errors (red arrows) and merge errors (yellow arrow). A fixed segmentation matches the cell borders.

All automatic methods make errors, and we are left with large data which needs *proofreading* by humans. This crucial task serves two purposes: 1) to correct errors in the segmentation, and 2) to provide a large body of labeled data to train better automatic segmentation methods. Recent proofreading tools provide intuitive user interfaces to browse segmentation data in 2D and 3D and to identify and manually correct errors [30, 13, 19, 10]. Many kinds of errors exist, such as inaccurate boundaries, but the most common are *split errors*, where a single segment is labeled as two, and *merge errors*, where two segments are labeled as one (Fig. 1). With user interaction, split errors can be joined, and the missing boundary in a merge error can be defined with manually-seeded watersheds [10]. However, even with semi-automatic correction tools, the visual inspection to find errors in the first place takes the majority of the time.

Our goal is to add automatic detection of split and merge errors to proofreading tools. Instead of the user visually inspecting the whole data volume carefully to spot errors, we design automatic classifiers that detect split and merge errors in 2D segmentations. Then, a proofreading tool can

recommend regions with a high probability of an error to the user, and suggest corrections to accept or reject. We call this process *guided proofreading* (GP).

As our main contribution, we introduce classifiers to detect merge- and split errors based on a convolutional neural network (CNN). We believe that this is the first time that deep learning is applied to the task of proofreading — especially in the field of connectomics. Our classifiers work on top of any existing automatic segmentation method to find potential errors and suggest corrections. Given a membrane segmentation from a fast automatic method, our classifiers operate on the boundaries of whole cell regions. Compared to techniques that must analyze every input pixel, we reduce the data analysis to the boundaries only. Explicitly, we train a CNN to detect only split errors. We then reuse the same network to also detect merge errors by generating possible boundaries within a cell and inverting the split error score. Correction suggestion for both types of errors is then trivial and our technique therefor reduces the proofreading operation to simple yes/no decisions.

We further propose a greedy algorithm to perform proofreading automatically and measure performance as the adapted Rand error which is a common metric for segmentation comparison [31]. Our system is integrated into an existing proofreading workflow for large connectomics data. For this, we also explore an active label suggestion approach in addition to the ranking obtained by guided proofreading. We quantitatively validate automatic and human-driven variations of guided proofreading on five different real-world connectomics datasets of mouse as well as fruit-fly (drosophila) brain. To study the performance of novice and expert proofreaders, we perform a between-subjects experiment and ask participants to proofread a publicly available dataset. For comparison, we establish two baselines: a recently published fully interactive proofreading tool named *Dojo* by Haehn *et al*. [10] and semi-automatic *focused proofreading* (FP) approach by Plaza [27]. In all experiments, we significantly outperform both interactive proofreading as well as Plaza's method. As a consequence, we are able to provide tools to proofread segmentations more efficiently, and so better tackle large volumes of connectomics imagery.

2. Related Work

Proofreading is performed on top of an existing automatic segmentation. First, this section gives an overview of state-of-the-art automatic segmentation methods. Then, we discuss interactive proofreading tools and recent developments in computer-aided proofreading systems.

Automatic Segmentation. Multi-terabyte EM brain volumes require automatic segmentation [12, 22, 24, 25], but can be hard to classify due to ambiguous intercellular space: the 2013 IEEE ISBI neurites 3D segmentation challenge [1] showed that existing algorithms which learn from expert-segmented training data still exhibit high error rates.

NeuroProof [2] tries to decrease error rates with interactive learning of agglomeration of over-segmentations of images, based on a random forest classifier. Vazquez-Reina *et al*. [33] propose automatic 3D segmentation by taking whole EM volumes into account rather than a per section approach, then solving a fusion problem with a global context. Kaynig *et al*. [16] propose a random forest classifier coupled with an anisotropic smoothing prior in a conditional random field framework with 3D segment fusion. It is also possible to learn segmentation classification features directly from images with CNNs. Ronneberger *et al*. [28] use a contracting/expanding CNN path architecture to enable precise boundary localization with small amounts of training data. Lee *et al*. [20] recursively train very deep networks with 2D and 3D filters to detect boundaries. Bogovic *et al*. [6] learn 3D features, and show even that unsupervised learning can produce better features than hand-designs. Our work was inspired by this paper and we extend the features reported by Bogovic *et al*. for our guided proofreading classifiers as described in section 3. These approaches make good progress; however, in general, proofreading is required to improve them through generating more ground-truth segmentations.

Interactive Proofreading. While proofreading is very time consuming, it is fairly easy for humans to perform corrections through splitting and merging segments. One way to perform such corrections is by using expert tools such as Raveler introduced by Chklovskii *et al*. [7, 13]. This software offers many parameters for tweaking the proofreading process. Created in 2010, Raveler is still used today by professional full-time proofreaders and many similar systems exist as stand-alone products or plugins to existing visualization system, *e.g*. V3D [26] or AVIZO [30]. In contrast to these expert tools, recent works attack the problem of proofreading massive datasets by novices through crowdsourcing [29, 4, 9]. A very popular platform is EyeWire presented by Kim *et al*. [18]. EyeWire is set up as an online game and participants earn virtual rewards for merging oversegmented labeling to reconstruct the retina cells. A range of proofreading tools exist in-between expert systems and online games such as Mojo and *Dojo* developed by Haehn *et al*. [10, 3]. Mojo provides a simple scribble interface for error correction, and Dojo extends this for distributed proofreading via a minimalistic web-based user interface. The authors define requirements for general proofreading tools, and then evaluate the accuracy and speed of Raveler, Mojo, and Dojo through a quantitative user study (Sec. 3 and 4) [10]. In this paper, we use the Dojo system as a baseline for interactive proofreading and extend the experiment reported by Haehn *et al*., where Raveler, Mojo, and Dojo are compared in terms of accuracy and speed.

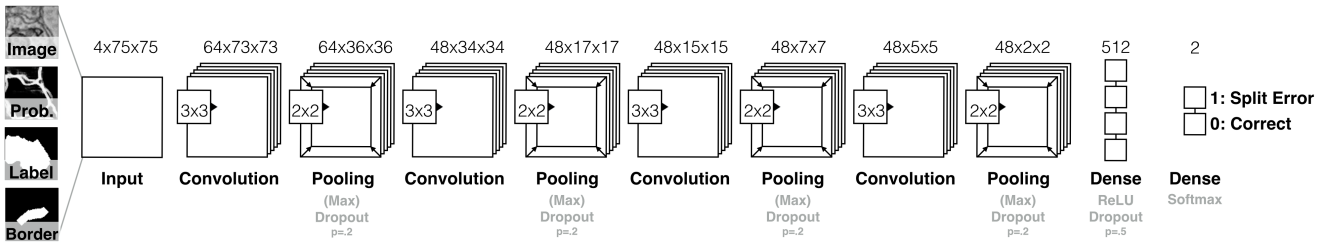

Figure 2. We build the guided proofreading classifiers using a traditional CNN architecture. The network is based on four convolutional layers, each followed by max pooling as well as dropout regularization. The 4-channel input patches are rated as either correct splits or as split errors.

All interactive proofreading solutions require the user to find potential errors manually which takes the majority of time [26, 10]. Recent works propose computer-aided proofreading systems which help with this visual search task.

Computer-aided Proofreading. To reduce the time spent looking for errors, Plaza proposed *focused proofreading* (FP) [27]. His approach finds split errors by analyzing segment size ratios across slices and then offers yes/no questions to correct these errors. Plaza reports that additional processing beyond FP is required to find merge errors. His method is freely available as open source software and is integrated into Raveler. This makes it feasible for us to use FP as a baseline for evaluating guided proofreading as described in section 4. A similar approach was published by Karimov *et al*. as guided volume editing [14]. Measuring differences in histogram distributions in image data enables to find potential split and merge errors in the corresponding segmentation. For merge errors, the authors generate possible boundaries using watershed which inspired our approach as described in section 3. Guided volume editing was designed to let expert users correct labeled computer-tomography datasets by performing several interactions per correction. While focused proofreading and guided volume editing both use a heuristical approach to analyze the image data, Uzunbas *et al*. showed that potential labeling errors can be found by considering the merge tree of an automatic segmentation method [32]. The authors track uncertainty throughout the automatic labeling by training a conditional random field. This method is really a segmentation technique but it is possible to use the uncertainty information to present potential regions for proofreading. Uzunbas' paper describes a great idea but requires further work to overcome the requirement of isotropic volumes, a property not given for most connectomics datasets. Our approach, guided proofreading, works on isotropic as well as anisotropic data, and finds merge and split errors.

3. Method

We first describe our classifier for detecting split errors which is based on a convolutional neural network (CNN). We detail the CNN architecture, input features and the training method. We then describe how the same classifier can be used to detect merge errors and how we create potential corrections. The classifiers are integrated into an existing proofreading workflow as reported after. Finally, we explore an active label suggestion method which reorders the ranking obtained by our classifiers and maximizes the information gain provided by each potential correction.

3.1. Split Error Detection

We build a split error classifier with output p using a convolutional neural network (CNN) to check whether an edge within an existing automatic segmentation is valid ($p = 0$) or not ($p = 1$). Rather than analyzing every input pixel, the classifier operates only on segment boundaries which requires less pixel context and is faster. Our approach was inspired by Bogovic *et al*. [6] but works with 2D slices rather than 3D volumes. This enables proofreading prior or in parallel to an expensive alignment of individual EM images.

Convolutional Neural Network Architecture. Split error detection of a given boundary is really a binary classification task since the boundary is either correct or erroneous. However, in reality the score p is between 0 and 1. The classification complexity arises from hundreds of different cell types in connectomics data rather than from the classification decision. Intuitively, this yields a wider (meaning more filters) rather than a deeper (meaning more layers) architecture. We explored different architectural configurations - including residual networks [11] - by performing a brute force parameter search and comparing precision and recall (see supplementary materials). Our final CNN configuration for split error detection is composed of four convolutional layers, each followed by max pooling as well as dropout regularization to prevent overfitting due to limited training data. Fig. 2 shows the CNN architecture for split error detection.

Classifier Inputs. To train the CNN for split error detection, we take boundary context information into consideration for the decision making process. For this,

we grab a $75x75$ pixel patch at the center of an existing boundary. This covers approximately 80% of all boundaries in real-world connectomics data with nanometer resolution. If the boundary edge is not fully covered, we sample up to 10 non-overlapping patches along the boundary and combine the resulting score by weighted averaging based on boundary length coverage per patch. In their paper, Bogovich *et al.* propose to use grayscale image data, corresponding boundary probabilities, and a single binary mask combining the two neighboring labels as features for their recursive neural network [6]. We are building on this set of features as inputs for our CNN and create a stacked pixel patch. However, we observed that the boundary probability information generated from EM images is often misleading due to noise or artefacts in the data. This can result in merge errors within the automatic segmentation. To better direct our classifier to train on the true boundary edge, we extract the border between two segments. We then dilate this border by 5 pixels to consider slight edge ambiguities and use this additional binary mask as another feature to create a 4-channel input patch. Fig. 3 shows examples of correct and erroneous feature patches and their corresponding automatic segmentation and ground truth.

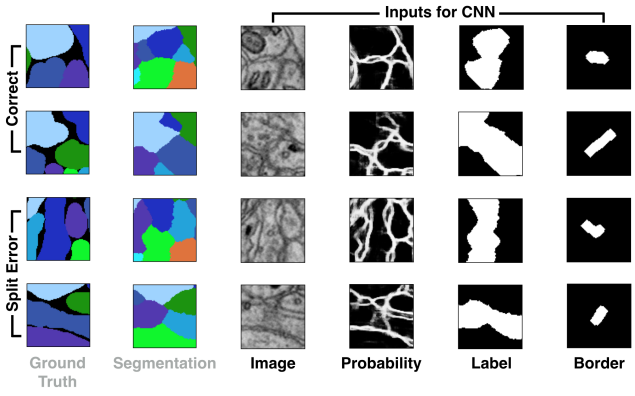

Figure 3. Example inputs for learning correct splits and split errors as reflected in the segmentation relative to the ground truth. Image, membrane probabilities, merged binary labels, and a dilated border mask are combined to 4-channel input patches.

Training. To initially train our network, we use the blue 3-cylinder mouse cortex volume of Kasthuri *et al.* [15] ($2048 \times 2048 \times 300$ voxels). The tissue is dense mammalian neuropil from layers 4 and 5 of the S1 primary somatosensory cortex of a healthy mouse. The resolution of our dataset is $3\,nm$ per pixel, and the section thickness is $30\,nm$. The image data and a manually-labeled expert segmentation is publicly available as ground truth for the entire dataset[1]. We use the first 250 sections of the data for training and validation and the last 50 for testing. We use a state-of-the-art method to create a dense automatic segmentation of the data. To generate training data, we identify correct regions and split errors in the automatic segmentation by intersection with ground truth regions. This is required since extracellular space is not labeled in the ground truth but in our dense automatic segmentation. From these regions, we sample 120,000 correct and 120,000 split error patches with 4-channels as described above. The patches are normalized and to further augment our training data, we rotate patches within each mini-batch by $k * 90$ degrees with randomly chosen k. The training parameters such as filter size, number of filters, learning rate, and momentum are the result of intuition and experience, studying recent machine learning research as well as a brute force parameter search within a limited range (see supplementary material). The final parameters and training results are listed in table 1. For baseline comparison, we also list the parameters and training results of focused proofreading in this table but elaborate on these further in section 4. Our CNN configuration results in approximately 170,000 learnable parameters. We assume that training has converged if the validation loss does not decrease for 50 epochs.

	cost [m]	Val. loss	Val. acc.	Test acc.	Prec./Recall	F1 Score
Guided Proofreading Filter size: 3x3 No. Filters 1: 64 No. Filters 2-4: 48 Dense units: 512 Learning rate: 0.03-0.00001 Momentum: 0.9-0.999 Mini-Batchsize: 128	383	0.0845	0.969	0.94	0.94/0.94	0.94
Focused Proofreading Iterations: 3 Learning strategy: 2 Mito agglomeration: Off Threshold: 0.2	43	?	?	0.839	?/?	?

Table 1. Training parameters, cost and results of our guided proofreading classifier versus focused proofreading by Plaza [27]. Both methods were trained on the same mouse brain dataset using the same hardware (Tesla K40 graphics card). While the training of our classifier is more expensive, testing accuracy is superior.

For performance comparison on data of a different species, in particular on fruitfly brain (drosophila), we retrain our network. The training procedure is according to our initial training and network architecture as well as parameters are not changed. We further elaborate on the drosophila datasets in section 4. Fig. 4 displays receiver operating characteristics (ROC) for guided proofreading trained on mouse and drosophila data, as well as our comparison baseline focused proofreading trained on these datasets respectively.

3.2. Merge Error Detection

Identification and correction of merge errors is more challenging, because we must look inside segmentation regions for missing or incomplete boundaries and then propose the correct boundary. However, we can reuse the same

[1]The Kasthuri 3-cylinder mouse cortex volume is available at https://software.rc.fas.harvard.edu/lichtman/vast/

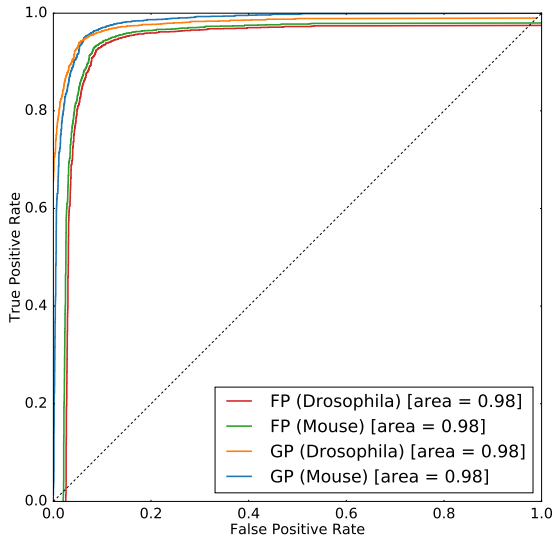

Figure 4. ROC performance of guided proofreading (GP) and focused proofreading (FP) trained separately on mouse and drosophila brain images. The area under the curve indicates better performance for GP.

trained CNN for this task. Similar to guided volume editing by Karimov *et al*. [14] we generate potential borders within a segment. For each segmentation label, we dilate the label by 20 pixel and generate 30 potential boundaries through the region by randomly placing watershed seed points at opposite sides of the label boundary. For watershed, we use the inverted gray scale EM image as features. This yields 30 corresponding splits. Dilation of the segment prior to watershed is motivated by our observation that the generated split then actually hogs the real membrane boundary. These boundaries are then individually rated using our split error classifier. For this, we invert the probability score meaning that a correct split (previously encoded as $p = 0$) is most likely a candidate for a merge error (now encoded as $p = 1$). In other words, if a generated boundary is ranked as correct, it probably should be in the segmentation. Fig. 5 illustrates this procedure.

3.3. Error Correction

We use the proposed classifiers in combination to perform corrections of split and merge errors in automatic segmentations. For this, we first perform merge error detection for all existing segments in a data set and store the inverted ranking $1 - p$. We then sort the rankings and loop through all of them in greedy fashion, starting with the most likely error. If the inverted score $1 - p$ of a merge error is higher than a threshold p_t, we mark this merge error for correction. The merge error detection yields the potential new boundary and we can modify the segmentation data to create a new segment accordingly. Depending on the variation of guided proofreading as described in section 4, we either

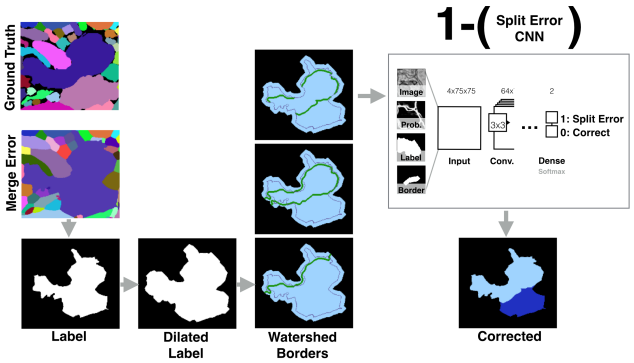

Figure 5. Merge errors are identified by generating randomly seeded watershed borders within a dilated label segment. These borders then are individually rated using the split error CNN by inverting the probability score. This way, a confident rating for a correct split most likely indicates the missing border of the merge error and can be used for correcting the labeling.

perform the correction directly (automatic GP) or we have a user accept or reject the correction (simulated GP or human novice/expert). Once all merge errors are corrected, we perform split error detection. For this, we perform split error detection and store the ranking p for all existing segments in the for merge errors corrected segmentation. We then sort the rankings and again loop through all of them - the most likely error first. If the score p is higher than a threshold p_t, we mark the split error for correction. Potential split errors are identified at the border of two segments. This reduces the correction to merging the segments and is therefore trivial. Similarly to merge errors, we either perform the correction directly or present a user with a yes/no decision. Our experiments have shown that the threshold p_t is the same for merge and split errors which makes sense for a balanced classifier. The only exception is when a user drives the correction process: we then set p_t for split errors to 0 to let the user inspect every possible split error. Inspecting all merge errors is not possible for users due to the sheer amount of generated borders.

3.4. Application

Guided proofreading is integrated into an existing workflow for large connectomics data. The GP system is web-based and is designed with a minimalistic user interface showing three components. First, we show the outline of the current labeling of a cell boundary and its proposed correction on top of the EM image data. For the user, it is not possible to distinguish the current labeling and the proposed correction to avoid selection bias. Second, we show a solid overlay of the current and proposed labeling. And finally, to provide context, we show a larger area of the EM image where the potentially erroneous region is highlighted. User interaction is simple and involves one mouse click on either the current labeling or the correction. After interaction, the

next potential error is shown. We provide a screenshot of the application as part of the supplementary material.

3.5. Active Label Suggestion

In an interactive setting, one way to present patches to the user for proofreading is to order them by the confidence probability of the GP classifier. However, in an active learning setting, where the network is retrained repeatedly on new label evidence, this approach is less likely to decrease segmentation error as, with the new labels, we are only reinforcing what the network already has a high confidence in. Instead, we apply active label suggestion to guide the user into labeling patches which will be more informative to retraining, and so overall decrease VI faster within the proofreading cycle of label $\rightarrow$ train $\rightarrow$ label. For each patch, we remove the softmax classification layer and look at the activation weights associated with the last dense layer. These become a high-dimensional feature vector. Then, we adapt Anon *et al*. [5] to provide label suggestions based on features from the learned CNN, which is based on maximizing the average information gain provided by a candidate patch to label. A second consideration is that each patch labeled by the user provides evidence to other patches, e.g., correcting a split error redefines an entire boundary, from which multiple candidate patch labelings could have been drawn. As such, when the user labels a patch, we consider all 'knock-on' effect patches as also being labeled, and feed these into the active label suggestion system similarly. In section 4, we report the difference in performance from using active label suggestion rather than confidence ordering when presenting patches to the user. These results are without retraining the network after new labelings: this should improve results, but would have to be batched to reduce computational load; hence, we leave this for future work.

4. Evaluation

We evaluate guided proofreading (GP) on multiple real-world connectomics datasets of different species. In particular, we evaluate GP on two datasets of mouse brain and three datasets of fruitfly brain (drosophila). For comparison, we choose the fully interactive proofreading software *Dojo* by Haehn *et al*. [10] as well as the aided proofreading framework *focused proofreading* (FP) by Plaza [27]. We first describe the evalation on mouse brain data and then the evaluation on drosophila brain.

4.1. Mouse Brain

Mouse brain is a common target for connectomics research because the structural proportions are similar to human brains [21]. For our first experiment we recruited novice and expert participants as part of a quantitative user study. Our second experiment is performed on a larger dataset and we evaluate a simulated user.

User study. Recently, Haehn *et al*. evaluated the interactive proofreading tools Raveler, Mojo, and Dojo as part of an experiment with novice users [10]. The participants corrected an automatic segmentation with merge and split errors. The dataset was the most representative sub-volume (based on object size histograms) of a larger connectomics dataset and $400x400x10$ voxels in size. The participants were given a fixed time frame of 30 minutes to perform the correction interactively. While participants clearly struggled with the proofreading task, the best performing tool in their evaluation was Dojo. The dataset including manually labeled ground truth and the results of Haehn *et al*. are publicly available. This means we are able to use their findings as a baseline for comparison of GP for novices. In particular, we use the best performing user of Dojo who was truly an outlier as reported by Haehn *et al*.

Since interactive proofreading most likely yields lower performance than aided proofreading, we also compare against FP by Plaza [27] which is integrated in Raveler and freely available. For FP we consulted an expert to obtain the best possible parameters as shown in table 1. Besides performance by novices, we are also interested in expert proofreading performance. Therefore, we design between-subjects experiments for 20 novice users and separately, for 6 expert users using the exact same conditions as Haehn *et al*. The recruiting, consent and debriefing process is further described in the supplementary material. We randomly assign 10 novices to GP with active label suggestion (GP*) and 10 novices to FP. For the expert experiment, we assign accordingly. In addition to human performance, we also evaluate automatic GP, automatic GP with active label suggestion (GP*) and automatic FP. Due to the automatic nature, we do not enforce the 30 minute time limit but we stop once our probability threshold of $p_t = .95$ is reached. This value was observed as stable in previous experiments using automatic GP (see supplementary material). To measure proofreading performance in comparison to ground truth, we use the adapted Rand error (aRE) metric [31]. aRE is a measure of dissimilarity, related to introduced errors, meaning lower scores are better.

The results of our comparisons are shown in the first row of Fig. 6. In all cases, GP* is able to correct the segmentation further than other methods (aRE measures: automatic GP XX, GP* XX, FP XX, novice Dojo XX, GP* XX, FP XX, expert Dojo XX, GP* XX, FP XX). This is not surprising since guided proofreading works for both merge and split errors while FP does not and in interactive Dojo the majority of time is spent finding errors which is minimized for aided proofreading solutions. In fact, the average correction time for novices is for GP* 3.6 (expert X), for FP Y (expert YY), and for Dojo 30 (expert ZZ) seconds.

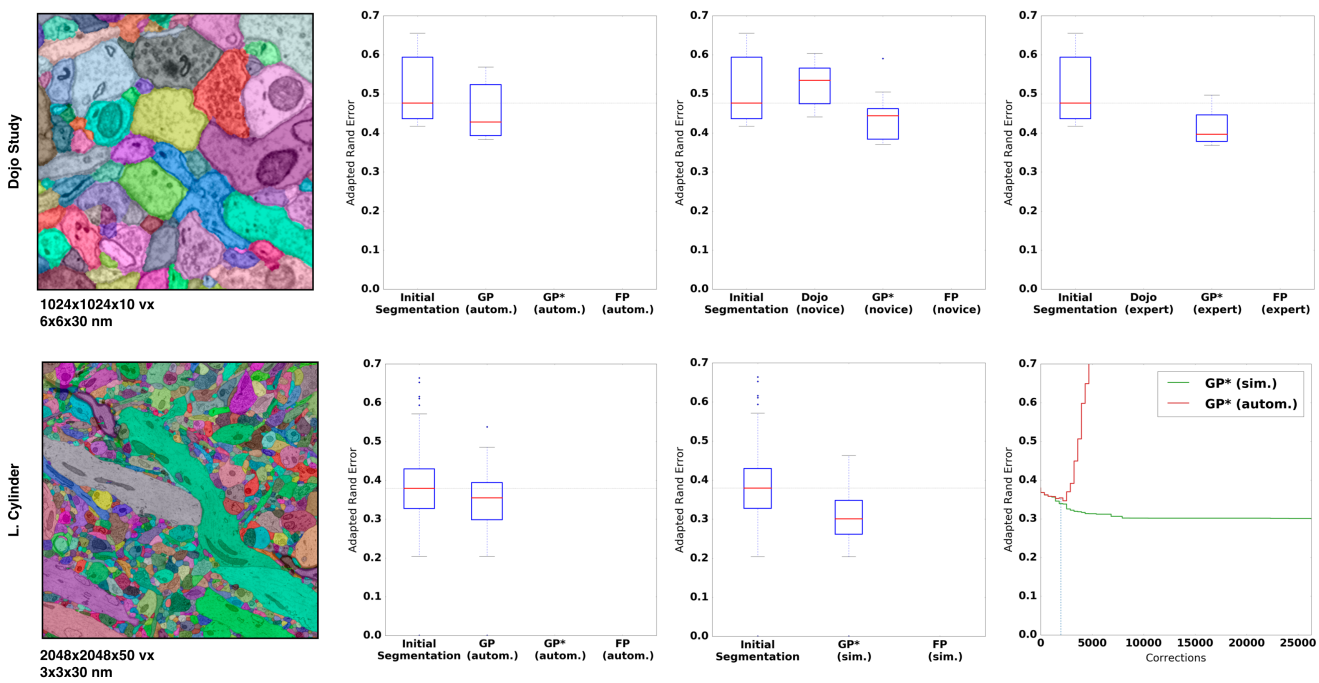

Figure 6. Performance evaluation of the classifiers on two mouse brain datasets measured as adapted Rand error (lower scores are better). We compare guided proofreading (GP), guided proofreading with active label suggestion (GP*) and focused proofreading. Proofreading is performed automatically (autom., with probability threshold $p_t = .95$), simulated as a perfect user (sim.), or by novice and expert users as indicated. The first row of images shows the results of a user study and includes comparisons to the interactive proofreading software Dojo by Haehn *et al.* [10]. GP* is able to correct the segmentation further than other methods. The second row shows the results of the simulated user compared to automatic GP* and FP performance. The bottom right graph compares automatic GP* and simulated GP* per individual correction. The blue dashed line here indicates the moment the probability threshold p_t is reached. The simulated user is able to correct the initial segmentation beyond this threshold while automatic GP* then introduces errors.

Simulated experiment. For our second experiment with mouse brain data, we proofread the last 50 slices of the blue 3-cylinder mouse cortex volume of Kasthuri *et al.* [15] which we also used for testing in section 3. The data was not seen by the network before and includes $2048x2048x50$ voxels with a total number of 17,560 labeled objects. Since an interactive evaluation of such a large dataset would consume a significant amount of time, we restrict our experiment to a simulated (perfect) user and to automatic corrections, both with GP, GP* and FP. Similar to our comparison study, the simulated user assess a stream of errors by comparing the adapted Rand error measure before and after each performed correction. The simulated user is designed to be perfect and only accepts corrections if the measure is reduced. This time, we do not enforce a time limit to see the lower bound of possible corrections. For automatic GP and GP*, we use our defined probability threshold $p_t = .95$.

The results of this experiment are shown in the second row of Fig. 6. GP* is again able to correct the segmentation further than other methods (aRE measures: automatic GP XX, GP* XX, FP XX, simulated GP* XX, FP XX). Again, the results are not surprising since GP* can correct merge and split errors.

4.2. Drosophila Brain

The drosophila brain is analyzed by connectomics researchers because of its small size and hence, a reasonable target to obtain a complete wiring diagram. Despite the size, fruit flies exhibit complex behaviors and are in general well studied. We evaluate the performance of our guided proofreading classifiers on three different datasets of adult fly brain. The datasets are publicly available as part of the MICCAI 2016 challenge on circuit reconstruction from electron microscopy images (CREMI)[2]. Each dataset consists of $1250x1250x125$ voxels of training data (A,B,C) as well as testing data (A+,B+,C+) of the same dimensions. Manually labeled ground truth is also available for A,B, and C but not for the testing data.

Since drosophila brain exhibits different cell structures than mouse brain, we retrain the guided proofreading classifiers (and our automatic segmentation pipeline) as well as focused proofreading combined on the three training

[2]The MICCAI CREMI challenge data is available at http://www.cremi.org

datasets. We use 300 slices of the A,B,C samples for training and validation, and 75 slices for testing. This results in YYY correct and ZZZ split error patches (respectively, XXX and YYY for testing). The architecture and all parameters of our classifiers stay the same. The trained GP classifier exhibits a reasonable performance on the testing data as seen in Fig. 4.

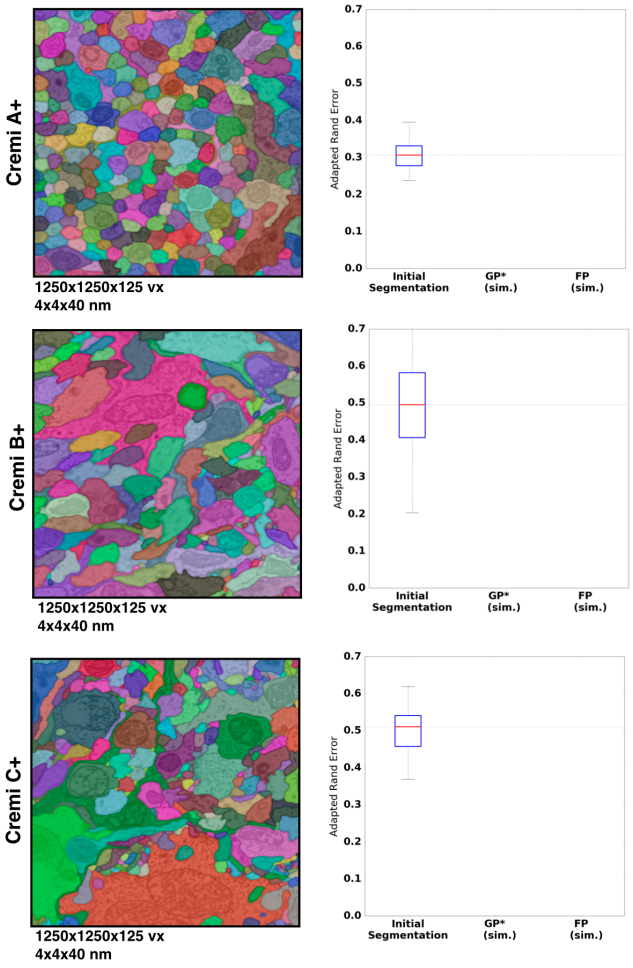

Figure 7. Results of guided proofreading with active label suggestion (GP*) and focused proofreading performed automatically on three drosophila datasets. The datasets are part of the MICCAI 2016 CREMI challenge and publicly available. We measure performance as adapted Rand error (the lower, the better). GP* is able to correct the initial segmentation further than FP. Our GP* scores places us XXnd on the CREMI leaderboard.

We then use the trained GP* and FP classifiers to evaluate proofreading automatically. Since ground truth labeling is not available, the evaluation is performed by submitting our results to the CREMI leaderboard. Again, we use adapted Rand error to quantify the performance. Fig. 7 shows the results for each of the A+,B+, and C+ datasets. The performance of GP* is significantly better than FP and places us XXnd on the CREMI leaderboard.

5. Conclusions

The task of automatic cell boundary segmentation is difficult, and trying to improve such segmentations automatically as a post-process through merge and split error correction is, in principle, no different than trying to improve the underlying cell boundary segmentation. Due to the task difficulty, manual proofreading of connectomics segmentations is necessary, but it is a time consuming and error-prone task. Humans are the bottleneck and minimizing the manual labor is the goal. We have addressed this problem through training a convolutional neural network to detect ambiguous regions from labeled data—in effect, by finding a non-linear mapping between image and segmentation data. This allows us to identify merge and split errors with better performance than existing systems. Our experiments have shown that guided proofreading has the potential to reduce the bottleneck in the analysis of large connectomics datasets. To encourage testing of our proposed architecture and replicate our experiments, we provide our framework and data as free and open research at (link omitted for review).

References

[1] IEEE ISBI challenge: SNEMI3D - 3D segmentation of neurites in EM images. http://brainiac2.mit.edu/SNEMI3D, 2013. Accessed on 11/01/2016. 1, 2

[2] Neuroproof: Flyem tool, hhmi / janelia farm research campus. https://github.com/janelia-flyem/NeuroProof, 2013. Accessed on 03/15/2106. 2

[3] A. K. Al-Awami, J. Beyer, D. Haehn, N. Kasthuri, J. W. Lichtman, H. Pfister, and M. Hadwiger. Neuroblocks - visual tracking of segmentation and proofreading for large connectomics projects. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):738–746, Jan 2016. 2

[4] J. Anderson, S. Mohammed, B. Grimm, B. Jones, P. Koshevoy, T. Tasdizen, R. Whitaker, and R. Marc. The Viking Viewer for connectomics: Scalable multi-user annotation and summarization of large volume data sets. *Journal of Microscopy*, 241(1):13–28, 2011. 2

[5] ANON. Anon. *ANON*, 2016. 6

[6] J. A. Bogovic, G. B. Huang, and V. Jain. Learned versus hand-designed feature representations for 3d agglomeration. *CoRR*, abs/1312.6159, 2013. 1, 2, 3, 4

[7] D. B. Chklovskii, S. Vitaladevuni, and L. K. Scheffer. Semi-automated reconstruction of neural circuits using electron microscopy. *Current Opinion in Neurobiology*, 20(5):667 – 675, 2010. Neuronal and glial cell biology New technologies. 2

[8] D. C. Ciresan, A. Giusti, L. M. Gambardella, and J. Schmidhuber. Deep neural networks segment neuronal membranes in electron microscopy images. In *NIPS*, 2012. 1

[9] R. J. Giuly, K.-Y. Kim, and M. H. Ellisman. DP2: Distributed 3D image segmentation using micro-labor workforce. *Bioinformatics*, 29(10):1359–1360, 2013. 2

[10] D. Haehn, S. Knowles-Barley, M. Roberts, J. Beyer, N. Kasthuri, J. Lichtman, and H. Pfister. Design and eval-

uation of interactive proofreading tools for connectomics. *IEEE Transactions on Visualization and Computer Graphics (Proc. IEEE SciVis 2014)*, 20(12):2466–2475, 2014. 1, 2, 3, 6, 7

[11] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 3

[12] V. Jain, B. Bollmann, M. Richardson, D. Berger, M. Helmstädter, 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. 1, 2

[13] Janelia Farm. Raveler. https://openwiki.janelia.org/wiki/display/flyem/Raveler, 2014. Accessed on 11/01/2016. 1, 2

[14] A. Karimov, G. Mistelbauer, T. Auzinger, and S. Bruckner. Guided volume editing based on histogram dissimilarity. *Computer Graphics Forum*, 34(3):91–100, May 2015. 3, 5

[15] N. Kasthuri, K. J. Hayworth, D. R. Berger, R. L. Schalek, J. A. Conchello, S. Knowles-Barley, D. Lee, A. Vázquez-Reina, V. Kaynig, T. R. Jones, et al. Saturated reconstruction of a volume of neocortex. *Cell*, 162(3):648–661, 2015. 1, 4, 7

[16] V. Kaynig, T. Fuchs, and J. Buhmann. Neuron geometry extraction by perceptual grouping in sstem images. In *Proc. IEEE CVPR*, pages 2902–2909, 2010. 2

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

[18] J. S. Kim, M. J. Greene, A. Zlateski, K. Lee, M. Richardson, S. C. Turaga, M. Purcaro, M. Balkam, A. Robinson, B. F. Behabadi, M. Campos, W. Denk, H. S. Seung, and EyeWirers. Space-time wiring specificity supports direction selectivity in the retina. *Nature*, 509(7500):331336, May 2014. 2

[19] 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. 1

[20] K. Lee, A. Zlateski, A. Vishwanathan, and H. S. Seung. Recursive training of 2d-3d convolutional networks for neuronal boundary detection. *arXiv preprint arXiv:1508.04843*, 2015. 2

[21] J. W. Lichtman and W. Denk. The big and the small: Challenges of imaging the brain's circuits. *Science*, 334(6056):618–623, 2011. 6

[22] T. Liu, C. Jones, M. Seyedhosseini, and T. Tasdizen. A modular hierarchical approach to 3D electron microscopy image segmentation. *Journal of Neuroscience Methods*, 226(0):88 – 102, 2014. 1, 2

[23] J. Masci, A. Giusti, D. C. Ciresan, G. Fricout, and J. Schmidhuber. A fast learning algorithm for image segmentation with max-pooling convolutional networks. In *ICIP*, 2013. 1

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

[25] J. Nunez-Iglesias, R. Kennedy, S. M. Plaza, A. Chakraborty, and W. T. Katz. Graph-based active learning of agglomeration (GALA): A python library to segment 2D and 3D neuroimages. *Frontiers in Neuroinformatics*, 8(34), 2014. 1, 2

[26] H. Peng, F. Long, T. Zhao, and E. Myers. Proof-editing is the bottleneck of 3D neuron reconstruction: The problem and solutions. *Neuroinformatics*, 9(2-3):103–105, 2011. 2, 3

[27] S. M. Plaza. Focused Proofreading: Efficiently Extracting Connectomes from Segmented EM Images, Sept. 2014. 2, 3, 4, 6

[28] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. *CoRR*, abs/1505.04597, 2015. 2

[29] S. Saalfeld, A. Cardona, V. Hartenstein, and P. Tomančák. CATMAID: collaborative annotation toolkit for massive amounts of image data. *Bioinformatics*, 25(15):1984–1986, 2009. 2

[30] R. Sicat, M. Hadwiger, and N. J. Mitra. Graph abstraction for simplified proofreading of slice-based volume segmentation. In *EUROGRAPHICS Short Paper*, 2013. 1, 2

[31] R. Unnikrishnan, C. Pantofaru, and M. Hebert. A measure for objective evaluation of image segmentation algorithms. pages 34–, 2005. 2, 6

[32] M. G. Uzunbas, C. Chen, and D. Metaxas. An efficient conditional random field approach for automatic and interactive neuron segmentation. *Medical Image Analysis*, 27:31 – 44, 2016. Discrete Graphical Models in Biomedical Image Analysis. 3

[33] A. Vázquez-Reina, M. Gelbart, D. Huang, J. Lichtman, E. Miller, and H. Pfister. Segmentation fusion for connectomics. In *Proc. IEEE ICCV*, pages 177–184, Nov 2011. 2