Proofreading and Automated Error Correction of Neuron Instance Segmentation

Carlotta Sophia Hölzle[1]

Abstract

With the increasing use of high-resolution electron microscopy, large-scale brain tissue data availability has expanded, making manual segmentation infeasible and necessitating automated segmentation of neuronal structures. These segmentations are error prone, with merge and split errors being the most common. This study compares existing tools and advancements in error identification and repair to identify improvements in (semi-) automated proofreading processes. The assessment looks for advancements that can reduce the amount of manual labour needed and increase proofreading efficiency.

1. Introduction

The development of high-resolution electron microscopy (EM) has vastly increased the size and availability of brain tissue data. This data is crucial for understanding neuronal connectivity and architecture, thus advancing our knowledge of cognition, behaviour, and disease (Seung, 2009). However, the sheer size of these datasets makes manual annotation nearly impossible, necessitating (semi-)automatic segmentation. Frequent errors, like false merges and splits of neurons, severely compromise the accuracy of connectomic analyses. Detecting and correcting these errors is challenging and labour-intensive; for example, manually proofreading just half of the centre of a fly brain can take up to 50 person-years of effort (Scheffer et al., 2020). Since segmentation proofreading is the main bottleneck in obtaining a connectome (Sheridan et al., 2023) for analysis and becomes even more complicated with larger datasets (Dorkenwald et al., 2022), automated error detection and (semi-) automated proofreading tools are essential research fields in neuron instance segmentation.

This paper compares tools for automating error correction and minimizing manual labour in the proofreading processes; it does not evaluate specific case studies. These tools are mainly used for research and crowd-sourcing projects. Since these projects have diverse natures and information on the widespread use of each tool is limited, making a direct comparison with real-life applications is currently not feasible. This overview and comparative analysis of existing tools aims to identify areas for research and technological innovation. The analysis is split into two parts: the first focuses on evaluating the efficacy of interactive proofreading tools, meaning tools that integrate the human into the workflow, and the other evaluates methods that try to exclude the human from the proofreading process.

2. Background

Proofreading in neuron instance segmentation involves correcting errors to ensure an accurate representation of neuronal structures. The most common errors are merge and split errors (Haehn et al., 2014). Merge errors occur when parts of different cells are incorrectly connected, resulting in a segment that includes parts of multiple neurons. Split errors happen when a single cell is erroneously divided into multiple parts, called segmentation fragments or supervoxels. Graph-based methods are frequently used to visualize and correct oversegmentation, which is primarily characterized by multiple split errors and, ideally, no merge errors. Here, fragments are represented as nodes and decisions about merging them are represented as edges. Synapse threshold proofreading sets correction criteria based on the expected number and distribution of synapses within a neuron. The system suggests or automatically performs corrections by merging or splitting segments to align with known synapse distributions. Volume threshold proofreading focuses on the physical size of neuronal segments, flagging anomalies that deviate from neuron sizes derived from biological data. Other types of errors, such as boundary errors, where the boundary of a labelled segment does not match the EM image structure, and orphan twigs, thin neural segments not assigned to larger structures, are hardly considered in (semi-)automated proofreading applications.

3. Interactive Proofreading

Improving connectome reconstruction necessitates advancing proofreading approaches to complement current automated segmentation methods (Schmidt et al., 2024). These approaches are versatile; some incorporate biological priors

[1]Technical University of Munich, Munich, Germany. Correspondence to: Carlotta Sophia Hölzle <carlotta.hoelzle@tum.de>.
Machine Learning for Connectomics Seminar, 2024. Copyright 2024 by the author(s).

for error detection, while others enable crowd-sourcing and ensure universal access (Meirovitch et al., 2016; Rolnick et al., 2017; Hubbard et al., 2020; Haehn et al., 2018). The degree of error detection and correction automation within a proofreading tool significantly determines the reduction of manual labour. Additionally, the usability of the tool's interface further decreases manual efforts by enabling more efficient spotting and correcting of errors, particularly in larger datasets. Table 2 provides an overview, and the following section evaluates each application that facilitates advancements in proofreading.

The following criteria were used to evaluate applications for reducing manual labour in proofreading:

1. **Collaborativeness and Accessibility:** Tools are assessed based on their real-time, collaborative proofreading support, required operational expertise, control mechanisms for multi-user applications, and accessibility, with a preference for web-based platforms and no hardware or software installation.

2. **Visualization:** Tools are assessed for their segment visualization capabilities, with a preference for 3D and those enabling easy toggling between rendering models or additional visualization methods.

3. **Error Detection and Correction:** The analysis assesses the level of automated error detection and correction, prioritizing tools requiring minimal manual proofreading intervention.

4. **Interface and Tools:** The application's GUIs and tools are evaluated for simplicity and ease of use. A minimalist and uncluttered GUI and simple tools are preferred for enhanced usability.

5. **Progress Tracking and History Visualization:** Tools are assessed based on their ability to mark and visualize proofreading progress or state and maintain a history of changes.

6. **Computational Expense** The main drivers for computational cost and the overall computational efficiency are evaluated.

7. **Scalability to Large Datasets** This point looks at the possibility of scaling the applications to large datasets.

8. **Input and Output Data Requirements:** The standard input is an automated EM segmentation, and the output is a proofread segmentation map. Tools deviating from this norm undergo analysis, penalizing those that require added manual effort for data processing.

Each criterion is rated on a scale from 0 to 4, with 0 signifying "does not decrease manual labour" and four signifying "eliminates manual labour." Table 2 details the assignment of these points, with each requirement receiving an individual score.

3.1. Graph abstraction for simplified proofreading of slice-based volume segmentation

The method introduced by Sicat et al. (2013) simplifies proofreading of slice-based segmentation into a graph abstraction. Each segment is represented as a node, and edges between nodes depict the connections between neural structures. Nodes store boundary contours and measure segmentation consistency, highlighting potential errors by weighting nodes based on local inconsistencies. Larger nodes indicate higher inconsistency, directing focus to problematic areas. This semi-automated error detection, paired with the intuitive GUI, facilitates efficient proofreading. The tool does not support tracking proofreading progress or visualizing the edit history. The tool is designed for single-user operation with a local installation, limiting collaborative proofreading capabilities and accessibility. While the 2D visualization provides a local understanding, the lack of 3D representation can limit error detection in more complex structures or the overall connectom. The application outputs a graph abstraction with stored inconsistent areas and the corrected segmentation map. Constructing the graph abstraction and measure of inconsistencies is computationally intensive, limiting the scalability to larger datasets. Furthermore, more complex data, e.g., with many intertwined neural structures, can make the graph more challenging to navigate and correct. Consequently, the overall reduction of manual labour is minimal, scoring one on the point scale.

3.2. Raveler

Raveler by Olbris et al. (2014) is a proofreading tool for neuron instance segmentation that divides datasets into discrete blocks using a quadtree-based system. This method enables multiple proofreaders to simultaneously proofread a dataset by locking blocks per user. However, the block-locking complicates scaling to larger datasets, makes segments spanning multiple blocks difficult to proofread effectively and increases overhead costs. Raveler offers extensive customization and advanced tools, making it challenging for non-specialists to use (Haehn et al., 2014). Its limited 3D capabilities and reliance on 2D slices hinder the detection of false merge errors and prevent a global understanding of the connectome. Error identification and correction are predominantly manual, increasing the workload and adding a risk of oversight. The tool lacks progress or history tracking features, hindering workflow management in multi-user settings. The tool streamlines proofreading through collaboration and customization, but its complexity and limited visualization capabilities pose challenges in error detection and correction, scoring a one on the point scale.

3.3. Mojo/Dojo

Knowles-Barley et al. (2013) developed Mojo and Dojo. Mojo is a desktop application that supports single users with advanced semi-automated proofreading capabilities. It is suited for in-depth, individual analysis but is limited by its desktop-bound nature, requiring installation. The primary offering of 2D slice-based visualization makes tracking 3D structures challenging. Therefore, the overall decrease in manual labour is minimal, with a point scale rating of 1. Dojo is a web-based platform designed to be a successor to Mojo. It facilitates real-time collaborative proofreading among multiple users. Its web-based nature eliminates the need for installation and offers a clean interface, 3D volume rendering, and user-friendly tools beneficial to non-experts. Dojo has limitations in error detection support, undo stack, and lack of history or progress tracking. Although it is an improvement compared to Mojo, the minimal level of automation in error detection and correction, along with other limitations, leads to an overall score of 2.

3.4. NeuroProof

Plaza (2014) developed a proofreading tool called NeuroProof. This tool emphasizes automated error detection and correction through volume and synapse threshold proofreading, orphan tracing, and user validation through binary (yes/no) error confirmation queries. It enables automated merging and splitting of neuronal segments, reducing human errors and manual efforts. The process can be used via a Raveler plugin or as a standalone tool installed locally from GitHub. NeuroProof lacks multi-user and web-based options and does not track proofreading progress or history. However, it features a minimalistic GUI guiding proofreaders with simple yes/no questions. NeuroProof is computationally efficient, does not require a global uncertainty model, and is easily parallelizable. It operates on data stacks suitable for image processing, leveraging C++ for performance, allowing extensive data processing. Scalability is only limited by hardware availability. According to the original publication, NeuroProof accelerates proofreading 3-5 times compared to manual methods. Manual oversight remains necessary for quality control and specific errors, balancing automated efficiency with human expertise, putting the tool with its limitations on a two on the labour reduction point scale.

3.5. NeuTU

NeuTu by Zhao et al. (2018) is a software package designed to proofread large multi-terabyte EM datasets, offering both 2D and 3D visualization and supporting multiple users through integration with DVID, a distributed, versioned, image-oriented data service. This integration enables segment locking to prevent simultaneous edits, enhancing consistency and reducing conflicts. The tool's 3D and detailed 2D visualization, combined with a seeded watershed algorithm for automatically detecting false merges and visualization hints for synapse confidence, enhances error detection and correction. Although limited to Linux and Mac operating systems, NeuTu provides an intuitive GUI that saves a history of changes. Furthermore, users can track their proofreading progress individually and across the connectome by adding a state 'to-do' or 'done' per area. Despite its efficiency with large datasets through sparse volume representation and multi-scale updating, NeuTu faces challenges with large data volumes and complex 3D structures due to its dependency on DVID for data management. This dependency requires data to be loaded into and exported from DVID, potentially complicating workflows and limiting flexibility with different data formats. NeuTu's robust platform excels in visualization, semi-automated error correction, and collaborative proofreading, yet system dependencies and data management complexities constrain its effectiveness, extensively reducing manual labour, a three on the point scale.

3.6. VICE

The code for VICE is not publicly available, and the web-based front end has yet to be published. Therefore, this author could not verify the results from the publication. VICE by Gonda et al. (2021) employs a web-based single-cell strategy for proofreading neural circuits, focusing on errors within individual cells and their connections. Its automated error detection system uses expert heuristics and a dynamic 3D visualization framework for error correction. VICE supports 2D and 3D visualizations, transforming segmentation into skeletons and clustering them for better error identification. While VICE's web-based front end is a notable feature, it lacks multi-user support, limiting collaborative proofreading. The multi-stage workflow of VICE is intuitive, simplifying proofreading for non-experts but presenting a high initial learning curve due to the variety of tools. Like most applications, it does not support tracking proofreading progress or a history of changes. VICE manages computational complexity by selectively rendering 3D elements in low resolution, switching to high resolution only when necessary. Paired with an on-demand data loading strategy, VICE is scalable to vast datasets without significant performance degradation. VICE outputs connectivity graphs alongside the proofread segmentation mask, making downstream analysis easy and reducing manual labour for additional verification steps. Overall, VICE excels in automated error detection and semi-automatic correction, earning a score of 2, as it would benefit from collaborative proofreading and progress tracking support.

3.7. FlyWire

FlyWire by Dorkenwald et al. (2022) is an online platform designed explicitly for proofreading and annotating the Drosophila melanogaster (fruit fly) brain. FlyWire's uniqueness lies in leveraging an open, community-based approach, allowing researchers and laypersons worldwide to contribute to the proofreading process. The quality control of user proofreading is based on the same mechanism of crowd wisdom as Wikipedia, namely iterative collaborative editing. FlyWire uses a data structure that breaks down extensive neural datasets into manageable segments within a supervoxel graph. Locking a segment before editing makes rapid, real-time corrections without overlapping or conflicting modifications possible. This data structure enables scaling to large datasets and proofreading communities. In addition, FlyWire integrates advanced algorithms, like the max-flow min-cut algorithm, to automate the proofreading process. The tool efficiently identifies the best ways to split neurons to maintain the neuronal structure's integrity. It also stores an edit history. FlyWire is currently limited to the Drosophila dataset, which restricts its application to human or other species' neuronal data. Additionally, FlyWire's conservative strategy excludes the proofreading of orphan twigs or boundary errors. Hence, the reduction in manual labor is quite advanced, scoring a 2.5 on the point scale.

4. Automated Proofreading Workflows

This section evaluates algorithms designed to automatically detect and correct segmentation errors before the final output, not including the human in the proofreading process. The goal is to improve the detection of inconsistencies using algorithms that recognize deviations from expected norms. Once detected, the systems automatically adjust segmentation boundaries and correct errors by merging or splitting regions. Table 1 provides an overview, and the following section evaluates each method independently.

4.1. RoboEM

RoboEM by Schmidt et al. (2024) is an AI-driven software tool that uses a self-steering 3D flight system trained on EM data to navigate neurites and detect segmentation errors. It employs autonomous driving-like technology to generate 3D steering signals from EM images, optimizing path tracing and error correction. Validation strategies like reciprocal tracing ensure neurite continuity and accuracy, reducing merge errors and enhancing overall segmentation quality. RoboEM has lowered computational annotation costs for cortical connectome analysis by approximately 400-fold compared to manual methods. RoboEM has proven effective in reducing merge and split errors. However, it relies on high-quality initial segmentation for accurate navigation and requires sophisticated neural network training, limiting its adaptability to diverse EM datasets and demanding high computational efforts for training.

4.2. Neuronal Subcompartment Classification and Merge Error Correction

Li et al. (2020) employ a 3D convolutional neural network to classify neuron fragments into three subcompartments and subsequently automate the correction of merge errors. The model inputs automated segmentations from flood-filling networks, enhanced with contrast-normalized images or organelle probability maps, and outputs probability scores for each subcompartment category. Segmentation errors are detected by categorizing each segment and checking for inconsistencies in these categories. Furthermore, heuristics are used to correct merge errors. Errors in axon/dendrite merges are assessed by identifying inconsistencies in cuts in neuron branches. Soma merge errors are corrected by examining the trajectory of branches in relation to the soma surface. Overall, the system detects a high percentage of merge errors with a low false positive rate, drastically limiting the need for manual proofreading. However, the model is limited to correcting merge errors and faces challenges with scaling as the field of view increases. Processing efficiency declines with larger volumes, suggesting a need for more optimized or alternative sparse 3D data representations.

4.3. NEURD

Celii et al. (2023) introduced NEURD, a tool for automated proofreading, error correction, and feature extraction of neuron segmentations. It converts the 3D neuron meshes into a graph-based format. This transformation allows heuristic rules and algorithms to identify and correct segmentation errors, such as false merges. The heuristic rules are based on standard error patterns, like unexpected branching or abrupt changes in neurite diameters. These heuristics guide the algorithm in redefining boundaries and disconnecting erroneously merged segments. Furthermore, the tool analyzes geometrical alignments and pre-computed feature sets to find inconsistencies. NEURD uses graph filters to specialize error-correction algorithms for various neuron types, allowing customized proofreading for different neurological structures. The graph-based representation is enhanced with features like cell types, compartments, and spine information, which aids downstream tasks like error correction, cell classification, and connectivity analysis. The computational complexity, similar to the two previously introduced methods, is significant. NEURD is mainly limited to correcting false merges and does not handle split errors, which impacts its overall effectiveness in automated proofreading.

5. Future Research Possibilities

The main objective for future advancements should be to scale fully automated methods to large datasets, make them computationally feasible and extend them to correct orphan twigs, split and boundary errors. Afterwards, these automated error correction algorithms should be merged with advanced proofreading tools into one workflow. Such a workflow should operate across diverse datasets and integrate into collaborative, multi-user environments.

Secondly, transforming complex neuron instance proofreading tasks into an interactive, gamified platform presents a viable strategy for enhancing proofreading accuracy, as exemplified by EyeWire (Tinati et al., 2017). Engaging a vast community in a user-friendly, web-based environment broadens scientific involvement and makes neurological data exploration more accessible. Future research could explore extending or reinventing the EyeWire crowed-sourcing paradigm to EM data.

These advancements could significantly enhance the field by streamlining workflows, integrating AI algorithms with global collaborative input, and thus further advancing neuron proofreading.

6. Conclusion

The comparative analysis reveals that fully automated workflows cannot detect all types of errors, are mainly limited to merge errors, and are not yet scalable to large EM datasets. While multiple interactive proofreading tools are available, none incorporate all specified requirements or achieve a score of 4, indicating full reduction of manual labor, on the point scale. More research is needed to address limitations such as detecting and correcting split errors and orphan twigs, scaling to large datasets, and supporting proofreading communities. The author recommends trying NeuTU first due to its user-friendliness, availability, unique hints for possible errors, and features for marking within the proofreading process.

As connectomic datasets are expected to grow significantly, the number of human proofreaders is not. Therefore, leveraging computational resources is crucial. We must scale proofreading tools to handle large datasets, fully utilize available computing power, and integrate various (semi-) automated proofreading methods. One unified workflow is needed to advance neuronal instance segmentation to the level of the human brain. Without this we cannot advance research of degenerative diseases such as Alzheimer's, which are becoming increasingly significant in our ageing society.

References

Celii, B., Papadopoulos, S., Ding, Z., Fahey, P. G., Wang, E., Papadopoulos, C., Kunin, A. B., Patel, S., Bae, J. A., Bodor, A. L., et al. Neurd: automated proofreading and feature extraction for connectomics. *BioRxiv*, 2023.

Dorkenwald, S., McKellar, C. E., Macrina, T., Kemnitz, N., Lee, K., Lu, R., Wu, J., Popovych, S., Mitchell, E., Nehoran, B., et al. Flywire: online community for whole-brain connectomics. *Nature methods*, 19(1):119–128, 2022.

Gonda, F., Wang, X., Beyer, J., Hadwiger, M., Lichtman, J. W., and Pfister, H. Vice: Visual identification and correction of neural circuit errors. In *Computer graphics forum*, volume 40, pp. 447–458. Wiley Online Library, 2021.

Haehn, D., Knowles-Barley, S., Roberts, M., Beyer, J., Kasthuri, N., Lichtman, J. W., and Pfister, H. Design and evaluation of interactive proofreading tools for connectomics. *IEEE transactions on visualization and computer graphics*, 20(12):2466–2475, 2014.

Haehn, D., Kaynig, V., Tompkin, J., Lichtman, J. W., and Pfister, H. Guided proofreading of automatic segmentations for connectomics. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 9319–9328, 2018.

Hubbard, P. M., Berg, S., Zhao, T., Olbris, D. J., Umayam, L., Maitin-Shepard, J., Januszewski, M., Katz, W. T., Neace, E. R., and Plaza, S. M. Accelerated em connectome reconstruction using 3d visualization and segmentation graphs. *BioRxiv*, pp. 2020–01, 2020.

Knowles-Barley, S., Roberts, M., Kasthuri, N., Lee, D., Pfister, H., and Lichtman, J. W. Mojo 2.0: Connectome annotation tool. *Frontiers in Neuroinformatics*, 60:1, 2013.

Li, H., Januszewski, M., Jain, V., and Li, P. H. Neuronal subcompartment classification and merge error correction. In *Medical Image Computing and Computer Assisted Intervention–MICCAI 2020: 23rd International Conference, Lima, Peru, October 4–8, 2020, Proceedings, Part V 23*, pp. 88–98. Springer, 2020.

Meirovitch, Y., Matveev, A., Saribekyan, H., Budden, D., Rolnick, D., Odor, G., Knowles-Barley, S., Jones, T. R., Pfister, H., Lichtman, J. W., et al. A multi-pass approach to large-scale connectomics. *arXiv preprint arXiv:1612.02120*, 2016.

Olbris, D., Winston, P., Plaza, S., Bolstad, M., Rivlin, P., Scheffer, L., and Chklovskii, D. Raveler: A proofreading tool for em reconstruction. 2014.

Plaza, S. M. Focused proofreading: efficiently extracting connectomes from segmented em images. *arXiv preprint arXiv:1409.1199*, 2014.

Rolnick, D., Meirovitch, Y., Parag, T., Pfister, H., Jain, V., Lichtman, J. W., Boyden, E. S., and Shavit, N. Morphological error detection in 3d segmentations. *arXiv preprint arXiv:1705.10882*, 2017.

Scheffer, L. K., Xu, C. S., Januszewski, M., Lu, Z., Takemura, S.-y., Hayworth, K. J., Huang, G. B., Shinomiya, K., Maitlin-Shepard, J., Berg, S., et al. A connectome and analysis of the adult drosophila central brain. *elife*, 9: e57443, 2020.

Schmidt, M., Motta, A., Sievers, M., and Helmstaedter, M. Roboem: automated 3d flight tracing for synaptic-resolution connectomics. *Nature Methods*, pp. 1–6, 2024.

Seung, H. S. Reading the book of memory: sparse sampling versus dense mapping of connectomes. *Neuron*, 62(1): 17–29, 2009.

Sheridan, A., Nguyen, T. M., Deb, D., Lee, W.-C. A., Saalfeld, S., Turaga, S. C., Manor, U., and Funke, J. Local shape descriptors for neuron segmentation. *Nature methods*, 20(2):295–303, 2023.

Sicat, R., Hadwiger, M., and Mitra, N. J. Graph abstraction for simplified proofreading of slice-based volume segmentation. In *Eurographics (Short Papers)*, pp. 77–80, 2013.

Tinati, R., Luczak-Roesch, M., Simperl, E., and Hall, W. An investigation of player motivations in eyewire, a gamified citizen science project. *Computers in Human Behavior*, 73:527–540, 2017.

Zhao, T., Olbris, D. J., Yu, Y., and Plaza, S. M. Neutu: software for collaborative, large-scale, segmentation-based connectome reconstruction. *Frontiers in Neural Circuits*, 12:101, 2018.

A. Comparison of Tools

Table 1. Comparison of RoboEM, NEURD, and Li et al. (2020)

	RoboEM	**NEURD**	**Li et al. (2020)**
Algorithm	3D CNN outputting 3D steering signals	automated pipeline based on graph decomposition	automated pipeline with 3D CNN for classification
Concept	3D flight system to annotated EM data	transformation of 3D meshes into graph-based format, automated proofreading via graph filters	automatic classification of 3D voxels, with proofreading of merge errors
Correction of Errors	Automated connecting and validation of individual segments; resolving split and merge errors; attaching singular spine heads	Heuristic rules for detection and correction of segmentation errors, redefining boundaries & disconnecting erroneously merged segments	Error correction through analyzing consistency scores and using heuristics
Input	high-quality 3D segmentation	3D meshes from large EM volumes	specifically prepared segmentation data, e.g., contrast-normalized images or organelle probability maps
Output	updated segmentation map	corrected segmentation & set of morphological and connectomic features	neuron subcompartments probability scores & corrected neuron segmentation
Scalability	limited by quality of initial segmentation & computational resources available	graph-based approach possible, limited by computational resources available, automated methods only validated on small datasets	problematic if field of view increases
Computational complexity	significant	significant	significant

Table 2. Comparative Analysis of Tools for Proofreading of Neuron Instance Segmentation

Requirements	**Raveler**	**Mojo**	**Dojo**	**NeuTu**	**Flywire**	**Sicat et al. (2013)**	**VICE**	**NeuroProof**
Collaborativeness and Accessibility	multi-user; user lock; compilation	single-user; installation	multi-user; real-time editing without user locks; web-based	multi-user; user lock; installation (Linux and Mac)	multi-user; lock per user; web-based	single-user; installation	single-user; web-based	single-user; installation
	1	**0**	**3**	**2**	**2**	**0**	**1**	**0**
Visualization	2D; limited 3D	2D	3D	3D; 2D	Graph; 3D	2D; Graph	2D; 3D; seg. skeletons	2D
	1	**0**	**3**	**3**	**3**	**1**	**3**	**0**
Error Detection and Correction	manual	manual	manual	semi-automated; automated hints (confidence level & verification status)	semi-automated: max-flow min-cut to improve neuron splits	semi-automated: highlights inconsistencies	manual & semi-automated: heuristics from domain experts;	(semi-) automated: volume and synaptic activity; yes/no error questions
	0	**0**	**0**	**3***	**3**	**1**	**3**	**3**
Interface & Tools	advanced tools; experts; very complex GUI	advanced tools; experts; complex GUI	easy & parameter-free tools; non-experts; intuitive GUI	non-experts; intuitive GUI	non-experts; intuitive GUI	non-experts; intuitive GUI; easy graph representation	proficient users; advanced tools; intuitive workflow	non-experts; guidance to potential errors; minimalistic GUI
	1	**1**	**3**	**3**	**3**	**3**	**2**	**3**
Progress & History Tracking	no	no	no	progress & history	history	no	no	no
	0	**0**	**0**	**3***	**2**	**0**	**0**	**0**
Computational Expenses	overhead costs for data division and reintegration proportionally to dataset size	easy parallelizable; mostly scales linear with the number of processors	server-side costs; only metadata at user-side; memory constraints; no GPU utilization	efficient data handling & fetching; semi-automated cost proportional to body size	server-side computational costs	cost proportional with dataset size for computation of raw graph abstraction & measure of inconsistencies	adaptive rendering; on-demand loading	efficient algorithm; no global model of uncertainty; easy to parallelize
	1	**2**	**2**	**3**	**2**	**0**	**3**	**3**
Scalability	medium	yes	yes	medium	yes	medium	yes	yes
	1	**3**	**3**	**2**	**3**	**1**	**3**	**3**
Standard Input & Output	yes	yes	yes	Input & Output in DVID	yes	output: graph abstraction with inconsistencies	additional output: connectivity graph	yes
	2	**2**	**2**	**1**	**2**	**2***	**2**	**2**
Reduction	1	1	2	3	2.5	1	2	2

0 = no reduction, 1 = minimal, 2 = advanced, 3 = extensive, 4 = full, * = advanced features