Global Neuron Shape Reasoning with Point Affinity Transformers

Jakob Troidl[1,2] Johannes Knittel[1] Wanhua Li[1] Fangneng Zhan[1] Hanspeter Pfister[1,*] Srinivas Turaga[2,*]
[1]Harvard University [2]HHMI Janelia
{jtroidl, jknittel, wanhuali, fnzhan, pfister}@g.harvard.edu turagas@janelia.hhmi.org

Abstract

Connectomics is a subfield of neuroscience that aims to map the brain's intricate wiring diagram. Accurate neuron segmentation from microscopy volumes is essential for automating connectome reconstruction. However, current state-of-the-art algorithms use image-based convolutional neural networks that are limited to local neuron shape context. Thus, we introduce a new framework that reasons over global neuron shape with a novel point affinity transformer. Our framework embeds a (multi-)neuron point cloud into a fixed-length feature set from which we can decode any point pair affinities, enabling clustering neuron point clouds for automatic proofreading. We also show that the learned feature set can easily be mapped to a contrastive embedding space that enables neuron type classification using a simple KNN classifier. Our approach excels in two demanding connectomics tasks: proofreading segmentation errors and classifying neuron types. Evaluated on three benchmark datasets derived from state-of-the-art connectomes, our method outperforms point transformers, graph neural networks, and unsupervised clustering baselines.

1. Introduction

The field of connectomics maps the wiring diagrams of biological neural networks using high-resolution 3D microscopy and image segmentation. The recent completion of the connectome of the entire fruit fly brain was a major triumph for neuroscience [8]. Assembling this connectome involved substantial manual proofreading to correct segmentation errors, even with state-of-the-art automatic segmentation algorithms; 622 researchers from 146 laboratories worldwide contributed 33 human-years of manual proofreading effort for just 10^5 neurons in the millimeter-sized fruit fly (Drosophila) brain. Scaling up connectomics to map the 500× larger mouse brain [1] will require orders of magnitude larger human effort with current technology. Therefore, improving the accuracy of automated neuron segmentation algorithms is of vital importance to the field of neuroscience.

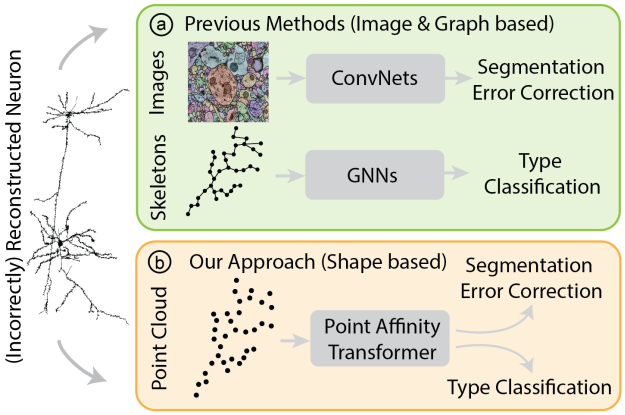
(Incorrectly) Reconstructed Neuron
ⓐ Previous Methods (Image & Graph based)
Images
ConvNets
Segmentation Error Correction
Skeletons
GNNs
Type Classification
ⓑ Our Approach (Shape based)
Point Cloud
Point Affinity Transformer
Segmentation Error Correction
Type Classification

Figure 1. **Overview & Comparison to Previous Work.** ⓐ Previous approaches use 3D ConvNets for improving neuron segmentation accuracy and are thus limited to local fields of view. Additionally, neuron type classification often requires auxiliary skeleton graphs. ⓑ We propose a new point affinity transformer model for neuron segmentation error correction and type classification.

Existing segmentation algorithms [12, 16, 24, 28–30, 40, 44] are largely CNN-based local methods which cannot reason over the shape of an entire neuron. Local methods have made significant progress towards accurately segmenting neurons, with error-free path-lengths of around 1.1 millimeters of neurite wire [16]. However, even a tiny fruit fly brain contains 145 meters of wiring and required significant manual proofreading effort to correct the machine-generated segmentation [8]. Human proofreaders use 3D renderings of candidate segmentations of entire neurons to detect and correct segmentation errors. These candidate segmentations sparsely span very large volumes with bounding boxes on the order of $10^4 \times 10^4 \times 10^4$ voxels, limiting the ability of image-based CNN to scale to global neuron shape. To address this challenge, we propose a novel type of point cloud transformer, called a point affinity transformer, to reason over global neuron shapes.

Point clouds can efficiently represent sparse yet expansive neuron shapes. We formulate neuron shape learning as a pairwise affinity prediction problem, where two points have an affinity value of 1 if both belong to the same neuron and 0 otherwise. Critically, our architecture does not require the model to explicitly represent spatial features (e.g., features over a grid), allowing it to implicitly represent sparse spatial neuron shapes flexibly. We apply our approach to automated neuron proofreading, which involves detecting and correcting errors in erroneous neuron point clouds. Additionally, we demonstrate our model's capability to represent global neuron structures by projecting its internal features into a contrastive embedding space, which enables the automatic assignment of type labels to neurons. Both tasks rely on a detailed understanding of global neuron morphology. We empirically evaluate our approach on three state-of-the-art connectome datasets [8, 36, 42]. We benchmark against other learning-based techniques, including graph neural networks (GNNs), alternative point cloud transformer architectures, and unsupervised clustering methods. Our results demonstrate superior performance for both automatic proofreading and neuron-type classification, achieving improvements in terms of common clustering and multi-class classification metrics. In summary, our work makes several contributions:

- We propose a novel neuron shape learning framework centered around pairwise point affinity prediction. Given a (multi-)neuron point cloud, our model's task is to disentangle individual neurons and background fragments by predicting correct affinities between point pairs. Our transformer-based model, the point affinity transformer, consists of two components: an encoder backbone (Fig. 3b), and an affinity decoder (Fig. 3c). This framework captures complex neuron morphologies.
- We apply this framework and model to automatic neuron proofreading. We correct reconstruction errors in multi-neuron point clouds using predicted pairwise point affinities and agglomerative clustering. Our approach outperforms baselines, such as graph neural networks (GNNs) and other point cloud transformers.
- Our model's internal neuron representations, trained only for affinity prediction, can easily be projected into a contrastive embedding space for neuron type classification.

2. Related Work

Point Clouds. Our rationale for choosing point clouds over other spatial data representations is two-fold. First, compared to volumetric representations, point clouds fit sparse and thin spatial data well. Second, point clouds are easy to obtain from segmentation volumes and do not require additional processing [20] like surface meshes. Approaches for segmentation and classification of point clouds can be broadly categorized into GNNs [41, 46], transformers-based approaches [13, 48, 49, 53–55], and set-based approaches [34, 35]. For the purpose of representing global neuron shapes, we argue that attention-based transformer models are better suited than GNNs. Neurons are thin, long, and sparse, making it hard for traditional message-passing GNNs to update point features based on distant points on the same neuron. In contrast, self-attention overcomes this issue by flexibly allowing node feature updates across long distances on the neuron. Within the space of point cloud transformers, there exist several key limitations for representing neuron shapes. First, several methods [52, 55] rely on spatial references within its internal feature space such as learned features over a grid-like structure, limiting flexibility when fitting to spatially sparse data. Our model's internal feature space has no spatial reference, allowing it to fit flexibly to thin and sparse neuron structures. Second, previous architectures [55] often depend on *a priori* knowledge about point cloud sizes. Our model easily adapts to variable point cloud input sizes in a single layer through a masked attention mechanism, enabling training on variable-size multi-neuron point clouds. Third, approaches for instance segmentation, such as Point Transformer V3 (PTv3) [48] combined with PointGroup (PG) [18] clustering, are designed for highly compact objects centered around object centroids. However, neurons are thin and tightly intertwined spatial structures. Thus neuron centroids are non-discriminative features.

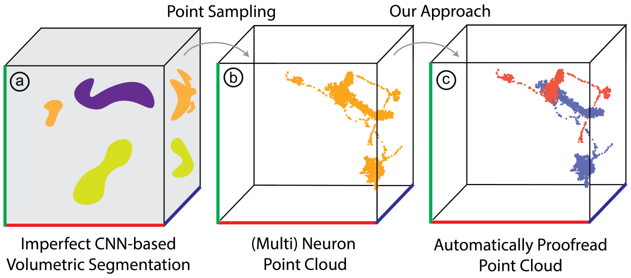
Point Sampling
Our Approach
ⓐ
ⓑ
ⓒ
Imperfect CNN-based Volumetric Segmentation
(Multi) Neuron Point Cloud
Automatically Proofread Point Cloud

Figure 2. **Automatic Proofreading Workflow.** ⓐ 3D CNN-based volumetric neuron segmentations contain errors. ⓑ We derive a (multi-)neuron point cloud through either skeletonizing or subsampling the segmentation volume. ⓒ Our approach enables automatic proofreading of such (multi-)neuron point clouds.

Neuron Error Correction. Substantial ongoing efforts manually proofread large-scale connectomes [5, 7, 8, 31, 39]. Another approach is to reduce the number of errors by increasing the robustness and accuracy of image-based segmentation models [12, 14–16, 29, 30, 56]. However, these 3D convolutional networks lack the global context of neuron morphology due to inherently localized convolution operations. Additionally, they must adjust for factors such as image noise and anisotropic image resolution. Instead, we show that point clouds are a robust and expressive 3D representation that allows the global neuron shape context to be taken into account. Prior work has made progress toward automatically proofreading connectome reconstructions. However, these approaches either rely on handcrafted heuristics [19], build on GNNs that require brittle auxiliary graphs [2], or use agent-based tracing algorithms [37].

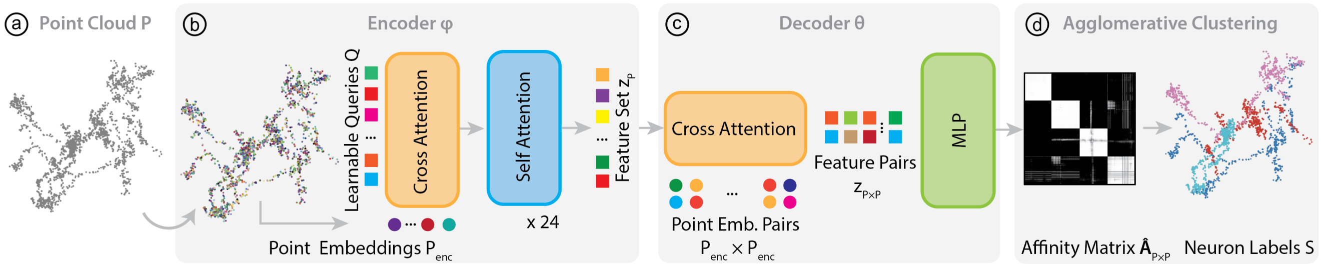

Figure 3. **Automatic Proofreading Pipeline.** ⓐ Our model encodes a point cloud with samples from N neurons ⓑ into a fixed length feature set using a series of cross- and self-attention layers (Sec. 3.2). ⓒ Next, we condition the decoder with this feature set and compute feature pairs of points from the input point cloud using a cross-attention module. These feature pairs are then decoded into affinity values using an MLP (Sec. 3.2). ⓓ We then extract neuron labels by performing feature agglomerative clustering on the affinity matrix (Sec. 3.3).

Neuron Classification. Approaches for neuron classification are separated into two groups. The first group focuses on neuron sub-compartment classification [6, 26], that is, assigning labels to subcellular structures such as dendrites, axons, or cell bodies. Notably, these approaches do not require global shape context to make reliable predictions. The second group focuses on neuron typing [3, 4, 17, 23, 38], which assigns a single label to a neuron that is indicative of its function. This is more challenging than sub-compartment classification since global neuron shape is vital for assigning a type. In practice, handcrafted feature extractors are still being used to group neurons into types [4, 22]. On the other hand, learning-based methods often operate on auxiliary graph data structures like skeletons [3, 11, 50], a global connectome graph [27], or multiview 2D projections [38] that do not fully capture 3D morphology. We argue that skeletons are brittle, sensitive to hyper-parameters, and accumulate errors in earlier steps. Manual parameter tuning is required to minimize small twigs, and gaps are easily introduced if the underlying image segmentation is discontinuous. Thus, we propose using point clouds for global neuron type classification, which are easy to obtain but also capture detailed shapes.

3. Method

Our goal is to learn an expressive feature space that can capture global neuron shapes. We demonstrate this capability in two challenging connectomics tasks: automatic neuron proofreading using our point affinity transformer model and neuron type classification.

3.1. Automatic Proofreading Workflow

3D CNNs for neuron segmentation generate voxel-based label volumes, where each label typically represents a ”supervoxel” — a small sub-segment of a neuron. Therefore, reconstructing the complete structure of a neuron requires agglomerating numerous supervoxels (Fig.2ⓐ). The specifics of the segmentation technique determine the approach to agglomeration; however, this process is generally susceptible to segmentation errors, resulting in either split or merge errors. The goal of proofreading is to correct these errors by separating incorrectly merged segments and joining incorrectly split ones. In practice, the balance between split and merge errors is influenced by the level of supervoxel agglomeration. To address both types of errors, our automated proofreading framework assumes a high degree of supervoxel agglomeration, which biases the segmentation toward merge errors. Consequently, we consider each point cloud as a superset containing all true neuron points (Fig.2ⓑ) alongside background fragments. We sample points from skeleton vertices and discard edge information. Our algorithm then focuses on partitioning the point cloud into individual neurons (Fig. 2ⓒ) while excluding any background fragments that do not correspond to full neurons.

3.2. Point Affinity Transformer

Problem Formulation. The main idea of our approach is to formulate global neuron shape learning as a pair-wise affinity prediction task on multi-neuron point clouds. Given an input point cloud $P = \{p_0, ..., p_n\} \subseteq \mathbb{R}^3$ (Fig. 3ⓐ) sampled from $\{1, ..., K\}$ neurons and a set of background fragments, our model M learns to predict an affinity matrix $\hat{\mathbf{A}}_{P\times P} \in \mathbb{R}^{n\times n}$ between all point pairs $P \times P$ as below:

$$\hat{\mathbf{A}}_{P\times P} = M(P, P \times P) = \{a_{(p_0,p_0)}, ..., a_{(p_n,p_n)}\}, \quad (1)$$

where an affinity value $a_{(p_i,p_j)}$ represents the likelihood that p_i and p_j belong to the same neuron. Given $\hat{\mathbf{A}}_{P\times P}$, we reconstruct segmentation labels $S = \{s_0, ..., s_n\}, s_i \subseteq \mathbb{N}$. $s_i \in S$ maps to the neuron identity of the respective point $p_i \in P$. Specifically, $s = 0$ denotes a background class, which covers fragments that do not assemble entire neurons. The goal is to learn a mapping such that all points with the same semantic label $s > 0$ belong to the same neuron. We derive S by clustering P using an affinity-based distance metric $\hat{\mathbf{D}}_{P\times P} = \mathbf{1} - \hat{\mathbf{A}}_{P\times P}$ (Sec. 3.3).

For model training, we use the binary cross entropy (BCE) loss between the predicted matrix $\hat{\mathbf{A}}$ and the binary ground truth matrix $\mathbf{A} \in \{0, 1\}^{n\times n}$, which we derived from proofread connectome data:

$$L_{\text{affinity}} = \text{BCE}(\hat{\mathbf{A}}, \mathbf{A}). \tag{2}$$

We reduce computational cost during training by decoding a randomly selected subset of point pairs $(P \times P)' \subset (P \times P)$ and thus also only predict a subset of the entire affinity matrix $\hat{\mathbf{A}}_{(P \times P)'} \subset \hat{\mathbf{A}}_{P \times P}$. At test time, we decode point pairs $P \times P$ to obtain the segmentation label set S for points P. Alternatively, we can also condition M using a query point $q \in P$, allowing us to compute affinities for a single neuron or background fragment $\hat{\mathbf{A}}_{P \times q} = M(P, q)$. We discuss the scalability in the supplement (Sec. D).

Point Cloud Encoding. Inspired by recent progress in neural fields [53], our point cloud encoding procedure consists of three steps. First, we project each point p using a positional frequency encoding γ [43] and a fully connected layer (FC) into D-dimensional embedding space as below:

$$p_{\text{enc}} = \text{FC}(\gamma(p), p),\ \mathbb{R}^3 \rightarrow \mathbb{R}^D. \tag{3}$$

We denote the encoded point cloud as P_{enc}. Second, P_{enc} is transformed into feature set representation $z^{(0)}$ using cross-attention [45] with a learnable queries set $Q \in \mathbb{R}^{D \times C}$ [53] and $P_{\text{enc}} \in \mathbb{R}^{D \times |P|}$ as the key-value pair:

$$z^{(0)} = \text{CrossAttn}(Q, P_{\text{enc}}) \in \mathbb{R}^{D \times C}. \tag{4}$$

Cross-attention is defined as:

$$\text{CrossAttn}(X, X') = \sigma\left(\frac{XW_Q(X'W_K)^T}{\sqrt{D}}\right) XW_V, \tag{5}$$

with X and X' being two input sequences, W_Q, W_K, and W_V denoting three trainable weight matrices, and σ representing the *softmax* function. Here, D is the dimensionality of all C features in the feature set. Q is a randomly initialized trainable parameter in our model. Third, we enable information exchange between all elements z_i in the feature set through a series of $j = 24$ sequential self-attention modules:

$$z^{(j)} = \text{SelfAttn}(z^{(j-1)}) = \text{CrossAttn}(z^{(j-1)}, z^{(j-1)}) \tag{6}$$

Here, $z^{(j)}$ is a feature representation of a (multi-)neuron point cloud after the j self-attention layer. From now on, we use $z^{(24)} = z_P$ for clarity. Note that, unlike other methods [33, 52], z_P has no explicit spatial reference and is thus well suited to fit thin and sparse spatial structures such as neurons flexibly. In summary, the encoder φ uses a series of cross- and self-attention layers to encode P into a feature set z_P (Fig. 3 ⓑ):

$$z_P = \varphi(P)\text{ , with } z_P = \{z_i \in \mathbb{R}^D\}_{i=1}^{C}. \tag{7}$$

In the following sections, we show how to decode z_P into point pair affinities, which are subsequently used for guiding clustering that leads to the label set S.

Affinity Decoding. Next, we input the feature set z_P and point embedding pairs $P_{\text{enc}} \times P_{\text{enc}}$ into the decoder θ to predict the affinity matrix $\hat{\mathbf{A}}_{P \times P}$ (Fig 3 ⓒ):

$$\hat{\mathbf{A}}_{P \times P} = \theta(z_P, P_{\text{enc}} \times P_{\text{enc}}). \tag{8}$$

We can either use all point pairs $P \times P$ to receive the full affinity matrix $\hat{\mathbf{A}}_{P \times P}$, or only use a subset of point pairs $(P \times P)' \subset (P \times P)$. Next, a cross-attention module maps the feature set $z_P \in \mathbb{R}^{D \times C}$ into a set of feature pairs $z_{P \times P} \in \mathbb{R}^{2D \times |P \times P|}$ given $P_{\text{enc}} \times P_{\text{enc}}$:

$$z_{P \times P} = \text{CrossAttn}(P_{enc} \times P_{\text{enc}}, z_P) \in \mathbb{R}^{2D \times |P \times P|}. \tag{9}$$

Here, $|P \times P|$ represents the number of point pairs. Finally, feature pairs $z_{P \times P}$ are decoded into the affinity matrix $\hat{\mathbf{A}}_{P \times P}$ using a MLP : $\mathbb{R}^{2D} \rightarrow \mathbb{R}$. The MLP has a sigmoid final layer to ensure all affinity values are in $[0, 1]$.

3.3. Affinity-guided Point Clustering

We derive the set of segmentation labels S from the predicted affinity matrix $\hat{\mathbf{A}}_{P \times P}$. Since affinities are computed per point pair, we interpret affinity values as a point proximity metric. Thus, we can cluster the point cloud P according to an affinity-based distance metric $\hat{\mathbf{D}}_{P \times P} = \mathbf{1.0} - \hat{\mathbf{A}}_{P \times P}$.

We use feature agglomerative hierarchical clustering [47] to iteratively merge points into clusters according to $\hat{\mathbf{D}}_{P \times P}$, an average linkage criterion and a termination threshold t (Fig. 3ⓓ). Upon convergence of the agglomerative clustering algorithm, all points in a cluster get the same segmentation label $s \in S$ assigned. This approach allows accurate reconstruction of segmentation labels, even if affinity predictions are imperfect. The background class ($s = 0$) is defined by the union of all clusters with less than 30 points. We find that 30 is a good threshold to summarize background fragments into the background class.

3.4. Contrastive Feature Embeddings

We use neuron type classification to demonstrate that our model learns to represent global neuron shapes, as the shape is among the most important contributing factors for neuron type. Given a pretrained affinity prediction model M (Sec. 3.1) and a single neuron point cloud P_s, we retrieve the respective feature set $z_{P_s} \in \mathbb{R}^{D \times C}$ from M (Sec. 3.2). Next, we train a Deep Set [51] $g : \mathbb{R}^{D \times C} \rightarrow \mathbb{R}^E$, that maps z_{P_s} into a lower dimensional contrastive embedding space with dimensionality E. The Deep Set g consists of a feature embedding MLP ζ, a permutation invariant summation operation $\sum$, and an output MLP ρ that projects into the contrastive embedding space:

$$g(z_{P_s}) = \rho(\sum_{i}^{C} \zeta(z_i)). \tag{10}$$

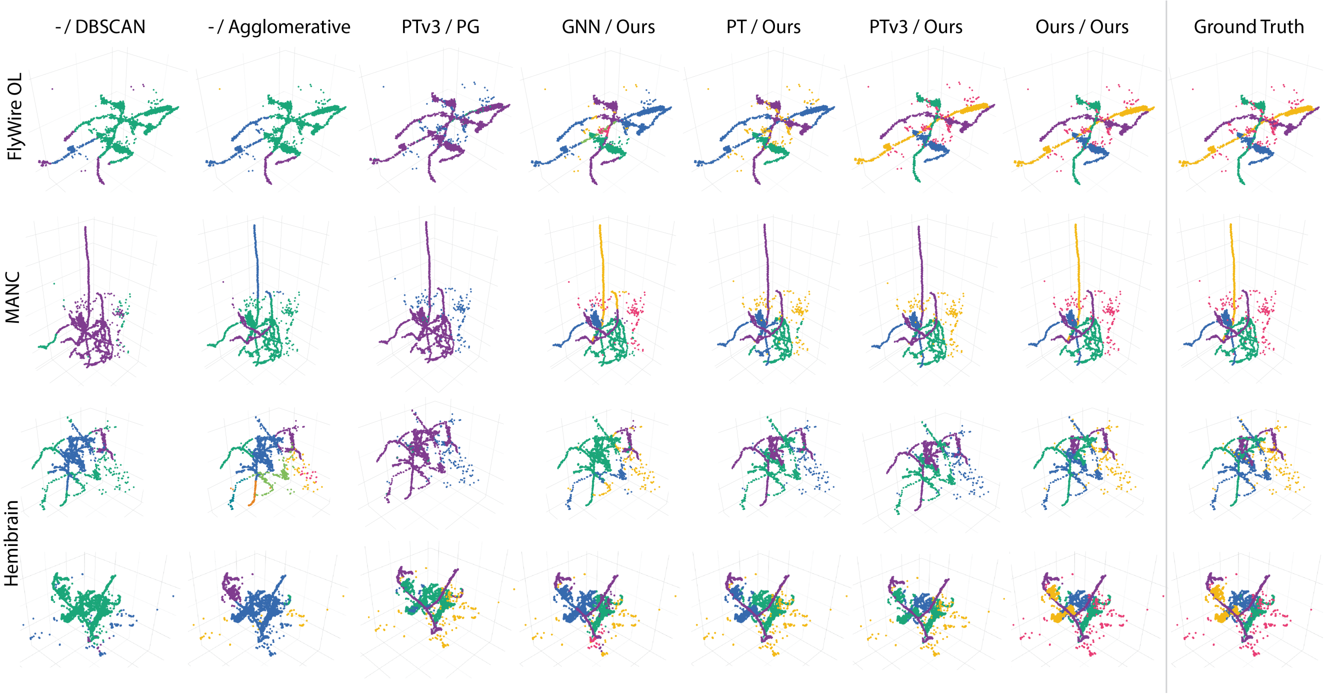

Figure 4. **Automatic Proofreading Examples.** Our approach can reliably correct segmentation errors in (multi-)neuron point clouds. Qualitative results across three datasets demonstrate our model's performance. We compare our approach against unsupervised clustering methods such as DBSCAN and agglomerative clustering, a graph neural network-based approach (Point-GNN), and the commonly used Point-Transformer architecture. Baselines are named as (Encoder/Cluster Decoding). The color represents segmentation labels S.

Additionally, we normalize contrastive embeddings, such that $|g(z_{P_s})| = 1$. While keeping all parameters of M frozen when retrieving z_{P_s}, we train g with a contrastive loss function :

$$L_{\text{contrastive}} = [d_{\text{pos}} - m_{\text{pos}}]_+ + [m_{\text{neg}} - d_{\text{neg}}]_+. \qquad (11)$$

We use neuron type labels to compute Euclidean distances d_{pos} between embeddings derived from point clouds of neurons from the same type. d_{neg} are embedding distances between point clouds from different neuron types. m_{pos} and m_{neg} are the positive and negative margin values, respectively. Finally, an Euclidean k-nearest-neighbors classifier computes neuron-type probabilities for a set of single-neuron contrastive embeddings. We evaluate the performance of our contrastive embeddings in Section 4.5.

4. Experiments

In this section, we evaluate our proposed approach on three benchmark datasets. All three benchmarks are derived from proofread connectomes [7, 36, 42], allowing us to evaluate our model's performance accurately. We report results for the automatic proofreading task and also demonstrate our model's global neuron shape reasoning capability by using its feature representation for neuron type classification.

4.1. Datasets

FlyWire [8]. The FlyWire optic lobe (OL) is part of the first-ever full adult Drosophila connectome and includes well-defined cell type families [32].

MANC [42]. The MANC connectome describes the neural wiring architecture of an adult male fruit fly's ventral nerve cord, responsible for signaling motor output to the fly's wings and legs and receiving sensory input.

Hemibrain [36]. The hemibrain dataset is a proofread connectome reconstruction of a fruit fly's central brain, responsible for navigation and integrating visual information.

Data Construction. Real-world, standardized datasets for automated proofreading do not exist. Therefore, we constructed three benchmark datasets programmatically, derived from state of the art connectomes. We construct artificial reconstruction errors by combining randomly selected neurons and background fragments into a single point cloud for training and testing. We randomly sample $15,000$ neurons for model training and $2,000$ for testing for each connectome. We sample point clouds for each neuron from its respective skeleton, by randomly selecting a subset of the skeleton's vertices and discarding edge information. Skeletons for all datasets are publicly available. Small background fragments (e.g., spine heads or arborizations) are common artifacts in CNN-based image segmentations

Dataset	Metrics	-/DBSCAN	-/Agglo	PTv3/PG	GNN/Ours	PT/Ours	PTv3/Ours	Ours/Ours
FlyWire [8]	VOI ↓	1.19	1.24	1.47	0.82	0.32	0.37	**0.17**
	VOI_s ↓	0.28	0.37	0.21	0.42	**0.07**	0.14	**0.07**
	VOI_m ↓	0.91	0.87	1.26	0.42	0.26	0.24	**0.10**
	ARE ↓	0.42	0.44	0.55	0.24	0.12	0.12	**0.04**
MANC [42]	VOI ↓	0.97	0.97	1.44	0.51	0.27	0.29	**0.20**
	VOI_s ↓	0.21	0.32	0.12	0.20	**0.07**	0.09	0.08
	VOI_m ↓	0.76	0.65	1.32	0.31	0.2	0.20	**0.13**
	ARE ↓	0.33	0.3	0.52	0.15	0.09	0.10	**0.05**
Hemibrain [36]	VOI ↓	1.12	1.06	1.42	0.77	0.39	0.35	**0.23**
	VOI_s ↓	0.40	0.48	0.31	0.39	**0.09**	0.11	**0.09**
	VOI_m ↓	0.72	0.58	1.11	0.38	0.30	0.23	**0.14**
	ARE ↓	0.34	0.35	0.49	0.23	0.14	0.11	**0.05**

Table 1. **Automated Proofreading Evaluation**. Our approach consistently outperforms baseline methods across all three datasets in terms of the VOI, including the split VOI_s and merge VOI_m variants, and the adjusted rand error (ARE). The best scores per row are **bolded**, and the second best scores are underlined. Baselines are named as (Encoder/Cluster Decoding). See Section 4.3 for details about baselines.

and must be attached to their respective parent segment. Background fragments are added to the input by randomly adding small (≤ 8 μm) terminal neuron branches. We randomly vary the number of merged neurons in the range of $[1, ..., 4] \in \mathbb{N}$. The supplemental material shows examples with more (≥ 4) input neurons (Fig. 8).

Data Augmentation. We apply a series of data augmentations to increase data diversity. Before combining neurons into a single point cloud, we first center each neuron around the origin, which helps the model to learn shapes rather than overfitting on global position. Next, we jitter each point by a random factor of $[0, 1\text{um}] \in \mathbb{R}$ and randomly rotate $[0, 200°]$ and translate individual neurons. Finally, we merge randomly selected neurons and background fragments and apply a global scaling factor to ensure all coordinates are within $[-1, 1]$.

4.2. Evaluation Metrics

Automatic Proofreading. We solve an affinity-guided clustering problem, where points of the same neuron form a cluster. Thus, we use standard clustering metrics such as the *variation of information* (VOI) and the *adjusted rand error* (ARE). Additionally, the VOI is decomposed into a split (VOI_s) and merge (VOI_m) component, which measures the degree of over- and under-segmentation, respectively.

Neuron Type Classification. For this multi-class classification problem, we report mean average precision (*mAP*), the Top1 Error, and the Top5 Error. The Top1 error indicates the likelihood that the correct type is *not* the top prediction. The Top5 Error measures the likelihood that the correct neuron type is *not* in the top 5 predictions.

4.3. Baseline Approaches.

We compare against six baselines and report results for different encoder and cluster decoding methods. Baselines are named as (Encoder/Cluster Decoding).

Unsupervised Clustering. We compare our affinity based clustering (Sec. 3.3), to traditional unsupervised approaches such as DBSCAN [10] and agglomerative clustering [47]. We first compute pairwise Euclidean distances between all point pairs $P \times P$ and predict neuron clusters with each respective algorithm. We obtain hyperparameters by performing a grid search on a small training dataset and picking the set of hyperparameters that yields the best VOI score.

GNNs. Graph-based approaches are a set of popular methods for point cloud processing and classification [25]. We compare our approach to Point-GNN [41], a popular message-passing model based on graph convolution. This approach predicts a feature per point, which is decoded to object labels in the original paper. For the purpose of affinity prediction, we add an MLP decoder to the model that decodes point feature pairs to affinity values (GNN/Ours). We reconstruct a radius graph from the point cloud using radius $= 32$ and a maximum of 64 neighbors per point.

Point Transformers. We compare to Point Transformer (PT) [55] and Point Transformer V3 [48] (PTv3). These architectures perform step-wise local neighborhood aggregation and predict features per point. We compare against the original PTv3 instance segmentation approach that uses Point Group [18] for clustering (PTv3/PG). This led to poor performance due to PG's reliance on object centroids for clustering, which are non-discriminative for tightly entangled neurons. We investigated if pairing PT and PTv3 encoders with our affinity-based cluster decoding could rescue their poor performance (PT/Ours, PTv3/Ours).

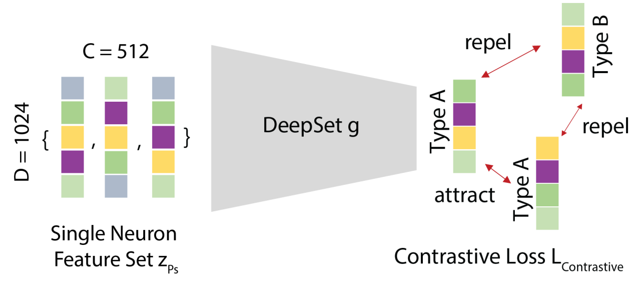

Figure 5. **Neuron Type Contrastive Embeddings.** We project a single neuron feature set z_{P_s} into a lower dimensional embedding space using a DeepSet g, trained using a contrastive loss function $L_{\text{contrastive}}$. In the embedding space, neuron types are predicted with a k-nearest-neighbor classifier using Euclidean distances.

Dataset	Metrics	DGCNN [46]	Ours
FlyWire [8]	mAP ↑	0.62	**0.77**
	Top1 Error ↓	0.39	**0.19**
	Top5 Error ↓	0.07	**0.03**
Hemibrain [42]	mAP ↑	0.53	**0.73**
	Top1 Error ↓	0.38	**0.23**
	Top5 Error ↓	0.08	**0.04**

Table 2. **Neuron Type Classification Results.** Our contrastive embeddings significantly outperform DGCNN-based [46] embeddings for neuron type classification. DGCNN is a GNN-based approach. The best scores are **bolded**.

Contrastive Embeddings. We compare our contrastive embeddings (Sec. 3.4) against a representative GNN called DGCNN [46]. This approach dynamically builds a graph from a point cloud by connecting nearby points with edges. DGCNN implements graph convolution and predicts a feature per single neuron point cloud P_s. Following the training procedure of our DeepSet g (Sec. 3.4), we optimize DGCNN using the contrastive loss function $L_{\text{contrastive}}$. In contrast to our method, which uses a pretrained encoder and only optimizes the deep set parameters, DGCNN is trained end to end (Sec. 3.4).

4.4. Implementation

Automated Proofreading. Each point cloud contains 4288 samples, which includes 1024 points per neuron and up to 6 background fragments, with each up to 32 points. We randomly sample $\{1, ..., 4\}$ neurons during affinity prediction training and testing. For cases where the number of neurons is < 4, zero padding is used to get an equal number of points per batch. We also report data ablation for OOD testing with > 4 neurons (Sec. 4.6). Generally, based on manual analysis of segmentation errors and the datasets used, we find ≤ 4 neurons to be a reasonable training distribution. The feature set z_P uses $C = 512$ channels, each with a dimensionality of $D = 1024$. The encoder φ includes a series of 24 self-attention modules. The decoder's MLP has two layers, each $2D = 2048$ units. The first layer uses ReLU activations, and the output layer uses a Sigmoid activation function. The convergence threshold for the agglomerative clustering algorithm is $t = 0.8$. All affinity prediction experiments are trained for 946 epochs and a batch size of 230 on 8 NVIDIA H100 GPUs. For testing, we used a single H100 GPU. We use five warmup epochs to linearly increase the learning rate from 0 to $lr_{\text{max}} = 1e^{-4}$ and use a cosine decay learning rate scheduler with $lr_{\text{min}} = 1e^{-6}$.

Neuron Type Classification. Here, we use single-neuron point clouds without background fragments, resulting in 1024 points. Using our pretrained encoder, the contrastive Deep Sets are trained for 100 epochs and a batch size of 650 on a single H100 GPU. The DGCNN baseline is trained end to end. The k-nearest neighbor classifier uses Euclidean distances and $k = 15$. For both experiments on FlyWire OL and the Hemibrain data (see Table. 4.5), we exclude neurons whose type has fewer than 40 instances in the respective dataset. Due to the imbalance of the neuron type distribution in both datasets, we rebalance the training dataset so that each type is equally represented. With these constraints, the Flywire training set has 35 unique types, each with 400 representative neurons. The hemibrain dataset has 41 unique neuron types, each with 300 instances. The DeepSet MLP ζ and ρ consist of two linear layers with ReLU activations. ζ uses 1024 input units and 256 units in the hidden and output layers. ρ has 256 input and 32 output units with no activation function in the last layer. The contrastive embedding space has a dimensionality of $E = 32$.

4.5. Main Results

Automated Proofreading. Table 1 shows quantitative comparisons of our approach to six baselines (Sec. 4.3). Our approach (Ours/Ours) consistently outperforms baselines. Broadly, learning-based approaches (GNNs, Transformers & Ours) outperform unsupervised clustering methods such as DBSCAN and Agglomerative Clustering. We also find that conventional instance segmentation techniques such as PTv3 with Point Group (PTv3/PG) [18] lead to poor performance, due to their reliance on object centroids for point clustering. However, combining the PTv3 encoder with our affinity based cluster decoding (PTv3/Ours) improves performance and achieves the second best scores in our experiments. Transformer architectures (PT/Ours, PTv3/Ours, Ours/Ours) are superior overall to GNNs (GNN/Ours). In contrast to Point Transformers, our method (Ours/Ours) produces significantly fewer neuron merge errors as shown in Table 1 (VOI$_m$) and in Figure 4. This observation is consistent across all three benchmark datasets. We attribute these performance gains

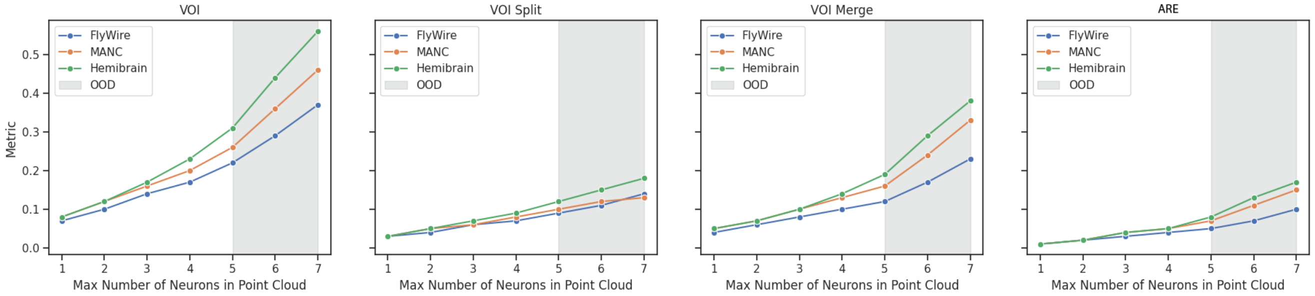

Figure 6. **Input Data Size Ablation.** We train our model with a maximum number of 4 neurons per point cloud. However, our model architecture is agnostic to input size. Hence, we report test set metrics like VOI, VOI$_s$, VOI$_m$, and ARE for individual evaluations with a maximum of 1-7 neurons per point cloud. The grey area indicates out-of-distribution (OOD) performance. While we observe a slow linear decrease in distribution (max 1-4 neurons) and a bias towards under-segmentation for OOD.

to our model's ability to learn a feature representation without explicit spatial references, allowing flexible fitting to thin and sparse spatial objects, such as neurons. Our method depends on a threshold t, which determines the convergence of the agglomerative affinity clustering (Sec. 3.3). We created a small training set for each method and identified t with the highest VOI through a grid search.

Neuron Type Classification. Table 2 shows neuron type classification metrics for both our contrastive embeddings (Sec. 3.4) and the embeddings from our GNN-based baseline DGCNN [46] (Sec. 4.3). Our approach outperforms the baselines for both datasets on all reported metrics. Specifically, the Top1 error metric improved, with a -0.2 for FlyWire and a -0.15 for the Hemibrain over the baseline.

4.6. Ablations

We report two ablation studies for the automatic proofreading task to answer the following questions:

- Question 1: Does our approach generalize to out-of-training distributions regarding the maximum number of neurons merged into a single point cloud?
- Question 2: Does a lower feature dimensionality D & C reduce performance numbers ?

OOD Data Ablation (Q1). We show our model performance on various numbers of neurons in a single point cloud for the automatic proofreading task (Fig 6). Our model was trained for only a maximum number of four neurons per point cloud. Thus, the grey area indicates OOD performance. As expected, we observe a linear performance decrease within distribution (max. 1-4 neurons per point cloud). For OOD evaluations (max. 5 - 7 neurons per point cloud), we identify a bias towards undersegmenting point clouds. In comparison, the VOI$_s$ increases linearly for OOD input data, and VOI$_m$ starts increasing more rapidly. This effect could be mitigated by decreasing the convergence threshold t of the affinity-based agglomerative clustering (Sec. 3.3).

Feature Size	**VOI$_s$↓**	**VOI$_m$↓**	**ARE↓**
$D = 1024, C = 512$	0.07	0.10	0.04
$D = 512, C = 512$	0.09	0.12	0.04
$D = 1024, C = 256$	0.09	0.12	0.04

Table 3. **Feature Size Ablation.** We ablate the size of the feature dimensionality D and the number of elements C in the feature set. Reducing both D or C equally degrades performance metrics.

Feature Set Size (Q2). Table 3 shows that reducing either feature dimensionality D or the number of features C equally decreases VOI$_s$ and VOI$_m$. The ARE error metric does not decrease.

5. Conclusion

The lack of neural network architectures capable of reasoning with the global shapes of neurons has been a major bottleneck for the complete automation of connectome generation. Solving this problem will enable connectome mapping of entire fly brains in mere weeks instead of years, making connectome mapping of entire mouse brains feasible. Our results show significant performance gains for both tasks over various baseline methods and datasets.

Our work can be extended in several ways. Point clouds could be augmented with additional feature vectors based on local image information, for instance, based on SegCLR [6] or synapse type [9]. Neuron embeddings generated by our general-purpose encoder based on a small number of annotations in one brain region or animal species could then be used for the unsupervised discovery of new cell types in other brain regions and species. Our method for proofreading can be extended to tracing neurons in light microscopic images by proofreading point clouds segmented with simple binary thresholding. Finally, decoders could be trained to generate skeleton graph representations and computationally efficient renderings (Gaussian splatting [21]) of neuronal shapes from sparsely sampled point clouds.

References

[1] Larry F Abbott, Davi D Bock, Edward M Callaway, Winfried Denk, Catherine Dulac, Adrienne L Fairhall, Ila Fiete, Kristen M Harris, Moritz Helmstaedter, Viren Jain, and others. The mind of a mouse. *Cell*, 182(6):1372–1376, 2020.

[2] Brendan Celii, Stelios Papadopoulos, Zhuokun Ding, Paul G Fahey, Eric Wang, Christos Papadopoulos, Alexander B Kunin, Saumil Patel, J Alexander Bae, Agnes L Bodor, and others. NEURD: A mesh decomposition framework for automated proofreading and morphological analysis of neuronal EM reconstructions. 2023.

[3] Hanbo Chen, Jiawei Yang, Daniel Maxim Iascone, Lijuan Liu, Lei He, Hanchuan Peng, and Jianhua Yao. TreeMoCo: contrastive neuron morphology representation learning. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, Red Hook, NY, USA, 2024.

[4] Marta Costa, James D Manton, Aaron D Ostrovsky, Steffen Prohaska, and Gregory SXE Jefferis. NBLAST: rapid, sensitive comparison of neuronal structure and construction of neuron family databases. *Neuron*, 91(2):293–311, 2016.

[5] Sven Dorkenwald, Claire E McKellar, Thomas Macrina, Nico Kemnitz, Kisuk Lee, Ran Lu, Jingpeng Wu, Sergiy Popovych, Eric Mitchell, Barak Nehoran, and others. FlyWire: online community for whole-brain connectomics. *Nature methods*, 19(1):119–128, 2022.

[6] Sven Dorkenwald, Peter H Li, Michał Januszewski, Daniel R Berger, Jeremy Maitin-Shepard, Agnes L Bodor, Forrest Collman, Casey M Schneider-Mizell, Nuno Maçarico da Costa, Jeff W Lichtman, and others. Multi-layered maps of neuropil with segmentation-guided contrastive learning. *Nature Methods*, 20(12):2011–2020, 2023.

[7] Sven Dorkenwald, Casey M Schneider-Mizell, Derrick Brittain, Akhilesh Halageri, Chris Jordan, Nico Kemnitz, Manual A Castro, William Silversmith, Jeremy Maitin-Shephard, Jakob Troidl, and others. CAVE: Connectome annotation versioning engine. *bioRxiv*, 2023.

[8] Sven Dorkenwald, Arie Matsliah, Amy R Sterling, Philipp Schlegel, Szi-Chieh Yu, Claire E McKellar, Albert Lin, Marta Costa, Katharina Eichler, Yijie Yin, and others. Neuronal wiring diagram of an adult brain. *Nature*, 634(8032): 124–138, 2024.

[9] Nils Eckstein, Alexander Shakeel Bates, Andrew Champion, Michelle Du, and et al. Neurotransmitter classification from electron microscopy images at synaptic sites in Drosophila melanogaster. *Cell*, 187(10):2574–2594.e23, 2024.

[10] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, and others. A density-based algorithm for discovering clusters in large spatial databases with noise. In *kdd*, pages 226–231, 1996. Issue: 34.

[11] Yimin Fan, Yaxuan Li, Yunhua Zhong, Liang Hong, Lei Li, and Yu Li. Learning meaningful representation of single-neuron morphology via large-scale pre-training. *Bioinformatics*, 40(Supplement_2):ii128–ii136, 2024.

[12] Jan Funke, Fabian Tschopp, William Grisaitis, Arlo Sheridan, Chandan Singh, Stephan Saalfeld, and Srinivas C Turaga. Large scale image segmentation with structured loss based deep learning for connectome reconstruction. *IEEE transactions on pattern analysis and machine intelligence*, 41(7):1669–1680, 2018.

[13] Benjamin Gallusser and Martin Weigert. Trackastra: Transformer-based cell tracking for live-cell microscopy. In *European Conference on Computer Vision*, pages 467–484. Springer, 2024.

[14] Felix Gonda, Xueying Wang, Johanna Beyer, Markus Hadwiger, Jeff W Lichtman, and Hanspeter Pfister. VICE: Visual identification and correction of neural circuit errors. In *Computer graphics forum*, pages 447–458, 2021. Issue: 3.

[15] Daniel Haehn, Verena Kaynig, James Tompkin, Jeff W Lichtman, and Hanspeter Pfister. Guided proofreading of automatic segmentations for connectomics. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9319–9328, 2018.

[16] Michał Januszewski, Jörgen Kornfeld, Peter H Li, Art Pope, Tim Blakely, Larry Lindsey, Jeremy Maitin-Shepard, Mike Tyka, Winfried Denk, and Viren Jain. High-precision automated reconstruction of neurons with flood-filling networks. *Nature methods*, 15(8):605–610, 2018.

[17] Jiaxiang Jiang, Michael Goebel, Cezar Borba, William Smith, and BS Manjunath. A robust approach to 3D neuron shape representation for quantification and classification. *BMC bioinformatics*, 24(1):366, 2023.

[18] Li Jiang, Hengshuang Zhao, Shaoshuai Shi, Shu Liu, Chi-Wing Fu, and Jiaya Jia. Pointgroup: Dual-set point grouping for 3d instance segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and Pattern recognition*, pages 4867–4876, 2020.

[19] Justin Joyce, Rupasri Chalavadi, Joey Chan, Sheel Tanna, Daniel Xenes, Nathanael Kuo, Victoria Rose, Jordan Matelsky, Lindsey Kitchell, Caitlyn Bishop, and others. A Novel Semi-automated Proofreading and Mesh Error Detection Pipeline for Neuron Extension. *bioRxiv*, 2023.

[20] Lida Kanari, Paweł Dłotko, Martina Scolamiero, Ran Levi, Julian Shillcock, Kathryn Hess, and Henry Markram. A topological representation of branching neuronal morphologies. *Neuroinformatics*, 16:3–13, 2018. Publisher: Springer.

[21] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3D Gaussian Splatting for Real-Time Radiance Field Rendering. 2023.

[22] Reem Khalil, Sadok Kallel, Ahmad Farhat, and Pawel Dlotko. Topological sholl descriptors for neuronal clustering and classification. *PLOS Computational Biology*, 18(6): e1010229, 2022.

[23] Longxin Le and Yimin Wang. 3D-aware neural network for analyzing neuron morphology. In *2024 5th International Conference on Computer Engineering and Application (ICCEA)*, pages 101–104, Hangzhou, China, 2024. IEEE.

[24] Kisuk Lee, Jonathan Zung, Peter Li, Viren Jain, and H. Sebastian Seung. Superhuman Accuracy on the SNEMI3D Connectomics Challenge, 2017. Version Number: 1.

[25] Dilong Li, Chenghui Lu, Ziyi Chen, Jianlong Guan, Jing Zhao, and Jixiang Du. Graph Neural Networks in Point Clouds: A Survey. *Remote Sensing*, 16(14):2518, 2024.

[26] Hanyu Li, Michał Januszewski, Viren Jain, and Peter H Li. 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*, pages 88–98. Springer, 2020.

[27] Minghui Liao, Guojia Wan, and Bo Du. Joint Learning Neuronal Skeleton and Brain Circuit Topology with Permutation Invariant Encoders for Neuron Classification. In *AAAI*, pages 197–205, 2024. Issue: 1.

[28] Zudi Lin, Donglai Wei, Won-Dong Jang, Siyan Zhou, and others. Two stream active query suggestion for active learning in connectomics. In *ECCV 2020*, pages 103–120. Springer, 2020.

[29] Zudi Lin, Donglai Wei, Aarush Gupta, Xingyu Liu, Deqing Sun, and Hanspeter Pfister. Structure-Preserving Instance Segmentation via Skeleton-Aware Distance Transform. In *MICCAI*, pages 529–539. Springer, 2023.

[30] Xiaoyu Liu, Miaomiao Cai, Yinda Chen, Yueyi Zhang, Te Shi, Ruobing Zhang, Xuejin Chen, and Zhiwei Xiong. Cross-dimension affinity distillation for 3d em neuron segmentation. In *CVPR*, pages 11104–11113, 2024.

[31] Yixiong Liu, Qihua Chen, and Xuejin Chen. Neuproofreader: An Interactive Proofreading System with Suggestive Prompts for Connectomics. In *2024 IEEE International Conference on Multimedia and Expo Workshops (ICMEW)*, pages 1–2. IEEE, 2024.

[32] Arie Matsliah, Szi-chieh Yu, Krzysztof Kruk, Doug Bland, Austin T Burke, Jay Gager, James Hebditch, Ben Silverman, Kyle Patrick Willie, Ryan Willie, and others. Neuronal parts list and wiring diagram for a visual system. *Nature*, 2024.

[33] Songyou Peng, Michael Niemeyer, Lars Mescheder, Marc Pollefeys, and Andreas Geiger. Convolutional occupancy networks. In *ECCV*, pages 523–540. Springer, 2020.

[34] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE CVPR*, 2017.

[35] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *NeurIPS*, 30, 2017.

[36] Louis K Scheffer, C Shan Xu, Michal Januszewski, Zhiyuan Lu, Shin-ya Takemura, Kenneth J Hayworth, Gary B Huang, and et al. A connectome and analysis of the adult Drosophila central brain. *eLife*, 9:e57443, 2020.

[37] Martin Schmidt, Alessandro Motta, Meike Sievers, and Moritz Helmstaedter. RoboEM: automated 3D flight tracing for synaptic-resolution connectomics. *Nature Methods*, 21(5):908–913, 2024.

[38] Philipp J Schubert, Sven Dorkenwald, Michał Januszewski, Viren Jain, and Joergen Kornfeld. Learning cellular morphology with neural networks. *Nature communications*, 10 (1):2736, 2019.

[39] Alexander Shapson-Coe, Michał Januszewski, Daniel R Berger, Art Pope, Yuelong Wu, Tim Blakely, Richard L Schalek, Peter H Li, Shuohong Wang, Jeremy Maitin-Shepard, and others. A petavoxel fragment of human cerebral cortex reconstructed at nanoscale resolution. *Science*, 384(6696):eadk4858, 2024.

[40] Arlo Sheridan, Tri M Nguyen, Diptodip Deb, Wei-Chung Allen Lee, Stephan Saalfeld, Srinivas C Turaga, Uri Manor, and Jan Funke. Local shape descriptors for neuron segmentation. *Nature methods*, 20(2):295–303, 2023.

[41] Weijing Shi, Ragunathan, and Rajkumar. Point-GNN: Graph Neural Network for 3D Object Detection in a Point Cloud, 2020. Version Number: 1.

[42] Shin-ya Takemura, Kenneth J Hayworth, Gary B Huang, Michal Januszewski, Zhiyuan Lu, Elizabeth C Marin, Stephan Preibisch, C Shan Xu, John Bogovic, Andrew S Champion, and others. A connectome of the male drosophila ventral nerve cord. *BioRxiv*, pages 2023–06, 2023.

[43] Matthew Tancik, Pratul Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. *NeurIPS*, 33:7537–7547, 2020.

[44] Hidetoshi Urakubo and Torsten Bullmann. UNI-EM: an environment for deep neural network-based automated segmentation of neuronal electron microscopic images. *Scientific reports*, 9(1), 2019.

[45] A Vaswani. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017.

[46] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *ACM Transactions on Graphics (tog)*, 38(5):1–12, 2019.

[47] Joe H Ward Jr. Hierarchical grouping to optimize an objective function. *Journal of the American statistical association*, 58(301):236–244, 1963. Publisher: Taylor & Francis.

[48] Xiaoyang Wu and et al. Point Transformer V3. In *CVPR*, 2024.

[49] Xiaoyang Wu, Yixing Lao, Li Jiang, Xihui Liu, and Hengshuang Zhao. Point transformer v2: Grouped vector attention and partition-based pooling. *Advances in Neural Information Processing Systems*, 35:33330–33342, 2022.

[50] Feng Xiong, Peng Xie, Zuohan Zhao, Yiwei Li, Sujun Zhao, Linus Manubens-Gil, Lijuan Liu, and Hanchuan Peng. DSM: Deep sequential model for complete neuronal morphology representation and feature extraction. *Patterns*, 5 (1):100896, 2024.

[51] Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Russ R Salakhutdinov, and Alexander J Smola. Deep sets. *NeurIPS*, 30, 2017.

[52] Biao Zhang, Matthias Nießner, and Peter Wonka. 3dilg: Irregular latent grids for 3d generative modeling. *Advances in Neural Information Processing Systems*, 2022.

[53] Biao Zhang, Jiapeng Tang, Matthias Niessner, and Peter Wonka. 3dshape2vecset: A 3d shape representation for neural fields and generative diffusion models. *ACM Transactions on Graphics (TOG)*, 42(4):1–16, 2023.

[54] Cheng Zhang, Haocheng Wan, Xinyi Shen, and Zizhao Wu. Patchformer: An efficient point transformer with patch attention. In *Proceedings of the IEEE/CVF CVPR*, pages 11799–11808, 2022.

[55] Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip Torr, and Vladlen Koltun. Point Transformer, 2020.

[56] Jonathan Zung, Ignacio Tartavull, Kisuk Lee, and H. Sebastian Seung. An Error Detection and Correction Framework for Connectomics, 2017. Version Number: 2.

Global Neuron Shape Reasoning with Point Affinity Transformers

Supplementary Material

A. Additional Ablations

Figure 9 illustrates the sensitivity of our method and the respective baselines to the convergence threshold of the affinity-guided agglomeration clustering (Sec. 3.3). Our method outperforms baselines across all possible thresholds in terms of the variation of information (VOI) metric, its split (VOI$_s$), and merges (VOI$_m$) decompositions, and the adjusted rand error (ARAND). For all metrics, lower values indicate better performance.

B. Additional Qualitative Results

Automatic Proofreading. We provide additional qualitative results for the automatic proofreading use case. Figure 10 and Figure. 11 show examples for single-, double-, three- and four-neuron point clouds. Irrespective of the number of neurons, our approach outperforms the respective baselines. Notably, our approach reliably avoids over-segmenting single neuron point clouds (Fig 10).

Results with Parallel Fibers. Figure 7 shows additional qualitative results for particularly challenging automated proofreading cases involving parallel neuronal fibers. We show a success and failure case (top/bottom row).

More Input Neurons. The number of merged neurons depends on the agglomeration strategy applied to the CNN-based volumetric segmentation, and is thus a hyperparameter in our framework. Figure 8 shows additional qualitative results for 7 and 8 input neurons.

Neuron Type Classification. Figure 13 shows the detailed confusion matrices for quantitative results reported in Table 5. The numbers in each tile represent the percentage (%) of neuron instances of a specific ground truth type, which were classified respectively. Additionally, Figure 12 shows three examples of different neuron morphologies. For each neuron, we also show the Top3 predicted neuron types based on our classifier (Sec. 3.4). The ground truth type is highlighted in green. For all three examples, the correct type is within the top two predicted neuron types.

Videos. We attach videos to the supplement that show qualitative results for our approach and baselines.

C. Details for Baseline Implementations

We use the default hyperparameter configuration for the Point-GNN [41], PT [55], PTv3 [48], PG [18] and DGCNN [46] baselines, as reported in the respective papers. The PTv3 encoder additionally uses a grid size parameter of 0.001 to achieve the best performance with our data. All affinity decoding models were trained using the binary cross entropy loss. The PT baseline uses a convergence threshold for the agglomeration clustering of $t = 0.9$, PTv3 uses $t = 0.8$, and Point-GNN uses $t = 0.7$. We determined those parameters using grid search on a small training set. The unsupervised clustering baselines (DBSCAN/Agglo) use a threshold of $t = 0.1$ and $t = 0.3$, respectively. Additionally, DBSCAN uses a *min_samples* hyperparameter of 150. All learning-based baselines were trained until the respective loss converged.

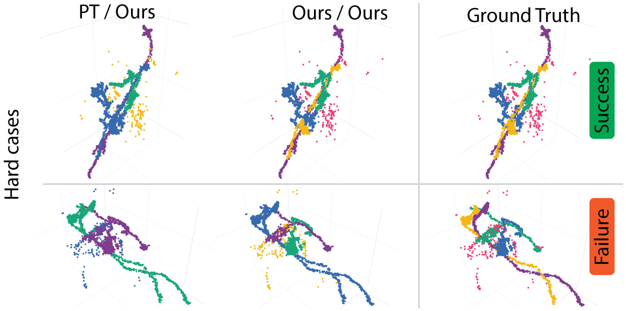

Figure 7. **Qualitative Results** involving parallel fibers. Top row shows a success case. The bottom row shows a failure case.

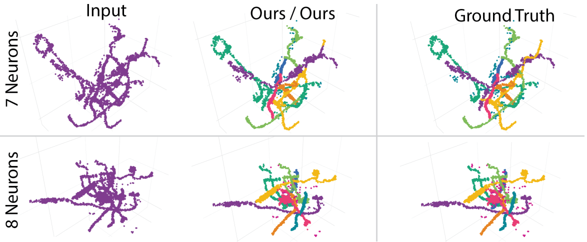

Figure 8. **More Neurons.** Examples with 7 / 8 complete neurons.

D. Scalability

Our model has a common point cloud encoder for both tasks and a task-dependent decoder. The encoder scales linearly with point cloud size through a constant sized feature embedding (Fig. 3b). The neuron typing decoder scales constantly with point cloud size. The affinity decoder naively scales quadratically with point cloud size. However, strategies that decode affinities between supervoxels only require computation scaling quadratically with supervoxels, not points. For instance, with $4,288$ input points, decoding affinities between $2,000$ supervoxels only takes 70 ms on one H100 GPU. For extremely large inputs, such as $30,000$ points grouped into $7,500$ supervoxels, inference takes around 750 ms using one H100 GPU.

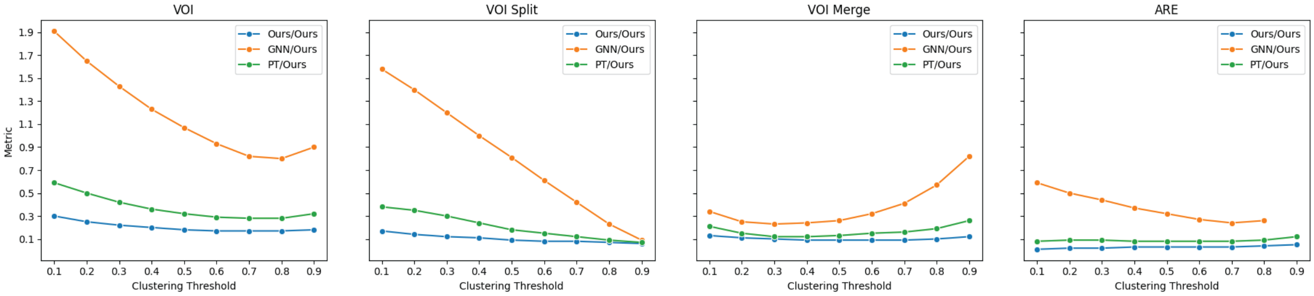

Figure 9. **Clustering Threshold Ablation.** These charts plot the agglomeration clustering threshold t (Sec. 3.3) relative to automated proofreading performance metrics for the FlyWire dataset. Here, we focus on learning-based approaches like GNN/Ours [41], PT/Ours [55], and our proposed approach. While performance fluctuations occur as the clustering threshold changes, our method consistently performs better than the reported baselines. All plots here are based on the FlyWire dataset.

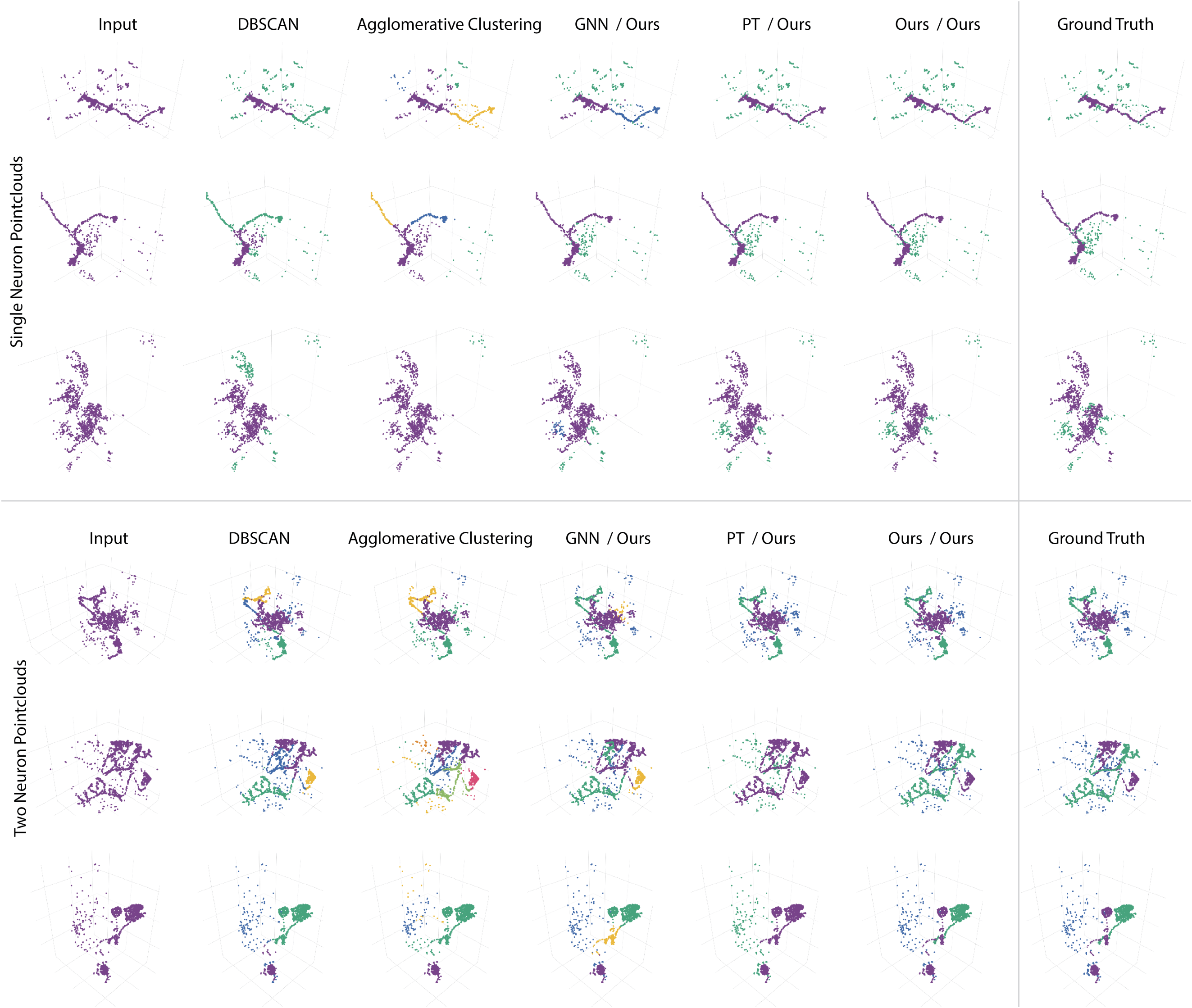

Figure 10. **Additional Qualitative Results for Single & Two Neuron Pointclouds.** We show additional examples for single neuron point clouds (top half) and two neuron point clouds (bottom half). Our approach reliably detects single neurons and does not oversegment them. Qualitative examples also show that our approach is more accurate for two-neuron point clouds than the shown baselines.

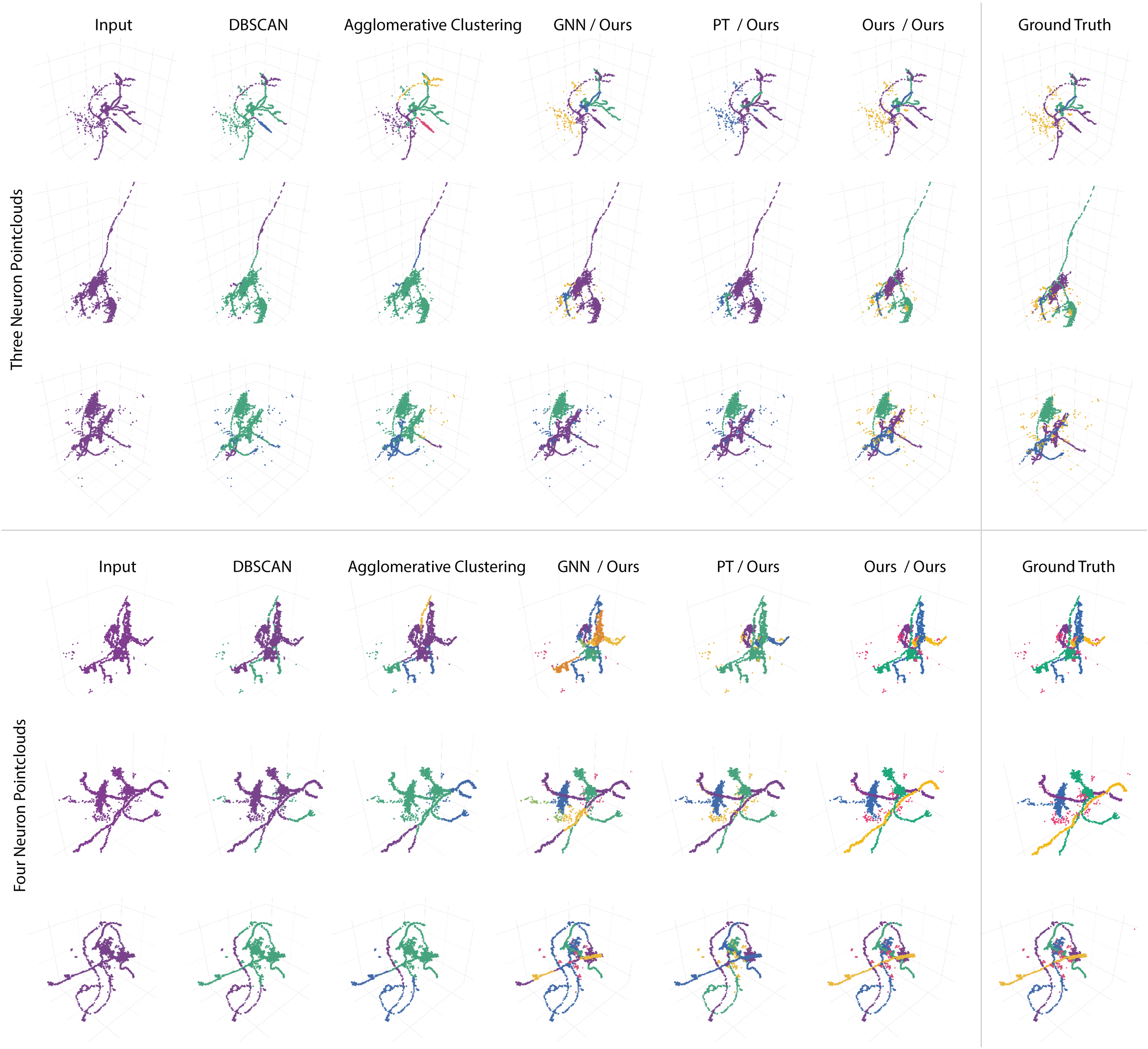

Figure 11. **Additional Qualitative Results for Three & Four Neuron Point Clouds.** The top half of the figure shows three neuron point clouds, while the lower half shows point clouds with four neurons. Our approach reliably detects whole neuron morphologies and can segment the point cloud into individual neurons. Notably, most baselines struggle with undersegmentation.

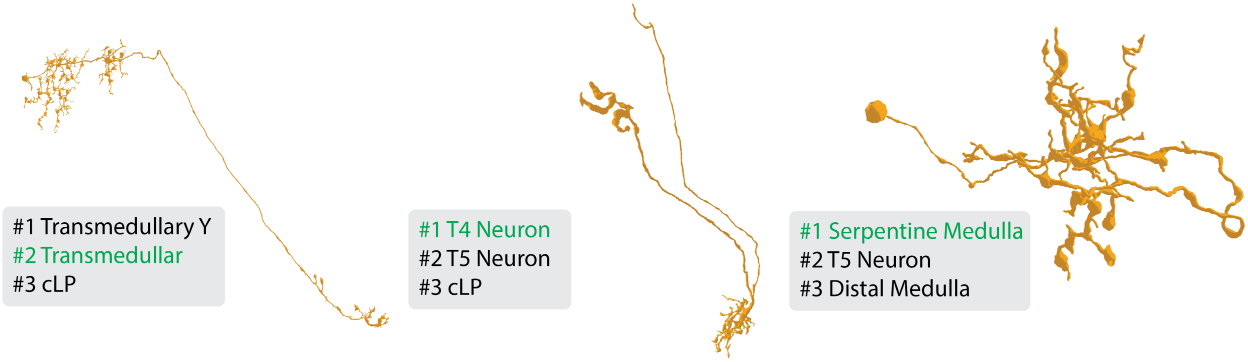

Figure 12. **Neuron Type Examples & Classifier Predictions.** We show three exemplary neuron morphologies and their Top3 predicted types using our KNN classifier. The ground truth type is shown in green. The correct type is within the top 2 predictions in all three examples. Table. 2 and Section 3.4 show more details and results.

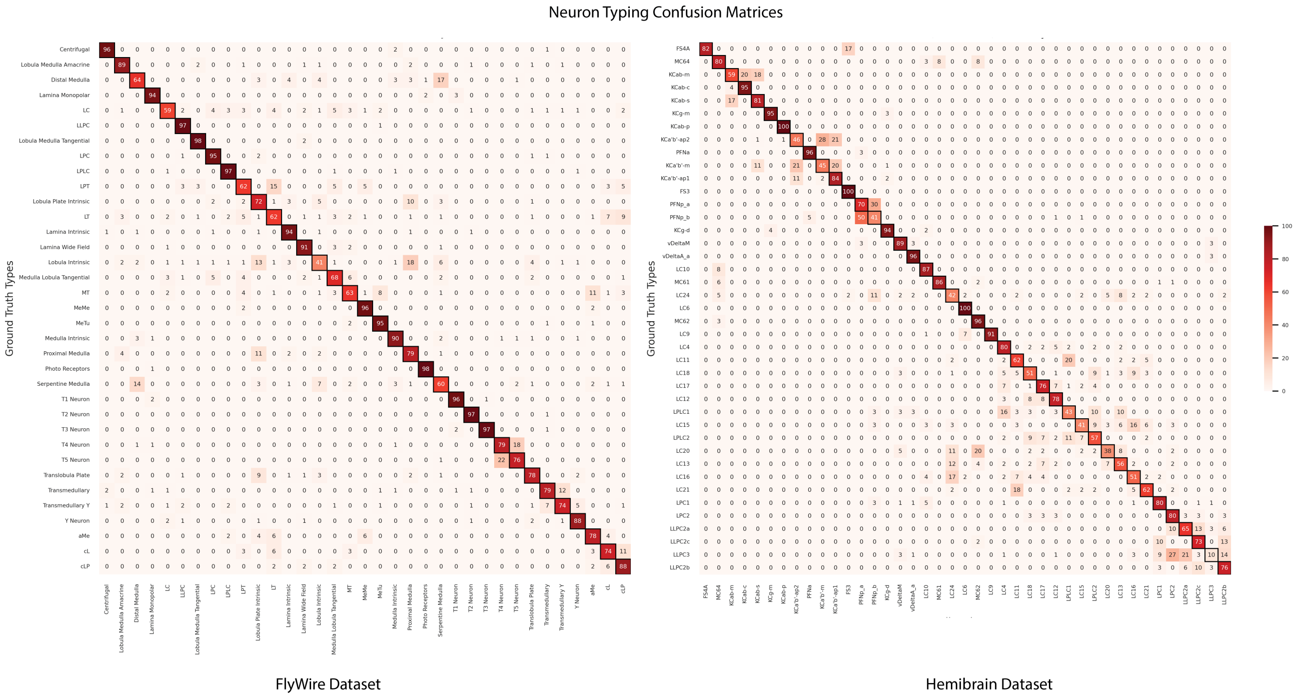

Figure 13. **Neuron Typing Confusion Matrices.** Here, we show neuron type classification confusion matrices for the FlyWire OL and the Hemibrain dataset. Both matrices are normalized, and percentage values (%) are shown in each matrix tile. The x-axis represents predicted types, while the y-axis represents ground truth types. More details are shown in Section 3.4.