# Deep Neural Networks to Register and Annotate the Cells of the *C. elegans* Nervous System

**Adam A. Atanas[1], Alicia Kun-Yang Lu[1], Jungsoo Kim[1], Saba Baskoylu[1], Di Kang[1], Talya S. Kramer[1], Eric Bueno[1], Flossie K. Wan[1], Steven W. Flavell[1,*]**

[1]Picower Institute for Learning & Memory, Department of Brain & Cognitive Sciences, Massachusetts Institute of Technology, Cambridge, MA, USA

[*]Corresponding Author: flavell@mit.edu

## ABSTRACT

Aligning and annotating the heterogeneous cell types that make up complex cellular tissues remains a major challenge in the analysis of biomedical imaging data. Here, we present a series of deep neural networks that allow for automatic non-rigid registration and cell identification in the context of the nervous system of freely-moving *C. elegans*. A semi-supervised learning approach was used to train a *C. elegans* registration network (BrainAlignNet) that aligns pairs of images of the bending *C. elegans* head with single pixel-level accuracy. When incorporated into an image analysis pipeline, this network can link neuronal identities over time with 99.6% accuracy. A separate network (AutoCellLabeler) was trained to annotate >100 neuronal cell types in the *C. elegans* head based on multi-spectral fluorescence of genetic markers. This network labels >100 different cell types per animal with 98% accuracy, exceeding individual human labeler performance by aggregating knowledge across manually labeled datasets. Finally, we trained a third network (CellDiscoveryNet) to perform unsupervised discovery and labeling of >100 cell types in the *C. elegans* nervous system by analyzing unlabeled multi-spectral imaging data from many animals. The performance of CellDiscoveryNet matched that of trained human labelers. These tools will be useful for a wide range of applications in *C. elegans* research and should be straightforward to generalize to many other applications requiring alignment and annotation of dense heterogeneous cell types in complex tissues.

## INTRODUCTION

Optical imaging of dense cellular tissues is widespread in biomedical research. Recently developed methods to label cells with highly multiplexed fluorescent probes should soon make it feasible to determine the heterogeneous cell types in any given sample[1–3]. However, it remains challenging to extract critical information about cell identity and position from fluorescent imaging data. Aligning images within or across animals that have non-rigid deformations can be inefficient and lack cellular-level accuracy. Additionally, annotating cell types in a given sample can involve time-consuming manual labeling and often only results in coarse labeling of the main cell classes, rather than full annotation of the vast number of defined cellular subtypes.

Deep neural networks provide a promising avenue for aligning and annotating complex images of fluorescently-labeled cells with high levels of efficiency and accuracy[4]. Deep learning has generated high-performance tools to segment cells from background in images[5,6]. In addition, deep learning approaches have proven useful for non-rigid image registration in the context of medical image alignment[7]. However, this has not been as widely applied to align images of fluorescently labeled cells, which requires micron-level accuracy. Automated cell annotation using clustering approaches, for example applied to single-cell RNA sequencing data, has been widely adopted[8]. Recent studies have also shown the feasibility of using deep learning applied on image features[9] or raw imaging data to label major cell classes[8,10,11]. However, these methods are still not sufficiently advanced to label the potentially hundreds of cellular subtypes in images of complex tissues. In addition, fully unsupervised discovery of the many distinct cell types in cellular imaging data remains an unsolved challenge.

There is considerable interest in using these methods to automatically align and annotate cells in the nervous system of *C. elegans*, which consists of 302 uniquely identifiable neurons[12–14]. The optical transparency of the animal enables *in vivo* imaging of fluorescent indicators of neural activity at brain-wide scale.[15,16] Advances in closed-loop tracking made this imaging feasible in freely-moving animals.[17,18] These approaches are being used to map the relationship between brain-wide activity and flexible behavior (reviewed in[19,20]). However, the animal bends and warps its head as it moves, resulting in non-rigid deformations of the densely-packed cells in its nervous system. Fully automating the alignment and annotation of cells in *C. elegans* imaging data would facilitate high-throughput and high-SNR brain-wide calcium imaging. These methods could also be applied to unsolved problems in quantifying reporter gene expression, developmental trajectories, and more.

Previous studies have described methods to align and annotate cells in multi-cellular imaging datasets from *C. elegans* and species with related imaging challenges like *Hydra*. Datasets from freely-moving animals pose an especially challenging case. Methods for aligning cells across timepoints in moving datasets include approaches that link neurons across adjacent timepoints[21–23], as well as approaches that use signal demixing[24], alignment of body position markers using anatomical constraints[25,26], or registration/clustering/matching based on features of the neurons, such as their centroid positions[27–32]. Targeted data augmentation combined with deep learning applied to raw images has recently been used to reduce manual labeling time during cell alignment.[33] Deep learning applied to raw images has also been used to identify specific image features, like multi-cellular structures in *C. elegans*.[34] We have previously applied non-rigid registration to full fluorescent images from brain-wide calcium imaging datasets to perform neuron alignment, but performing this complex image alignment via gradient descent is very slow, taking multiple days to process a single animal's data even on a computing cluster[35]. In summary, all of these current methods for neuron alignment are constrained by a tradeoff between alignment accuracy and time spent processing each dataset, either due to manually labeling subsets of neurons or computing the complex alignments that actually yield >95% alignment accuracy.

For *C. elegans* neuron class annotation, ground-truth measurements of neurons' locations in the head have allowed researchers to develop atlases describing the statistical likelihood of finding a given neuron in a given location[36–42]. Some of these atlases have utilized the NeuroPAL transgene in which four fluorescent proteins are expressed in genetically-defined sets of cells, allowing users to manually determine their identity based on multi-spectral fluorescence and neuron position[40–42]. However, this manual labeling is time-consuming (hours per dataset), and statistical approaches to automate neuron annotation based on manual labeling have still not achieved human-level performance (>95% accuracy).

Here we describe deep neural networks that solve these alignment and annotation tasks. First, we trained a neural network (BrainAlignNet) that can perform non-rigid registration to align images of the worm's head from different timepoints in freely-moving data. It is >600-fold faster than our previous gradient descent-based approach using elastix[35] and aligns neurons with 99.6% accuracy. Second, we trained a neural network (AutoCellLabeler) that annotates the identity of each *C. elegans* neuron in the head based on multi-spectral NeuroPAL labels. This network achieves 98% accuracy; versions trained on subsets of the fluorescent channels in NeuroPAL also achieve high performance. Finally, we trained a different network (CellDiscoveryNet) that can perform unsupervised discovery and labeling of >100 cell types of the *C. elegans* nervous system by comparing unlabeled NeuroPAL images across animals. Overall, our results reveal how to train neural networks to align and annotate cells in complex cellular imaging data with high performance.

## RESULTS

### BrainAlignNet: a neural network that registers cells in the deforming head of freely-moving *C. elegans*

When analyzing neuronal calcium imaging data, it is essential to accurately link neurons' identities over time to construct reliable calcium traces. This task is challenging in freely-moving animals where the nervous system is warped on sub-second timescales by animal movement. Therefore, we sought to develop a fast and accurate method to perform non-rigid image registration that can deal with these warping images. Previous studies have described such methods for non-rigid registration of point clouds (e.g. neuron centroid positions)[28–30,43], but, as we describe below, we found that performing full image alignment allows for higher accuracy neuron position alignments.

To solve this task, we used a previously-described network architecture[44,45] that takes as input a pair of 3-D images (i.e. volumes of fluorescent imaging data of the head of the worm) from different timepoints of the same neural recording (Fig. 1A). The network is tasked with determining how to warp one 3-D image (termed the "moving image") so that it resembles the other 3-D image (termed the "fixed" image). Specifically, the network outputs a dense displacement field (DDF), a pixel-wise coordinate transformation function designed to indicate which points in the moving and fixed images are the same (see Methods). The moving image is then transformed through this DDF to create a warped moving image, which should look like the fixed image. This network was selected because its LocalNet architecture (a modified 3-D U-Net) allows it to do the feature extraction and image reconstruction necessary to solve the task. To train and evaluate the network, we used data from freely-moving animals expressing both pan-neuronal NLS-GCaMP and NLS-tagRFP, but only provided the tagRFP images to the network, since this fluorophore's brightness should remain static over time. Since Euler registration of images (rotation and translation) is simple, we performed Euler registration on the images using a GPU-accelerated grid search prior to inputting them into the network. During training, we also provided the network with the locations of the centroids of matched neurons found in both images, which were available for these training and validation data since we had previously used gradient descent to solve those registration problems ("registration problem" here is defined as a single image pair that needs to be aligned) and link neurons' identities[35]. The centroid locations are only used for network training and are not required for the network to solve registration problems after training. The loss function that the network was tasked with minimizing had three components: (1) image loss: the Local squared zero-Normalized Cross-Correlation (LNCC) of the fixed and warped moving RFP images, which takes on a lower value when the images are more similar; (2) centroid alignment loss: the average of the Euclidean distances between the matched centroid pairs, where lower values indicate better alignment; and (3) regularization loss: a term that increases the overall loss the more that the images are deformed in a non-rigid manner (in particular, penalizing image scaling and scrambling of adjacent pixels; see Methods).

We trained and validated the network on 5,176 and 1,466 image pairs, respectively, over 300 epochs, at which point the validation loss plateaued (Fig. 1B). We then evaluated network performance on a separate set of 447 image pairs reserved for testing that were recorded from five different animals. On average, the network improved the Normalized Cross-Correlation (NCC) from 0.577 in the input image pairs to 0.947 in the registered image pairs – the maximum achievable score is 1 (Fig. 1C shows example of centroid positions; Fig. 1D shows image example; Fig. 1E shows both). The average distance between aligned centroids was 1.45 pixels (Fig. 1F). These results were only modestly different depending on the animal or the exact registration problem being solved (Extended Data Fig. 1A-C).

To determine which features of the network were critical for its performance, we trained the network under conditions where we omitted either the centroid alignment loss, the regularization loss, or the image loss. In the first case, the network would not be able to learn based on whether the neuron centroids were well-aligned; in the second case, there would be no constraints on the network performing any type of deformation to solve the task; in the third case, the deformations that the network learned to apply could only be learned from the alignment of the centroids, not the raw tagRFP images. Registration performance of each network was evaluated using the NCC and centroid distance, which quantify the quality of tagRFP image alignment and centroid alignment, respectively (Fig. 1F). While the NCC scores were similar for the full network and the no-regularization and no-centroid alignment networks, other performance metrics like centroid distance were significantly impaired by the absence of centroid alignment loss or regularization loss (Fig. 1E-F). This suggests that in the absence of centroid alignment loss or regularization loss, the network learns how to align the tagRFP images, but does so using unnatural deformations that do not reflect how the worm bends. In the case of the no-image loss network, all performance metrics, including both image and centroid alignment, were impaired compared to the full network (Fig. 1F). This suggests that allowing the network to learn how to warp the RFP images also enhances the network's ability to learn how to align the neuron positions (i.e. centroids).

The finding that the centroid positions were precisely aligned by the full network indicates that the centers of the neurons were correctly registered by the network. However, it does not ensure that all of the pixels that comprise a neuron are being correctly registered, which could be important for subsequent feature extraction from the aligned images. For example, it is formally possible to have perfect RFP image alignment in a context where the pixels from one neuron in the moving RFP image are scrambled to multiple neuron locations in the warped moving RFP image. In fact, we observed this in our first efforts to build such a network, where the loss function was only composed of the image loss. As an additional control to test for this possibility in our trained networks, we examined the network's performance on data from a different strain that expresses pan-neuronal NLS-mNeptune (analogous to the pan-neuronal NLS-tagRFP) and *eat-4*::NLS-GFP, which is expressed in ~40% of the neurons in the *C. elegans* head (Fig. 1G shows example image). If the pixels within the neurons are being correctly registered, then applying image registration to the GFP channel for these image pairs should result in highly correlated images (i.e., a high NCC value close to 1). If the pixels within neurons are being scrambled, then these images should not be well-aligned. We used the DDF that the network learned from pan-neuronal mNeptune data to register the corresponding *eat-4*::NLS-GFP images from the same timepoints and found that this resulted in high-quality GFP image alignment (Fig. 1H). In contrast, while the no-centroid alignment and no-regularization networks output a DDF that successfully aligned the RFP images, applying this DDF to corresponding GFP images resulted in poor GFP image registration (Fig. 1H shows that the no-centroid alignment network aligns the RFP channel, but not the GFP channel, in the *eat-4*::NLS-GFP strain). This further suggests that these reduced networks lacking centroid alignment or regularization loss are aligning the RFP images through unnatural image deformations. Altogether, these results suggest that the full ***Brain Align****ment Neural* ***Net****work* (**BrainAlignNet**) can perform non-rigid registration on pairs of images from freely-moving brain-wide calcium imaging data.

The registration problems included in the training, validation, and test data above were pulled from a set of registration problems that we had been able to solve with gradient descent (example images in Extended Data Fig. 1D). These problems did not include the most challenging cases, for example when the two images to be registered had the worm's head bent in opposite directions (though we note that it did include substantial non-rigid deformations). We next asked whether a network trained on arbitrary registration problems, including those that were not solvable with gradient descent (example images in Extended Data Fig. 1E), could obtain high performance. For this test, we also omitted the Euler registration step that we performed in advance of network training, since the goal was to test whether this network architecture could solve any arbitrary *C. elegans* head alignment problem. For this analysis, we used the same loss function as the successful network described above. We also increased the amount of training data from 5,176 to 335,588 registration problems. The network was trained for 300 epochs, at which point the validation loss plateaued. However, the test performance of the network was not high in terms of image alignment or centroid alignment (Extended Data Fig. 1F). This suggests that additional approaches may be necessary to solve these more challenging registration problems. Overall, our results suggest that, provided that there is an appropriate loss function, a deep neural network can perform non-rigid registration problems to align neurons across the *C. elegans* head with high speed and accuracy.

## Integration of BrainAlignNet into a complete calcium imaging processing pipeline

The above results suggest that BrainAlignNet can perform high quality image alignments. These alignments are only valuable if they enable accurate linking of neurons over time. To test whether performance was sufficient for this, we incorporated BrainAlignNet into our existing image analysis pipeline for brain-wide calcium imaging data and compared the results to our previously-described pipeline, which used gradient descent to solve image registration[35]. This image analysis pipeline, the **A**utomated **N**euron **T**racking **S**ystem for **U**nconstrained **N**ematodes (ANTSUN), includes steps for neuron segmentation (via a 3D U-Net), image registration, and linking of neurons' identities (Fig. 2A). Several steps are required to link neurons' identities based on image registration. First, image registration defines a coordinate transformation between the two images, which is then applied to the segmented neuron ROIs, warping them into a common coordinate frame. To link neurons' identities over time, we then build a N-by-N matrix (where N is the number of all segmented neuron ROIs at all timepoints in a given recording) with the following structure: (1) Enter zero if the ROIs were in an image pair that was not registered (we do not attempt to solve all registration problems, as this is unnecessary); (2) Enter zero if the ROIs were from a registered image pair, but the registration-warped ROI did not overlap with the fixed ROI; and (3) Otherwise, enter a heuristic value indicating the confidence that the ROIs are the same neurons based on several ROI features. These features include similarity of ROI positions and sizes, similarity of red channel brightness, registration quality (computed as NCC of the red channel images), a penalty for overly nonlinear registration transformations, and a penalty if ROIs were displaced over large distances during alignment. Finally, custom hierarchical clustering is applied to the matrix to generate clusters consisting of the ROIs that reflect the same neuron recorded at different timepoints. Calcium traces are then constructed from all of these timepoints, normalizing the GCaMP signal to the tagRFP signal (Fig. 2B-D shows example GCaMP dataset and GFP control datasets). We term the ANTSUN pipeline with gradient descent registration ANTSUN 1.4[35,46] and the version with BrainAlignNet registration ANTSUN 2.0 (Fig. 2A).

We ran a series of control datasets through both versions of ANTSUN to benchmark their results. The first was from the previously-described animals with pan-neuronal NLS-mNeptune and *eat-4*::NLS-GFP. The resulting GFP traces from these recordings allow us to quantify the number of timepoints where the neuron identities are not accurately linked together into a single trace (Fig. 2B shows example dataset). Specifically, in this strain, this type of error can be easily detected since it can result in a low-intensity GFP neuron (*eat-4*-) suddenly having a high-intensity value when the trace mistakenly incorporates data from a high-intensity neuron (*eat-4*+), or vice versa. We computed this error rate, taking into account the overall similarity of GFP intensities (i.e. since we can only observe errors when GFP- and GFP+ neurons are combined into the same trace). For both versions of ANTSUN, the error rates were <0.5%, suggesting that >99.5% of timepoints reflect correctly linked neurons (Fig. 2E).

We next estimated the SNR of the data collected from ANTSUN 2.0, as compared to ANTSUN 1.4. Here, we processed data from three pan-neuronal GCaMP animals and compared them to three animals expressing pan-neuronal GFP, in place of GCaMP. The relative signal fluctuations in GCaMP traces versus GFP traces (the GFP traces should ideally be flat) can provide an indication of the entire recording/processing pipeline's SNR (Fig. 2C-D show examples). Results were similar for ANTSUN 1.4 and 2.0, which indicates that incorporating BrainAlignNet did not impair the SNR of the data (Fig. 2F). ANTSUN 2.0 also successfully extracted traces from a similar number of neurons (Fig. 2G). However, while ANTSUN 1.4 requires 250 CPU days per dataset for registration, ANTSUN 2.0 only requires 9 GPU hours, reflecting a >600-fold increase in computation speed (Fig. 2H). These results suggest that ANTSUN 2.0, which uses BrainAlignNet, provides a massive speed improvement in extracting neural data from GCaMP recordings without compromising the SNR or accuracy of the data.

## AutoCellLabeler: a neural network that automatically annotates >100 neuron classes in the *C. elegans* head from multi-spectral fluorescence

We next turned our attention to annotating the identities of the recorded neurons in these brain-wide calcium imaging data. *C. elegans* neurons have fairly stereotyped positions in the heads of adult animals, though fully accurate inference of neural identity from position alone has not been shown to be possible. Fluorescent reporter gene expression using well-defined genetic drivers can provide additional information to assist with neuron annotation. The NeuroPAL strain is especially useful in this regard. It expresses pan-neuronal NLS-tagRFP, but also has expression of NLS-mTagBFP2, NLS-CyOFP1, and NLS-mNeptune2.5 under a set of well-chosen genetic drivers (example image in Fig. 3A)[40]. With proper training, humans can manually label the identities of most neurons in this strain using neuron position and multi-spectral fluorescence. For most of the brain-wide recordings collected using our calcium imaging platform, we used a previously characterized strain with a pan-neuronal NLS-GCaMP7F transgene crossed into NeuroPAL[35] . While freely-moving recordings were conducted with only NLS-GCaMP and NLS-tagRFP data acquisition, animals were immobilized at the end of each recording in order to capture multi-spectral fluorescence. Humans could manually label many neurons' identities in these multi-spectral images, and the image registration approaches described above could map the ROIs in the immobilized data to ROIs in the freely-moving recordings to match neuron identity to GCaMP traces.

Manual annotation of NeuroPAL images is time-consuming. First, to perform accurate labeling, the individual needs substantial amounts of training. Even after being fully trained, labeling all the ROIs in one NeuroPAL animal can take 3-5 hours. In addition, different individuals have different degrees of knowledge or confidence in labeling certain cell classes. For these reasons, it was desirable to automate NeuroPAL labeling, using datasets that had previously been labeled by a panel of human labelers. In particular, the labels that they provided with a high degree of confidence in their accuracy would be most useful for training an automated labeling network. Previous studies have developed statistical approaches for semi-automated labeling to label neural identity from NeuroPAL images, but the maximum precision that we are aware of is 90% without manual correction[40].

We trained a 3-D U-Net[47] to label the *C. elegans* neuron classes in a given NeuroPAL 3-D image. As input, the network received four fluorescent 3-D images from the head of each worm: pan-neuronal NLS-tagRFP, plus the NLS-mTagBFP2, NLS-CyOFP1, and NLS-mNeptune2.5 images that label stereotyped subsets of neurons (Fig. 3A). During training, the network also received the human-annotated labels of which pixels belong to which neurons. Humans provided ROI-level labels and the boundaries of each ROI were determined using a previously-described neuron segmentation network[35] trained to label all neurons in a given image (agnostic to their identity). Finally, during training the network also received an array indicating the relative weight to assign each pixel during training (Fig. 3B). This was incorporated into a pixel-weighted cross-entropy loss function (lower values indicate more accurate labeling of each pixel), summing across the pixels in a weighted manner. Pixel weighting was adjusted as follows: (1) background was given extremely low weight; (2) ROIs that humans were not able to label were given low weight; (3) all other ROIs received higher weight, proportional to the subjective confidence that the human had in assigning the label to the ROI and the rarity of the label. Regarding this latter point, neurons that were less frequently labeled by human annotation received higher weight so that the network could potentially learn how to classify these neurons from fewer labeled examples.

We trained the network over 300 epochs using a training set of 81 annotated images and a validation set of 10 images (Fig. 3C). Because the size of the training set was fairly small, we augmented the training data using both standard image augmentations (rotation, flipping, adding gaussian noise, etc.) and a custom augmentation where the images were warped in a manner to approximate worm head bending (see Methods). Because this ***Auto****matic* ***Cell Label****ing Network* (**AutoCellLabeler**) labels individual pixels, it was necessary to convert these pixel-wise classifications into ROI-level classifications. AutoCellLabeler outputs its confidence in its label for each pixel, and we noted that the network's confidence for a given ROI was highest near the center of the ROI (Fig. 3D). Therefore, to determine ROI-level labels, we took a weighted average of the pixel-wise labels within an ROI, weighing the center pixels more strongly. The overall confidence of these pixel scores was also used to compute a ROI-level confidence score, reflecting the network's confidence that it labeled the ROI correctly. Finally, after all ROIs were assigned a label, heuristics were applied to identify and delete problematic labels. Labels were deleted if (1) the network already labeled another ROI as that label with higher confidence; (2) the label was present too infrequently in the network's training data; (3) the network labeled that ROI as something other than a neuron (e.g. a gut granule or glial cell, which we supplied as valid labels during training); or (4) the network confidently predicted different parts of the ROI as different labels.

We evaluated the performance of the network on 11 separate datasets that were reserved for testing. We assessed the accuracy of AutoCellLabeler on the subset of ROIs with high-confidence human labels (subjective confidence scores of 4 or 5, on a scale from 1-5). On these neurons, average network confidence was 96.8% and its accuracy was 97.1%. We furthermore observed that the network was more confident in its correct labels (average confidence 97.3%) than its incorrect labels (average confidence 80.7%; Fig. 3E). More generally, AutoCellLabeler confidence was highly correlated with its accuracy (Fig. 3F). Indeed, excluding the neurons where the network assigns low (<75%) confidence increased its accuracy to 98.1% (Extended Data Fig. 2A displays the full accuracy-recall tradeoff curve). Under this confidence threshold cutoff, AutoCellLabeler still assigned a label to 90.6% of all ROIs with high-confidence human labels, so we elected to delete the low-confidence (<75%) labels from the set of valid network output labels (see Extended Data Fig. 2A for rationale for the 75% cutoff value).

We also examined model performance on data where humans had either low confidence or did not assign a neuron label. In these cases, it was harder to estimate the ground truth. Overall, model confidence was much lower for neurons that humans labeled with low confidence (87.3%) or did not assign a label (81.3%). The concurrence of AutoCellLabeler relative to low-confidence human labels was also lower (84.1%; we note that this is not truly a measure of accuracy since these 'ground-truth' labels had low confidence). Indeed, overall the network's concurrence versus human labels scaled with the confidence of the human label (Fig. 3G).

We carefully examined the subset of ROIs where the network had high confidence (>75%), but humans had either low-confidence or entered no label at all. This was quite a large set of ROIs: AutoCellLabeler identified significantly more high confidence neurons (119/animal) than the original human labelers (83/animal), and this could conceivably reflect a highly accurate pool of labels exceeding human performance. To determine whether this was the case, we obtained new human labels (by different human labelers) for a random subset of these neurons. Whereas some human labels remained low-confidence, others were now labeled with high confidence (20.9% of this group of ROIs). The new human labelers also labeled neurons that were originally labeled with high confidence so that we could compare the network's performance on relabeled data where the original data was unlabeled, low confidence, or high confidence. AutoCellLabeler's performance on all three groups was similar (88%, 86.1%, and 92.1%, respectively), which was comparable to the accuracy of humans relabeling data relative to the original high-confidence labels (92.3%). The slightly lower accuracy on these re-labeled data is likely due to the human labeling of the original training, validation, and testing data being highly vetted and thoroughly double-checked, whereas the re-labeling that we performed just for this analysis was done in a single pass. Overall, these analyses indicate that the high-confidence network labels (119/animal) have similar accuracy regardless of whether the original data had been labeled by humans as un-labelable, low confidence, or high confidence. This indicates that AutoCellLabeler can confidently label more neurons per dataset than individual human labelers.

We also split out model performance by cell type. This largely revealed similar trends. Model labeling accuracy and confidence were variable among the neuron types, with highest accuracy and confidence for the cell types where there were higher confidence human labels and a higher frequency of human labels (Fig. 3K). For the labels where there were high confidence network and human labels, we generated a confusion matrix to see if AutoCellLabeler's mistakes had recurring trends (Extended Data Fig. 2B). While mistakes of this type were very rare, we observed that the ones that occurred could mostly be categorized as either mislabeling a gut granule as the neuron RMG, or mislabeling the dorsal/ventral categorization of the neurons IL1 and IL2 (e.g.: mislabeling IL2D as IL2). Together, these categories accounted for 50% of all AutoCellLabeler's mistakes. We also observed that across cell types, AutoCellLabeler's confidence was highly correlated with human confidence (Extended Data Fig. 2C), suggesting that the main limitations of model accuracy are due to human labeling accuracy and confidence.

To provide better insights into which network features were critical for its performance, we trained additional networks lacking some of AutoCellLabeler's key features. To evaluate these networks, we considered both the number of high confidence labels assigned by AutoCellLabeler and the accuracy of those labels measured against high-confidence human labels. Surprisingly, a network that was trained with only standard image augmentations (i.e. lacking the custom augmentation to bend the images in a manner that approximates a worm head bend) had similar performance (Fig. 3I). However, a network that was trained without a pixel-weighting scheme (i.e. where all pixels were weighted equally) provided far fewer high-confidence labels. This suggests that devising strategies for pixel weighting is critical for model performance, though our custom augmentation was not important. Interestingly, all trained networks had similar accuracy (Fig. 3J) on their high-confidence labels, suggesting that the network architecture in all cases is able to accurately assess its confidence.

## Automated annotation of *C. elegans* neurons from fewer fluorescent labels and in different strains

We examined whether the full group of fluorophores were critical for AutoCellLabeler performance. This is a relevant question because (i) it is laborious to make, inject, and annotate a large number of plasmids driving fluorophore expression, and (ii) the large number of plasmids in the NeuroPAL strain has been noted to adversely impact the animals' growth and behavior[35,40,48]. To test whether fewer fluorescent labels could still facilitate automatic labeling, we trained four additional networks: one that only received the pan-neuronal tagRFP image as input, and three that received pan-neuronal tagRFP plus a single other fluorescent channel (CyOFP, tag-mBFP2, or mNeptune). As we still had the ground-truth labels based on humans viewing the full set of fluorophores, the supervised labels were identical to those supplied to the full network.

We evaluated the performance of these models by quantifying the number of high-confidence labels that each network provided in each testing dataset (Fig. 4A) and the accuracy of these labels measured against high-confidence human labels (Fig. 4B). We found that all four networks had attenuated performance relative to the full AutoCellLabeler network, which was almost entirely explainable by these networks having lower confidence in their labels, since network accuracy was always consistent with its confidence (Extended Data Fig. 3A). This means that labels from any version of the network can be treated equivalently, where the confidence of a given label can be taken as an indication of its accuracy. Additionally, of the four attenuated networks, the tagRFP+CyOFP network performance (107 neurons per animal labeled at 97.4% accuracy) was quite close to the full network in its performance. Given that there are >20 mTagBFP2 and mNeptune plasmids in the full NeuroPAL strain, these results raise the possibility that a smaller set of carefully chosen plasmids could permit training of a network with equal performance to the full network that we trained here.

We did not expect the tagRFP-only network to perform well, since the task of labeling tagRFP-only images is nearly impossible for humans. Surprisingly, this network still exhibited relatively high performance, with an average of 94 high-confidence neurons per animal and 94.8% accuracy on those neurons. On most neuron classes, it behaved nearly as well as the full network, though there are 10-20 neuron classes that it is much worse at labeling, such as ASG, IL1, and RMG (Fig. 4C). Since this network only requires the red channel fluorescence, it could in theory be used directly on freely-moving data, which has only GCaMP and tagRFP channel data. Potentially, network performance could be increased by evaluating it on many different timepoints from the freely-moving data to allow it to see the worm in many different postures. Since the tagRFP-only network was trained only on high-SNR images collected from immobilized animals, we first checked that the network was able to generalize outside its training distribution to single images with lower SNR (example images in Extended Data Fig. 3B). It was able to label 79 high-confidence neurons per animal at 95.2% accuracy on the lower SNR images (Fig. 4A-B, right). We then investigated whether allowing the network to access different postures of the same animal improved its accuracy. Specifically, we evaluated the tagRFP-only network on 100 randomly-selected timepoints in the freely-moving data of each animal (example images in Extended Data Fig. 3B). We then related these 100 network labels to the human labels, which could be easily determined, since ANTSUN registers free-moving images back to the immobilized NeuroPAL images that had been labeled by humans. We averaged the 100 network labels to obtain the most likely network label for each neuron, as well as the average confidence for that label. To properly compare network versions, we determined how many neurons could be labeled at any given target labeling accuracy – for example, how many neurons the network can label and still achieve 95% accuracy (Fig. 4D; changing the threshold network confidence value to include a given label allowed us to determine these full curves). This analysis revealed that averaging network labels across the 100 timepoints improved network performance, though only modestly. These results suggest that single color labels can be used to train networks to a high level of performance, but additional fluorescence channels further improve performance.

The strong performance of the tagRFP-only network on out-of-domain lower SNR images suggest an impressive ability of the AutoCellLabeler network to generalize across different modalities of data. This raised the possibility that it may be possible to use this network architecture to build a foundation model of *C. elegans* neuron annotation that works across strains and imaging conditions. As a first step to explore this, we investigated to what extent the tagRFP-only network could generalize to other strains of *C. elegans* besides the NeuroPAL strain. We used our previously-described SWF415 strain, which contains pan-neuronal NLS-GCaMP7F, pan-neuronal NLS-mNeptune2.5, and sparse tagRFP expression[35]. Notably, the pan-neuronal promoter utilized in this strain for NLS-mNeptune expression (P*rimb-1*) is distinct from the pan-neuronal promoter that drives NLS-tagRFP expression in NeuroPAL (a synthetic promoter). Since humans do not know how to label neurons in SWF415, we did a more limited analysis by analyzing network labels for a subset of neurons that have highly reliable activity dynamics with respect to behavior (AVA, AVE, RIM, and AIB encode reverse locomotion; RIB, AVB, RID, and RME encode forward locomotion; SMDD encodes dorsal head curvature; and SMDV and RIV encode ventral head curvature)[35,49–56]. Specifically, we asked whether neurons labeled with high confidence by the network had the behavior encoding properties typical of the neuron, assessed via analysis of the GCaMP traces from that neuron. Our previously-described CePNEM model[35] was used to determine whether each labeled neuron encoded forward/reverse locomotion or dorsal/ventral head curvature. The network provided high-confidence labels for an average of 7.4/21 of these neurons per animal, and the encoding properties of these neurons matched expectations 68% of the time (randomly labeled neurons had a match of 19%). However, it was possible for the network to (i) incorrectly label a neuron as another neuron that happened to have the same encoding; or (ii) correctly label a neuron that CePNEM lacked statistical power to declare an encoding for. We accounted for these effects via simulations (see Methods), which estimated that the actual labeling accuracy of the network on SWF415 was 69% (Fig. 4E). This is substantially lower than this network's accuracy on similar images from the NeuroPAL strain (i.e. the strain used to train the network), where an average of 12.5 of these neurons per animal were labeled with 97.1% accuracy. Nevertheless, this analysis indicates that AutoCellLabeler has a reasonable ability to generalize to strains with different genetic drivers and fluorophores, suggesting that in the future it may be worthwhile to pursue building a foundation model that labels *C. elegans* neurons across many strains.

### A neural network (CellDiscoveryNet) that facilitates unsupervised discovery of >100 cell types by aligning data across animals

Annotation of cell types via supervised learning is fundamentally limited by prior knowledge and humans' ability to label multi-spectral imaging data. In principle, unsupervised approaches that can automatically identify stereotyped cell types would be preferable. Thus, we next sought to train a neural network to perform unsupervised discovery of the cell types of *C. elegans* nervous system (Fig. 5A). If successful, these approaches could be useful for labeling of mutant genotypes, new permutations of NeuroPAL, or even related species. In addition, such an approach would be useful in more complex animals that do not yet have complete catalogs of cell types.

To facilitate unsupervised cell type discovery, we trained a network to register different animals' multi-spectral NeuroPAL imaging data to one another. Successful alignment of cells across all recorded animals would amount to unsupervised cell type annotation, since the cells that align across animals would be the same cell type identified in different animals. The architecture of this network was similar to BrainAlignNet, but the training data here consisted of pairs of 4-color NeuroPAL images from two different animals and the network was tasked with aligning all four fluorescent channels (Fig. 5B). No cell type positions (i.e. centroids) or neuronal identities were provided to the network during training. Regularization and augmentation were similar to that of BrainAlignNet (see Methods). Training and validation data were comprised of 91 animals' datasets, which gave rise to 3285 unique pairs for alignment; 11 animals were withheld for testing (the same test set as for AutoCellLabeler). The validation loss plateaued after 600 epochs (Fig. 5C) and we characterized the network that had the minimum validation loss (at epoch 596). In the analyses below, we characterize performance on training data and withheld testing data, describing any differences. We note that, in contrast to the networks described above, high performance on training data is still useful in this case, since the only criterion for success in unsupervised learning is successful alignment (i.e. even if all data need to be used for training to do so). Strong performance on testing data is still more desirable though, since it is less efficient to train different networks over and over as new data are incorporated into the full dataset.

We first characterized the ability of this Unsupervised **Cell Discovery Net**work (**CellDiscoveryNet**) to align images across different animals. Image alignment was reasonably high for all four fluorescent NeuroPAL channels with a median NCC of 0.80 overall (Fig. 5D). Alignment accuracy was nearly equivalent in training and testing data (Fig. 5D). We also examined how well the centroid positions of defined cell types were aligned, utilizing our prior knowledge of neurons' locations – i.e. the human labels (Fig. 5E). We computed this metric only on cell types that were identified with high confidence in both of the images of a given registration problem. The median centroid distance was 7.2 pixels, with similar performance on training and testing data. This was initially rather disappointing, as it suggested that the majority of neurons were not being placed at their correct locations. However, we observed two important properties of the centroid alignments. First, the distribution of centroid distances was bimodal – the 20th percentile centroid distance was only 1.4 pixels, which corresponds to a correct neuron alignment. Second, the median centroid distance decreased to 3.3 for registration problems with high (> 90th percentile = 0.85) NCC scores on the images. Together, these observations suggest that CellDiscoveryNet correctly aligns neurons some of the time.

We next sought to differentiate the neuron alignments where CellDiscoveryNet was correct from those where it was incorrect. Effectively, we wanted to treat CellDiscoveryNet as a hypothesis generator for which neurons might be the same, and then algorithmically separate good hypotheses from bad ones, stitching together the accurate hypotheses into a full set of neuron alignments. To accomplish this, we adapted our ANTSUN pipeline (described in Fig. 2) to use CellDiscoveryNet instead of BrainAlignNet. This modified ANTSUN 2U (**U**nsupervised) takes as input multi-spectral data from many animals instead of monochrome images from different time points of the same animal. This approach then allows us to effectively cluster neurons that might be the same neuron found in different animals. Thus, we ran CellDiscoveryNet on pairs of images and used the resulting DDFs to align the corresponding segmented neuron ROIs. We then constructed a N-by-N matrix where N is all segmented neurons detected across all of the NeuroPAL images (i.e. all neurons in all animals). Entries in the matrix are zero if the two neurons were in an image pair that was never registered or if the two neurons did not overlap at all in the registered image pair. Otherwise, a heuristic value indicating the likelihood that the neurons are the same was entered into the matrix. This heuristic included the same information as in ANTSUN 2.0 (described above), such as registration quality and ROI position similarity. The only difference was that the heuristic for tagRFP brightness similarity was replaced with a heuristic for 4-channel color similarity (see Methods). Custom hierarchical clustering of the rows of this matrix then identified groups of ROIs hypothesized to be the same cell type identified in different animals.

To determine the performance of this unsupervised cell type discovery approach, we quantified both the number of cell types that were discovered (i.e. number of clusters) and the accuracy of cell type labeling within each cluster. Here, accuracy was computed by first determining the most frequent neuron label for each cell type, based on the human labels. We then determined the number of correct versus incorrect detections of this cell type for all cells that fell within the cluster, where a correct detection was defined to be when the human label for that cell matched the most frequent label for that cell's cluster. The number of cell types identified and the labeling accuracy are directly related: more permissive clustering identifies more cell types, but at the cost of lower accuracy. A full curve revealing this tradeoff is shown in Fig. 5F (the parameter $w_7$ controls the restrictiveness of the clustering; see Methods). Based on this curve, we selected the clustering parameter $w_7 = 10^{-9}$ that identified 125 cell types with 93% labeling accuracy. Not every cell type is detected in every animal. On the testing data, the CellDiscoveryNet-powered ANTSUN 2U roughly matched human-level performance in terms of accuracy and number of neurons labeled per animal (Fig. 5G-H). However, it fell slightly short of AutoCellLabeler (Fig.

5G-H). Overall, this analysis reveals that CellDiscoveryNet facilitates unsupervised cell type discovery with a high level of performance, matching trained human labelers.

We examined whether the accuracy of cell identification was different across cell types or across animals. Fig. 5I shows the accuracy of labeling for each cell type (see Extended Data Figure 4A for per-animal accuracy). Indeed, results were mixed: some cell types had highly accurate detections across animals (eg: OLQD and RME), whereas a smaller subset of cell types were detected with lower accuracy (eg: AIZ and ASG), and yet other cell types were harder to assess accuracy due to a smaller number of human labels (eg: AIM and I4). In addition, there were five clusters which did not contain a sufficient number of human-labeled ROIs to be given a cell type label (<3 cells in these clusters had matching human labels; these are labeled “NEW 1” through “NEW 5”). To examine which neurons these might correspond to, we examined the high-confidence AutoCellLabeler labels for ROIs in these clusters. This produced enough labels to categorize four of these five clusters as SAAD, SMBD, VB02, and VB02. The repeated VB02 label is likely an indication of under-clustering (ie: the two VB02 clusters should have been merged into the same cluster). The identity of the fifth cluster was unclear, as the ROIs in that cluster were not well labeled by either humans or AutoCellLabeler.

Finally, we examined whether CellDiscoveryNet was able to label cells not detected via AutoCellLabeler. Specifically, we determined the fraction of the cells detected by CellDiscoveryNet that were labeled by AutoCellLabeler, which was 86%. The new unsupervised detections (the remaining 14%) included: new labels for cells that were otherwise well-labeled by AutoCellLabeler (e.g.: M3); the detection and labeling of several cell types that were uncommonly labeled by AutoCellLabeler (e.g.: RMEV); and the previously-mentioned cell type that could not be identified. This suggests that the unsupervised approach that we describe here is able to provide cell annotations that were not possible via human labeling or AutoCellLabeler.

## DISCUSSION

Aligning and annotating the cell types that make up complex tissues remains a key challenge in computational image analysis. We trained a series of deep neural networks that allow for automated non-rigid registration and neuron identification in the context of brain-wide calcium imaging in freely-moving *C. elegans*. This provides an appealing test case for the development of such tools. *C. elegans* movement creates major challenges with tissue deformation and the animal has >100 defined cell types in its nervous system. We describe BrainAlignNet, which can perform non-rigid registration of the neurons of the *C. elegans* head, allowing for 99.6% accuracy in aligning individual neurons. We also describe AutoCellLabeler, which can automatically label >100 neuronal cell types with 98% accuracy, exceeding the performance of individual human labelers by aggregating their knowledge. Finally, CellDiscoveryNet aligns data across animals to perform unsupervised discovery of stereotyped cell types, identifying >100 cell types of the *C. elegans* nervous system from unlabeled data. These tools should be useful for a wide range of applications in *C. elegans* and should be straightforward to generalize to analyses of other complex tissues.

Our newly-described network for freely-moving worm registration on average aligns neurons with single pixel-level accuracy. Incorporating the network into a full image processing pipeline indicates that it allows us to link neurons across time with 99.6% accuracy. Training a network to achieve this high performance highlighted a series of general challenges. For example, our attempt to train the network in a fully unsupervised manner (i.e. to simply align two images with no further information) failed. While the resulting networks aligned RFP images of testing data nearly perfectly, it turned out that the image transformations underlying this registration reflected a scrambling of pixels and that the network was not warping the images in the manner that the animal actually bends. We note that it was only possible to detect this failure mode through unique control datasets that we had available to us, namely a strain that also had GFP in a sparse subset of neurons and prior knowledge of ROI locations in the images. A semi-supervised training procedure that utilized information about ROI locations ultimately prevented this failure mode. While this approach is quite feasible for our use case, other types of datasets may not have additional features such as ROI centroids to serve as supervised labels. It is possible that image augmentation[33] might be able to assist in such cases.

Another limitation was that even with the semi-supervised approach, we were only able to train networks to register images from reasonably well initialized conditions. Specifically, we provided Euler-registered image pairs that were selected to have moderately similar head curvature (though we note that these examples still had fairly dramatic non-rigid deformations; see Figure 1). Solving this problem was sufficient to fully align neurons from freely-moving *C. elegans* brain-wide calcium imaging, since clustering could effectively be used to link identity across all timepoints even if our image registration only aligned a subset of the image pairs. Our attempts to train a network to register all timepoints to one another was unsuccessful, though a variety of approaches could conceivably improve upon this moving forward.

The AutoCellLabeler network that we describe here now automates a task that previously required several hours of manual labeling per dataset. It achieves 98% accuracy in cell identification and labels more neurons per dataset than individual human labelers. This performance required a pixel weighting scheme where the network was trained to be especially sensitive to high-confidence labels of neurons that were not ubiquitously labeled by all human labelers. In other words, the network could aggregate knowledge across human labelers and example animals to achieve high performance. While the high performance of AutoCellLabeler is extremely useful from a practical point of view, we note that AutoCellLabeler still cannot label all ROIs in a given image, which would be the highest level of desirable performance. Our analyses suggest that it is currently bounded by human labeling of training data, which in turn is bounded by our NeuroPAL image quality and the ambiguity of labeling certain neurons in the NeuroPAL strain.

While improvements in human labeling could improve performance of the network, this analysis also highlighted that it would be highly desirable to perform fully unsupervised cell labeling, where the cell types could be inferred and labeled in multispectral images even without any human labeling. To accomplish this, we developed CellDiscoveryNet, which aligns NeuroPAL images across animals. Together with a custom clustering approach, this enabled us to identify 125 neuron classes, labeling them with 93% accuracy in a completely unsupervised manner. This approach could be very useful within the *C. elegans* system, since it is extremely time consuming to perform human labeling and it is conceivable that the NeuroPAL labels may change in different genotypes or if the NeuroPAL transgene is modified. Beyond *C. elegans*, these unsupervised approaches should be useful, since the vast majority of tissues in larger animals do not yet have a full catalog of cell types and, therefore, would greatly benefit from unsupervised discovery. In this spirit, other recent studies have started to develop approaches for unsupervised labeling of imaging data[11,57,58], though these efforts were not aimed at identifying the full set of cellular subtypes (>100) in individual images, which was the chief objective of CellDiscoveryNet.

We also note the importance of our post-processing clustering approach to improving the robustness of neural networks in solving these image registration problems. BrainAlignNet and especially CellDiscoveryNet will sometimes generate incorrect solutions to individual neuron mappings between a single pair of images. Relying solely on the network output for an individual pair of images would thus be prone to inaccuracy. However, by treating the networks as hypothesis generators across many images and using clustering to consider more likely hypotheses first, we can generate highly accurate linkages across all images. We speculate that this strategy might generalize across disciplines to many problems where it is possible to use deep neural networks to generate large numbers of hypotheses whose likelihoods can be heuristically evaluated.

We trained alternative versions of AutoCellLabeler with subsets of the spectral information in NeuroPAL, which provides some insights into the possibility of performing high-accuracy neural identification in strains with less fluorophores than NeuroPAL. On the one hand, all networks that were trained with fewer than the full set of 4 fluorescent channels exhibited poorer performance. However, it is notable that the network trained with only pan-neuronal RFP still achieved 95% accuracy in labeling 94 neurons per image. It is important to note that this is the performance of a network that was only trained to evaluate a single static image. It is conceivable that there could be an improvement in performance if the network were trained on pan-neuronal RFP images from all freely-moving timepoints, since this might allow the network to infer identity based on the full range of movement and deformations that a given neuron exhibits, which is quite stereotyped[36–40]. The fact that AutoCellLabeler exhibited surprisingly good out-of-domain performance on images with different SNRs and on different strains suggests that it may also be possible to improve performance across strains by building a foundation model similar to AutoCellLabeler that has been specifically engineered to solve the general task of labeling the cell types of the *C. elegans* brain in a wide range of images and strains (data are starting to be aggregated into data repositories[42]). Future efforts should be able to build upon the tools described here to lead to these types of improvements.

It should also be possible to combine the tools that we describe here to great effect. For example, the unsupervised cell labels from CellDiscoveryNet could be used to train AutoCellLabeler in order to obtain more unsupervised labels at higher accuracy. Moreover, this process could potentially be multiplexed to achieve better cell annotation from TagRFP only. For example, multiple multi-spectral transgenic lines like NeuroPAL could be subject to CellDiscoveryNet labeling and fed into parallel AutoCellLabeler variants that only use the red channel for prediction. These networks could then be combined to potentially achieve high-performance cell identification from TagRFP only, utilizing knowledge gained from multiple multi-spectral lines.

These approaches for registering and annotating cells in dense tissues should be straightforward to generalize to other species. For example, variants of BrainAlignNet could be trained to facilitate alignment of tissue sections or to register imaging data onto a common anatomical reference atlas. Our results suggest that training these networks on subsets of data with labeled feature points, such as cell centroids (i.e. the semi-supervised approach we use here), will facilitate more accurate solutions that, after training, can still be applied to datasets without any labeled feature points. In addition, variants of AutoCellLabeler could be trained on any multi-color cellular imaging data with manual labels. A pixel-wise labeling approach, together with appropriate pixel weighting during training, should be generally useful to build models for automatic cell labeling in a range of different tissues and animals. Finally, models similar to CellDiscoveryNet could be broadly useful to identify previously uncharacterized cell types in many tissues. It is conceivable that hybrid or iterative versions of AutoCellLabeler and CellDiscoveryNet could lead to even higher performance cell type discovery and labeling.

## ACKNOWLEDGMENTS

We thank members of the Flavell lab for critical reading of the manuscript. T.S.K. acknowledges funding from a MathWorks Science Fellowship. S.W.F. acknowledges funding from NIH (NS131457, GM135413, DC020484); NSF (Award #1845663); the McKnight Foundation; Alfred P. Sloan Foundation; The Picower Institute for Learning and Memory; and The JPB Foundation.

## AUTHOR CONTRIBUTIONS

Conceptualization, A.A.A., J.K., S.W.F. Methodology, A.A.A., J.K., A.KY.L. Software, A.A.A., J.K., A.KY.L.  Formal analysis, A.A.A., A.KY.L. Investigation, A.A.A., J.K., A.KY.L.. T.S.K., S.B., E.B., F.K.W., D.K. Writing – Original Draft, A.A.A. and S.W.F. Writing – Review & Editing, A.A.A., J.K., A.KY.L., and S.W.F. Funding Acquisition, S.W.F.

## DECLARATION OF INTERESTS

The authors have no competing interests to declare

## MATERIALS AND METHODS

### *C. elegans* Strains and Genetics

All data were collected from one-day old adult hermaphrodite *C. elegans* animals raised at 22C on standard nematode growth medium (NGM) plates.

For the GCaMP-expressing animals without NeuroPAL, two transgenes were present: (1) *flvIs17*: *tag-168::NLS-GCaMP7F + NLS-tagRFPt* expressed under a small set of cell-specific promoters: *gcy-28.d, ceh-36, inx-1, mod-1, tph-1(short), gcy-5, gcy-7*; and (2) *flvIs18*: *tag-168::NLS-mNeptune2.5*. This resulting strain SWF415, has been previously characterized[35].

For the GCaMP-expressing animals with NeuroPAL, two transgenes were present in the strain: (1) *flvIs17*: described above; and (2) *otIs670*: low-brightness NeuroPAL. This resulting strain, named SWF702, has been previously characterized[35].

The animals with *eat-4::NLS-GFP* and *tag-168::NLS-GFP* were also previously described[35]. As is described in the strain list, *tag-168::NLS-mNeptune2.5* was also co-injected with each of these plasmids to generate the two strains: SWF360 (*eat-4::NLS-GFP; tag-168::NLS-mNeptune2.5*) and SWF467 (*tag-168::NLS-GFP; tag-168::NLS-mNeptune2.5*).

We provide here a list of these four strains:

**SWF415** *flvIs17[tag-168::NLS-GCaMP7F, gcy-28.d::NLS-tag-RFPt, ceh-36:NLS-tag-RFPt, inx-1::tag-RFPt, mod-1::tag-RFPt, tph-1(short)::NLS-tag-RFPt, gcy-5::NLS-tag-RFPt, gcy-7::NLS-tag-RFPt]; flvIs18[tag-168::NLS-mNeptune2.5]; lite-1(ce314); gur-3(ok2245)*

**SWF702** *flvIs17; otIs670 [low-brightness NeuroPAL]; lite-1(ce314); gur-3(ok2245)*

**SWF360** *flvEx450[eat-4::NLS-GFP, tag-168::NLS-mNeptune2.5]; lite-1(ce314); gur-3(ok2245)*

**SWF467** *flvEx451[tag-168::NLS-GFP, tag-168::NLS-mNeptune2.5]; lite-1(ce314); gur-3(ok2245)*

## Microscope and Recording Conditions

Data used to train and evaluate the models include previously-published datasets[35,46,59] and newly-collected data. These animals were recorded under similar recording conditions to those described in our previous study[35]. There were two types of datasets collected, relevant to this study: freely-moving GCaMP/TagRFP data, and immobilized NeuroPAL data.

Briefly, all neural data (free-moving and NeuroPAL) were acquired on a dual light-path microscope that was previously described[35]. The light path used to image GCaMP, mNeptune, and the fluorophores in NeuroPAL at single cell resolution is an Andor spinning disk confocal system with Nikon ECLIPSE Ti microscope. Light supplied from a 150 mW 488 nm laser, 50 mW 560 nm laser, 100 mW 405 nm laser, or 140 mW 637 nm laser passes through a 5000 rpm Yokogawa CSU-X1 spinning disk unit with a Borealis upgrade (with a dual-camera configuration). A 40x water immersion objective (CFI APO LWD 40X WI 1.15 NA LAMBDA S, Nikon) with an objective piezo (P-726 PIFOC, Physik Instrumente (PI)) was used to image the volume of the worm's head (a Newport NP0140SG objective piezo was used in a subset of the recordings). A custom quad dichroic mirror directed light emitted from the specimen to two separate sCMOS cameras (Zyla 4.2 PLUS sCMOS, Andor), which had in-line emission filters (525/50 for GCaMP/GFP, and 570 longpass for tagRFP/mNeptune in freely-moving recordings; NeuroPAL filters described below). Data was collected at a volume rate of 1.7 Hz (1.4 Hz for the datasets acquired with the Newport piezo).

For recordings, L4 worms were picked 18-22 hours before the imaging experiment to a new NGM agar plate seeded with OP50 to ensure that we recorded one day-old adult animals. Animals were recorded a thin, flat NGM agar pad (2.5cm x 1.8cm x 0.8mm). On the 4 corners of the agar pad, we placed a single layer of microbeads with a diameter of 80um to alleviate the pressure of the coverslip (#1.5) on the worm. Animals were transferred to the agar pad in a drop of M9, after which the coverslip was added.

For NeuroPAL data collection, animals were immobilized via cooling, after which multi-spectral information was captured. For cooling, the slide was mounted with a thermoelectric cooling element attached to it, set to cool the agar temperature to 1 °C. A closed-loop temperature controller (TEC200C, Thorlabs) with a micro-thermistor (SC30F103A, Amphenol) embedded in the agar kept the agar temperature at the 1 °C set point. Once the temperature reached the set point, we waited 5 minutes for the worm to be fully immobilized before imaging.

We obtained a series of images from each recorded animal while the animal was immobilized (this has been previously described[35]):

(1-3) Spectrally isolated images of mTagBFP2, CyOFP1, and mNeptune2.5. We excited CyOFP1 using the 488nm laser at 32% intensity under a 585/40 bandpass filter. mNeptune2.5 was recorded next using a 637nm laser at 48% intensity under a 655LP-TRF filter, in order to not contaminate this recording with TagRFP-T emission. Finally, mTagBFP2 was isolated using a 405nm laser at 27% intensity under a 447/60 bandpass filter.

(4) An image with TagRFP-T, CyOFP1, and mNeptune2.5 (all of the “red” markers) in one channel, and gCaMP7f in the other channel. As described in our previous study, this image was used for neuronal segmentation and registration to both the freely moving recording and individually isolated marker images. We excited TagRFP-T and mNeptune2.5 via 561nm laser at 15% intensity and CyOFP1 and gCaMP6f via 488nm laser at 17% intensity. TagRFP-T, mNeptune2.5, and CyOFP1 were imaged with a 570LP filter and gCaMP6f was isolated using a 525/50 bandpass filter.

All isolated images were recorded for 60 timepoints. We increased the signal to noise ratio for each of the images by first registering all timepoints within a recording to one another and then averaging the transformed images. For manual labeling of these datasets, we created a composite, 3-dimensional RGB image by setting the mTagBFP2 image to blue, CyOFP1 image to green, and mNeptune2.5 image to red as done by Yemini et al. (2021) and manually adjusting the intensity of each channel to optimally match their manual.

## Availability of Code

All code is freely and publicly available (use main/master branches unless otherwise specified):

- BrainAlignNet: https://github.com/flavell-lab/BrainAlignNet and https://github.com/flavell-lab/DeepReg (main branch)
- GPU-accelerated Euler registration: https://github.com/flavell-lab/euler_gpu
- ANTSUN 2.0: https://github.com/flavell-lab/ANTSUN (branch v2.1.0); see also https://github.com/flavell-lab/flv-c-setup and https://github.com/flavell-lab/FlavellPkg.jl/blob/master/src/ANTSUN.jl for auxiliary package installation.
- AutoCellLabeler: https://github.com/flavell-lab/pytorch-3dunet and https://github.com/flavell-lab/AutoCellLabeler
- CellDiscoveryNet: https://github.com/flavell-lab/DeepReg (multicolor branch)
- ANTSUN 2U: https://github.com/flavell-lab/ANTSUN-Unsupervised

## BrainAlignNet

### Network architecture

BrainAlignNet's architecture is derived from the DeepReg software package, which uses a variation of a 3-D U-Net architecture termed a LocalNet[44,45]. BrainAlignNet first has a concatenation layer that concatenates the moving and fixed images together along a new, channel dimension. The resulting $284 \times 120 \times 64 \times 2$ image is then passed as input to the LocalNet, which outputs a $284 \times 120 \times 64 \times 3$ dense displacement field (DDF). The DDF defines a coordinate transformation from fixed image coordinates to moving image coordinates, relative to the fixed image coordinate system. So, for instance, if $DDF[x, y, z] = (\Delta x, \Delta y, \Delta z)$, it means that the coordinates $(x, y, z)$ in the fixed image are mapped to the coordinates $(x + \Delta x, y + \Delta y, z + \Delta z)$ in the moving image. The network has a final warping layer that applies the DDF to transform the moving image into a predicted fixed image whose pixel at location $(x, y, z)$ contains the moving image pixel at location $(x, y, z) + DDF[x, y, z]$. It also has another final warping layer that transforms the fixed image centroids $(x, y, z)$ into predicted moving image centroids $(x, y, z) + DDF[x, y, z]$. The network's loss function causes it to seek to minimize the difference between its predictions and the corresponding input data.

The LocalNet is at its core a 3-D U-Net with an additional output layer that receives inputs from multiple output levels. In more detail, it has 3 input levels and 3 output levels, with $16 \cdot 2^i$ feature channels at the $i$th level for $i \in \{0,1,2\}$. It contains an encoder block mapping the input to level 0, followed by two more encoder blocks mapping input level $i$ to level $i + 1$ for $i \in \{0,1\}$. Each of these three encoder blocks contains a convolutional block, a residual convolutional block, and a $2 \times 2 \times 2$ max-pool layer. The convolutional block consists of a 3-D convolutional layer with kernel size 3 that doubles the number of feature channels, followed by a batch normalization layer, followed by a ReLU activation function. The residual convolutional block consists of two convolutional blocks in sequence, except that the input (to the residual convolutional block) is added to the output of the second convolutional block right before its ReLU activation function. The bottom block comes after the encoder block at level 2, mapping input level 2 to output level 2. It has the same architecture as a single convolutional block; notably, it does not contain the max-pool layer.

There are three decoder blocks receiving inputs from the three encoder blocks described above. The first two decoder blocks map output level $i + 1$ to output level $i$ for $i \in \{1,0\}$; the third one maps output level 0 to the preliminary output with the same $(x, y, z)$ dimensions as the input. Each decoding block consists of an upsampling block, a skip-connection layer, a convolutional block, and a residual convolutional block. The upsampling block contains a transposed 3D convolutional layer with kernel size 3 that halves the number of feature channels and an image resizing layer (run independently on the upsampling block's input) using bilinear interpolation to double each dimension of the image. The output of the resizing layer is then split into two equal pieces along the channel axis and summed, and then added to the output of the transposed convolutional layer. The skip-connection layer appends the output of the mirrored encoder block $i$ (for the third decoder block, this corresponds the first encoder block) right before that encoder block's max pool layer. The skip-connection layer appends this output to the channel dimension, doubling its size. The convolutional and residual convolutional blocks are identical to those in the encoding block, except that the convolutional block halves the number of input channels instead of doubling it.

Finally, there is the output layer. It takes as input the output of the bottom block, as well as the output of every decoder block. To each of these inputs, it applies a 3D convolutional layer that outputs exactly 3 channels, followed by an upsampling layer that uses bilinear interpolation to increase the dimensions to the size of the original input images. It then averages together all of these images to compute the final $284 \times 120 \times 64 \times 3$ DDF.

**Preprocessing**

To train and validate a registration network that aligns neurons across time series in freely-moving *C. elegans*, we took several steps to prepare the calcium imaging datasets with images and their corresponding centroids. The preprocessing procedure consisted of (i) selecting two different time points from a single video (fixed and moving time points) at which to obtain RFP images (all images given to the network are from the red channel, which contains the signal from NLS-TagRFP) and neuron centroids; (ii) cropping all RFP images to a consistent size; (iii) performing Euler registration (translation and rotation) to align neurons from the image at the moving time point (moving image) to the image at the fixed time point (fixed image); (iv) creating image centroids for the network, which consist of matched lists of centroid positions of all the neurons in both the fixed and moving images.

***(i) Selection of registration problems.***

We refer to the task of solving the transformation function that aligns neurons from the moving image to the fixed image as a registration problem. We selected our registration problems based on previously constructed[35] image registration graphs using ANTSUN 1.4. In these registration graphs, the time points of a single calcium imaging recording served as vertices. An edge between two time points indicates a registration problem that we will attempt to solve. Edges were preferentially created between time points with higher worm posture similarities.

In ANTSUN 1.4, we selected approximately 13,000 pairs of time points (fixed and moving) per video that had sufficiently high worm posture similarity. These registration problems were solved by gradient descent using our old image processing pipeline, and ANTSUN clustering yielded linked neuron ROIs across frames that were the basis of constructing calcium traces[35]. To train BrainAlignNet here, we randomly sampled about 100 problems across a total of 57 animals, ultimately compiling 5,176 registration problems for training (some registration problems were discarded during subsequent preprocessing steps). To prepare the validation datasets, we sampled 1,466 problems across 22 animals. Testing data was 447 problems from 5 animals.

***(ii) Cropping.***

The registration network requires all 3D image volumes in training, validation, and testing to be of the same size. Therefore, a crucial step in preprocessing was to crop or pad the images along the $x$, $y$, $z$ dimensions to a consistent size of (284, 120, 64). Before reshaping the images, we first subtracted the median pixel value from each image (both fixed and moving) and set the negative pixels to zero. Then, we either cropped or padded with zeros around the centers of mass of these images to make the $x$ dimension 284, the $y$ dimension 120, and the $z$ dimension 64.

***(iii) Euler registration.***

Through experimentation with various settings of the network, we have found that it is difficult for the network to learn large rotations and translations at the same time as smaller nonlinear deformations. Euler registration is far more computationally tractable than nonlinear deformation, so we solved Euler registration for the images before providing them to the network. In Euler registration, we rotate or translate the moving images by a certain amount, aiming to maximize their normalized cross-correlation (NCC) with the fixed image. The optimal parameters of translation and rotation that resulted in the highest NCC were determined using a brute-force, GPU-accelerated parameter grid search. To further accelerate the grid search, we projected the fixed and moving images onto the *x*-*y* plane using a maximum-intensity projection along the *z*-axis. We also downsampled the fixed and moving images by a factor of 4 after the *z* maximal projection. The best parameters identified for transforming the projected images were then applied to each z-slice to transform the entire 3D image. This approach was feasible because the vast majority of worm movement occurs along the *x*-*y* axes.

***(iv) Creating image centroids.***

We obtained the neuronal ROI images for both the fixed and moving RFP images, designating them as the fixed and moving ROI images respectively. The full sets of ROIs in each image were obtained using ANTSUN 1.4's image segmentation and watershedding functions. ROI images were them constructed as follows. Each pixel in an ROI image contains an index value: 0 for background, or a positive integer for a neuron. All pixels belonging to a specific neuron have the same index, and pixels belonging to any other neuron have a different index. Since the ROI images are created independently at each time point, their neuronal indices are not *a priori* consistent across time points. Therefore, we used previous runs of ANTSUN 1.4 to link the ROI identities across time points, and generated new ROI images with consistent indices across time points – for example, all pixels with value 6 in one time point correspond to the same neuron as pixels with value 6 in any other time point. We deleted any ROIs with indices that were not present in both the moving and fixed images.

We then cropped these ROI images to the same size and subjected them to Euler transformations using the same parameters as their corresponding fixed and moving RFP images. Next, we computed the centroids of each neuron index in the resulting moving and fixed ROI images. The centroid was defined to be the mean *x*, *y*, and *z* coordinates of all pixels of a given ROI. We stored these centroids as two lists of equal length (typically, around 110). Note that these lists are now the matched positions of neurons in the fixed and moving images.

Since the network expects image centroids to be of the same size, all neuronal centroids in the fixed and moving images were padded and aggregated into arrays of shape (200, 3), ensuring the same ordering of neurons. The extra entries that do not contain neurons are filled with (-1, -1, -1) to make the total number of neurons equal to 200. We designate the neuronal centroid positions in the fixed and moving ROI images as fixed and moving centroids, respectively.

**Loss functions**

Our main custom modifications to the DeepReg network focus on the design of the loss function. In particular, we implemented a new supervised centroid alignment loss component and new regularization loss sub-components. Overall, the loss function consists of three major components:

- **Image loss** $L_I$ captures the difference between the warped moving image and the ground-truth fixed image.
- **Centroid alignment loss** $L_C$ is a supervised portion of the loss function. Given pre-labeled centroids corresponding to ground-truth information about neuron positions in the fixed and moving images, this loss component captures the difference between the predicted moving centroids and the ground-truth moving centroids.
- **Regularization loss** $L_R$ captures the prior that the "simplest" DDF that achieves the desired transform outcome is the best. For example, it's implausible that a pair of neurons that start close together end up on opposite sides of the worm, so a DDF that generates such a transformation would have a high value of regularization loss.

The total loss is then computed as $Loss = w_I L_i + w_C L_C + w_R L_R$. We set $w_I = 1$, $w_C = 0.1$, and $w_R = 1$.

***(i) Image loss.***

The image loss is the negative of the local squared zero-normalized cross-correlation (LNCC) between the fixed and warped moving RFP images. We designate the fixed image as $X_{true}$ and the warped moving image as $X_{pred}$. Define $E(X)$ as a function that computes the discrete expectation of image $X$ within a sliding cube of side length $n$=16:

$$E(X)[x,y,z] = \frac{1}{n^3} \sum_{i=x}^{x+n-1} \sum_{j=y}^{y+n-1} \sum_{k=z}^{z+n-1} X[i,j,k]$$

We then can compute the discrete sliding variance as

$$V(X) = E(X^2) - E(X)^2$$

The image loss (i.e., negative LNCC) is then defined as

$$\boldsymbol{L_I} = -\mathrm{LNCC} = -\frac{\left[E\left(X_{true} \circ X_{pred}\right) - E\left(X_{true}\right) \circ \mathrm{E}\left(X_{pred}\right)\right]^2}{V\left(X_{true}\right) \circ V\left(X_{pred}\right) + \epsilon}$$

***(ii) Centroid alignment loss.***

The centroid alignment loss is calculated as the negative of the sum of the Euclidean distances between the moving centroids and the network's predicted moving centroids, averaged across the number of centroids available. We designate the ground-truth and network predicted centroids as $N \times 3$ matrices $y_{true}$ and $y_{pred}$ respectively, where $N$ is the number of centroids, and the $i$th row of each matrix represents the coordinates of neuron $i$'s centroid. Centroid alignment loss in the overall loss function is then expressed as follows:

$$\boldsymbol{L_C} = \frac{1}{N}\sum_{i=0}^{N-1}\sqrt{\sum_{d=0,1,2}\left(y_{true}[i,d] - y_{pred}[i,d]\right)^2}$$

***(iii) Regularization loss.***

Our regularization loss function consists of four terms that seek to penalize DDFs that do not correspond to possible physical motion of the worm. Of these terms, gradient norm is unchanged from its previous implementation in the DeepReg package, while the other three components are our additions:

- **Gradient norm loss** $L_{Grad}$ penalizes transformations for being nonuniform.
- **Difference norm loss** $L_{Diff}$ penalizes transformations for moving pixels too far.
- **Axis difference norm loss** $L_{AxisDiff}$ penalizes transformations for moving pixels too far along the *z*-dimension, which is less plausible than movement along the *x*- and *y*-dimensions in our recordings.
- **Nonrigid penalty loss** $L_{Nonrigid}$ penalizes transformations for being nonrigid (i.e., not translation and rotation). (Note that unlike the gradient norm loss, this loss function will not penalize DDFs that apply rigid-body rotations.)

We then set $\boldsymbol{L_R} = 0.02\ L_{Grad} + 0.005 L_{Diff} + 0.001\ L_{AxisDiff} + 0.02\ L_{Nonrigid}$

*Gradient Norm.* The gradient norm computes the average gradient of the DDF by summing up the central finite difference of the DDF as the approximation of derivatives along the *x*, *y*, and *z* axes. Specifically, we first approximate the partial derivatives for $\boldsymbol{m} \in \{\boldsymbol{0}, \boldsymbol{1}, \boldsymbol{2}\}$ as follows:

$$\frac{\partial D_m}{\partial x} \approx \frac{1}{2}\left(\mathrm{D}[2{:}\mathrm{X},\ 1{:}\mathrm{Y}-1,\ 1{:}\mathrm{Z}-1,\ m] - D[0{:}X-2,\ 1{:}Y-1,\ 1{:}Z-1,\ m]\right)$$

$$\frac{\partial D_m}{\partial y} \approx \frac{1}{2}\left(D[1{:}X-1,\ 2{:}Y,\ 1{:}Z-1,\ m] - D[1{:}X-1,\ 0{:}Y-2,\ 1{:}Z-1,\ m]\right)$$

$$\frac{\partial D_m}{\partial z} \approx \frac{1}{2}\left(D[1{:}X-1,\ 1{:}Y-1,\ 2{:}Z,\ m] - D[1{:}X-1,\ 1{:}Y-1,\ 0{:}Z-2,\ m]\right)$$

These results are then stacked to obtain $\frac{\partial D}{\partial x}, \frac{\partial D}{\partial y}$, and $\frac{\partial D}{\partial z}$. The gradient norm is calculated as the squared sum of these derivatives, averaged across all elements:

$$\boldsymbol{L_{Grad}} = \frac{1}{3(X-2)(Y-2)(Z-2)}\sum_{i=0}^{X-3}\sum_{j=0}^{Y-3}\sum_{k=0}^{Z-3}\sum_{m=0}^{2}\left[\left(\frac{\partial D}{\partial x}\right)^2 + \left(\frac{\partial D}{\partial y}\right)^2 + \left(\frac{\partial D}{\partial z}\right)^2\right]_{i,j,k,m}$$

*Difference Norm.* The difference norm computes the average squared displacement of a pixel under the DDF ***D***:

$$\boldsymbol{L_{Diff}} = \frac{1}{3XYZ}\sum_{i=0}^{X-1}\sum_{j=0}^{Y-1}\sum_{k=0}^{Z-1}\sum_{m=0}^{2}(D[i,j,k,m])^2$$

where $X, Y, Z$ are the sizes of the image along the *x*, *y*, and *z* axes respectively.

*Axis Difference Norm*. Axis difference norm of the DDF $D$ calculates the average squared displacement of a pixel along the $z$-axis:

$$D_z = D[:,:,:,2] \qquad \boldsymbol{L_{AxisDiff}} = \frac{1}{XYZ}\sum_{i=0}^{X-1}\sum_{i=0}^{Y-1}\sum_{k=0}^{Z-1}(D_z[i,j,k])^2$$

*Nonrigid penalty.* This term penalizes nonrigid transformations of the neurons by utilizing the gradient information of the DDF. Unlike the approach used in computing the gradient norm, where global rotations would have nonzero gradient, here we are interested in penalizing specifically nonrigid transforms. We accomplish this by constructing a reference DDF, denoted as $D_{ref}$, which warps the entire image to the origin: $D_{ref}[x, y, z, :] = [-x, -y, -z]$. Then the difference DDF $D_{diff} = D - D_{ref}$ has the property that the magnitude of its gradient is rotation-invariant. We can then compute $\frac{\partial D_{diff}}{\partial x}$, $\frac{\partial D_{diff}}{\partial y}$, and $\frac{\partial D_{diff}}{\partial z}$ as for the gradient norm and define the gradient magnitude:

$$M = \left(\frac{\partial D_{diff}}{\partial x}\right)^2 + \left(\frac{\partial D_{diff}}{\partial y}\right)^2 + \left(\frac{\partial D_{diff}}{\partial z}\right)^2$$

Under any rigid-body transform, $M = 1$. Thus, the nonrigid penalty is calculated as

$$\boldsymbol{L_{Nonrigid}} = \frac{1}{3(X-2)(Y-2)(Z-2)}\sum_{i=0}^{X-3}\sum_{j=0}^{Y-3}\sum_{k=0}^{Z-3}\sum_{m=0}^{2}\left|M + \frac{1}{M} - 2\right|_{i,j,k,m}$$

In this way, rigid-body transforms will have 0 loss while any nonrigid transform will have a positive loss.

**Data augmentation**

During training, input data was subject to augmentation. We used random affine transformations for augmentation. Each transformation was generated by perturbing the corner points of a cube by random amounts, and computing the affine transformation resulting in that perturbation. The same transformation was then applied to the moving image, fixed image, moving centroids, and fixed centroids.

**Optimizer**

BrainAlignNet was trained using the Adam optimizer with a learning rate of $10^{-4}$.

**Configuration file**

The full configuration file we used during network training is available at https://github.com/flavell-lab/BrainAlignNet/tree/main/configs

## Automatic Neuron Tracking System for Unconstrained Nematodes (ANTSUN) 2.0

We integrated BrainAlignNet into our previously-described ANTSUN pipeline[35,46] (also applied in[59]). Briefly, the pipeline: (i) performs some image pre-processing such as shear-correction and cropping; (ii) segments the images into neuron ROIs via a 3D U-Net; (iii) finds time points where the worm postures are similar; (iv) performs image registration to define a coordinate mapping between these time points; (v) applies that coordinate mapping to the ROIs; (vi) constructs an ROI similarity matrix storing how likely different ROIs are to correspond to the same neuron; (vii) clusters that matrix to extract neuron identity; (viii) maps the linked ROIs onto the GCaMP data to extract neural traces; and (ix) performs some postprocessing such as background-subtraction and bleach correction to extract neural traces.

The differences in ANTSUN 2.0 compared with our previously-published version of this pipeline, ANTSUN 1.4, are that in ANTSUN 2.0 we use BrainAlignNet to perform image registration rather than the gradient descent-based elastix, and we modified the heuristic function used to construct the ROI similarity matrix. We only replaced the freely-moving registration with BrainAlignNet; the immobilized registrations, channel alignment registration, and freely-moving to immobilized registration are still performed with elastix. These remaining elastix-based registrations are much less computationally expensive, taking only about 2% of the total computation time of the original ANTSUN 1.4 pipeline. They will also likely be replaced with BrainAlignNet in a future release of ANTSUN, after further diagnostics and controls are run.

The heuristic function used to compute the ROI similarity matrix was updated to add additional terms specific to BrainAlignNet, including regularization and an additional ROI displacement term that serves to implement our prior that ROIs which moved less far in the registration are more likely to be correctly registered. Letting $i$ and $j$ be two different ROIs in our recording at time points $t_i$ (moving) and $t_j$ (fixed), the full expression for the ROI similarity matrix is:

$$M_{ij} = R_{t_i t_j} \frac{1}{1 + w_1 d_i} q_{t_i t_j}^{w_2} r_{ij}^{w_3} e^{-\left(w_4 a_{ij} + w_5 c_{ij} + w_6 n_{t_i t_j}\right)}$$

Where:

$R_{t_i t_j}$ is 1 if there exists a registration mapping $t_i$ to $t_j$, and 0 otherwise.

$d_i$ is the displacement of the centroid of ROI $i$ under the DDF registration between $t_i$ and $t_j$.

$q_{t_i t_j}$ is the registration quality, computed as the NCC of warped moving image $t_i$ vs fixed image $t_j$.

$r_{ij}$ is the fractional overlap of warped moving ROI $i$ and fixed ROI $j$ (intersection / max size).

$a_{ij}$ is the absolute difference in marker channel activity (i.e. tagRFP brightness) between ROIs $i$ and $j$, normalized to mean activity at the corresponding timepoints $t_i$ and $t_j$.

$c_{ij}$ is the distance between the centroids of warped moving ROI $i$ and fixed ROI $j$.

$n_{t_i t_j}$ is the (unweighted) nonrigid penalty loss of the DDF registration from $t_i$ to $t_j$.

$w_i$ are weights controlling how important each variable is.

Additionally, the matrix is forced to be symmetrical by setting $M_{ji} = M_{ij}$ whenever $M_{ji} = 0$ and $M_{ij} \neq 0$. It is also sparse since $R_{t_i t_j}$ and $r_{ij}$ are usually 0. Finally, there are two additional hyperparameters in the clustering algorithm, $w_7$ and $w_8$. $w_7$ controls the minimum height the clustering algorithm will reach (effectively, $w_7$ is a cap on how low $M_{ij}$ values can get, or how low the heuristic value can fall before determining that the ROIs are not the same neuron) and $w_8$ controls the acceptable collision fraction (a collision is defined by a cluster containing multiple ROIs from the same timepoint, which should not happen since each neuron should correspond to only one ROI at each time point).

We determined the weights $w_i$ by performing a grid search through 2,912 different combinations of weights on three *eat-4*::NLS-GFP datasets. To evaluate the outcome of each combination, we computed the error rate (rate of incorrect neuron linkages) and number of detected neurons. The error rate was computed as previously described[35]: since the strain *eat-4*::NLS-GFP expresses GFP in some but not all neurons, we can quantify registration errors as instances where a GFP-positive neuron lacked GFP in a time point and vice versa, as these correspond to neuron mismatches. We then selected the combination of parameters that maximize the number of detected neurons while minimizing the error rate. One *eat-4*::NLS-GFP dataset (the one shown in Figure 2) was used as a withheld testing animal to determine this optimal set of parameters. The pan-neuronal GFP and pan-neuronal GCaMP animals were not included in this parameter search.

The values of the parameters we used were:

$$w_1 = 2$$

$$w_2 = 25$$

$$w_3 = 1$$

$$w_4 = 3$$

$$w_5 = 1$$

$$w_6 = 1$$

$$w_7 = 0.0001$$

$$w_8 = 0.05$$

## **AutoCellLabeler**

### *Network Architecture*

AutoCellLabeler uses a 3-D U-Net architecture[35,47], with input dimensions $4 \times 64 \times 120 \times 284$ (fluorophore channel, $z$, $y$, $x$) and output dimensions $185 \times 64 \times 120 \times 284$ (label channel, $z$, $y$, $x$). The 3D U-Net has 4 input levels and 4 output levels, with $64 \cdot 2^i$ feature channels at the $i$th level for $i \in \{0,1,2,3\}$.

There is an encoder block that maps an input image to the 0th input level, followed by three additional encoder blocks that map input level $i$ to input level $i + 1$ for $i \in \{0,1,2\}$. Each encoder block consists of two convolutional blocks followed by a $2 \times 2 \times 2$ max pool layer, with the exception of the first encoder layer which does not have the max pool layer. The first convolutional block in each encoder increases the number of channels by a factor of 2 and the second leaves it unchanged.

Each convolutional block consists of a GroupNorm layer with group size 16 (except for the first convolutional layer in the first encoder, which has group size 1), followed by a 3D convolutional layer with kernel size 3 and the appropriate number of input and output channels, followed by a ReLU activation layer.

After the encoder, the 3D U-Net then has three decoder blocks mapping output level $i + 1$ and input level $i$ to output level $i$ for $i \in \{0,1,2\}$. Output level 3 is defined to be the same as input level 3. Each decoder layer consists of an $2 \times 2 \times 2$ upsampling layer which upsamples output level $i$ via interpolation, followed by a concatenation layer that concatenates it to input level $i - 1$ along the channel axis, followed by two convolutional blocks. The first convolutional block decreases the number of channels by a factor of 2 and the second convolutional block leaves the number of channels unchanged. After the final decoder layer, a $1 \times 1$ convolutional layer is applied to increase the number of output channels to the desired 185.

*Training Inputs*

We trained the AutoCellLabeler network on a set of 81 human-annotated NeuroPAL images, with 10 images withheld for validation and another 11 withheld for testing. Each training dataset contained three components: image, label, and weight. The images were $4 \times 64 \times 120 \times 284$ , with the first dimension corresponding to channel: we spectrally isolated each of the four fluorescent proteins NLS-mNeptune 2.5, NLS-CyOFP1, NLS-mTagBFP2, and NLS-tagRFP using our previously described imaging setup[35], described in detail above. The training images were then created by registering all of the images to the NLS-tagRFP image as described above, cropping all of them to $64 \times 120 \times 284$ dimensions $(z, y, x)$, and then stacking them along the channel axis to be $4 \times 64 \times 120 \times 284$ (in the reverse order that they were for the BrainAlignNet).

To create the labels, we ran our segmentation U-Net on each such image to generate ROIs corresponding to neurons in these images. Humans then manually annotated the images and assigned a label and a confidence to these ROIs. These confidence values ranged from 1-5, with 5 being the maximum. For network training, only confidence-1 labels were excluded while all labels from confidence 2 through 5 were included. We then made a list $\ell$ of length 185: the background label, and all 184 labels that were ever assigned in any of the human-annotated images. This list contained all neurons expected to be in the *C. elegans* head with the exceptions of ADFR, AVFR, RMHL, RMHR, and SABD, as these neurons were not labeled in any dataset. The list also contained six other possible classes corresponding to neurons in the anterior portion of the ventral cord: VA01, VB01, VB02, VD01, DD01, and DB02, as well as the classes “glia” and “granule” to denote non-neuronal objects that fluoresce (and might be labeled with an ROI), and the class "RMH?" as the human labelers were never able to disambiguate whether their "RMH" labels corresponded to RMHL or RMHR.

Due to a data processing glitch, labels for 2 of the 81 training datasets were imported incorrectly; validation and testing datasets were unaffected. This resulted in those datasets effectively having random labels during training. We are currently re-training all versions of the AutoCellLabeler network and expect their performance to modestly increase once this is rectified.

For each image, the human labels were transformed into matrices $L$ with dimensions $185 \times 64 \times 120 \times 284$ via one-hot encoding, so that $L[n, z, y, x]$ denotes whether the pixel at position $(x, y, z)$ has label $\ell[n]$. Specifically, we set $L[n, z, y, x]$ for $n > 0$ to be 1 if the pixel at position $(x, y, z)$ corresponded to an ROI that the human labeled as $\ell[n]$, and 0 otherwise. For example, the fourth element of $\ell$ was I2L (i.e., $\ell[3] = "I2L"$), so $L[3, z, y, x]$ would be 1 in the ROI labeled as I2L and 0 everywhere else. The first label (i.e., $n = 0$) corresponded to the background, which was 1 if all other channels were 0, and 0 otherwise.

Finally, we create a weight matrix $W$ of dimensions $64 \times 120 \times 284$ (in the code, this matrix has dimensions $185 \times 64 \times 120 \times 284$, but the loss function is mathematically equivalent to the version presented here). The entries of $W$ are determined by the following set of rules for weighting each corresponding pixel in the human label matrix $L$:

- $W[z, y, x] = 1$ for all $x, y, z$ with the background label, i.e. $L[0, z, y, x] = 1$
- $W[z, y, x] = \frac{130}{N(l_r)} f(c_r)$ if there is an ROI at $(x, y, z)$ with label $l_r$ that has confidence $c_r$. Here $N(l_r)$ is the number of ROIs across all datasets (train, validation and testing) with the label $l_r$. This makes neurons with fewer labels more heavily weighted in training. Additionally, $f$ is a function that weighs labels based on human confidence score $c_r$, where $c_r \in \{2, 3, 4, 5\}$. Specifically, $f(2) = 50$, $f(3) = 600$, $f(4) = 900$, and $f(5) = 1000$. The number 130 was the maximum number of times that any neuronal label (e.g.: not "granule" or "glia") was detected across all of the training datasets.

For the "no weight" network described in Figure 4, all entries of this matrix were set to 1.

*Loss function*

The loss function is pixel-wise weighted cross-entropy loss. This is computed as:

$$\textbf{Loss} = -\frac{1}{d_x d_y d_z K} \sum_{n=0}^{K-1} \sum_{x=0}^{d_x-1} \sum_{y=0}^{d_y-1} \sum_{z=0}^{d_z-1} W[z, y, x] L[n, z, y, x] \log\left(\frac{e^{P[n,z,y,x]}}{\sum_{m=0}^{K-1} e^{P[m,z,y,x]}}\right)$$

Here $(d_z, d_y, d_x)$ are the image dimensions (64, 120, 284), $K$ is the number of total labels (i.e., the length of $\ell$), and $(n, x, y, z)$ are indices within label and image dimensions. $W$ and $L$ are as defined above, and $P$ is the prediction (output) of the network. In this way, the network has a lower loss if $P[n, z, y, x]$ is high when $L[n, z, y, x] = 1$ (ie: the network got the label right), as then the softmax $\log\left(\frac{e^{P[n,z,y,x]}}{\sum_{m=0}^{K-1} e^{P[m,z,y,x]}}\right)$ term will be close to 0 and therefore multiply $L[n,z,y,x]$ by a small (negative) number, resulting in an overall small (positive) loss. The $W[z,y,x]$ term makes it so the network cares more about pixels and labels with high weight – in particular, it cares more about foreground labels $n > 0$ and about higher-confidence and rarer labels.

*Evaluation metric*

The evaluation metric is weighted mean intersection-over-union (IoU) across channels. Let $A$ be the network's argmax label matrix. Specifically, $A[n,z,y,x] = 1$ when $P[n,z,y,x] = \max_m P[m,z,y,x]$ and $A[n,z,y,x] = 0$ otherwise. Then the evaluation metric is defined as:

$$\mathbf{MeanIoU} \approx \frac{1}{K}\sum_{n=0}^{K-1} \frac{\sum_{x=0}^{d_x-1}\sum_{y=0}^{d_y-1}\sum_{z=0}^{d_z-1} W[z,y,x]\cdot L[n.z,y,x]\cdot A[n,z,y,x]}{\sum_{x=0}^{d_x-1}\sum_{y=0}^{d_y-1}\sum_{z=0}^{d_z-1} W[z,y,x]\cdot \max\left(L[n,z,y,x], A[n,z,y,x]\right)}$$

In this manner, if the network is always correct, $A = L$, the numerator and denominator will be equal, and the evaluation score will be 1. Similarly, if the network is always wrong, the evaluation score will be 0. (In the code, this metric is slightly different from the version presented here due to additional complexity with the $W$ matrix having a nonuniform extra dimension, but they act very similarly.)

*Optimizer*

The network was optimized with the Adam optimizer with a learning rate of $10^{-4}$.

*Data augmentation*

The following data augmentations are performed on the training data. One augmentation is generated for each iteration, in the following order. The same augmentation is applied to the image, label, and weight matrices, except that contrast adjustment and noise are not used for the label and weight matrices. Missing pixels are set to the median of the image, or to 0 for the label and weight matrices. Interpolation is linear for the images and nearest-neighbors for label and weight. Full parameter settings such as strength or range of each augmentation are given in the parameter file (see below).

- **Rotation.** The rotations in the *xy* plane and *yz* plane are much larger than the rotation in the *xz* plane because the worm is oriented to lay roughly along the *x* axis, and the physics of the coverslip are such that it cannot rotate about the *y* axis.
- **Translation.** The image is translated.
- **Scaling.** The image is scaled.
- **Shearing.** The image is sheared.
- **B-Spline Deformation.** Evenly-spaced control points are chosen and a random piecewise-cubic B-Spline deformation is generated between them. Additionally, a second B-Spline deformation with the same control points is generated that focuses on deformations in the *xy* plane designed to resemble worm bending. The two transforms are added and then executed.

- **Rotation by multiples of 90 degrees.** The image is rotated.
- **Contrast adjustment.** Each channel is adjusted separately.
- **Gaussian blur.** Gaussian blur is added to the image, in a gradient along the $z$-axis. The gradient is intended to mimic the optical effect of the image becoming blurrier farther away from the objective.
- **Gaussian noise.** Added to the image, with each pixel being sampled independently.
- **Poisson noise.** Added to the image, with each pixel being sampled independently.

*“Less aug” network training*

We trained a version of the network with some of our custom augmentations disabled, to see how important they were to the overall performance, compared with the other more standard data augmentations. The specific augmentations that were disabled were:

- The second B-Spline deformation focusing on deformations in the $xy$ plane
- Contrast adjustment
- Gaussian blur

*Parameter file*

The full parameter files are available at:

https://github.com/flavell-lab/pytorch-3dunet/tree/master/AutoCellLabeler_parameters

They include augmentation hyperparameters and various other settings not listed here. There is a different parameter file for each version of the network, though in most cases the differences are simply the number of input channels. If a user installs the pytorch-3dunet package from that GitHub repository and replace the paths to the training and validation data with their locations on your computer, they can train it with the exact settings we used here. Training will require a GPU with at least 48GB of VRAM.

*Evaluation*

During evaluation, an additional softmax layer is applied to convert network output into probabilities. Let $I$ be the input image and let $P$ be the network’s output (after the softmax layer). Then at every pixel $(x, y, z)$, the network’s output array $P[n, z, y, x]$ represents the probability that this pixel has label $\ell[n]$.

*ROI image creation*

To convert the output into labels, we first ran our previously-described neuron segmentation network[35,46] on the tagRFP channel of the NeuroPAL image. Specifically, since this segmentation network was trained on lower-SNR freely-moving data, we ran it on a lower-SNR copy of the tagRFP channel. (This copy was one of the 60 images we averaged together to get the higher-SNR image fed to AutoCellLabeler.)

The segmentation network and subsequent watershed post-processing[35] were then used to generate a matrix $R$ with dimensions $284 \times 120 \times 64$ (same as the original tagRFP image). Each pixel in $R$ contains an index, either 0 for background or a positive integer indicating a specific neuron. The segmentation network and watershed algorithms were designed such that all pixels belonging to a specific neuron have the same index, and pixels belonging to any other neuron have different indices. We define an ROI $R_i = \{(x, y, z) \mid R[x, y, z] = i\}$.

*ROI label assignment*

We now wish to use AutoCellLabeler to assign a label to ROI $R_i$. To do this, we first generate a mask matrix $M_i$ with the same dimensions as $R$, defined by:

- $M_i[x, y, z] = 0$ if $R[x, y, z] \neq i$
- $M_i[x, y, z] = 0.01$ if $R[x, y, z] = i$ and there exists $(X, Y, Z)$ face-adjacent to $(x, y, z)$ such that $R[X, Y, Z] \neq i$.
- $M_i[x, y, z] = 1$ otherwise.

Here, the 0.01 entries are provided to the edges of the ROI so as to weight the central pixels of each ROI more heavily when determining the neuron's identity.

Finally, we define a prediction matrix $D$ that allows us to determine the label of each ROI and the corresponding confidence of each label. Letting $V$ be the number of distinct nonzero values in $R$ (ie: the number of ROIs) and $K = 185$ be the number of possible labels (as before), we define a $V \times K$ prediction matrix $D$ whose $(i, n)$th entry represents the probability that ROI $R_i$ has label $n$ as follows:

$$D[i, n] = \frac{\sum_{xyz} M[i, x, y, z] P[n, z, y, x]}{\sum_{xyz} M[i, x, y, z]}$$

Here the sums are taken over all pixels in the image.

Note that because of the additional softmax layer, we have $\sum_n D[i, n] = 1$ for all $i$. From this, we can then define the label index of ROI $R_i$ to be $n_i = \mathrm{argmax}_n D[i, n]$. From this, we can define its label to be $\ell[n_i]$, and the confidence of that label to be $D[i, n_i]$.

*ROI Label Postprocessing*

After all ROIs are assigned a label, they are sorted by confidence in descending order. The ROIs are iterated through in this order, with each ROI being assigned its most likely label and the set of all assigned labels being tracked. If an ROI $R_i$ has its most likely label $l_i$ already assigned to a different ROI $R_j$, the distance between the centroids of ROIs $R_i$ and $R_j$ is computed. If this distance is small enough, the collision is likely due to over-segmentation by the segmentation U-Net (i.e., ROIs $R_i$ and $R_j$ are actually the same neuron). In this case, they are assigned the same label. Otherwise, the collision is likely due to a mistake on the part of AutoCellLabeler, and the label for ROI $R_i$ is deleted. (i.e. the higher-confidence label for ROI $R_j$ is kept and the lower-confidence label $R_i$ is discarded.)

Additionally, ROIs are checked for under-segmentation. This rarely happens when the segmentation U-Net incorrectly merges two neurons into the same ROI. This is assessed by checking how many pixels in the ROI $R_i$ have predictions other than the full ROI label index $n_i$. Specifically, we count the number of pixels with $P[n, x, y, z] > 0.75$ within $R_i$ for some $n \neq n_i$. If there exists at least 10 pixels with label $n \neq n_i$, or 20% of the pixels in the ROI are labeled as

$n \neq n_i$ in this way, it is plausible that the ROI contains parts of another neuron. In this case, the label for that ROI is deleted.

Most neuron classes in the *C. elegans* brain are bilaterally symmetric and have two distinct cell bodies on the left and right part of the animal. These are genetically identical and therefore have exactly the same shape and color, which can often make it difficult to distinguish between them. For most applications, it is also usually unnecessary to distinguish between them since they typically have nearly-identical activity and function. In some cases, AutoCellLabeler can be confident in the neuron class but uncertain about the L/R subclass, assigning a probability of >10% to both L and R subclasses. In this case, we do not assign a specific subclass, instead assigning a label only for the main class with the sum of its confidence for either of the two subclasses. We note that this is only done for the L/R subclass – other neurons can also have D/V subclasses, but these are typically functionally distinct, so we require the network to disambiguate D/V for all neuron classes.

Finally, certain neuron classes were present few times in our manually-labeled data, making it more likely for the network to mislabel them due to lack of training data, and simultaneously making it difficult for us to assess its performance on these neuron classes due to the lack of testing data where they were labeled. We deleted any AutoCellLabeler labels corresponding to one of these classes, which were ADF, AFD, AVF, AVG, DB02, DD01, RIF, RIG, RMF, RMH, SAB, SABV, SIAD, SIBD, VA01, and VD01. Additionally, there are other fluorescent cell types in the worm's head. AutoCellLabeler was trained to label them as either "glia" or "granule", to avoid mislabeling them as neurons, and any AutoCellLabeler labels of "glia" or "granule" were deleted to ensure all of our analyses are based on actual neuron labels.

Altogether, these postprocessing heuristics resulted in deleting network labels for only 6.3% of ROIs with confidence 4 or greater human neuron labels (ie: not "granule" or "glia").

## CePNEM Simulation Analysis (Figure 4E)

To assess performance of our AutoCellLabeler network on the SWF415 strain, we could not compare its labels to human labels since humans do not know how to label neurons in this strain. Therefore, we used functional information about neuron activity patterns to assess accuracy of the network. We used our previously-described CePNEM model to do this[35]. Briefly, CePNEM fits a single neural activity trace to the animal's behavior to extract parameters about how that neuron represents information about the animal's behavior. CePNEM fits a posterior distribution for each parameter, and statistical tests run on that posterior are used to determine encoding of behavior. For example, if nearly all parameter sets in the CePNEM posterior for a given neuron have the property that they predict the neuron's activity is higher when the animal is reversing, then CePNEM would assign a reversal encoding to that neuron.

By doing this analysis in NeuroPAL animals where the identity of each neural trace is known, we have previously created an atlas of neural encoding of behavior[35]. This atlas revealed a set of neurons that have consistent encodings across animals: AVA, AVE, RIM, and AIB encode reverse locomotion; RIB, AVB, RID, and RME encode forward locomotion; SMDD encodes dorsal head curvature; and SMDV and RIV encode ventral head curvature. Based on this prior knowledge, we decided to quantify the fraction $f$ of labeled neurons with the expected activity-behavior coupling. For example, if AutoCellLabeler labeled 10 neurons as AVA and 7 of them encoded reverse locomotion when fit by CePNEM, this fraction would be 0.7.

However, this fraction is not necessarily an accurate estimate of AutoCellLabeler's accuracy. For example, it might have been possible for AutoCellLabeler to mislabel a neuron as AVA that happened to encode reverse locomotion in that animal, thus making the incorrect label appear accurate. On the other hand, CePNEM is limited by statistical power, and can sometimes fail to detect the appropriate encoding. This could make a correct label appear inaccurate.

To correct for these factors, we ran a simulation analysis to try to estimate the fraction $p$ of labels that were correct. To do this, we iterated through every one of AutoCellLabeler's labels that was one of the consistent-encoding neuron classes (i.e. one of the neurons listed above). In each simulation, we assign labels to neurons in the following manner: with probability $p_{sim}$ (i.e., the fraction of labels estimated by our simulation to be correct), the label was reassigned to a random neuron that was given that label by a human in a NeuroPAL animal (at confidence 3 or greater); with probability $1 - p_{sim}$, the label was reassigned to a random neuron in the same (SWF415) animal. In this way, the simulation controls for both of the possible inaccuracies outlined above. Then the fraction $f_{sim}$ of labeled neurons with the expected encoding was computed for each simulation. 1000 simulation trials were run for each value of $p_{sim}$, which ranged from 0 to 100 – the mean and standard deviation of these trials are shown in Figure 4E. We then computed the probability $p_{sim}$ for which $f_{sim}$ was in closest agreement to $f$, which was 69% (dashed vertical line). This is our estimate for the ground-truth correct label probability $p$.

## CellDiscoveryNet

### *Network Architecture*

The architecture of CellDiscoveryNet uses the same LocalNet backbone from DeepReg that BrainAlignNet uses, with the following modifications to the architecture and training procedure (these modifications are currently in the multicolor branch):

- The input images to CellDiscoveryNet are $284 \times 120 \times 64 \times 4$ instead of $284 \times 120 \times 64$.
- The image concatenation layer in CellDiscoveryNet concatenates the moving and fixed images along the existing channel dimension instead of adding a new channel dimension. Effectively, this means that the output of that layer (and input to the LocalNet backbone) is now $284 \times 120 \times 64 \times 8$ instead of $284 \times 120 \times 64 \times 2$.
- The affine data augmentation procedure was adjusted to first construct a 3D affine transformation, then independently apply that same transformation to each channel in the 4D input images.

- In the output warping layer, the DDF is now applied independently to each channel of the moving image to create the predicted fixed image.

*Loss function*

The loss function in CellDiscoveryNet is a weighted sum of image loss and regularization loss. As this is an entirely unsupervised learning procedure, label loss is not used.

The image loss component has weight 1 and uses GNCC instead of LNCC used by BrainAlignNet. The GNCC loss is computed as:

$$\mathbf{GNCC} = \frac{1}{C}\sum_{c=1}^{C}\frac{\sum_{x=0}^{d_x-1}\sum_{y=0}^{d_y-1}\sum_{z=0}^{d_z-1}(F[x,y,z]-\mu_{Fc})(P[x,y,z]-\mu_{Pc})}{d_x d_y d_z \sigma_{Fc}^2 \sigma_{Pc}^2}$$

Where $(d_x, d_y, d_z, C)$ are the dimensions of the image, $F$ is the fixed image, $P$ is the predicted fixed image (i.e.: DDF-transformed moving image), $C$ is the number of channels, $\mu_{Ic}$ is the mean of image $I$ in channel $c$, and $\sigma_{Ic}^2$ is the variance of image $I$ in channel $c$.

The regularization loss terms are as before for BrainAlignNet, except with weights 0 for the axis difference norm, 0.05 for the gradient norm, 0.05 for the nonrigid penalty, and 0.0025 for the difference norm.

*Training data*

CellDiscoveryNet was trained on 3,240 pairs of $284 \times 120 \times 64 \times 4$ images, comprising every possible pair combination of 81 distinct images. These were the same 81 images used to train AutoCellLabeler. Each pair consisted of a moving image and a fixed image. Both images were pre-processed by setting the dynamic range of pixel intensities to [0, 1], independently for each channel.

Each moving image was additionally pre-processed by using our previously-described GPU-accelerated Euler registration to coarsely align it to the corresponding fixed image. This registration was run on the NLS-tagRFP channel, and the Euler transform fit to that channel was then independently applied to each other channels to generate the full transformed moving image.

There were 45 validation image pairs (from 10 validation images), and 1,866 testing image pairs. The testing image pairs added 11 additional images, and consisted of all pairs not present in either the training or validation data. (So, for example, a registration problem between an image in the training data and an image in the validation data would count as a testing image pair, since the network never saw that image pair in training or validation.) The split of images in the validation and testing data was identical to that for AutoCellLabeler.

The network was trained for 600 epochs with the Adam optimizer and a learning rate of $10^{-4}$. Full training parameter settings are available at https://github.com/flavell-lab/DeepReg/blob/multicolor/CellDiscoveryNet/train_config.yaml

## **ANTSUN 2U**

To convert the CellDiscoveryNet registration outputs into neuron labels across animals, we created a modified version of our ANTSUN image processing pipeline. We skipped the pre-processing steps since the images were already pre-processed, used 102 four-channel images instead of 1600 one-channel images, set the registration graph to be the complete graph (except each pair of images is only registered once and not once in each direction), substituted BrainAlignNet with CellDiscoveryNet for the nonrigid registration step, and skipped the trace extraction steps of ANTSUN (stopping after it computed linked neuron IDs).

We also modified the heuristic function in the matrix that was subject to clustering to better account for the nature of this multi-spectral data. Specifically, we removed the marker channel brightness heuristic $a_{ij}$ since brightness of neurons relative to the mean ROI is not likely to be well conserved across different animals. We replaced it with a more problem-specific heuristic: color. Specifically, the color $C_i$ of an ROI $R_i$ was defined as the 4-vector of its brightness in each of the four channels, normalized to the average brightness of that ROI across the four channels. We then define

$$a_{ij} = \sum_{k=1}^{4} \left| C_{ik} - C_{jk} \right|$$

where $i, j$ each indicates an ROI label.

In this way, $a_{ij}$ will be small if the ROIs have similar colors and large if they have different colors. We use this new color-based $a_{ij}$ in the same way in the heuristic function that we used the original brightness-based $a_{ij}$, except that we set its weight $w_4 = 7$.

We did not run hyperparameter search on any of the other weight parameters $w_i$ for this dataset to avoid overfitting to the 102 animals included in it, instead leaving them all at their default values from the original ANTSUN 2.0 pipeline (with the one exception of $w_8$ which we set to 0 here in light of having much fewer animals than we did timepoints). We hypothesize that performance may increase even further upon hyperparameter search, though this would likely require considerably more data for testing. The only exception was that we varied parameter $w_7$, which controls the precision vs recall tradeoff. Larger values of $w_7$ result in more, but less accurate, detected clusters; each cluster corresponds to a single neuron class label. We elected to use a value of $w_7 = 10^{-9}$ for all displayed results; the full tradeoff curve is available in Figure 5f.

### *Accuracy metric*

By construction, clusters in ANTSUN 2U should correspond to individual neuron classes. To compute its accuracy, we checked whether clusters indeed only correspond to single neuron classes. Let $L(r, a)$ be the function mapping ROI $r$ in animal $a$ to its human label (ignoring L/R), and let $C_i$ be a set of $(r, a)$ values belonging to the same cluster. We can then define $L_i$ to be the set of labels in $C_i$: $L_i = \{L(r,a) | (r,a) \in C_i, L(r,a) \neq \text{UNKNOWN}\}$. Then let $F_i$ be the most frequent label in $L_i$. We can then define the accuracy of ANTSUN 2U as follows:

$$\textbf{Accuracy} = \frac{\sum_{i \in S} \sum_{l \in L_i} \delta_{lF_i}}{\sum_{i \in S} |L_i|}$$

Here $|L_i|$ is the number of elements in the set $L_i$, $S$ is the set of all clusters with $|L_i| > 2$ (which included all but one cluster in our data with $w_7 = 10^{-9}$), and $\delta$ is the Kronecker delta function.

## REFERENCES

1. Alon, S., Goodwin, D.R., Sinha, A., Wassie, A.T., Chen, F., Daugharthy, E.R., Bando, Y., Kajita, A., Xue, A.G., Marrett, K., et al. (2021). Expansion sequencing: Spatially precise in situ transcriptomics in intact biological systems. Science *371*, eaax2656. https://doi.org/10.1126/science.aax2656.

2. Chen, K.H., Boettiger, A.N., Moffitt, J.R., Wang, S., and Zhuang, X. (2015). RNA imaging. Spatially resolved, highly multiplexed RNA profiling in single cells. Science *348*, aaa6090. https://doi.org/10.1126/science.aaa6090.

3. Ke, R., Mignardi, M., Pacureanu, A., Svedlund, J., Botling, J., Wählby, C., and Nilsson, M. (2013). In situ sequencing for RNA analysis in preserved tissue and cells. Nat. Methods *10*, 857–860. https://doi.org/10.1038/nmeth.2563.

4. Moen, E., Bannon, D., Kudo, T., Graf, W., Covert, M., and Van Valen, D. (2019). Deep learning for cellular image analysis. Nat. Methods *16*, 1233–1246. https://doi.org/10.1038/s41592-019-0403-1.

5. Stirling, D.R., Swain-Bowden, M.J., Lucas, A.M., Carpenter, A.E., Cimini, B.A., and Goodman, A. (2021). CellProfiler 4: improvements in speed, utility and usability. BMC Bioinformatics *22*, 433. https://doi.org/10.1186/s12859-021-04344-9.

6. Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., Xiao, T., Whitehead, S., Berg, A.C., Lo, W.-Y., et al. (2023). Segment Anything. Preprint at arXiv, https://doi.org/10.48550/arXiv.2304.02643 https://doi.org/10.48550/arXiv.2304.02643.

7. Zou, J., Gao, B., Song, Y., and Qin, J. (2022). A review of deep learning-based deformable medical image registration. Front. Oncol. *12*, 1047215. https://doi.org/10.3389/fonc.2022.1047215.

8. Zidane, M., Makky, A., Bruhns, M., Rochwarger, A., Babaei, S., Claassen, M., and Schürch, C.M. (2023). A review on deep learning applications in highly multiplexed tissue imaging data analysis. Front. Bioinforma. *3*, 1159381. https://doi.org/10.3389/fbinf.2023.1159381.

9. Geuenich, M.J., Hou, J., Lee, S., Ayub, S., Jackson, H.W., and Campbell, K.R. (2021). Automated assignment of cell identity from single-cell multiplexed imaging and proteomic data. Cell Syst. *12*, 1173-1186.e5. https://doi.org/10.1016/j.cels.2021.08.012.

10. Amitay, Y., Bussi, Y., Feinstein, B., Bagon, S., Milo, I., and Keren, L. (2023). CellSighter: a neural network to classify cells in highly multiplexed images. Nat. Commun. *14*, 4302. https://doi.org/10.1038/s41467-023-40066-7.

11. Brbić, M., Cao, K., Hickey, J.W., Tan, Y., Snyder, M.P., Nolan, G.P., and Leskovec, J. (2022). Annotation of spatially resolved single-cell data with STELLAR. Nat. Methods *19*, 1411–1418. https://doi.org/10.1038/s41592-022-01651-8.

12. White, J.G., Southgate, E., Thomson, J.N., and Brenner, S. (1986). The structure of the nervous system of the nematode Caenorhabditis elegans. Philos. Trans. R. Soc. Lond. B. Biol. Sci. *314*, 1–340. https://doi.org/10.1098/rstb.1986.0056.

13. Witvliet, D., Mulcahy, B., Mitchell, J.K., Meirovitch, Y., Berger, D.R., Wu, Y., Liu, Y., Koh, W.X., Parvathala, R., Holmyard, D., et al. (2021). Connectomes across development reveal principles of brain maturation. Nature *596*, 257–261. https://doi.org/10.1038/s41586-021-03778-8.

14. Cook, S.J., Jarrell, T.A., Brittin, C.A., Wang, Y., Bloniarz, A.E., Yakovlev, M.A., Nguyen, K.C.Q., Tang, L.T.-H., Bayer, E.A., Duerr, J.S., et al. (2019). Whole-animal connectomes of both Caenorhabditis elegans sexes. Nature *571*, 63–71. https://doi.org/10.1038/s41586-019-1352-7.

15. Prevedel, R., Yoon, Y.-G., Hoffmann, M., Pak, N., Wetzstein, G., Kato, S., Schrödel, T., Raskar, R., Zimmer, M., Boyden, E.S., et al. (2014). Simultaneous whole-animal 3D imaging of neuronal activity using light-field microscopy. Nat. Methods *11*, 727–730. https://doi.org/10.1038/nmeth.2964.

16. Schrödel, T., Prevedel, R., Aumayr, K., Zimmer, M., and Vaziri, A. (2013). Brain-wide 3D imaging of neuronal activity in Caenorhabditis elegans with sculpted light. Nat. Methods *10*, 1013–1020. https://doi.org/10.1038/nmeth.2637.

17. Nguyen, J.P., Shipley, F.B., Linder, A.N., Plummer, G.S., Liu, M., Setru, S.U., Shaevitz, J.W., and Leifer, A.M. (2016). Whole-brain calcium imaging with cellular resolution in freely behaving Caenorhabditis elegans. Proc. Natl. Acad. Sci. U. S. A. *113*, E1074-1081. https://doi.org/10.1073/pnas.1507110112.

18. Venkatachalam, V., Ji, N., Wang, X., Clark, C., Mitchell, J.K., Klein, M., Tabone, C.J., Florman, J., Ji, H., Greenwood, J., et al. (2016). Pan-neuronal imaging in roaming Caenorhabditis elegans. Proc. Natl. Acad. Sci. U. S. A. *113*, E1082-1088. https://doi.org/10.1073/pnas.1507109113.

19. Flavell, S.W., and Gordus, A. (2022). Dynamic functional connectivity in the static connectome of Caenorhabditis elegans. Curr. Opin. Neurobiol. *73*, 102515. https://doi.org/10.1016/j.conb.2021.12.002.

20. Kramer, T.S., and Flavell, S.W. (2024). Building and integrating brain-wide maps of nervous system function in invertebrates. Curr. Opin. Neurobiol. *86*, 102868. https://doi.org/10.1016/j.conb.2024.102868.

21. Lagache, T., Hanson, A., Pérez-Ortega, J.E., Fairhall, A., and Yuste, R. (2021). Tracking calcium dynamics from individual neurons in behaving animals. PLoS Comput. Biol. *17*, e1009432. https://doi.org/10.1371/journal.pcbi.1009432.

22. Hanson, A., Reme, R., Telerman, N., Yamamoto, W., Olivo-Marin, J.-C., Lagache, T., and Yuste, R. (2024). Automatic monitoring of neural activity with single-cell resolution in behaving Hydra. Sci. Rep. *14*, 5083. https://doi.org/10.1038/s41598-024-55608-2.

23. Wen, C., Miura, T., Voleti, V., Yamaguchi, K., Tsutsumi, M., Yamamoto, K., Otomo, K., Fujie, Y., Teramoto, T., Ishihara, T., et al. (2021). 3DeeCellTracker, a deep learning-based pipeline for segmenting and tracking cells in 3D time lapse images. eLife *10*, e59187. https://doi.org/10.7554/eLife.59187.

24. Nejatbakhsh, A., Varol, E., Yemini, E., Venkatachalam, V., Lin, A., Samuel, A.D.T., and Paninski, L. (2020). Extracting neural signals from semi-immobilized animals with deformable non-negative matrix factorization. Preprint at bioRxiv, https://doi.org/10.1101/2020.07.07.192120 https://doi.org/10.1101/2020.07.07.192120.

25. Christensen, R.P., Bokinsky, A., Santella, A., Wu, Y., Marquina-Solis, J., Guo, M., Kovacevic, I., Kumar, A., Winter, P.W., Tashakkori, N., et al. (2015). Untwisting the Caenorhabditis elegans embryo. eLife *4*, e10070. https://doi.org/10.7554/eLife.10070.

26. Ardiel, E.L., Lauziere, A., Xu, S., Harvey, B.J., Christensen, R.P., Nurrish, S., Kaplan, J.M., and Shroff, H. (2022). Stereotyped behavioral maturation and rhythmic quiescence in C. elegans embryos. eLife *11*, e76836. https://doi.org/10.7554/eLife.76836.

27. Ryu, J., Nejatbakhsh, A., Torkashvand, M., Gangadharan, S., Seyedolmohadesin, M., Kim, J., Paninski, L., and Venkatachalam, V. (2024). Versatile multiple object tracking in sparse 2D/3D videos via deformable image registration. PLoS Comput. Biol. *20*, e1012075. https://doi.org/10.1371/journal.pcbi.1012075.

28. Nguyen, J.P., Linder, A.N., Plummer, G.S., Shaevitz, J.W., and Leifer, A.M. (2017). Automatically tracking neurons in a moving and deforming brain. PLoS Comput. Biol. *13*, e1005517. https://doi.org/10.1371/journal.pcbi.1005517.

29. Yu, X., Creamer, M.S., Randi, F., Sharma, A.K., Linderman, S.W., and Leifer, A.M. (2021). Fast deep neural correspondence for tracking and identifying neurons in C. elegans using semi-synthetic training. eLife *10*, e66410. https://doi.org/10.7554/eLife.66410.

30. Wu, Y., Wu, S., Wang, X., Lang, C., Zhang, Q., Wen, Q., and Xu, T. (2022). Rapid detection and recognition of whole brain activity in a freely behaving Caenorhabditis elegans. PLoS Comput. Biol. *18*, e1010594. https://doi.org/10.1371/journal.pcbi.1010594.

31. Nejatbakhsh, A., and Varol, E. (2021). Neuron Matching in C. elegans With Robust Approximate Linear Regression Without Correspondence. In, pp. 2837–2846.

32. Deng, H., Yu, J., and Venkatachalam, V. (2024). Neuron tracking in C. elegans through automated anchor neuron localization and segmentation. In Computational Optical Imaging

and Artificial Intelligence in Biomedical Sciences (SPIE), pp. 84–95. https://doi.org/10.1117/12.3001982.

33. Park, C.F., Barzegar-Keshteli, M., Korchagina, K., Delrocq, A., Susoy, V., Jones, C.L., Samuel, A.D.T., and Rahi, S.J. (2024). Automated neuron tracking inside moving and deforming C. elegans using deep learning and targeted augmentation. Nat. Methods *21*, 142–149. https://doi.org/10.1038/s41592-023-02096-3.

34. Wang, D., Lu, Z., Xu, Y., Wang, Z.I., Santella, A., and Bao, Z. (2019). Cellular structure image classification with small targeted training samples. IEEE Access Pract. Innov. Open Solut. *7*, 148967–148974. https://doi.org/10.1109/access.2019.2940161.

35. Atanas, A.A., Kim, J., Wang, Z., Bueno, E., Becker, M., Kang, D., Park, J., Kramer, T.S., Wan, F.K., Baskoylu, S., et al. (2023). Brain-wide representations of behavior spanning multiple timescales and states in C. elegans. Cell *186*, 4134-4151.e31. https://doi.org/10.1016/j.cell.2023.07.035.

36. Toyoshima, Y., Wu, S., Kanamori, M., Sato, H., Jang, M.S., Oe, S., Murakami, Y., Teramoto, T., Park, C., Iwasaki, Y., et al. (2020). Neuron ID dataset facilitates neuronal annotation for whole-brain activity imaging of C. elegans. BMC Biol. *18*, 30. https://doi.org/10.1186/s12915-020-0745-2.

37. Bubnis, G., Ban, S., DiFranco, M.D., and Kato, S. (2019). A probabilistic atlas for cell identification. Preprint at arXiv, https://doi.org/10.48550/arXiv.1903.09227 https://doi.org/10.48550/arXiv.1903.09227.

38. Chaudhary, S., Lee, S.A., Li, Y., Patel, D.S., and Lu, H. (2021). Graphical-model framework for automated annotation of cell identities in dense cellular images. eLife *10*, e60321. https://doi.org/10.7554/eLife.60321.

39. Skuhersky, M., Wu, T., Yemini, E., Nejatbakhsh, A., Boyden, E., and Tegmark, M. (2022). Toward a more accurate 3D atlas of C. elegans neurons. BMC Bioinformatics *23*, 195. https://doi.org/10.1186/s12859-022-04738-3.

40. Yemini, E., Lin, A., Nejatbakhsh, A., Varol, E., Sun, R., Mena, G.E., Samuel, A.D.T., Paninski, L., Venkatachalam, V., and Hobert, O. (2021). NeuroPAL: A Multicolor Atlas for Whole-Brain Neuronal Identification in C. elegans. Cell *184*, 272-288.e11. https://doi.org/10.1016/j.cell.2020.12.012.

41. Varol, E., Nejatbakhsh, A., Hobert, O., Mena, G., Paninski, L., Yemini, E., and Sun, R. (2020). Statistical atlas of C. elegans neurons (Springer).

42. Sprague, D.Y., Rusch, K., Dunn, R.L., Borchardt, J.M., Bubnis, G., Chiu, G.C., Wen, C., Suzuki, R., Chaudhary, S., Dichter, B., et al. (2024). Unifying community-wide whole-brain imaging datasets enables robust automated neuron identification and reveals determinants of neuron positioning in C. elegans. Preprint at bioRxiv, https://doi.org/10.1101/2024.04.28.591397 https://doi.org/10.1101/2024.04.28.591397.

43. Ma, J., Zhao, J., and Yuille, A.L. (2016). Non-Rigid Point Set Registration by Preserving Global and Local Structures. IEEE Trans. Image Process. *25*, 53–64. https://doi.org/10.1109/TIP.2015.2467217.

44. Fu, Y., Brown, N.M., Saeed, S.U., Casamitjana, A., Baum, Z.M. c, Delaunay, R., Yang, Q., Grimwood, A., Min, Z., Blumberg, S.B., et al. (2020). DeepReg: a deep learning toolkit for medical image registration. J. Open Source Softw. *5*, 2705. https://doi.org/10.21105/joss.02705.

45. Hu, Y., Modat, M., Gibson, E., Ghavami, N., Bonmati, E., Moore, C.M., Emberton, M., Noble, J.A., Barratt, D.C., and Vercauteren, T. (2018). Label-driven weakly-supervised learning for multimodal deformable image registration. In 2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018), pp. 1070–1074. https://doi.org/10.1109/ISBI.2018.8363756.

46. Pradhan, S., Madan, G.K., Kang, D., Bueno, E., Atanas, A.A., Kramer, T.S., Dag, U., Lage, J.D., Gomes, M.A., Lu, A.K.-Y., et al. (2024). Pathogen infection induces sickness behaviors by recruiting neuromodulatory systems linked to stress and satiety in C. elegans. Preprint at bioRxiv, https://doi.org/10.1101/2024.01.05.574345 https://doi.org/10.1101/2024.01.05.574345.

47. Wolny, A., Cerrone, L., Vijayan, A., Tofanelli, R., Barro, A.V., Louveaux, M., Wenzl, C., Strauss, S., Wilson-Sánchez, D., Lymbouridou, R., et al. (2020). Accurate and versatile 3D segmentation of plant tissues at cellular resolution. eLife *9*, e57613. https://doi.org/10.7554/eLife.57613.

48. Sharma, A.K., Randi, F., Kumar, S., Dvali, S., and Leifer, A.M. (2024). TWISP: A Transgenic Worm for Interrogating Signal Propagation in C. elegans. Genetics, iyae077. https://doi.org/10.1093/genetics/iyae077.

49. Kato, S., Kaplan, H.S., Schrödel, T., Skora, S., Lindsay, T.H., Yemini, E., Lockery, S., and Zimmer, M. (2015). Global brain dynamics embed the motor command sequence of Caenorhabditis elegans. Cell *163*, 656–669. https://doi.org/10.1016/j.cell.2015.09.034.

50. Li, Z., Liu, J., Zheng, M., and Xu, X.Z.S. (2014). Encoding of both analog- and digital-like behavioral outputs by one C. elegans interneuron. Cell *159*, 751–765. https://doi.org/10.1016/j.cell.2014.09.056.

51. Chalfie, M., Sulston, J.E., White, J.G., Southgate, E., Thomson, J.N., and Brenner, S. (1985). The neural circuit for touch sensitivity in Caenorhabditis elegans. J. Neurosci. Off. J. Soc. Neurosci. *5*, 956–964.

52. Gordus, A., Pokala, N., Levy, S., Flavell, S.W., and Bargmann, C.I. (2015). Feedback from network states generates variability in a probabilistic olfactory circuit. Cell *161*, 215–227. https://doi.org/10.1016/j.cell.2015.02.018.

53. Luo, L., Wen, Q., Ren, J., Hendricks, M., Gershow, M., Qin, Y., Greenwood, J., Soucy, E.R., Klein, M., Smith-Parker, H.K., et al. (2014). Dynamic encoding of perception, memory, and

movement in a C. elegans chemotaxis circuit. Neuron *82*, 1115–1128. https://doi.org/10.1016/j.neuron.2014.05.010.

54. Wang, Y., Zhang, X., Xin, Q., Hung, W., Florman, J., Huo, J., Xu, T., Xie, Y., Alkema, M.J., Zhen, M., et al. (2020). Flexible motor sequence generation during stereotyped escape responses. eLife *9*, e56942. https://doi.org/10.7554/eLife.56942.

55. Ben Arous, J., Tanizawa, Y., Rabinowitch, I., Chatenay, D., and Schafer, W.R. (2010). Automated imaging of neuronal activity in freely behaving Caenorhabditis elegans. J. Neurosci. Methods *187*, 229–234. https://doi.org/10.1016/j.jneumeth.2010.01.011.

56. Lim, M.A., Chitturi, J., Laskova, V., Meng, J., Findeis, D., Wiekenberg, A., Mulcahy, B., Luo, L., Li, Y., Lu, Y., et al. (2016). Neuroendocrine modulation sustains the C. elegans forward motor state. eLife *5*, e19887. https://doi.org/10.7554/eLife.19887.

57. Kim, J., Rustam, S., Mosquera, J.M., Randell, S.H., Shaykhiev, R., Rendeiro, A.F., and Elemento, O. (2022). Unsupervised discovery of tissue architecture in multiplexed imaging. Nat. Methods *19*, 1653–1661. https://doi.org/10.1038/s41592-022-01657-2.

58. Liu, C.C., Greenwald, N.F., Kong, A., McCaffrey, E.F., Leow, K.X., Mrdjen, D., Cannon, B.J., Rumberger, J.L., Varra, S.R., and Angelo, M. (2023). Robust phenotyping of highly multiplexed tissue imaging data using pixel-level clustering. Nat. Commun. *14*, 4618. https://doi.org/10.1038/s41467-023-40068-5.

59. Dag, U., Nwabudike, I., Kang, D., Gomes, M.A., Kim, J., Atanas, A.A., Bueno, E., Estrem, C., Pugliese, S., Wang, Z., et al. (2023). Dissecting the functional organization of the C. elegans serotonergic system at whole-brain scale. Cell *186*, 2574-2592.e20. https://doi.org/10.1016/j.cell.2023.04.023.

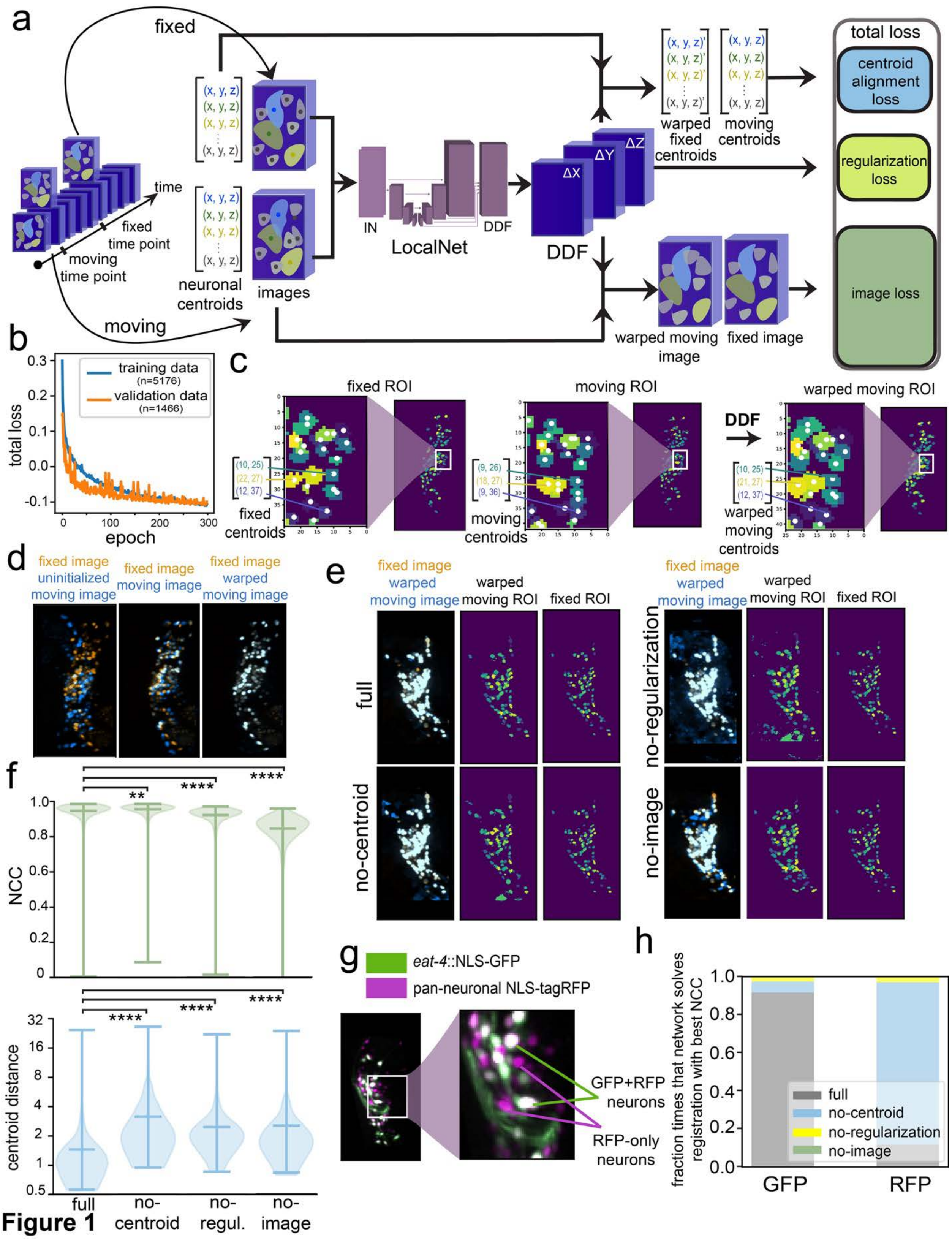

a
fixed
time
fixed time point
moving time point
moving
(x, y, z)
neuronal centroids
images
IN
LocalNet
DDF
ΔX
ΔY
ΔZ
DDF
(x, y, z)'
warped fixed centroids
moving centroids
warped moving image
fixed image
total loss
centroid alignment loss
regularization loss
image loss
b
training data (n=5176)
validation data (n=1466)
total loss
0.3
0.2
0.1
0.0
-0.1
0
100
200
300
epoch
c
fixed ROI
moving ROI
DDF
warped moving ROI
(10, 25)
(22, 27)
(12, 37)
fixed centroids
(9, 26)
(18, 27)
(9, 36)
moving centroids
(10, 25)
(21, 27)
(12, 37)
warped moving centroids
d
fixed image
uninitialized moving image
fixed image
moving image
fixed image
warped moving image
e
fixed image
warped moving image
warped moving ROI
fixed ROI
full
no-centroid
no-regularization
no-image
f
**
****
****
NCC
1.0
0.8
0.6
0.4
0.2
0
****
****
****
centroid distance
32
16
8
4
2
1
0.5
full
no-centroid
no-regul.
no-image
g
eat-4::NLS-GFP
pan-neuronal NLS-tagRFP
GFP+RFP neurons
RFP-only neurons
h
fraction times that network solves registration with best NCC
1.0
0.8
0.6
0.4
0.2
0.0
full
no-centroid
no-regularization
no-image
GFP
RFP

**Figure 1**

**Figure 1. BrainAlignNet can perform non-rigid registration to align the neurons in the *C. elegans* head**

**(A)** Network training pipeline. The network takes in a pair of images and a pair of centroid position lists corresponding to the images at two different time points (fixed and moving). (In the LocalNet diagram, this is represented as "IN". Intermediate cuboids represent intermediate representations of the images at various stages of network processing. In reality, the cuboids are four-dimensional, but we represent them with three dimensions (up/down is $x$, left/right is $y$, in/out is channel, and we omit $z$) for visualization purposes. Spaces and arrows between cuboids represent network blocks, layers, and information flow. See Methods for a detailed description of network architectures.) Image pairs were selected based on the similarity of worm postures (see Methods). The fixed and moving images were pre-registered using an Euler transformation, translating and rotating the moving images to maximize their cross-correlation with the fixed images. The fixed and moving neuron centroid positions were obtained by computing the centers of the same neurons in both the fixed and moving images as a list of ($x$, $y$, $z$) coordinates. This information was available since we had previously extracted calcium traces from these videos using a previous, slow version of our image analysis pipeline. The network outputs a Dense Displacement Field (DDF), a 4-D tensor that indicates a coordinate transformation from fixed image coordinates to moving image coordinates. The DDF is then used to transform the moving images and fixed centroids to resemble the fixed images and moving centroids. During training, the network is tasked with learning a DDF that transforms the centroids and images in a way that minimizes the centroid alignment and image loss, as well as the regularization loss (see Methods). Note that, after training, only images (not centroids) need to be input into the network to align the images.

**(B)** Network loss curves. The training and validation loss curves show that validation performance plateaued around 300 epochs of training.

**(C)** Example of registration outcomes on neuronal ROI images. The network-learned DDF warps the neurons in the moving image ('moving ROIs'). The warped-moving ROIs are meant to be closer to the fixed ROIs. Each neuron is uniquely colored in the ROI images to represent its identity. The centroids of these neurons are represented by the white dots. Here, we take a $z$-slice of the 3-D fixed and moving ROI blocks on the $x$-$y$ plane to show that the DDF can warp the $x$ and $y$ coordinates of the moving centroids to align with the $x$ and $y$ coordinates of the fixed centroids with one-pixel precision.

**(D)** Example of registration outcomes on tagRFP images. We show the indicated image blocks as Maximal Intensity Projections (MIPs) along the z-axis, overlaying the fixed image (orange) with different versions of the moving image (blue). While the fixed image remains untransformed, the uninitialized moving image (left) gets warped by an Euler transformation (middle) and a network-learned DDF (right) to overlap with the fixed image.

**(E)** Registration outcomes shown on example tagRFP and ROI images for four different trained networks. We randomly selected one registration problem from one of the testing datasets and tasked the trained networks with creating a DDF to warp the moving (RFP) image and moving ROI onto the fixed (RFP) image and fixed ROI. The full network with full loss function aligns neurons in both RFP and ROI images almost perfectly. For the networks trained without the centroid alignment loss, regularization loss, or image loss—while keeping the rest of the training configurations identical—the resulting DDF is unable to fully align the neurons and displays unrealistic deformation (closely inspect the warped moving ROI images).

**(F)** Evaluation of registration performance on testing datasets under four different network configurations. Here, we evaluated 80-100 problems per animal for all animals in the testing data. Two performance metrics are shown. Normalized cross-correlation (NCC, top) quantifies alignment of the fixed and warped moving RFP images, where a score of one indicates perfect alignment. Centroid distance (bottom) is measured as the mean Euclidean distance between the centroids of all neurons in the fixed ROI and the centroids of their corresponding neurons in the warped moving ROI; a distance of 0 indicates perfect alignment. All violin plots are accompanied by lines indicating the minimum, mean, and maximum values. $^{**}p<0.01$, $^{***}p<0.001$, $^{****}p<0.0001$, distributions of registration metrics (NCC and centroid distance) were compared pairwise across all four versions of the network with the Wilcoxon signed rank test.

**(G)** Example image of the head of an animal from a strain that expresses both pan-neuronal NLS-tagRFP and eat-4::NLS-GFP. The neurons expressing both NLS-tagRFP and eat-4::NLS-GFP is a subset of all the neurons expressing pan-neuronal NLS-tagRFP.

**(H)** A comparison of the registration qualities of the four trained registration networks: full network, no-centroid alignment loss, no-regularization loss, no-image loss. Each network was evaluated on four datasets in which both pan-neuronal NLS-tagRFP and *eat-4*::NLS-GFP are expressed, examining 3927 registration problems per dataset. For a total of 15,708 registration problems, each network was tasked with registering the tagRFP images. The resulting DDFs from the tagRFP registrations were also used to register the *eat-4*::GFP images. For each channel in each problem, we determined which of the four networks had the highest performance (i.e. highest NCC). Note that the no-centroid alignment network performs the best of the RFP channel, but not in the GFP channel. Instead, the full network performs the best in the GFP channel. This suggests that the network without the centroid alignment loss deforms RFP images in a manner that does not accurately move the neurons to their correct locations (i.e. scrambles the pixels).

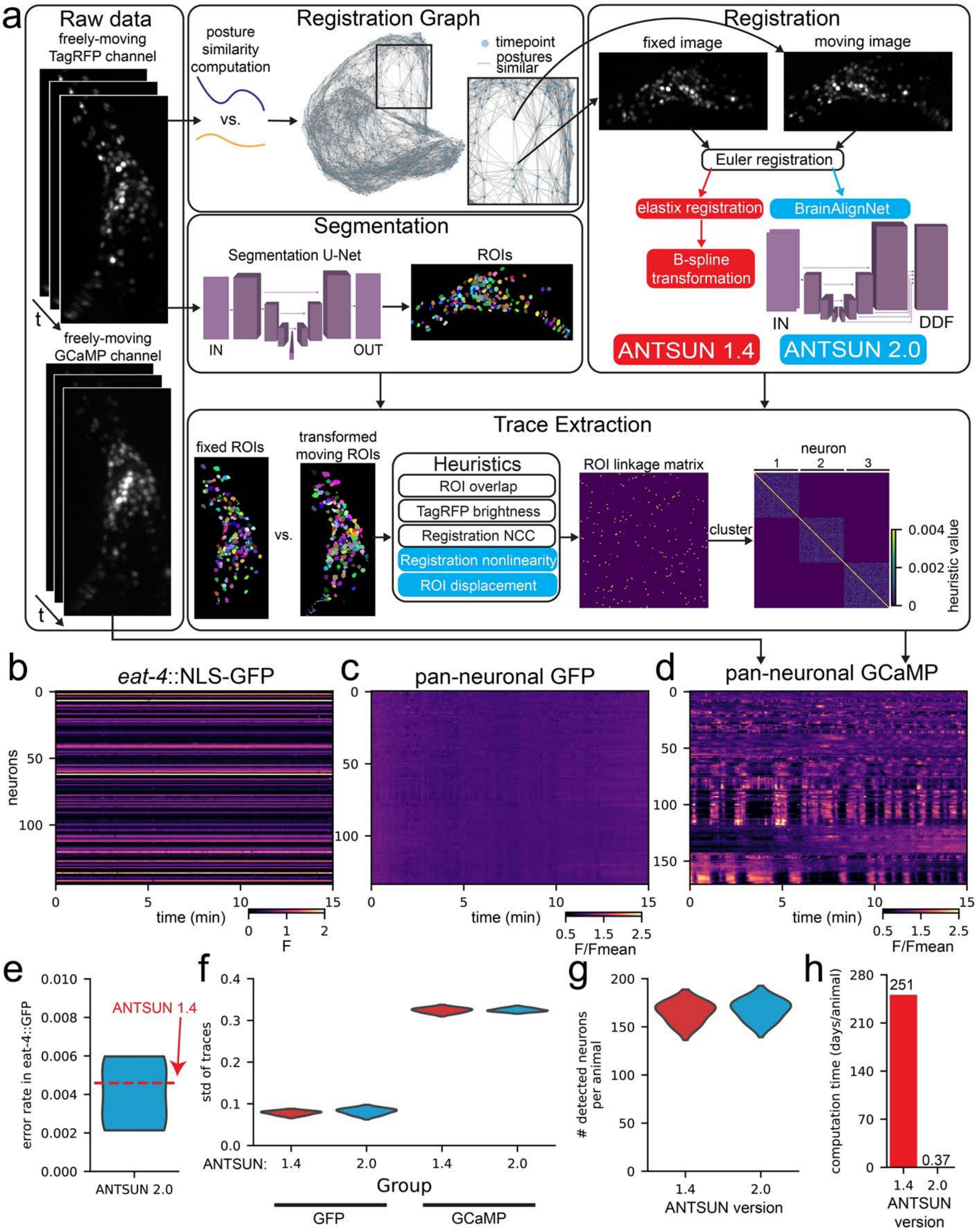

a
Raw data
freely-moving
TagRFP channel
t
freely-moving
GCaMP channel
t
Registration Graph
posture
similarity
computation
vs.
timepoint
postures
similar
Registration
fixed image
moving image
Euler registration
elastix registration
BrainAlignNet
B-spline
transformation
IN
DDF
ANTSUN 1.4
ANTSUN 2.0
Segmentation
Segmentation U-Net
ROIs
IN
OUT
Trace Extraction
fixed ROIs
transformed
moving ROIs
vs.
Heuristics
ROI overlap
TagRFP brightness
Registration NCC
Registration nonlinearity
ROI displacement
ROI linkage matrix
cluster
neuron
1
2
3
0.004
0.002
0
heuristic value
b
eat-4::NLS-GFP
neurons
0
50
100
0
5
10
15
time (min)
0
1
2
F
c
pan-neuronal GFP
0
50
100
0
5
10
15
time (min)
0.5
1.5
2.5
F/Fmean
d
pan-neuronal GCaMP
0
50
100
150
0
5
10
15
time (min)
0.5
1.5
2.5
F/Fmean
e
error rate in eat-4::GFP
0.010
0.008
0.006
0.004
0.002
0.000
ANTSUN 1.4
ANTSUN 2.0
f
std of traces
0.4
0.3
0.2
0.1
0.0
ANTSUN:
1.4
2.0
1.4
2.0
Group
GFP
GCaMP
g
# detected neurons
per animal
200
150
100
50
0
1.4
2.0
ANTSUN version
h
computation time (days/animal)
280
210
140
70
0
251
0.37
1.4
2.0
ANTSUN
version

**Figure 2**

**Figure 2. BrainAlignNet supports calcium trace extraction with high accuracy and high SNR**

**(A)** Diagram of ANTSUN 1.4 and 2.0, which are two full calcium trace extraction pipelines that only differ with regards to image registration. Raw tagRFP channel data is input into the pipeline, which submits image pairs with similar worm postures for registration using either elastix (ANTSUN 1.4; red) or BrainAlignNet (ANTSUN 2.0; blue). The registration is used to transform neuron ROIs identified by a segmentation U-Net (the cuboid diagram is represented as in Figure 1A). These are input into a heuristic function (ANTSUN 2.0-specific heuristics shown in blue) which defines an ROI linkage matrix. Clustering this matrix then yields neuron identities.

**(B)** Sample dataset from an *eat-4*::NLS-GFP strain, showing ratiometric (GFP/tagRFP) traces without any further normalization. This strain has some GFP+ neurons (bright horizontal lines) as well as some GFP- neurons (dark horizontal lines, which have F~0). Registration artifacts between GFP+ and GFP- neurons would be visible as bright points in GFP- traces or dark points in GFP+ traces.

**(C)** Sample dataset from a pan-neuronal GFP strain, showing F/Fmean fluorescence. Any variation visible here is noise.

**(D)** Sample dataset from a pan-neuronal GCaMP strain, showing F/Fmean fluorescence. Robust calcium dynamics are visible in most neurons.

**(E)** Violin plot of the error rate of ANTSUN 2.0 registration across four *eat-4*::NLS-GFP animals, computed based on mismatches between GFP+ and GFP- neurons in the *eat-4*::NLS-GFP strain. Dashed red line shows the error rate of ANTSUN 1.4. Note that all error rates are <1%.

**(F)** Violin plots of the standard deviation of traces across three animals per strain (pan-neuronal GFP or pan-neuronal GCaMP).

**(G)** Violin plots of the number of detected neurons across three pan-neuronal GCaMP animals for the two different ANTSUN versions (1.4 or 2.0).

**(H)** Computation time to process one animal based on ANTSUN version (1.4 or 2.0). ANTSUN 1.4 was run on a computing cluster that provided an average of 32 CPU cores per registration problem; computation time is the total number of CPU hours used (ie: the time it would have taken to run ANTSUN 1.4 registration locally on a comparable 32-core machine). ANTSUN 2.0 was run locally on NVIDIA A4000, A5500, and A6000 graphics cards.

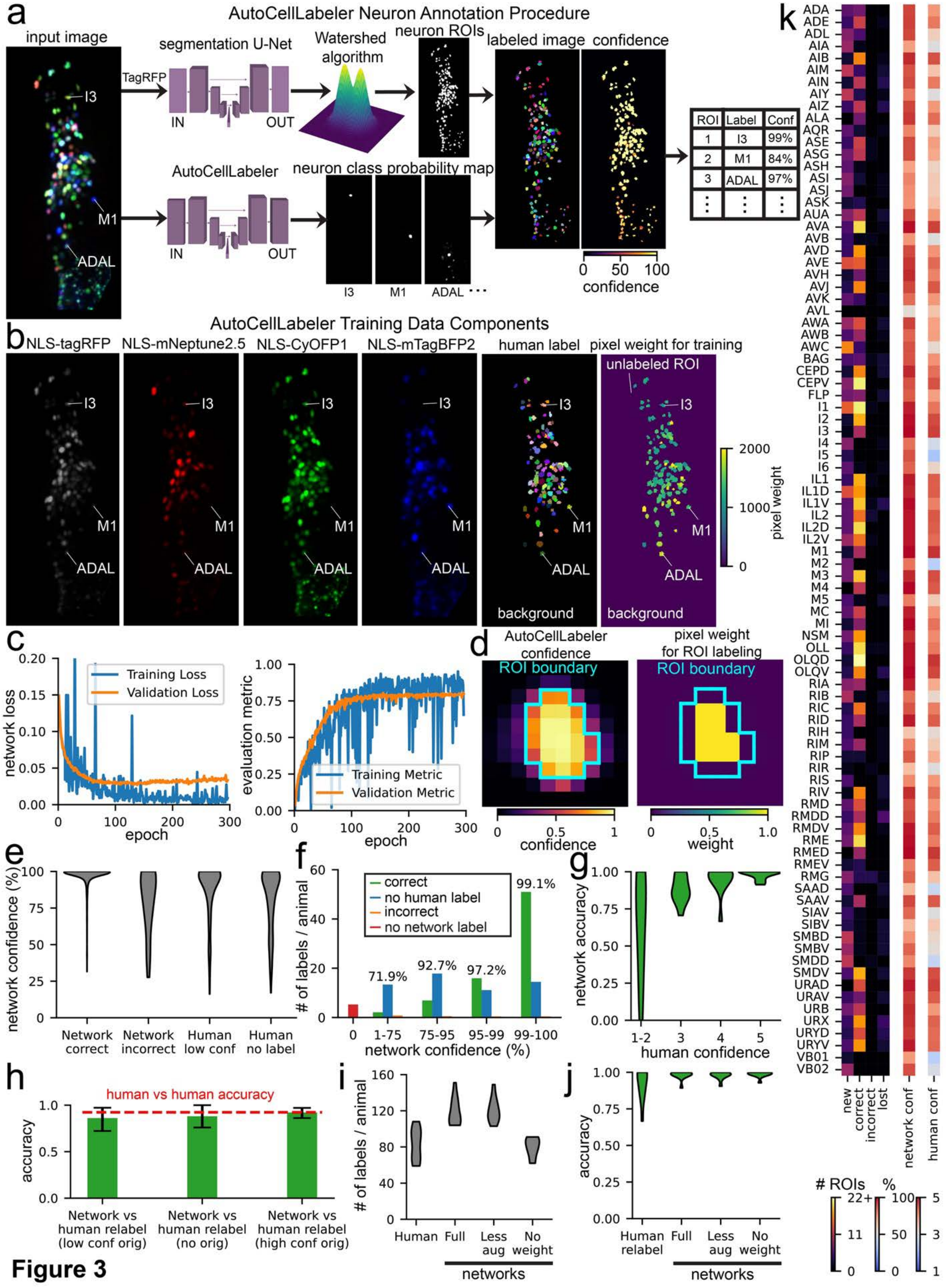

a
AutoCellLabeler Neuron Annotation Procedure
input image
I3
M1
ADAL
TagRFP
segmentation U-Net
IN
OUT
Watershed algorithm
neuron ROIs
labeled image
confidence
AutoCellLabeler
neuron class probability map
I3
M1
ADAL
0
50
100
confidence
ROI | Label | Conf
1 | I3 | 99%
2 | M1 | 84%
3 | ADAL | 97%
b
AutoCellLabeler Training Data Components
NLS-tagRFP
NLS-mNeptune2.5
NLS-CyOFP1
NLS-mTagBFP2
human label
pixel weight for training
unlabeled ROI
background
2000
1000
0
pixel weight
c
Training Loss
Validation Loss
network loss
epoch
Training Metric
Validation Metric
evaluation metric
d
AutoCellLabeler confidence
ROI boundary
pixel weight for ROI labeling
confidence
weight
e
network confidence (%)
Network correct
Network incorrect
Human low conf
Human no label
f
correct
no human label
incorrect
no network label
71.9%
92.7%
97.2%
99.1%
# of labels / animal
0
1-75
75-95
95-99
99-100
network confidence (%)
g
network accuracy
1-2
3
4
5
human confidence
h
human vs human accuracy
accuracy
Network vs human relabel (low conf orig)
Network vs human relabel (no orig)
Network vs human relabel (high conf orig)
i
# of labels / animal
Human
Full
Less aug
No weight
networks
j
accuracy
Human relabel
Full
Less aug
No weight
networks
k
ADA ADE ADL AIA AIB AIM AIN AIY AIZ ALA AQR ASE ASG ASH ASI ASJ ASK AUA AVA AVB AVD AVE AVH AVJ AVK AVL AWA AWB AWC BAG CEPD CEPV FLP I1 I2 I3 I4 I5 I6 IL1 IL1D IL1V IL2 IL2D IL2V M1 M2 M3 M4 M5 MC MI NSM OLL OLQD OLQV RIA RIB RIC RID RIH RIM RIP RIR RIS RIV RMD RMDD RMDV RME RMED RMEV RMG SAAD SAAV SIAV SIBV SMBD SMBV SMDD SMDV URAD URAV URB URX URYD URYV VB01 VB02
new
correct
incorrect
lost
network conf
human conf
# ROIs
%
22+
11
0
100
50
0
5
3
1

Figure 3

**Figure 3. The AutoCellLabeler Network can automatically annotate >100 neuronal cell types in the *C. elegans* head**

- **(A)** Procedure by which AutoCellLabeler generates labels for neurons. First, the tagRFP component of a multi-spectral image is passed into a segmentation neural network, which extracts neuron ROIs, labeling each pixel as an arbitrary number with one number per neuron. Then, the full multi-spectral image is input into AutoCellLabeler, which outputs a probability map. This probability map is applied to the ROIs to generate labels and confidence values for those labels. The network cuboid diagrams are represented as in **Figure 1A**.
- **(B)** AutoCellLabeler's training data consists of a set of multi-spectral images (NLS-tagRFP, NLS-mNeptune2.5, NLS-CyOFP1, and NLS-mTagBFP2), human neuron labels, and a pixel weighting matrix based on confidence and frequency of the human labels that controls how much each pixel is weighted in AutoCellLabeler's loss function.
- **(C)** Pixel-weighted cross-entropy loss and pixel-weighted IoU metric scores for training and validation data. Cross-entropy loss captures the discrepancy between predicted and actual class probabilities for each pixel. The IoU metric describes how accurately the predicted labels overlap with the ground truth labels.
- **(D)** During the label extraction procedure, AutoCellLabeler is less confident of its label on pixels near the edge of ROI boundaries. Therefore, we allow the central pixels to have much higher weight when determining the overall ROI label from pixel-level network output.
- **(E)** Distributions of AutoCellLabeler's confidence across test datasets based on the relationship of its label to the human label ("Correct" = agree, "Incorrect" = disagree, "Human low conf" = human had low confidence, "Human no label" – human did not even guess a label for the neuron).
- **(F)** Categorization of neurons in test datasets based on AutoCellLabeler's confidence. Here "Correct" and "Incorrect" are as in **(E)**, but "No human label" also includes low-confidence human labels. Printed percentage values are the accuracy of AutoCellLabeler on the corresponding category, computed as $\frac{correct}{correct+incorrect}$
- **(G)** Distributions of accuracy of AutoCellLabeler's high confidence (>75%) labels on neurons across test datasets based on the confidence of the human labels.
- **(H)** Accuracy of AutoCellLabeler compared with high-confidence labels from new human labelers on neurons in test datasets that were labeled at low confidence, not at all, or at high confidence by the original human labelers. Error bars are bootstrapped 95% confidence intervals. Dashed red line shows accuracy of new human labelers relative to the old human labelers, when both gave high confidence to their labels.
- **(I)** Distributions of number of high-confidence labels per animal over test datasets. High confidence was 4-5 for human labels and >75% for network labels.
- **(J)** Distributions of accuracy of high-confidence labels per animal over test datasets, relative to the original human labels.
- **(K)** Number of ROIs per neuron class labeled at high confidence in test datasets that fall into each category, along with average confidence for all labels for each neuron class in those test datasets. "New" represents ROIs that were labeled by the network as the neuron and were not labeled by the human. "Correct" represents ROIs that were labeled by both AutoCellLabeler and the human as that neuron. "Incorrect" represents ROIs that were labeled by the network as that neuron and were labeled by the human as something else.

"Lost" represents ROIs that were labeled by the human as that neuron and were not labeled by the network. "Network conf" represents the average confidence of the network for all its labels of that neuron. "Human conf" represents the average confidence of the human labelers for all their labels of that neuron. Neuron classes with high values in the "Correct" column and low values in the "Incorrect" column indicate a very high degree of accuracy in AutoCellLabeler's labels for those classes. If those classes also have a high value in the "New" column, it could indicate that AutoCellLabeler is able to find the neuron with high accuracy in animals where humans were unable to label it.

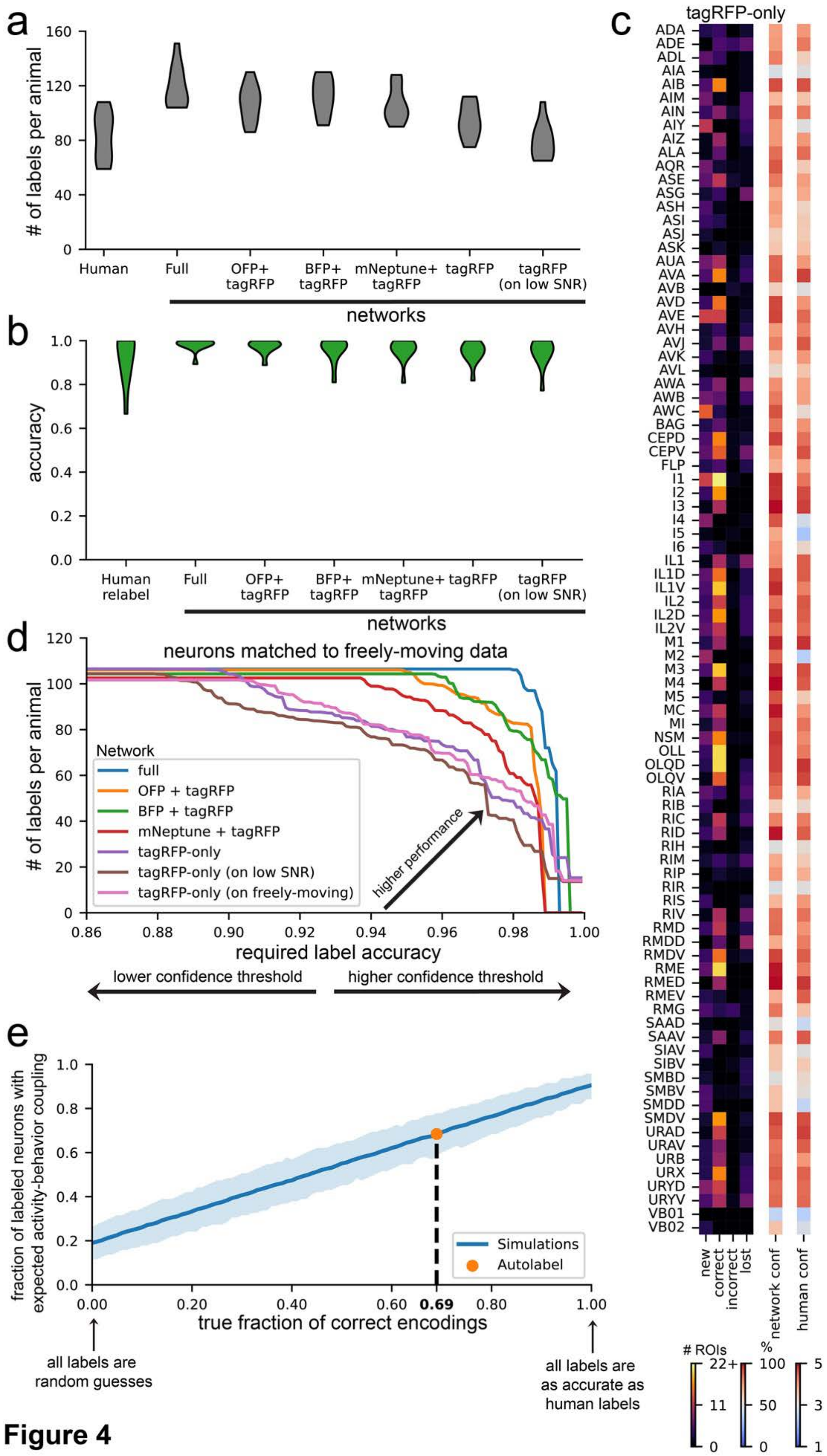

a
# of labels per animal
0
40
80
120
160
Human
Full
OFP+ tagRFP
BFP+ tagRFP
mNeptune+ tagRFP
tagRFP
tagRFP (on low SNR)
networks
b
accuracy
0.0
0.2
0.4
0.6
0.8
1.0
Human relabel
Full
OFP+ tagRFP
BFP+ tagRFP
mNeptune+ tagRFP
tagRFP
tagRFP (on low SNR)
networks
c
tagRFP-only
ADA
ADE
ADL
AIA
AIB
AIM
AIN
AIY
AIZ
ALA
AQR
ASE
ASG
ASH
ASI
ASJ
ASK
AUA
AVA
AVB
AVD
AVE
AVH
AVJ
AVK
AVL
AWA
AWB
AWC
BAG
CEPD
CEPV
FLP
I1
I2
I3
I4
I5
I6
IL1
IL1D
IL1V
IL2
IL2D
IL2V
M1
M2
M3
M4
M5
MC
MI
NSM
OLL
OLQD
OLQV
RIA
RIB
RIC
RID
RIH
RIM
RIP
RIR
RIS
RIV
RMD
RMDD
RMDV
RME
RMED
RMEV
RMG
SAAD
SAAV
SIAV
SIBV
SMBD
SMBV
SMDD
SMDV
URAD
URAV
URB
URX
URYD
URYV
VB01
VB02
new
correct
incorrect
lost
network conf
human conf
# ROIs
0
11
22+
%
0
50
100
1
3
5
d
neurons matched to freely-moving data
# of labels per animal
0
20
40
60
80
100
120
Network
full
OFP + tagRFP
BFP + tagRFP
mNeptune + tagRFP
tagRFP-only
tagRFP-only (on low SNR)
tagRFP-only (on freely-moving)
higher performance
0.86
0.88
0.90
0.92
0.94
0.96
0.98
1.00
required label accuracy
lower confidence threshold
higher confidence threshold
e
fraction of labeled neurons with expected activity-behavior coupling
0.0
0.2
0.4
0.6
0.8
1.0
Simulations
Autolabel
0.00
0.20
0.40
0.60
0.69
0.80
1.00
true fraction of correct encodings
all labels are random guesses
all labels are as accurate as human labels

**Figure 4**

**Figure 4. Variants of AutoCellLabeler can annotate neurons from fewer fluorescent channels and in different strains**

- **(A)** Distributions of number of high-confidence labels per animal over test datasets for the networks trained on the indicated set of fluorophores. The "tagRFP (on low SNR)" column corresponds to a network that was trained on high-SNR, tagRFP-only data and tested on low-SNR tagRFP data due to shorter exposure times in freely-moving animals.
- **(B)** Distributions of accuracy of high-confidence labels per animal over test datasets for the networks trained on the indicated set of fluorophores. The "tagRFP (on low SNR)" column is as in **(A)**.
- **(C)** Same as **Figure 3K**, except for the tagRFP-only network.
- **(D)** Accuracy vs detection tradeoff for various AutoCellLabeler versions. For each network, we can set a confidence threshold above which we accept labels. By varying this threshold, we can produce a tradeoff between accuracy of accepted labels (*x*-axis) and number of labels per animal (*y*-axis) on test data. Each curve in this plot was generated in this manner. The "tagRFP-only (on low SNR)" values are as in **(A)**. The "tagRFP-only (on freely-moving)" values come from evaluating the tagRFP-only network on 100 randomly-chosen timepoints in the freely-moving (tagRFP) data for each test dataset. The final labels were then computed on each immobilized ROI by averaging together the 100 labels and finding the most likely label. To ensure fair comparison to other networks, only immobilized ROIs that were matched to the freely-moving data were considered for any of the networks in this plot (unlike Extended Data Figure 2A, which used all available ROIs).
- **(E)** Evaluating the performance of tagRFP-only AutoCellLabeler on data from another strain SWF415, where there is pan-neuronal NLS-GCaMP7f and pan-neuronal NLS-mNeptune2.5. Notably, the pan-neuronal promoter used for NLS-mNeptune2.5 differs from the pan-neuronal promoter used for NLS-tagRFP in NeuroPAL. Performance here was quantified by computing the fraction of network labels with the correct expected activity-behavior relationships in the neuron class (y-axis; quantified by whether an encoding model showed significant encoding; see Methods). For example, when the label was the reverse-active AVA neuron, did the corresponding calcium trace show higher activity during reverse? The blue line shows the expected fraction as a function of the true accuracy of the network (x-axis), computed via simulations (see Methods). Orange circle shows the actual fraction when AutoCellLabeler was evaluated on SWF415. Based on this, the dashed line shows estimated true accuracy of this labeling.

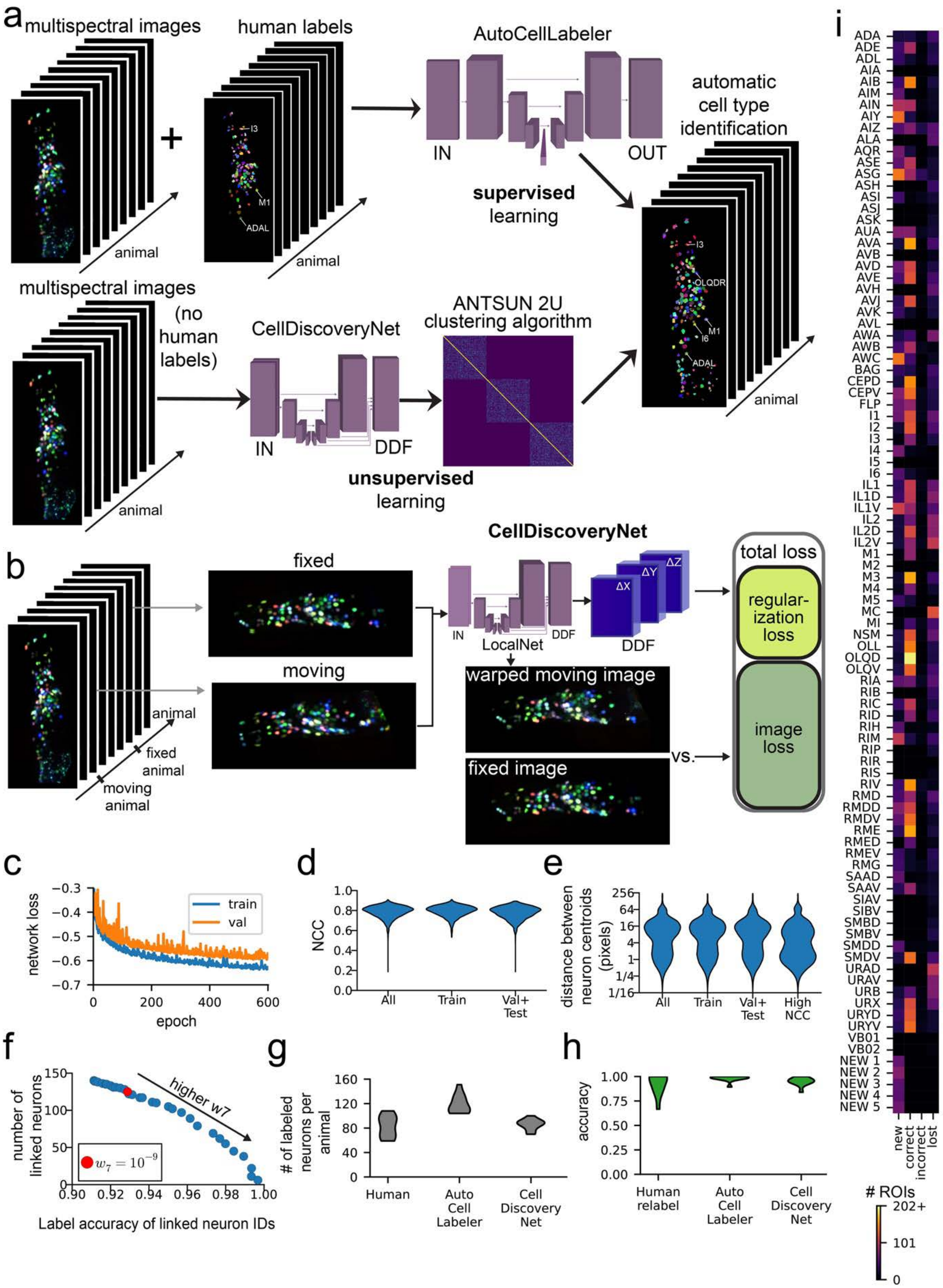

a
multispectral images
+
animal
human labels
I3
M1
ADAL
animal
AutoCellLabeler
IN
OUT
supervised
learning
automatic
cell type
identification
I3
OLQDR
M1
I6
ADAL
animal
multispectral images
(no
human
labels)
animal
CellDiscoveryNet
IN
DDF
unsupervised
learning
ANTSUN 2U
clustering algorithm
b
fixed
moving
animal
fixed
animal
moving
animal
CellDiscoveryNet
IN
LocalNet
DDF
ΔX
ΔY
ΔZ
DDF
warped moving image
fixed image
vs.
total loss
regular-
ization
loss
image
loss
c
network loss
−0.3
−0.4
−0.5
−0.6
−0.7
0
200
400
600
epoch
train
val
d
NCC
1.0
0.8
0.6
0.4
0.2
0.0
All
Train
Val+
Test
e
distance between
neuron centroids
(pixels)
256
64
16
4
1
1/4
1/16
All
Train
Val+
Test
High
NCC
f
number of
linked neurons
150
100
50
0
0.90
0.92
0.94
0.96
0.98
1.00
Label accuracy of linked neuron IDs
higher w7
$w_7 = 10^{-9}$
g
# of labeled
neurons per
animal
160
120
80
40
0
Human
Auto
Cell
Labeler
Cell
Discovery
Net
h
accuracy
1.00
0.75
0.50
0.25
0.00
Human
relabel
Auto
Cell
Labeler
Cell
Discovery
Net
i
ADA
ADE
ADL
AIA
AIB
AIM
AIN
AIY
AIZ
ALA
AQR
ASE
ASG
ASH
ASI
ASJ
ASK
AUA
AVA
AVB
AVD
AVE
AVH
AVJ
AVK
AVL
AWA
AWB
AWC
BAG
CEPD
CEPV
FLP
I1
I2
I3
I4
I5
I6
IL1
IL1D
IL1V
IL2
IL2D
IL2V
M1
M2
M3
M4
M5
MC
MI
NSM
OLL
OLQD
OLQV
RIA
RIB
RIC
RID
RIH
RIM
RIP
RIR
RIS
RIV
RMD
RMDD
RMDV
RME
RMED
RMEV
RMG
SAAD
SAAV
SIAV
SIBV
SMBD
SMBV
SMDD
SMDV
URAD
URAV
URB
URX
URYD
URYV
VB01
VB02
NEW 1
NEW 2
NEW 3
NEW 4
NEW 5
new
correct
incorrect
lost
# ROIs
202+
101
0

**Figure 5**

**Figure 5. CellDiscoveryNet and ANTSUN 2U can perform unsupervised cell type discovery by analyzing data across different *C. elegans* animals**

**(A)** A schematic comparing the approaches of AutoCellLabeler and CellDiscoveryNet. AutoCellLabeler uses supervised learning, taking as input both images and manual labels for those images, and learns to label neurons accordingly. CellDiscoveryNet uses unsupervised learning, and can learn to label neurons after being trained only on images (with no labels provided).

**(B)** CellDiscoveryNet training pipeline. The network takes as input two multi-spectral NeuroPAL images from two different animals. It then outputs a Dense Displacement Field (DDF), which is a coordinate transformation between the two images. It warps the moving image under this DDF, producing a warped moving image that should ideally look very similar to the fixed image. The dissimilarity between these images is the image loss component of the loss function, which is added to the regularization loss that penalizes non-linear image deformations present in the DDF.

**(C)** Network loss curves. Both training and validation loss curves start to plateau around 600 epochs.

**(D)** Distributions of normalized cross-correlation (NCC) scores comparing the CellDiscoveryNet predictions (warped moving images) and the fixed images for each pair of registered images. These NCCs were computed on all four channels simultaneously, treating the entire image as a single 4D matrix for this purpose. The "Train" distribution contains the NCC scores for all pairs of images present in CellDiscoveryNet's training data, while the "Val+Test" distribution contains any pair of images that was not present in its training data.

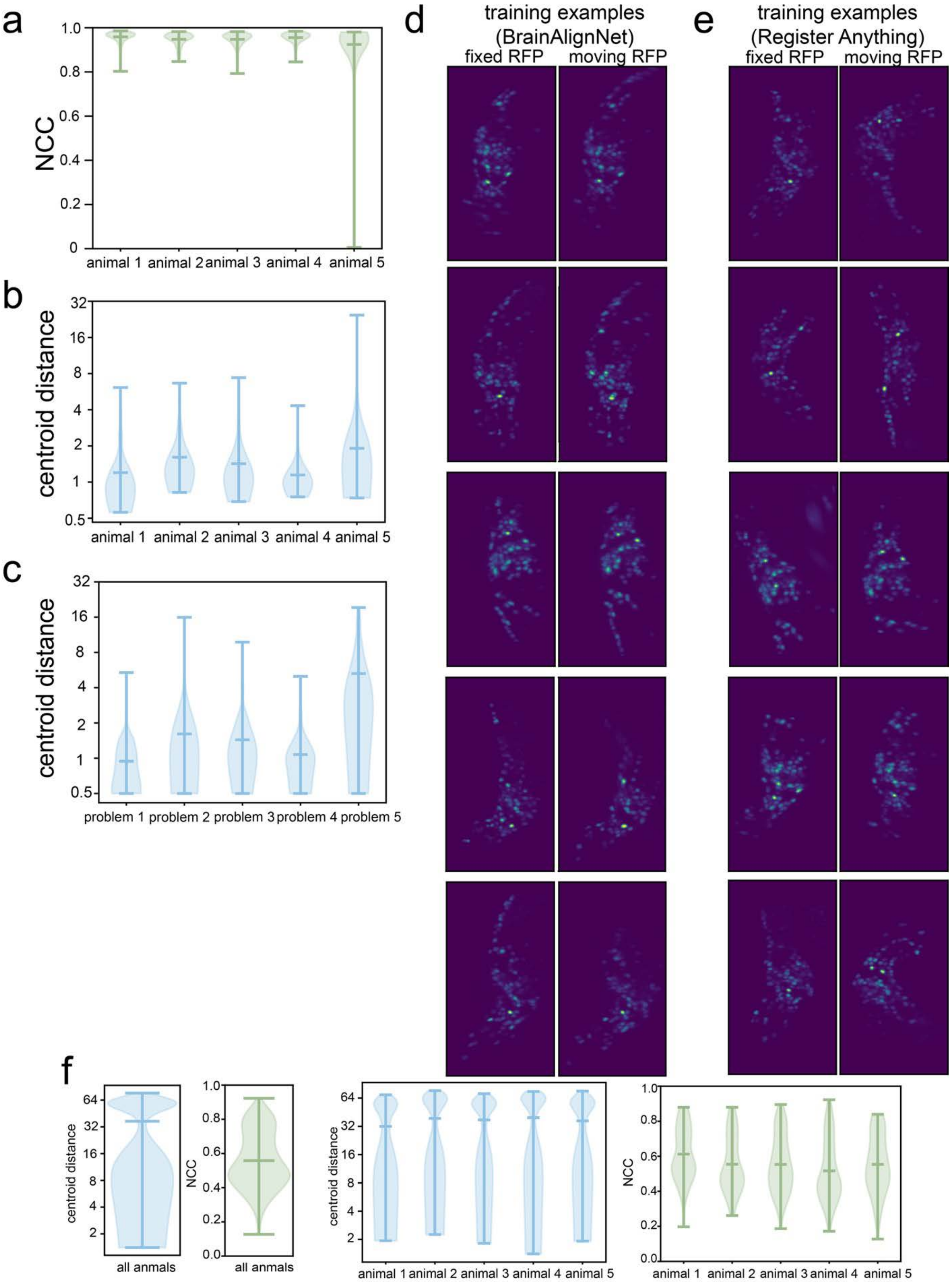
**(E)** Distributions of centroid distance scores based on human labels. These are computed over all (moving, fixed) image pairs on all neurons with high-confidence human labels in both moving and fixed images. The centroid distance scores represent the Euclidean distance from the network's prediction for where the neuron was and its correct location as labeled by the human. Values of a few pixels or less likely roughly indicate that the neuron was mapped to its correct location, while large values mean the neuron was mis-registered. The "Train" and "Val+Test" distributions are as in **(D)**. The "High NCC" distribution is from only (moving, fixed) image pairs where the NCC score was greater than the 90th percentile of all such NCC scores.

**(F)** Labeling accuracy vs number of linked neurons tradeoff curve. Accuracy is the fraction of linked ROIs with labels matching their cluster's most frequent label (see Methods). Number of linked neurons is the total number of distinct clusters; each cluster must contain an ROI in more than half of the animals to be considered a cluster. The parameter $w_7$ describes when to terminate the clustering algorithm – higher values mean the clustering algorithm terminates earlier, resulting in more accurate but fewer detections. Red dot is the selected value $w_7 = 10^{-9}$ where 125 clusters were detected with 93% labeling accuracy.

**(G)** Number of neurons labeled per animal in the 11 testing datasets. This plot compares the number of neurons labeled as follows: human labels with 4-5 confidence, AutoCellLabeler labels with 75% or greater confidence, and CellDiscoveryNet with ANTSUN 2U labels with parameter $w_7 = 10^{-9}$.

**(H)** Accuracy of neuron labels in the 11 testing datasets. This plot defines the original human confidence 4-5 labels as ground truth. "Human relabel" are confidence 4-5 labels done by different humans (independently from the first set of human labels). AutoCellLabeler are confidence 75% or greater labels. CellDiscoveryNet labels were created by running ANTSUN 2U with $w_7 = 10^{-9}$, and defining the correct label for each cluster to be its most frequent label.

**(I)** Same as **Figure 4(K)**, except using labels from CellDiscoveryNet with ANTSUN 2U. The neurons "NEW 1" through "NEW 5" are clusters that were not labeled frequently enough by humans to be able to determine which neuron class they corresponded to, as described in the main text.



**Extended Data Figure 1**

**Extended Data Figure 1. Example images and performance of network trained to register arbitrary image pairs.**

**(A)** Performance of image registration in five different animals in the testing set. Normalized Cross-Correlation (NCC) scores of aligned tagRFP images are shown, which indicate the extent of image alignment (best achievable score is 1). 90-100 registration problems examined per animal are shown as violin plots with the overlaying lines indicating minimum, mean, and maximum values.

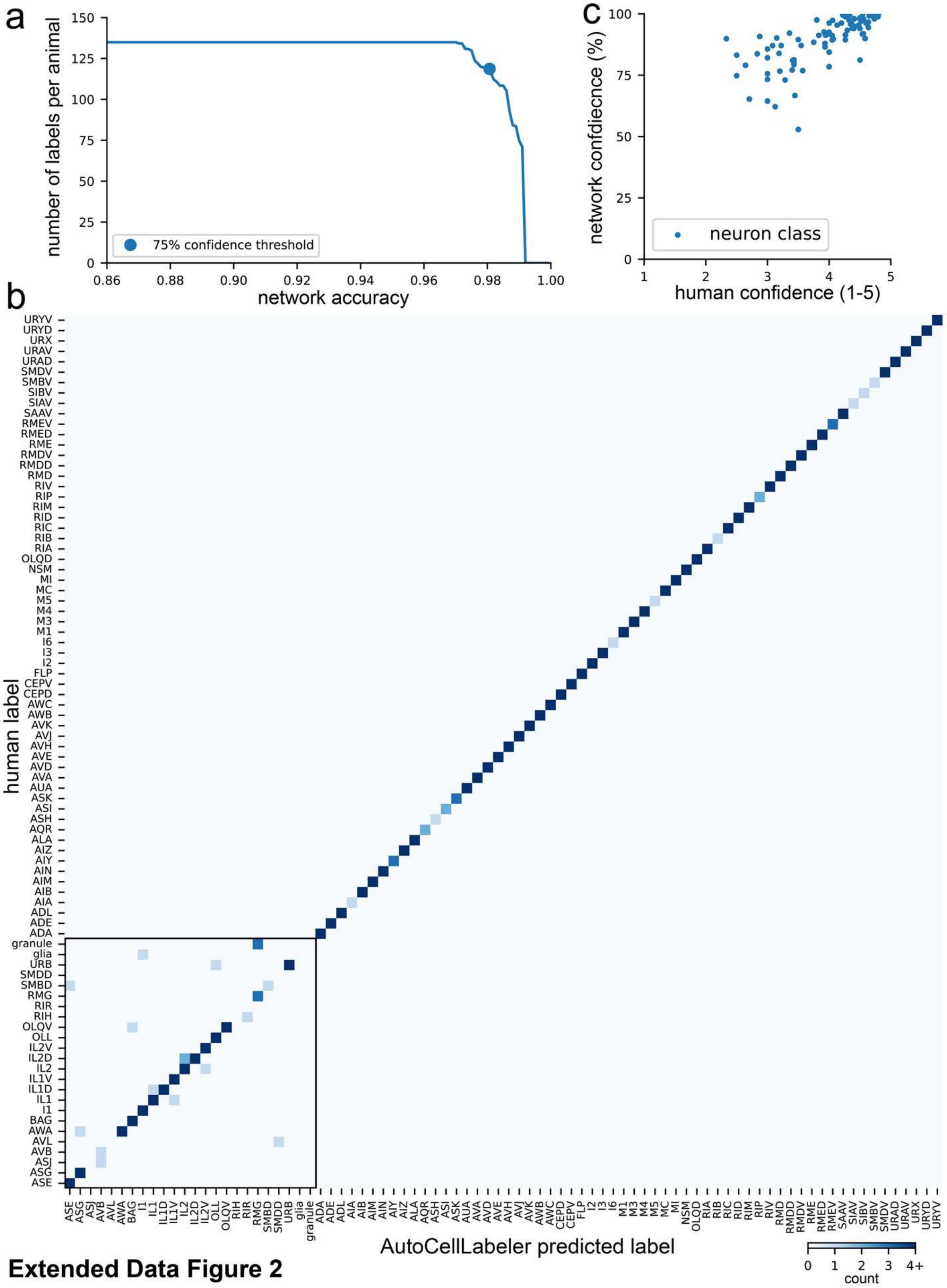
**(B)** Performance of image registration in five different animals in the testing set. Centroid distance is the average Euclidean distance between the centroids of matched neurons in each image (best achievable score is 0). 90-100 registration problems examined per animal are shown as violin plots with the overlaying lines indicating minimum, mean, and maximum values.

**(C)** Performance of image registration in five different registration problems (i.e. image pairs) from one example animal. Centroid distance is the average Euclidean distance between the centroids of matched neurons in that image pair (best achievable score is 0). All the centroid position distances for each registration problem as shown as violin plots with the overlaying lines indicating minimum, mean, and maximum values.

**(D)** Five example image pairs in the training set for BrainAlignNet. These are maximum intensity projections of the tagRFP channel, showing two different timepoints that were selected to be the fixed and moving images in each of these five registration problems.

**(E)** Five example image pairs in the training set for the network trained to align arbitrary image pairs, including much more challenging problems. Note that the head bending is more dissimilar for these image pairs, as compared to those in (D). Data are shown as in (D).

**(F)** Performance of the network trained to register arbitrary image pairs. Quantification is for testing data. We quantify centroid distance (average alignment of neuron centroids) and NCC (image similarity) as in panels (A-C). By both metrics, this network's performance is far worse than that of the BrainAlignNet presented in Fig. 1. The two panels on the right show that results are qualitatively similar for different animals in the testing set.



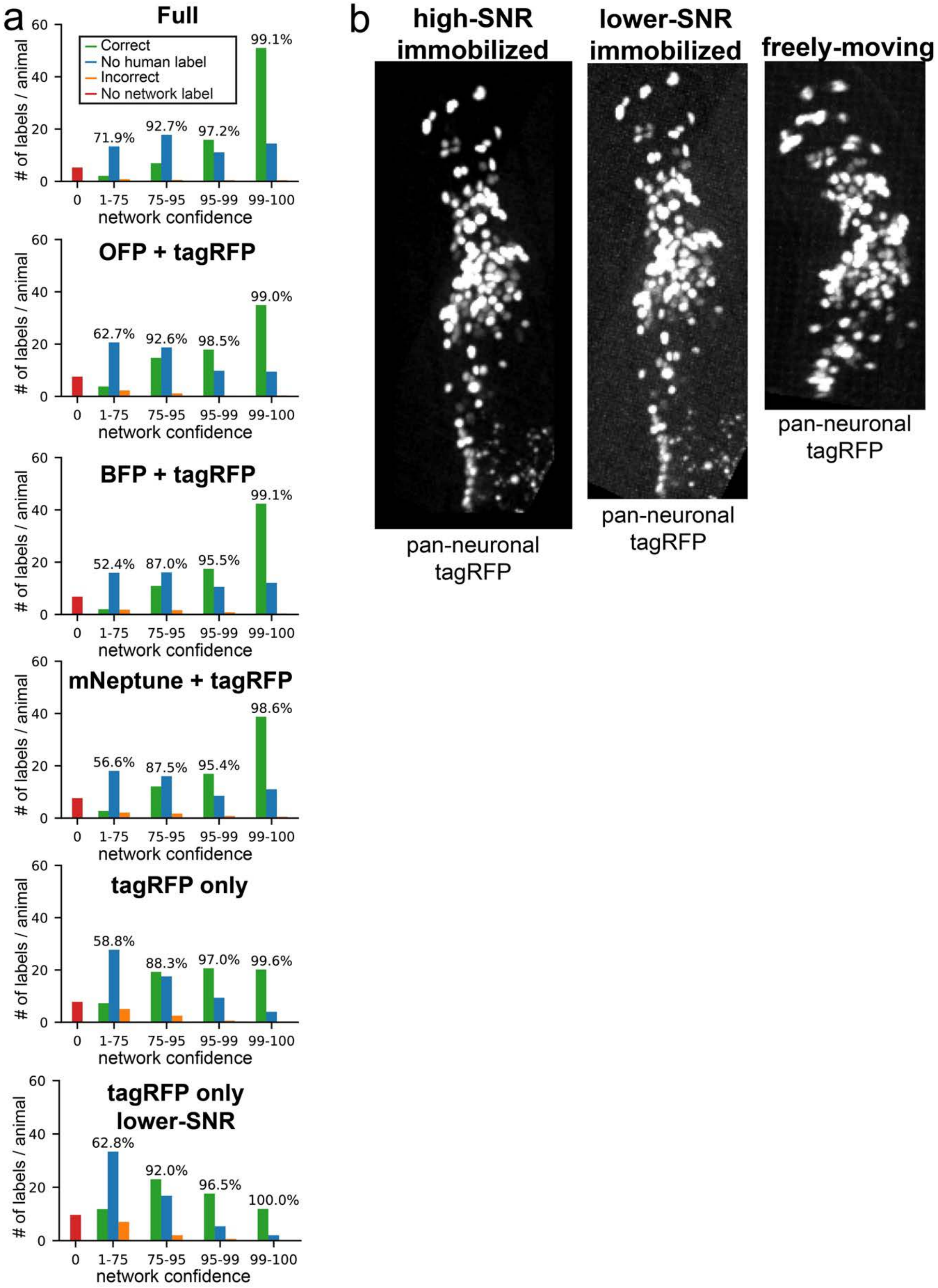
**Extended Data Figure 2**

**Extended Data Figure 2. Further characterization of the AutoCellLabeler network**

**(A)** Tradeoff of network labeling accuracy (*x*-axis) and number of neurons labeled (*y*-axis) for the full AutoCellLabeler network. The number of neurons labeled can be varied by adjusting the threshold confidence that the network needs to achieve to label an ROI. By varying this threshold, we were able to generate this curve. This full curve captures the tradeoff and shows the 75% confidence threshold (blue circle) that we selected to use in our analyses.

**(B)** Confusion matrix showing which neurons could potentially be confused for one another by AutoCellLabeler. Note that, except for the diagonal, the matrix is mostly white, reflecting that it is mostly (98%) accurate. Neurons with some inaccuracies were clustered to the lower left (boxed region). Note that with a linear color scale the diagonal would be off-scale bright with correct labels. So we capped the colorbar range at 4 counts so as to not block the ability to see actual confusion entries. For reference, the actual mean value across the diagonal is 9.7.

**(C)** Positive correlation between human and autolabel confidence across the neuronal cell types (each cell type is a blue dot). This plot also highlights that a subset of cells are more difficult for human labelers and, therefore, also for AutoCellLabeler (i.e. the cells that are not clustered in the upper right).

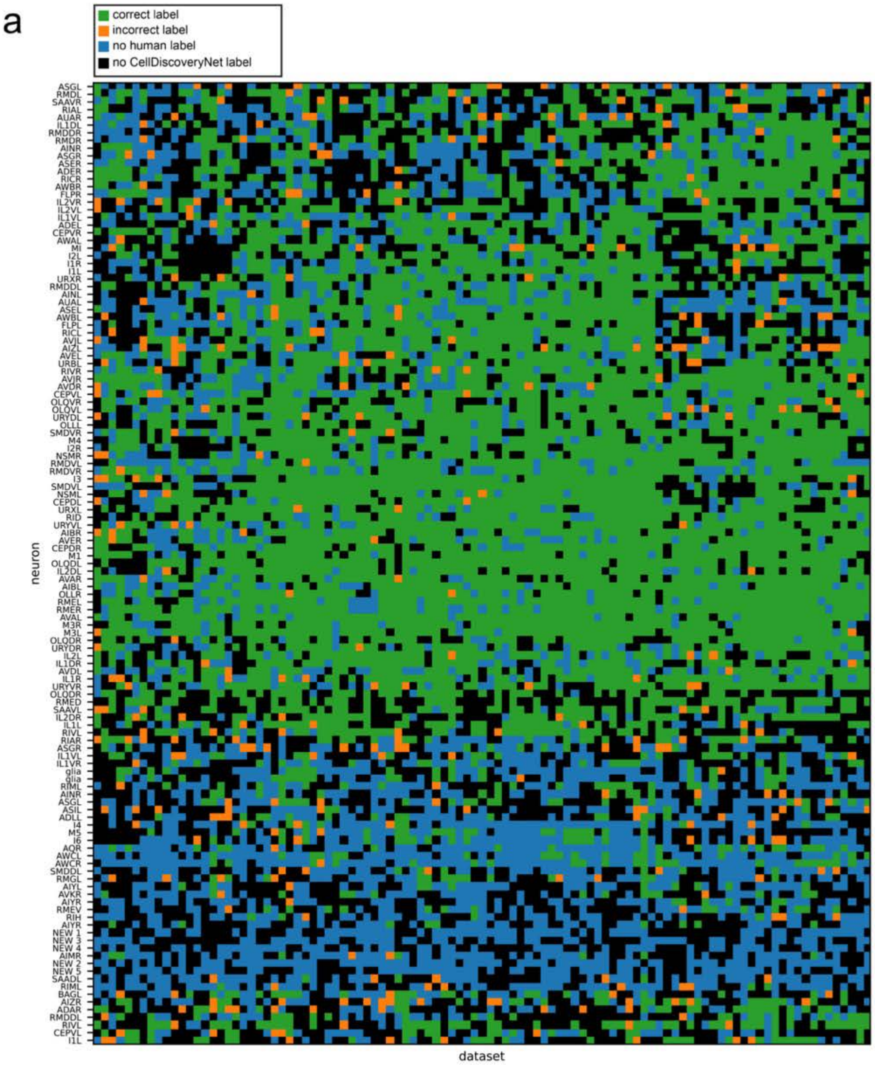


**Extended Data Figure 3**

**Extended Data Figure 3. Further characterization of the different AutoCellLabeler variants.**

**(A)** These plots show the performance of different indicated cell annotation networks (trained and/or evaluated on different fluorophores, as indicated). Data is displayed to show network performance on different ROIs that it labels with different levels of confidence. Printed percentage values are the accuracy of AutoCellLabeler on the corresponding category, computed as $\frac{correct}{correct+incorrect}$. Note that the lower performing networks (for example, tagRFP-only) are still accurate for their high-confidence labels, and that their decreased accuracy is mostly due to a lower fraction of high-confidence labels (i.e. more cell types where the networks had low confidence in their annotations).

**(B)** Example maximum intensity projection images of the worm in the tagRFP channel under three different imaging conditions: immobilized high-SNR (created by averaging together 60 immobilized lower-SNR images together, our typical condition for NeuroPAL imaging); immobilized lower-SNR (i.e. one of those 60 images); and freely-moving (which was taken with the same imaging settings as immobilized lower-SNR but in a freely-moving animal)



## Extended Data Figure 4

**Extended Data Figure 4. Further characterization of CellDiscoveryNet and ANTSUN 2U performance**

**(A)** Matrix of all clusters generated by running ANTSUN 2U. Each row is a distinct cluster (i.e. inferred cell type), while each column is a distinct animal. Black entries mean that the given cluster did not include any ROIs in the given animal (ie: ANTSUN 2U failed to label that cluster in that animal). Non-black entries mean that the cluster contained an ROI in that animal. Row names correspond to the most frequent human label among ROIs in the cluster (this was defined by first disambiguating most frequent neuron class, and then disambiguating L from R). Green entries correspond to cases when the given ROI's label matched the most frequent class label (row name ignoring L/R), orange entries correspond to the case when the given ROI's label did not match the most frequent class label, and blue entries mean that the given ROI did not have a high-confidence human label. The neurons "NEW 1" through "NEW 5" are clusters that were not labeled frequently enough by humans to be able to determine which neuron class they corresponded to, as described in the main text.