Structural Biology and Molecular Biophysics

Ais: streamlining segmentation of cryo-electron tomography datasets

Mart GF Last ✉, **Leoni Abendstein, Lenard M Voortman, Thomas H Sharp** ✉

Department of Cell and Chemical Biology, Leiden University Medical Center, Leiden, The Netherlands • Institute of Science and Technology Austria (ISTA), Klosterneuburg, Austria • School of Biochemistry, University of Bristol, Bristol, United Kingdom

Reviewed Preprint
v2 • November 8, 2024
Revised by authors

Reviewed Preprint
v1 • June 21, 2024

eLife Assessment

This work describes a new software platform for machine-learning-based segmentation of and particle-picking in cryo-electron tomograms. The program and its corresponding online database of trained models will allow experimentalists to conveniently test different models and share their results with others. The paper provides **convincing** evidence that the software will be **valuable** to the community.

https://doi.org/10.7554/eLife.98552.2.sa4

Abstract

Segmentation is a critical data processing step in many applications of cryo-electron tomography. Downstream analyses, such as subtomogram averaging, are often based on segmentation results, and are thus critically dependent on the availability of open-source software for accurate as well as high-throughput tomogram segmentation. There is a need for more user-friendly, flexible and comprehensive segmentation software that offers an insightful overview of all steps involved in preparing automated segmentations. Here, we present Ais: a dedicated tomogram segmentation package that is geared towards both high performance and accessibility, available at github.com/bionanopatterning/Ais. In this report, we demonstrate two common processing steps that can be greatly accelerated with Ais: particle picking for subtomogram averaging, and generating many-feature segmentations of cellular architecture based on *in situ* tomography data. Featuring comprehensive annotation, segmentation, and rendering functionality, as well as an open repository for trained models at aiscryoet.org, we hope that Ais will help accelerate research and dissemination of data involving cryoET.

Introduction

Segmentation is a key step in processing cryo-electron tomography (cryoET) datasets that entails identifying specific structures of interest within the volumetric data and marking them as distinct features. This process forms the basis of many subsequent analysis steps, including particle

picking for subtomogram averaging and the generation of 3D macromolecular feature maps that help visualize the ultrastructure of the sample.

Although it is a common task, the currently available software packages for tomogram segmentation often leave room for improvement in either scope, accessibility, or open availability of the source code. Some popular programs, such as Amira™ (Thermo Fisher Scientific), are not free to use, while other more general purpose EM data processing suites offer limited functionality in terms of visualization and user interaction. Specifically, we found that a deficit existed of software that is easy to use, competitively performing, freely available, and dedicated to segmentation of cryoET datasets for downstream processing.

In this report we present Ais, an open-source tool that is designed to enable any cryoET user – whether experienced with software and segmentation or a novice – to quickly and accurately segment their cryoET data in a streamlined and largely automated fashion. Ais comprises a comprehensive and accessible user interface within which all steps of segmentation can be performed, including: the annotation of tomograms and compiling datasets for the training of convolutional neural networks (CNNs), training and monitoring performance of CNNs for automated segmentation, 3D visualization of segmentations, and exporting particle coordinates or meshes for use in downstream processes. To help generate accurate segmentations, the software contains a library of various neural network architectures and implements a system of configurable interactions between different models. Overall, the software thus aims to enable a streamlined workflow where users can interactively test, improve, and employ CNNs for automated segmentation. To ensure compatibility with other popular cryoET data processing suites, Ais employs file formats that are common in the field, using .mrc files for volumes, tab-separated .txt or .star files for particle datasets, and the .obj file format for exporting 3D meshes.

To demonstrate the use of Ais, we outline its use in two such tasks: first, to automate the particle picking step of a subtomogram averaging workflow, and secondly for the generation of rich three-dimensional visualizations of cellular architecture, with ten distinct cellular components, based on cryoET datasets acquired on cellular samples.

Results and discussion

The first step in image segmentation using CNNs is to manually annotate a subset of the data for use as a training dataset. Ais facilitates this step by providing a simple interface for browsing data, drawing overlays, and selecting boxes to use as training data (**Fig. 1A**, Fig. S1-S5). Multiple features, such as membranes, microtubules, ribosomes, and mitochondrial granules, can be segmented and edited at the same time across multiple datasets (even hundreds). These annotations are then extracted and used as ground truth labels upon which to condition multiple neural networks, each trained to segment a single feature type, with which one can automatically segment the same or any other dataset (**Fig. 1B**). Segmentation in Ais is performed *on-the-fly* and can achieve interactive framerates, depending on the size of the datasets and network architectures that are used. With a little experience, users can generate a training dataset and then train, apply, and asses the quality of a model within a few minutes (**Fig. 1C**), including on desktop or laptop Windows and Linux systems with relatively low-end GPUs (e.g., we often use an NVIDIA T1000).

Many cryoET datasets look alike, especially for cellular samples. A model prepared by one user to segment, for example, ribosomes in a dataset with a pixel size of 10 Å, might also be adequate for another user's ribosome segmentation at 12 Å per pixel. To facilitate this sort of reuse and sharing of models, we launched an open model repository at *aiscryoet.org* where users can freely upload and download successfully trained models (**Fig. 1D**). Models that pass screening become public, are labelled with relevant metadata (**Fig. 1E**), and can be downloaded in a format that allows

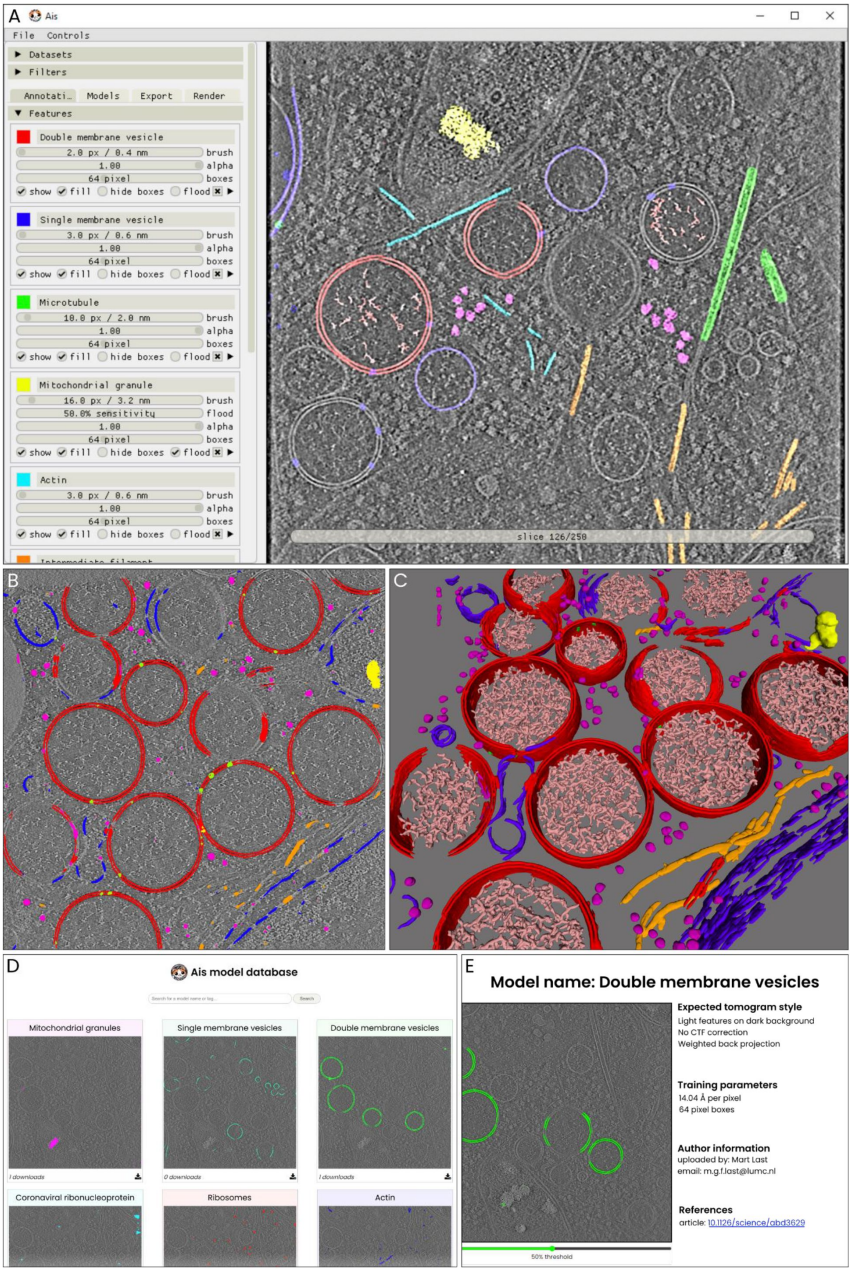


Figure 1

An overview of the user interface and functionalities.

The various panels represent sequential stages in the Ais processing workflow, including annotation (A), testing CNNs (B), visualizing segmentation (C). These images (A-C) are unedited screenshots of the software. **A)** The interface for annotation of datasets. In this example a tomographic slice has been annotated with various features – a detailed explanation follows in **Fig. 5**. **B)** After annotation, multiple neural networks are set up and trained on the aforementioned annotations. The resulting models can then be used to segment the various distinct features. In this example, double membrane vesicles (DMVs, red), single membranes (blue), ribosomes (magenta), intermediate filaments (orange), mitochondrial granules (yellow), and molecular pores in the DMVs (lime) are segmented. **C)** After training or downloading the required models and exporting segmented volumes, the resulting segmentations are immediately available within the software for 3d rendering and inspection. **D)** The repository at *aiscryoet.org* facilitates sharing and reuse of trained models. After validation, submitted models can be freely downloaded by anyone. **E)** Additional information, such as the pixel size and the filtering applied to the training data, is displayed alongside all entries in the repository, in order to help a user identify whether a model is suited to segment their datasets.

for direct use in Ais. Thus, users can skip the annotation and training steps of the segmentation workflow. To kickstart the repository, all 27 models that are presented in this article have been uploaded to it.

Our software is not the first to address the challenge of segmenting cryoET datasets; established suites such as EMAN2[1], MIB[2], SuRVOS[3], or QuPath[4] also provide some or most of the functionality that is available in Ais. Each comes equipped with one or various choices of neural network architectures to use, and many more designs for neural networks for semantic image segmentation can be found in the literature. Therefore, as well as creating a package geared specifically towards ease of use and fast results, we also wanted to include functionality that enables a user to quickly compare different models in order to facilitate determining which models are best suited for a particular segmentation task. The software thus includes a library of a number of well-performing architectures, including adaptations of single-model CNN architectures such as InceptionNet[5], ResNet[6], various UNets[7], VGGNet[8], and the default architecture available in EMAN2[1], as well as the more complex generative-adversarial network Pix2pix[9]. This library can also be extended by copying any Python file that adheres to a minimal template into the corresponding directory of the project (Supplementary Note 1).

A library of neural network architectures supports varied applications

To illustrate how useful it can be to rapidly test various architectures before selecting one that is well suited for the segmentation of any particular feature, we used six different architectures for the segmentation of three distinct features within the same tomogram, and analyzed the results (**Table 1**). We used a cryoET dataset that we have previously acquired[10], which contained liposomes with membrane bound Immunoglobulin G3 (IgG3) antibodies that form an elevated Fragment crystallizable (Fc) platform, prepared on a lacey carbon substrate. The features of interest for segmentation were the membranes, antibody platforms, and carbon support film (**Fig. 2A**).

After training, we applied the resulting models to a different tomogram containing the same features, that was not previously used to generate the training data (**Fig. 2B**), so that there was no overlap between the training and testing datasets. Next, we compared the training times, relative loss values, and quality of the segmentations.

Based on the loss values (the loss is a metric of how well predictions match the ground truth labels), VGGNet was best suited for the segmentation of membranes, while UNet performed best on the carbon support film and antibody platforms. However, the loss values do not capture model performance in the same way as human judgement (**Fig. 2C**). For the antibody platform, the model that would be expected to be one of the worst based on the loss values, Pix2pix, actually generates segmentations that are well-suited for the downstream processing tasks. It also output fewer false positive segmentations for sections of membranes than many other models, including the lowest-loss model UNet. Moreover, since Pix2pix is a relatively large network, it might also be improved further by increasing the number of training epochs. We thus decided to use Pix2pix for the segmentation of antibody platforms, and increased the size of the antibody platform training dataset (from 58 to 170 positive samples) to train a much improved second iteration of the network for use in the following analyses (Fig. S6).

When taking the training and processing speeds in to account as well as the segmentation results, there is no overall best architecture. We therefore included multiple well-performing model architectures in the final library, in order to allow users to select from these models to find one that works well for their specific datasets. Although it is not necessary to screen different network architectures and users may simply opt to use the default (VGGNet), these results thus show that it can be useful to test different networks to identify one that is best. Moreover, these results also

Architecture	*Parameters*	*Training time*[a]	*Processing time*[a, b]	*Membrane*[c]	*Carbon*[c]	*Antibody*[c]
EMAN2	378,081	38 s	50 ms	.044	.072	.010
InceptionNet	550,529	462 s	295 ms	.046	.120	.008
UNet	922,881	132 s	110 ms	.013	**.007**	**.001**
VGGNet	1,493,059	91 s	85 ms	**.011**	.125	.013
ResNet	4,887,265	1533 s	980 ms	.047	.060	.014
Pix2pix[d]	29,249,409	793 s	225 ms	.050[d]	.129[d]	.016[d]

Table 1

Comparison of some of the default models available in Ais.

[a] The computational cost is only roughly proportional to the number of model parameters, which is reported in the software. The specifics of the network architecture affect the processing speed more significantly. [b] Time required to process one 511 ×720 pixel sized tomographic slice. [c] These columns list the loss values after training, calculated as the binary cross-entropy (bce) between the prediction and original annotation. The loss is a (rough) metric of how well a trained network performs (see Methods). [d] Unlike the other architectures, Pix2pix is not trained to minimize the bce loss but uses a different loss function instead. The bce loss values shown here were computed after training and may not be entirely comparable.

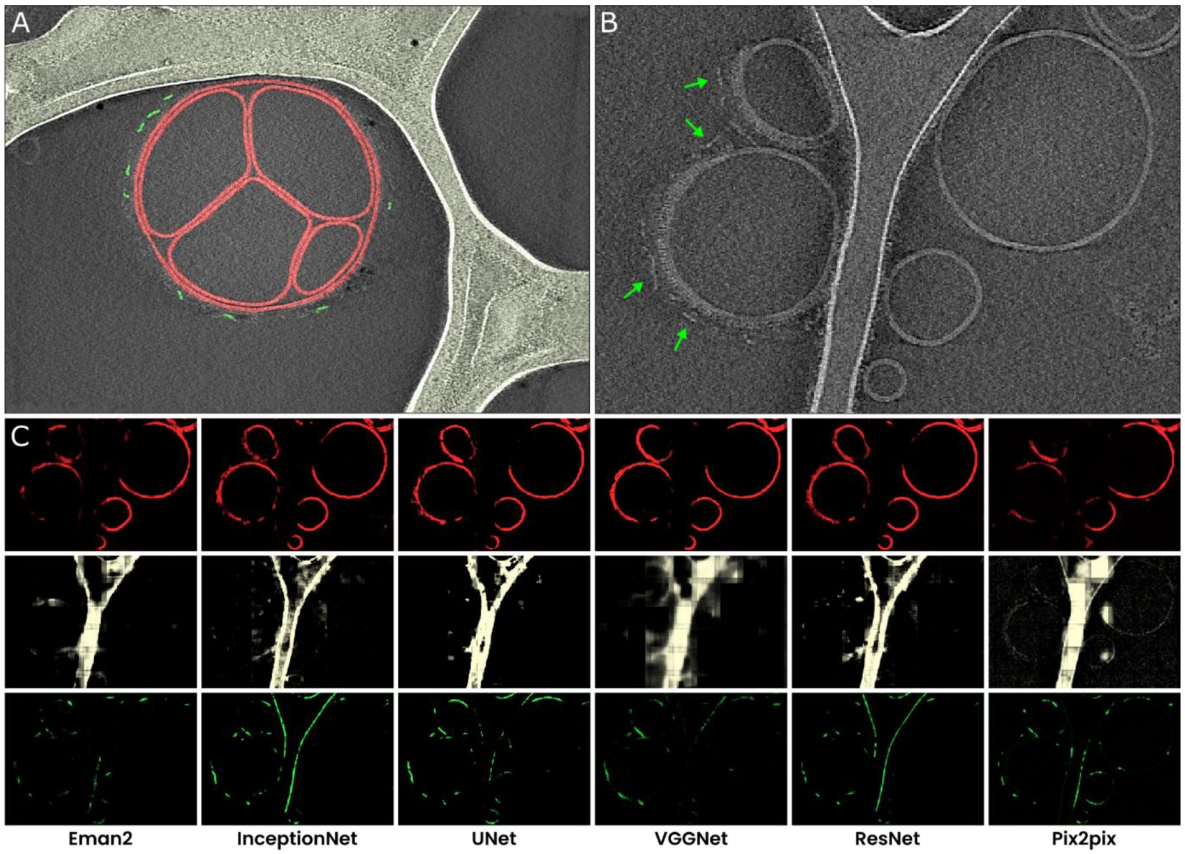


Figure 2

A comparison of different neural networks for tomogram segmentation .

A) A representative example of the manual segmentation used to prepare training datasets. Membranes are annotated in red, carbon film in bright white, and antibody platforms in green. For the antibody training set, we used annotations prepared in multiple slices of the same tomogram, but for the carbon and membrane training set the slice shown here comprised all the training data. **B)** A tomographic slice from a different tomogram that contains the same features of interest, also showing membrane bound antibodies with elevated Fc platforms that are adjacent to carbon (red arrowheads). **C)** Results of segmentation of membranes (top ; red), carbon (middle; white), and antibody platforms (bottom; green), with the six different neural networks.

highlight the utility of preparing well-performing models by iteratively improving training datasets and re-training models in a streamlined interface. To aid in this process, the software displays the loss value of a network during training and allows for the application of models to datasets during training. Thus, users can inspect how a model's output changes during training and decide whether to interrupt training and improve the training data or choose a different architecture.

Fine-tuning segmentation results with *model interactions*

Although the above results go some way towards distinguishing the three different structures, they also show demonstrate a common limitation encountered in automated tomogram segmentation: some parts of the image are assigned a high segmentation value by multiple of the networks, leading to false classifications and ambiguity in the results. For example, the InceptionNet and ResNet antibody platform models falsely label edges of the carbon film.

To further improve the segmentation results, we decided to implement a system of proximity-based 'model interactions' of two types, colocalization and avoidance (**Fig. 3A**), using which the output of one model can be adjusted based on the output of other models. In a colocalization interaction, the predictions of one model (the *child*) are suppressed wherever the prediction value of another model (the *parent)* is below some threshold. In an avoidance interaction, suppression occurs wherever the parent model's prediction value is above a threshold.

These interactions are implemented as follows: first, a binary mask is generated by thresholding the *parent* model's predictions using a user-specified threshold value. Next, the mask is then dilated using a circular kernel with a radius R, a parameter that we call the interaction radius. Finally, the *child* model's prediction values are multiplied with this mask.

Besides these specific interactions between two models, the software also enables pitching multiple models against one another in what we call 'model competition'. Models can be set to 'emit' and/or 'absorb' competition from other models. Here, to emit competition means that a model's prediction value is included in a list of competing models. To absorb competition means that a model's prediction value will be compared to all values in that list, and that this model's prediction value for any pixel will be set to zero if any of the competing models' prediction value is higher. On a pixel-by-pixel basis, all models that absorb competition are thus suppressed whenever their prediction value for a pixel is lower than that of any of the emitting models.

With the help of these model interactions it is possible to suppress common erroneous segmentation results. For example, an interaction like 'absorbing **membrane model** avoids emitting **carbon model** with R = 10 nm' is effective at suppressing the prediction of edges of the carbon film as being membranes (**Fig. 3B**). Another straightforward example of the utility of membrane interactions is the segmentation of membrane-bound particles. By defining the following two interactions: '**antibody platform model** avoids **membrane model** with R = 10 nm' followed by '**antibody platform model** colocalizes with **membrane model** with R = 30 nm', the Fc-platforms formed by IgG3 at a distance of ~22 nm from the membrane are retained, while false positive labelling of features such as the membrane or carbon is suppressed (**Fig. 3C**).

By conditionally combining and editing the prediction results of multiple neural networks, model interactions can thus be helpful in fine-tuning segmentations to be better suited for downstream applications. To illustrate this, we generated a comparison of segmentation results using i) no interactions, ii) model competition only, and iii) model competition as well as model interactions (**Fig. 3D**), which demonstrates the degree to which false positives can be reduced by the use of model interactions (although at times at the expense of increasing the rate of false negatives).

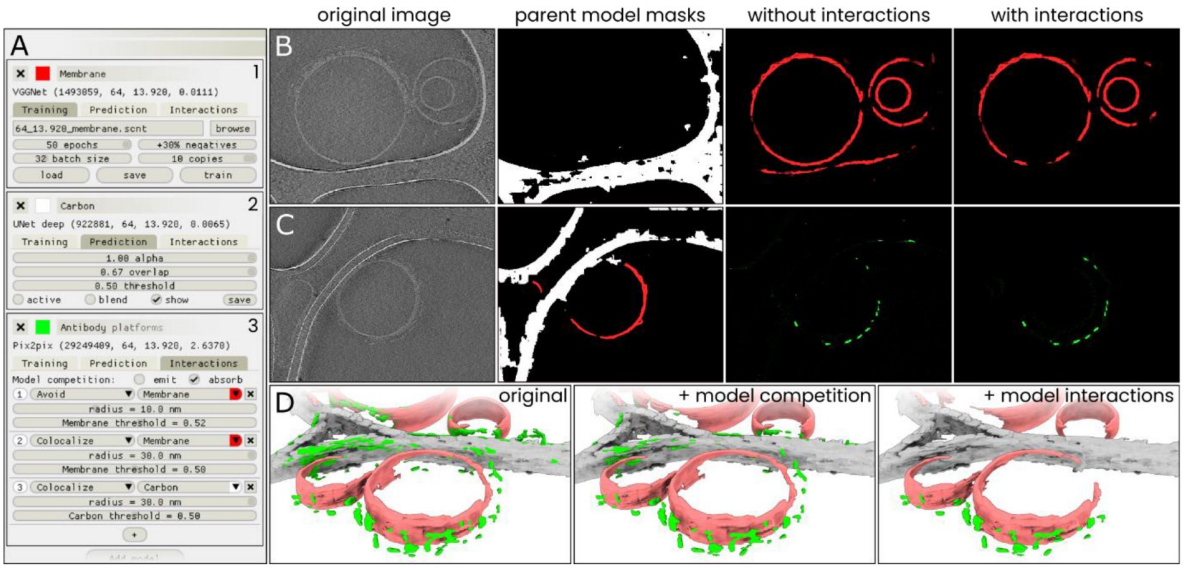


Figure 3

Model interactions can significantly increase segmentation accuracy.

A) An overview of the settings available in the 'Models' menu in Ais. Three models: 1) ' membrane' (red), 2) ' carbon' (white), and 3) ' antibody platforms' (green) are active, with each showing a different section of the model settings: the training menu (1), prediction parameters (2), and the interactions menu (3). **B)** A section of a tomographic slice is segmented by two models, carbon (white; *parent* model) and membrane (red; *child* model), with the membrane model showing a clear false positive prediction on an edge of the carbon film (panel ' without interactions'). By configuring an avoidance interaction between the membrane model that is conditional upon the carbon model' s prediction, this false positive is avoided (panel ' with interactions'). **C)** By setting up multiple model interactions, inaccurate predictions by the ' antibody platforms' model are suppressed. In this example, the membrane model avoids carbon while the antibody model is set to colocalize with the membrane model. **D)** 3D renders (see Methods) of the same dataset as used in **Fig. 2** processed three ways: without any interactions (left), using model competition only (middle), or by using model competition as well as multiple model interactions (right).

Automating particle picking for subtomogram averaging

Protein structure determination by cryoET requires the careful selection of many subtomograms (i.e., sub-volumes of a tomogram that all contain the same structure of interest), and aligning and averaging these to generate a 3D reprojection of the structure of interest with a significantly increased signal to noise ratio. The process of selecting these sub-volumes is called 'particle picking', and can be done either manually or in an automated fashion. Much time can be saved by automating particle picking based on segmentations. In many cases, though, segmentation results are not readily usable for particle picking, as they can often introduce numerous false positives. This is particularly the case with complex, feature-rich datasets such as those obtained within cells, where the structures of interest can visually appear highly similar to other structures that are also found in the data, or when the structures of interest are located close to other features and are therefore hard to isolate. An example of this latter case is the challenge of picking membrane-bound particles.

Recently, we have used cryoET and subtomogram averaging to determine the structures of membrane bound IgG3 platforms and of IgG3 interacting with the human complement system component 1 (C1) on the surface of lipid vesicles[10]. The reconstructions of the antibody platforms alone and of the antibody-C1 complex were prepared using 1193 and 2561 manually selected subtomograms, extracted from 55 and 101 tomograms, respectively. Manual picking of structures of interest, although very precise, is very time consuming and in this particular case took approximately 20 hours of work.

To demonstrate the utility of our software for particle picking, we re-analyzed these same datasets, this time using Ais to automate the picking of the two structures: antibody platforms and antibody-C1 complexes. For the antibody platforms, we used the same models and model interactions as described above, while we trained an additional neural network to identify C1 complexes for the segmentation of the antibody-C1 complexes. To prepare a training dataset for this latter model, we opened all 101 tomograms in Ais and browsed the data to select and annotate slices where one or multiple antibody-C1 complexes were clearly visible (**Fig. 4A**). The training dataset thus consisted of samples taken from multiple different tomograms; the annotation and data selection in this case took around 1 hour of work.

The antibody platform and antibody-C1 complex models were then applied to the respective datasets, in combination with the membrane and carbon models and the model interactions described above (**Fig. 4B**): the membrane avoiding carbon, and the antibody platforms colocalizing with the resulting membranes. We then launched a relatively large batch segmentation process (3 models × 156 tomograms) and left it to complete overnight.

Once complete, we used Ais to inspect the segmented volumes and original datasets in 3D, and adjusted the threshold value so that, in as far as possible, only the particles of interest remained visible and the number of false positive particles was minimized (**Fig. 4C**). After selecting these values, we then launched a batch particle picking process to determine lists of particle coordinates based on the segmented volumes.

Particle picking in Ais comprises a number of processing steps (Fig. S7). First, the segmented (.mrc) volumes are thresholded at a user-specified level. Second, a distance transform of the resulting binary volume is computed, in which every nonzero pixel in the binary volume is assigned a new value, equal to the distance of that pixel to the nearest zero-valued pixel in the mask. Third, a watershed transform is applied to the resulting volume, so that the sets of pixels closest to any local maximum in the distance transformed volume are assigned to one group. Fourth, groups that are smaller than a user-specified minimum volume are discarded. Fifth, the remaining groups are assigned a weight value, equal to the sum of the prediction value (i.e. the corresponding pixel value in the input .mrc volume) of the pixels in the group. For every group found within close

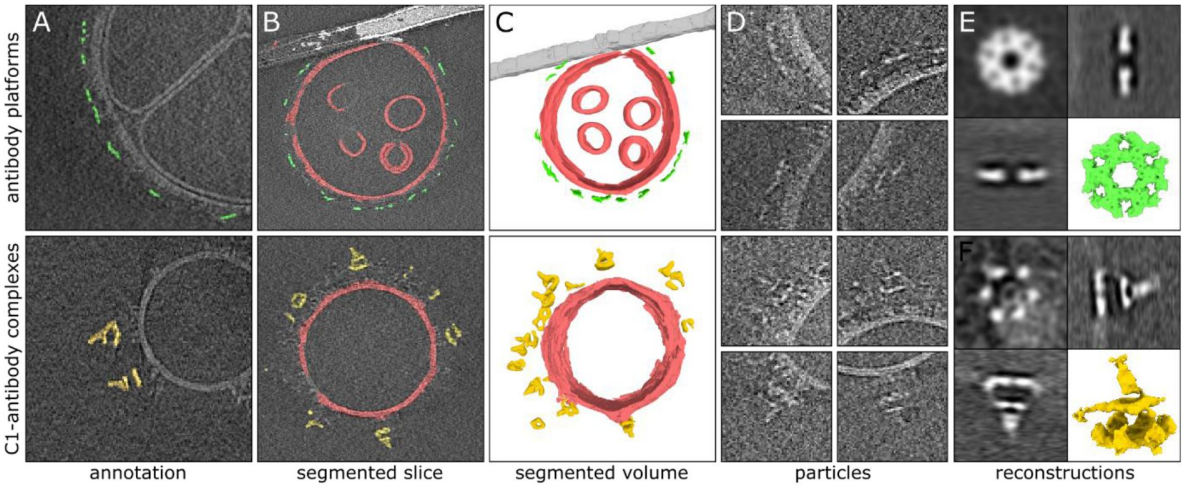


Figure 4

Automated particle picking for sub-tomogram averaging of antibody complexes.

A) Manually prepared annotations used to train a neural network to recognize antibody platforms (top) or antibody-C1 complexes (bottom). **B)** Segmentation results as visualized within the software. Membranes (red) and carbon support film (white) were used to condition the antibody (green) and antibody-C1 complex (yellow) predictions using model interactions. **C)** 3 D representations of the segmented volumes rendered in Ais. **D)** Tomographic slices showing particles picked automatically based on the segmented volume shown in panel c. **E)** Subtomogram averaging result of the 2499 automatically picked antibody platforms. **F)** Subtomogram averaging result obtained with the 602 automatically picked antibody-C1 complexes. The quadrants in panels e and f show orthogonal slices of the reconstructed density maps and a 3D isosurface model (the latter rendered in ChimeraX).

proximity to another group (using a user-specified value for the minimum particle spacing), the group with the lower weight value is discarded. Finally, the centroid coordinate of the grouped pixels is considered the final particle coordinate, and the list of all coordinates is saved in a tab-separated text file with values rounded to the nearest integer.

As an alternative output format, segmentations can also be converted to and saved as triangulated meshes (.obj file format), which can then be used for, e.g., membrane-guided particle picking[11]. After picking particles, the resulting coordinates are immediately available for inspection in the Ais 3D renderer (Fig. S8).

After picking, we used EMAN2[1] (spt_boxer.py) to extract volumes using the particle coordinates as an input (**Fig. 4D**), which resulted in 2499 volumes for the antibody platform reconstruction and 602 for the antibody-C1 complex (*n.b.* these numbers can be highly dependent on the threshold value). These volumes, or subtomograms, were used as the input for subtomogram averaging using EMAN2[1] and Dynamo[12] (see Methods), without further curation – i.e., we did not manually discard any of the extracted volumes.

After applying the same approach to subtomogram averaging as used previously[10], the resulting averages were indeed highly similar to the original reconstructions (**Figs. 4EF**, Fig. S9). These results demonstrate that Ais can be successfully used to automate particle picking, and thus to significantly reduce the amount of time spent on what is often a laborious processing step.

Many-feature segmentations of complex *in situ* datasets

Aside from particle picking, segmentation is also often used to visualize and study the complex internal structure of a sample, as for example encountered when applying tomography to whole cells. Here too, the accuracy of a segmentation can be a critical factor in the success of downstream analyses such as performing measurements on the basis of 3D feature maps generated *via* segmentation.

A challenging aspect of segmentation of cellular samples is that these datasets typically contain many features that are biologically distinct, but visually and computationally difficult to distinguish. For example, one challenge that is often encountered is that of distinguishing between various linearly shaped components: lipid membranes, actin filaments, microtubules, and intermediate filaments, which all appear as linear features with a relatively high density. To show the utility of Ais for the accurate segmentation of complex cellular tomograms, we next demonstrate a number of examples of such feature-rich segmentations.

The first example is a segmentation of seven distinct features observed in the base of *Chlamydomonas reinhardtii* cilia (**Fig. 5A**), using the data by van den Hoek et al.[13] that was deposited in the Electron Microscopy Public Image Archive (EMPIAR)[14] with accession number 11078. The features are: membranes, ribosomes, microtubule doublets, axial microtubules, non-microtubular filaments, interflagellar transport trains (IFTs), and glycocalyx. This dataset was particularly intricate (the supplementary information to the original publication lists more than 20 features that can be identified across the dataset) and some rare features, such as the IFTs, required careful annotation across all tomograms before we could compile a sufficiently large training dataset. The final segmentation correctly annotates most of the selected characteristics present in the sample: the ribosome exclusion zone that surrounds the ciliary base[13] is clearly recognizable, and the structures of the glycocalyx, membranes, and microtubule doublets within the cilia are well defined. Some fractions of the meshwork of stellate fiber and Y-link proteins are also detected within the cilium.

In the second example, we show a segmentation of six features found in and around a mitochondrion in a mouse neuron (**Fig. 5B**), using the data by Wu et al.[15] as available in the Electron Microscopy Data Bank (EMDB)[16] with accession number 29207. In the original

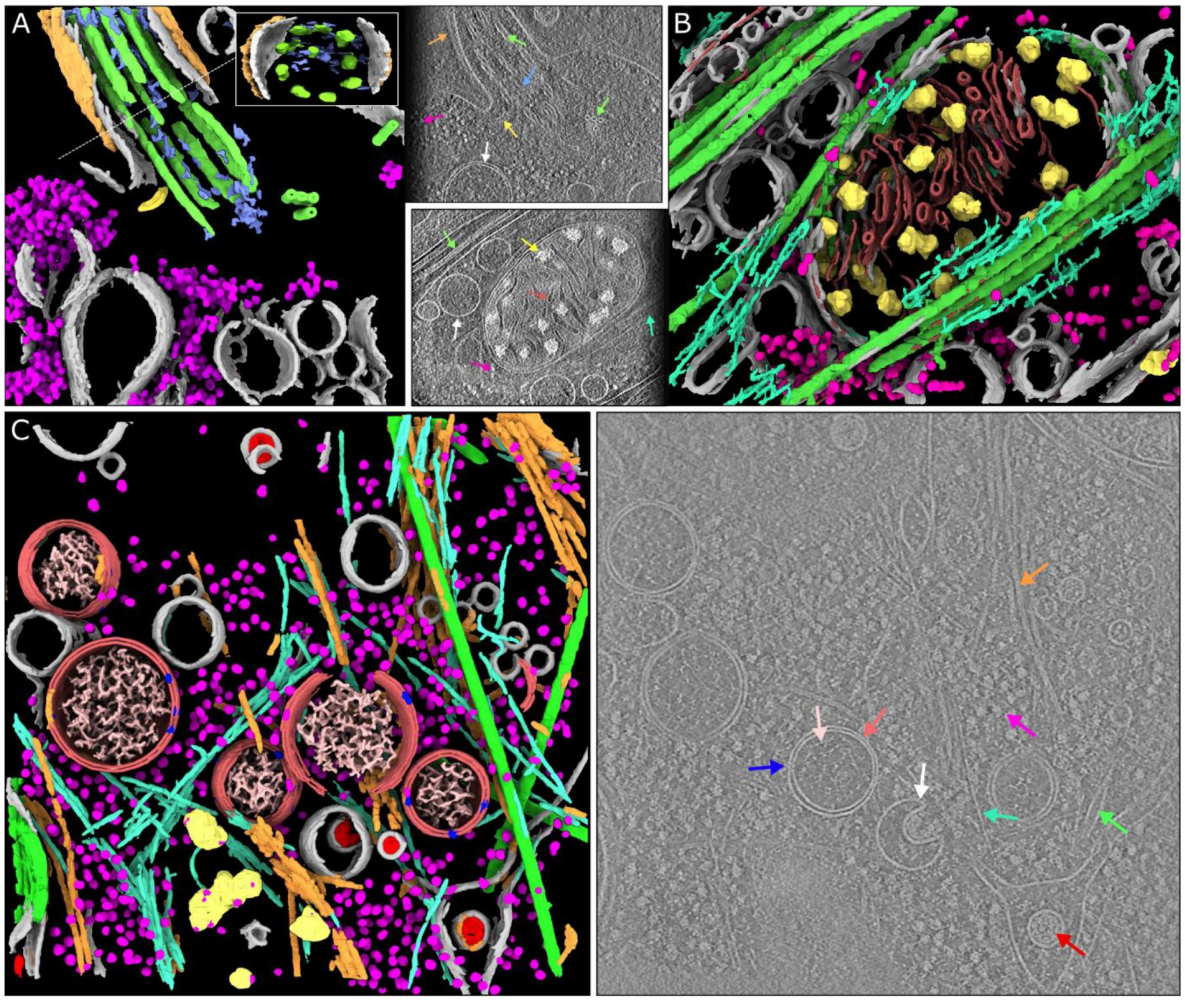


Figure 5

Segmentations of complex *in situ* tomograms.

A) A segmentation of seven distinct features observed in the base of *C. reinhardtii* cilia[13] (EMPIAR-11078 , tomogram 12): membranes (gray), ribosomes (magenta), microtubule doublets (green) and axial microtubules (green), non-microtubular fi laments within the cilium (blue), interflagellar transport trains (yellow), and glycocalyx (orange). Inset: a perpendicular view of the axis of the cilium. The arrows in the adjacent panel indicate these structures in a tomographic slice. **B)** A segmentation of six features observed in and around mitochondria in a mouse neuron with Huntington disease phenotype [15] (EMD-29207): membranes (gray), mitochondrial granules (yellow), membranes of the mitochondrial cristae (red), microtubules (green), actin (turquoise), and ribosomes (magenta). **C)** Left: a segmentation of ten different cellular components found in tomograms of coronavirus infected mammalian cells [17] : double membrane vesicles (DMVs, light red), single membranes (gray), viral nucleocapsid proteins (red), viral pores in the DMVs (blue), nucleic acids in the DMVs (pink), microtubules (green), actin (cyan), intermediate fi laments (orange), ribosomes (magenta), and mitochondrial granules (yellow). Right: a representative slice, with examples of each of the features (except the mitochondrial granules) indicated by arrows of the corresponding colour.

publication, the authors developed a segmentation method to detect and perform measurements on the granules found within mitochondria. Using Ais, we were able to prepare models to segment these granules, as well as microtubules, actin filaments, ribosomes, and to distinguish between the highly similar vesicular membranes on the one hand, and the membranes of the mitochondrial cristae on the other.

Lastly, we used Ais to generate a 3D visualization of ten distinct cellular features observed in coronavirus infected mammalian cells (**Fig. 5C**, Fig. S10), using data by Wolff et al.[17]: single membranes, double membrane vesicles (DMVs), actin filaments, intermediate filaments, microtubules, mitochondrial granules, ribosomes, coronaviral nucleocapsid proteins, coronaviral pores in the DMVs, and the nucleic acids within the DMV replication organelles. The software was able to accurately distinguish between single membranes and double membranes, as well as to discriminate between the various filaments of the cytoskeleton. Moreover, we could identify the molecular pores within the DMV, and pick sets of particles that might be suitable for use in subtomogram averaging (see Fig. S11). We could also segment the nucleocapsid proteins, thus distinguishing viral particles from other, similarly sized, single membrane vesicles, as well as detect the nucleic acids found within the DMVs.

Although manual annotation took some hours in this case, the processing of the full volume took approximately 3 minutes per feature once the models were trained, and models can of course be applied to other volumes without requiring additional training.

To conclude, our aim with the development of Ais was to simplify and improve the accuracy of automated tomogram segmentation in order to make this processing step more accessible to many cryoET users. Here, we have attempted to create an intuitive and organized user interface that streamlines the whole workflow from annotation, to model preparation, to volume processing and particle picking and inspecting the results. Additionally, the model repository at *aiscryoet.org* is designed to aid users in achieving results even faster, by removing the need to generate custom models for common segmentation tasks. To help users become familiar with the software, documentation and tutorials are available at *ais-cryoet.readthedocs.org* and video tutorials can be accessed via *youtube.com/@scNodes*. By demonstrating the use of Ais in automating segmentation and particle picking for subtomogram averaging, and making the software available as an open-source project, we thus hope to help accelerate research and dissemination of data involving cryoET.

Methods

Neural network comparisons

The comparison presented in **Table 1** was prepared with the use of the same training datasets for all architectures, consisting of 53/52/58 positive images showing membranes/carbon/antibody platforms and corresponding annotations alongside 159/51/172 negative images that did not contain the feature of interest, but rather the other two features, reconstruction artefacts, isolated protein, or background noise. Images were 64 × 64 pixels in size and the dataset was resampled, in random orientations, such that every positive image was copied 10 times and that the ratio of negatives to positives was 1.3:1. Networks were trained for 50 epochs with 32 images per batch, with the exception of the ResNet and Pix2pix antibody platform models, which were trained for 30 epochs to avoid a divergence that occurred during training after a larger number of epochs, due to the large number of network parameters and relatively low number of unique input images.

The reported loss is that calculated on the training dataset itself, i.e., no validation split was applied. During regular use of the software, users can specify whether to use a validation split or not. By default, a validation split is not applied in order to prioritize training accuracy by making

full use of the input set of ground truth annotations. When the training dataset is sufficiently large, we recommend increasing the size of the validation split. Depending on the chosen split size, the software reports either the overall training loss or the validation loss during training.

Data visualization

Images shown in the figures were either captured within the software (**Fig. 1**, **Fig. 3BC** 'original image', **Fig. 4ABC**), output from the software that was colourized in Inkscape (**Fig. 2C**, **Fig. 3BC**), or output from the software that was rendered using ChimeraX[18] (**Fig. 3D**, **Fig. 4EF**, **Fig. 5**). For the panels in **Fig. 3D**, segmented volumes were rendered as isosurfaces at a manually chosen suitable isosurface level and with the use of the 'hide dust' function (the same settings were used for each panel, different settings used for each feature). This 'dust' corresponds to small (in comparison to the segmented structures of interest) volumes of false positive segmentations, which are present in the data due to imperfections in the used models. The rate and volume of false positives can be reduced either by improving the models (typically by including more examples of the images that would be false negatives or positives in the training data) or, if the dust particles are indeed smaller than the structures of interest, they can simply be discarded by filtering particles based on their volume, as applied here. In particle picking a 'minimum particle volume' is specified – particles with a smaller volume are considered 'dust'.

Hardware

The software does not require a GPU, but works optimally when a CUDA capable GPU is available. For the measurements shown in **Table 1** we used an NVIDIA Quadro P2200 GPU on a PC with an Intel i9-10900K CPU. We've also extensively used the software on a less powerful system equipped with an NVIDIA T1000 and an Intel i3-10100 CPU, as well as on various systems with intermediate specifications, and found that the software reaches interactive segmentation rates in most cases. For batch processing of many volumes, a more powerful GPU is useful.

Tomogram reconstruction and subtomogram averaging

Data collection and subtomogram averaging (**Figs. 3** and **4**) was performed as described in a previously published article[10]. Briefly, tilt series were collected on a Talos Arctica 200 kV system equipped with a Gatan K3 detector with energy filter at a pixel size of 1.74 Å per pixel using a dose-symmetric tilt scheme with range ±57° and tilt increments of 3° with a total dose of 60 e/Å^2. Tomograms were reconstructed using IMOD[19]. Particle picking was done in Ais (and is explained in more detail in the online documentation). Subtomogram averaging was done using a combination of EMAN[1] and Dynamo[12]. For a detailed description of the subtomogram averaging procedure, see Abendstein et al. (2023)[10].

Open-source software

This project depends critically on a number of open-source software components, including: Python, Tensorflow[20], numpy[21], scipy[22], scikit-image[23], mrcfile[24], and imgui[25].

Software availability

A standalone version of the software is available as 'Ais-cryoET' on the Python package index and at *github.com/bionanopatterning/Ais*. We have also integrated the functionality into scNodes[26], our dedicated processing suite for correlated light and electron microscopy. In the combined package, the segmentation editor contains additional features for visualization of fluorescence

data and the scNodes correlation editor can be used to prepare correlated datasets for segmentation. Documentation for both versions of Ais can be found at *ais-cryoet.readthedocs.org*. Video tutorials are available via *youtube.com/@scNodes*.

Acknowledgements

We thank A. Koster and M. Barcena for helpful discussions and providing us with access to the coronaviral replication organelle datasets. We are also grateful to van den Hoek et al.[13] and Wu et al.[15], for uploading the data that we used for **Fig. 5** onto EMPIAR and EMDB, as well as to the authors of various other datasets uploaded to these databases that are not discussed in this manuscript but that were useful for testing the software. We also thank the reviewers, whose comments were very helpful in improving the manuscript and the software. We are also grateful to a number of Ais users who provided us with feedback on the software and reported issues. This research was supported by the following grants to THS: European Research Council H202 Grant 759517; European Union's Horizon Europe Program IMAGINE grant 101094250, and the Netherlands Organization for Scientific Research Grant VI.Vidi.193.014.

Additional information

Author Contributions

MGFL and LA collected and processed the data. MGFL wrote the software with input from all other authors. THS and LMV supervised the project.

Competing Interests Statement

The authors declare no competing interests.

References

1 Galaz-Montoya J. G., Flanagan J., Schmid M. F., Ludtke S. J. (2015) **Single particle tomography in EMAN2** *J Struct Biol* **190**:279–290 https://doi.org/10.1016/j.jsb.2015.04.016

2 Belevich I., Joensuu M., Kumar D., Vihinen H., Jokitalo E. (2016) **Microscopy image browser: a platform for segmentation and analysis of multidimensional datasets** *PLoS biology* **14** https://doi.org/10.1371/journal.pbio.1002340

3 Luengo I., et al. (2017) **SuRVoS: Super-Region Volume Segmentation workbench** *J Struct Biol* **198**:43–53 https://doi.org/10.1016/j.jsb.2017.02.007

4 Bankhead P., et al. (2017) **QuPath: Open source software for digital pathology image analysis** *Sci Rep* **7** https://doi.org/10.1038/s41598-017-17204-5

5 Szegedy C., et al. (2014) **Going deeper with convolutions** *arXiv* https://doi.org/10.48550/arXiv.1409.4842

6 Szegedy C., Ioffe S., Vanhoucke V., Alemi A. (2016) **Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning** *arXiv* https://doi.org/10.48550/arXiv.1602.07261

7 Ronneberger O., Fisher P., Brox T. (2015) **U-Net: Convolutional Networks for Biomedical Image Segmentation** *arXiv* https://doi.org/10.48550/arXiv.1505.04597

8 Simonyan K., Zisserman A. (2015) **Very Deep Convolutional Networks for Large-scale Image Regognition** *arXiv* https://doi.org/10.48550/arXiv.1409.1556

9 Isola P., Zhu J.-Y., Zhou T., Efros A. A. (2017) **Image-to-Image Translation with Conditional Adversarial Networks** *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* :5967–5976 https://doi.org/10.1109/CVPR.2017.632

10 Abendstein L., et al. (2023) **Complement is activated by elevated IgG3 hexameric platforms and deposits C4b onto distinct antibody domains** *Nat Commun* **14** https://doi.org/10.1038/s41467-023-39788-5

11 Pyle E., Hutchings J., Zanetti G. (2022) **Strategies for picking membrane-associated particles within subtomogram averaging workflows** *Faraday Discuss* **240**:101–113 https://doi.org/10.1039/d2fd00022a

12 Castano-Diez D., Kudryashev M., Arheit M., Stahlberg H. (2012) **Dynamo: a flexible, user-friendly development tool for subtomogram averaging of cryo-EM data in high-performance computing environments** *J Struct Biol* **178**:139–151 https://doi.org/10.1016/j.jsb.2011.12.017

13 van den Hoek H., et al. (2022) **In situ architecture of the ciliary base reveals thestepwise assembly of intraflagellar transport trains** *Science* **377**:543–548 https://doi.org/10.1126/science.abm6704

14 Iudin A., et al. (2023) **EMPIAR: the electron microscopy public image archive** *Nucleic Acids Research* **51**:D1503–D1511

15 Wu G. H., et al. (2023) **CryoET reveals organelle phenotypes in huntington disease patient iPSC-derived and mouse primary neurons** *Nat Commun* **14** https://doi.org/10.1038/s41467-023-36096-w

16 Lawson C. L., et al. (2015) **EMDataBank unified data resource for 3DEM** *Nucleic Acids Research* **44**:D396–D403 https://doi.org/10.1093/nar/gkv1126

17 Wolff G., et al. (2020) **A molecular pore spans the double membrane of the coronavirus replication organelle** *Science* **369**:1395–1398 https://doi.org/10.1126/science.abd3629

18 Goddard T. D., et al. (2018) **UCSF ChimeraX: Meeting modern challenges in visualization and analysis** *Protein Sci* **27**:14–25 https://doi.org/10.1002/pro.3235

19 Mastronarde D. N., Held S. R. (2017) **Automated tilt series alignment and tomographic reconstruction in IMOD** *J Struct Biol* **197**:102–113 https://doi.org/10.1016/j.jsb.2016.07.011

20 Abadi M., et al. (2015) **Large-Scale Machine Learning on Heterogeneous Distributed Systems** *arXiv* https://doi.org/10.48550/arXiv.1603.04467

21 Harris C. R., et al. (2020) **Array programming with NumPy** *Nature* **585**:357–362 https://doi.org/10.1038/s41586-020-2649-2

22 Virtanen P., et al. (2020) **SciPy 1.0: fundamental algorithms for scientific computing in Python** *Nat Methods* **17**:261–272 https://doi.org/10.1038/s41592-019-0686-2

23 van der Walt S., et al. (2014) **scikit-image: image processing in Python** *PeerJ* **2** https://doi.org/10.7717/peerj.453

24 Burnley T., Palmer C. M., Winn M. (2017) **Recent developments in the CCP-EM software suite** *Acta Crystallographica Section D* **73**:469–477 https://doi.org/10.1107/S2059798317007859

25 Cornut O. Dear Imgui (2023) **GitHub repository**

26 Last M. G. F., Voortman L. M., Sharp T. H. (2023) **scNodes: a correlation and processing toolkit for super-resolution fluorescence and electron microscopy** *Nat Methods* **20**:1445–1446 https://doi.org/10.1038/s41592-023-01991-z

Author information

Mart GF Last
Department of Cell and Chemical Biology, Leiden University Medical Center, Leiden, The Netherlands
ORCID iD: 0000-0002-3739-8863

For correspondence: mgflast@gmail.com

Leoni Abendstein
Department of Cell and Chemical Biology, Leiden University Medical Center, Leiden, The Netherlands, Institute of Science and Technology Austria (ISTA), Klosterneuburg, Austria
ORCID iD: 0000-0001-7634-5353

Lenard M Voortman
Department of Cell and Chemical Biology, Leiden University Medical Center, Leiden, The Netherlands
ORCID iD: 0000-0001-9794-067X

Thomas H Sharp
Department of Cell and Chemical Biology, Leiden University Medical Center, Leiden, The Netherlands, School of Biochemistry, University of Bristol, Bristol, United Kingdom
ORCID iD: 0000-0002-1990-2333

For correspondence: t.sharp@bristol.ac.uk

Editors

Reviewing Editor
Sjors Scheres
MRC Laboratory of Molecular Biology, Cambridge, United Kingdom

Senior Editor
Felix Campelo
Institute of Photonic Sciences, Barcelona, Spain

Reviewer #1 (Public review):

This paper describes "Ais", a new software tool for machine-learning based segmentation and particle picking of electron tomograms. The software can visualise tomograms as slices and allows manual annotation for the training of a provided set of various types of neural networks. New networks can be added, provided they adhere to a python file with an (undescribed) format. Once networks have been trained on manually annotated tomograms, they can be used to segment new tomograms within the same software. The authors also set up an online repository to which users can upload their models, so they might be re-used by others with similar needs. By logically combining the results from different types of segmentations, they further improve the detection of distinct features. The authors demonstrate the usefulness of their software on various data sets. Thus, the software appears to be a valuable tool for the cryo-ET community that will lower the boundaries of using a variety of machine-learning methods to help interpret tomograms.

https://doi.org/10.7554/eLife.98552.2.sa3

Reviewer #2 (Public review):

Summary:

Last et al. present Ais, a new deep learning based software package for segmentation of cryo electron tomography data sets. The distinguishing factor of this package is its orientation to the joint use of different models, rather than the implementation of a given approach:

Notably, the software is supported by an online repository of segmentation models, open to contributions from the community.

The usefulness of handling different models in one single environment is showcased with a comparative study on how different models perform on a given data set; then with an explanation on how the results of several models can be manually merged by the interactive tools inside Ais.

The manuscripts presents two applications of Ais on real data sets; one oriented to showcase its particle picking capacities on a study previously completed by the authors; a second one refers to a complex segmentation problem on two different data sets (representing different geometries as bacterial cilia and mitochondria in a mouse neuron), both from public databases.

The software described in the paper is compactly documented in its website, additionally providing links to some youtube videos (less than an hour it toral) where the authors videocapture and comment major workflows.

In short, the manuscript describes a valuable resource for the community of tomography practitioners.

Strengths:

Public repository of segmentation models; easiness of working with several models and comparing/merging the results.

https://doi.org/10.7554/eLife.98552.2.sa2

Reviewer #3 (Public review):

Summary:

In this manuscript, Last and colleagues describe Ais, an open-source software package for the semi-automated segmentation of cryo-electron tomography (cryo-ET) maps. Specifically, Ais provides a graphical user interface (GUI) for the manual segmentation and annotation of specific features of interest. These manual annotations are then used as input ground-truth data for training a convolutional neural network (CNN) model, which can then be used for automatic segmentation. Ais provides the option of several CNNs so that users can compare their performance on their structures of interest in order to determine the CNN that best suits their needs. Additionally, pretrained models can be uploaded and shared to an online database.

Algorithms are also provided to characterize "model interactions" which allows users to define heuristic rules on how the different segmentations interact. For instance, a membrane adjacent protein can have rules where it must colocalize a certain distance away from a membrane segmentation. Such rules can help reduce false positives; as in the case above, false negatives predicted away from membranes are eliminated.

The authors then show how Ais can be used for particle picking and subsequent subtomogram averaging and for segmentation of cellular tomograms for visual analysis. For subtomogram averaging, they used a previously published dataset and compared the averages of their automated picking with the published manual picking. Analysis of cellular tomogram segmentations were primarily visual.

Strengths:

CNN-based segmentation of cryo-ET data is a rapidly developing area of research, as it promises substantially faster results than manual segmentation as well as the possibility for higher accuracy. However, this field is still very much in the development and the overall performance of these approaches, even across different algorithms, still leaves much to be desired. In this context, I think Ais is an interesting packages, as it aims to provide both new and experienced users streamlined approaches for manual annotation, access to a number of CNNs, and methods to refine the outputs of CNN models against each other. I think this can be quite useful for users, particularly as these methods develop.

https://doi.org/10.7554/eLife.98552.2.sa1

Author response:

The following is the authors' response to the original reviews.

We would like to thank the reviewers for helping us improve our article and software. The feedback that we received was very helpful and constructive, and we hope that the changes that we have made are indeed effective at making the software more accessible, the manuscript clearer, and the online documentation more insightful as well. A number of comments related to shared concerns, such as:

• the need to describe various processing steps more clearly (e.g. particle picking, or the nature of 'dust' in segmentations)

• describing the features of Ais more clearly, and explaining how it can interface with existing tools that are commonly used in cryoET

• a degree of subjectivity in the discussion of results (e.g. about Pix2pix performing better than other networks in some cases.)

We have now addressed these important points, with a focus on streamlining not only the workflow within Ais but also making interfacing between Ais and other tools easier. For instance, we explain more clearly which file types Ais uses and we have added the option to export .star files for use in, e.g., Relion, or meshes instead of coordinate lists. We also include information in the manuscript about how the particle picking process is implemented, and how false positives ('dust') can be avoided. Finally, all reviewers commented on our notion that Pix2pix can work 'better' despite reaching a higher loss after training. As suggested, we included a brief discussion about this idea in the supplementary information (Fig. S6) and used it to illustrate how Ais enables iteratively improving segmentation results.

Since receiving the reviews we have also made a number of other changes to the software that are not discussed below but that we nonetheless hope have made the software more reliable and easier to use. These include expanding the available settings, slight changes to the image processing that can help speed it up or avoid artefacts in some cases, improving the GUI-free usability of Ais, and incorporating various tools that should help make it easier to use Ais with remote data (e.g. doing annotation on an office PC, but model training on a more powerful remote PC). We have also been in contact with a number of users of the software, who reported issues or suggested various other miscellaneous improvements, and many of whom had found the software via the reviewed preprint.

> ***Reviewer 1 (Public Review):***
>
> *This paper describes "Ais", a new software tool for machine-learning-based segmentation and particle picking of electron tomograms. The software can visualise tomograms as slices and allows manual annotation for the training of a provided set of various types of*

> *neural networks. New networks can be added, provided they adhere to a Python file with an (undescribed) format. Once networks have been trained on manually annotated tomograms, they can be used to segment new tomograms within the same software. The authors also set up an online repository to which users can upload their models, so they might be re-used by others with similar needs. By logically combining the results from different types of segmentations, they further improve the detection of distinct features. The authors demonstrate the usefulness of their software on various data sets. Thus, the software appears to be a valuable tool for the cryo-ET community that will lower the boundaries of using a variety of machine-learning methods to help interpret tomograms.*

We thank the reviewer for their kind feedback and for taking the time to review our article. On the basis of their comments, we have made a number of changes to the software, article, and documentation, that we think have helped improve the project and render it more accessible (especially for interfacing with different tools, e.g. the suggestions to describe the file formats in more detail). We respond to all individual comments one-by-one below.

> ***Recommendations:***
>
> *I would consider raising the level of evidence that this program is useful to *convincing* if the authors would adequately address the suggestions for improvement below.*
>
> *(1) It would be helpful to describe the format of the Python files that are used to import networks, possibly in a supplement to the paper.*

We have now included this information in both the online documentation and as a supplementary note (Supplementary Note 1).

> *(2) Likewise, it would be helpful to describe the format in which particle coordinates are produced. How can they be used in subsequent sub-tomogram averaging pipelines? Are segmentations saved as MRC volumes? Or could they be saved as triangulations as well? More implementation details like this would be good to have in the paper, so readers don't have to go into the code to investigate.*

Coordinates: previously, we only exported arrays of coordinates as tab-separated .txt files, compatible with e.g. EMAN2. We now added a selection menu where users can specify whether to export either .star files or tsv .txt files, which together we think should cover most software suites for subtomogram averaging.

Triangulations: We have now improved the functionality for exporting triangulations. In the particle picking menu, there is now the option to output either coordinates or meshes (as .obj files). This was previously possible in the Rendering tab, but with the inclusion in the picking menu exporting triangulations can now be done for all tomograms at once rather than manually one by one.

Edits in the text: the output formats were previously not clear in the text. We have now included this information in the introduction:

“[…] To ensure compatibility with other popular cryoET data processing suites, Ais employs file formats that are common in the field, using .mrc files for volumes, tab-separated .txt or .star files for particle datasets, and the .obj file format for exporting 3D meshes.”

> *(3) In Table 2, pix2pix has much higher losses than alternatives, yet the text states it achieves fewer false negatives and fewer false positives. An explanation is needed as to why that is. Also, it is mentioned that a higher number of epochs may have improved the results. Then why wasn't this attempted?*

The architecture of Pix2pix is quite different from that of the other networks included in the test. Whereas all others are trained to minimize a binary cross entropy (BCE) loss, Pix2pix uses a composite loss function that is a weighted combination of the generator loss and a discriminator penalty, neither of which employ BCE. However, to be able to compare loss values, we do compute a BCE loss value for the Pix2pix generator after every training epoch. This is the value reported in the manuscript and in the software. Although Pix2pix' BCE loss does indeed diminish during training, the model is not actually optimized to minimize this particular value and a comparison by BCE loss is therefore not entirely fair to Pix2pix. This is pointed out (in brief) in the legend to the able:

"Unlike the other architectures, Pix2pix is not trained to minimize the bce loss but uses a different loss function instead. The bce loss values shown here were computed after training and may not be entirely comparable."

Regarding the extra number of epochs for Pix2pix: here, we initially ran in to the problem that the number of samples in the training data was low for the number of parameters in Pix2pix, leading to divergence later during training. This problem did not occur for most other models, so we decided to keep the data for the discussion around Table 1 and Figure 2 limited to that initial training dataset. After that, we increased the sample size (from 58 to 170 positive samples) and trained the model for longer. The resulting model was used in the subsequent analyses. This was previously implicit in the text but is now mentioned explicitly and in a new supplementary figure.

"For the antibody platform, the model that would be expected to be one of the worst based on the loss values, Pix2pix, actually generates segmentations that are seem well-suited for the downstream processing tasks. It also output fewer false positive segmentations for sections of membranes than many other models, including the lowest-loss model UNet. Moreover, since Pix2pix is a relatively large network, it might also be improved further by increasing the number of training epochs. We thus decided to use Pix2pix for the segmentation of antibody platforms, and increased the size of the antibody platform training dataset (from 58 to 170 positive samples) to train a much improved second iteration of the network for use in the following analyses (Fig. S6)."

> *(4) It is not so clear what absorb and emit mean in the text about model interactions. A few explanatory sentences would be useful here.*

We have expanded this paragraph to include some more detail.

"Besides these specific interactions between two models, the software also enables pitching multiple models against one another in what we call 'model competition'. Models can be set to 'emit' and/or 'absorb' competition from other models. Here, to emit competition means that a model's prediction value is included in a list of competing models. To absorb competition means that a model's prediction value will be compared to all values in that list, and that this model's prediction value for any pixel will be set to zero if any of the competing models' prediction value is higher. On a pixel-by-pixel basis, all models that absorb competition are thus suppressed whenever their prediction value for a pixel is lower than that of any of the emitting models."

> *(5) Under Figure 4, the main text states "the model interactions described above", but because multiple interactions were described it is not clear which ones they were. Better to just specify again.*

Changed as follows:

“The antibody platform and antibody-C1 complex models were then applied to the respective datasets, in combination with the membrane and carbon models and the model interactions described above (Fig. 4b): the membrane avoiding carbon, and the antibody platforms colocalizing with the resulting membranes”.

(6) The next paragraph mentions a "batch particle picking process to determine lists of particle coordinates", but the algorithm for how coordinates are obtained from segmented volumes is not described.

We have added a paragraph to the main text to describe the picking process:

“This picking step comprises a number of processing steps (Fig. S7). First, the segmented (.mrc) volumes are thresholded at a user-specified level. Second, a distance transform of the resulting binary volume is computed, in which every nonzero pixel in the binary volume is assigned a new value, equal to the distance of that pixel to the nearest zero-valued pixel in the mask. Third, a watershed transform is applied to the resulting volume, so that the sets of pixels closest to any local maximum in the distance transformed volume are assigned to one group. Fourth, groups that are smaller than a user-specified minimum volume are discarded. Fifth, groups are assigned a weight value, equal to the sum of the prediction value (i.e. the corresponding pixel value in the input .mrc volume) of the pixels in the group. For every group found within close proximity to another group (using a user-specified value for the minimum particle spacing), the group with the lower weight value is discarded. Finally, the centroid coordinate of the grouped pixels is considered the final particle coordinate, and the list of all

coordinates is saved in a tab-separated text file.

“As an alternative output format, segmentations can also be converted to and saved as triangulated meshes, which can then be used for, e.g., membrane-guided particle picking. After picking particles, the resulting coordinates are immediately available for inspection in the Ais 3D renderer (Fig. S8).“

The two supplementary figures are pasted below for convenience. Fig. S7 is new, while Fig. S8 was previously Fig. S10 -the reference to this figure was originally missing in the main text, but is now included.

> *(7) In the Methods section, it is stated that no validation splits are used "in order to make full use of an input set". This sounds like an odd decision, given the importance of validation sets in the training of many neural networks. Then how is overfitting monitored or prevented? This sounds like a major limitation of the method.*

In our experience, the best way of preparing a suitable model is to (iteratively) annotate a set of training images and visually inspect the result. Since the manual annotation step is the bottleneck in this process, we decided not to use validation split in order to make full use of an annotated training dataset (i.e. a validation split of 20% would mean that 20% of the manually annotated training data is not used for training)

We do recognize the importance of using separate data for validation, or at least offering the possibility of doing so. We have now added a parameter to the settings (and made a Settings menu item available in the top menu bar) where users can specify what fraction (0, 10, 20, or 50%) of training datasets should be set aside for validation. If the chosen value is not 0%, the software reports the validation loss as well as the size of the split during training, rather than (as was done previously) the training loss. We have, however, set the default value for the validation split to 0%, for the same reason as before. We also added a section to the online documentation about using validation splits, and edited the corresponding paragraph in the methods section:

"The reported loss is that calculated on the training dataset itself, i.e., no validation split was applied. During regular use of the software, users can specify whether to use a validation split or not. By default, a validation split is not applied, in order to make full use of an input set of ground truth annotations. Depending on the chosen split size, the software reports either the overall training loss or the validation loss during training."

> *(8) Related to this point: how is the training of the models in the software modelled? It might be helpful to add a paragraph to the paper in which this process is described, together with indicators of what to look out for when training a model, e.g. when should one stop training?*

We have expanded the paragraph where we write about the utility of comparing different networks architectures to also include a note on how Ais facilitates monitoring the output of a model during training:

"When taking the training and processing speeds in to account as well as the segmentation results, there is no overall best architecture. We therefore included multiple well-performing model architectures in the final library, in order to allow users to select from these models to find one that works well for their specific datasets. Although it is not necessary to screen different network architectures and users may simply opt to use the default (VGGNet), these results thus show that it can be useful to test different networks in order to identify one that is best. Moreover, these results also highlight the utility of preparing well-performing models by iteratively improving training datasets and re-training models in a streamlined interface. To aid in this process, the software displays the loss value of a network during training and allows for the application of models to datasets during training. Thus, users can inspect how a model's output changes during training and decide whether to interrupt training and improve the training data or choose a different architecture."

> *(9) Figure 1 legend: define the colours of the different segmentations.*

Done

> *(10) It may be better to colour Figure 2B with the same colours as Figure 2A.*

We tried this, but the effect is that the underlying density is much harder to see. We think the current grayscale image paired with the various segmentations underneath is better for visually identifying which density corresponds to membranes, carbon film, or antibody platforms.

> ***Reviewer 2 (Public Review):***
>
> *Summary:*
>
> *Last et al. present Ais, a new deep learning-based software package for the segmentation of cryo-electron tomography data sets. The distinguishing factor of this package is its orientation to the joint use of different models, rather than the implementation of a given approach. Notably, the software is supported by an online repository of segmentation models, open to contributions from the community.*
>
> *The usefulness of handling different models in one single environment is showcased with a comparative study on how different models perform on a given data set; then with an explanation of how the results of several models can be manually merged by the interactive tools inside Ais.*

> *The manuscripts present two applications of Ais on real data sets; one is oriented to showcase its particlepicking capacities on a study previously completed by the authors; the second one refers to a complex segmentation problem on two different data sets (representing different geometries as bacterial cilia and mitochondria in a mouse neuron), both from public databases.*
>
> *The software described in the paper is compactly documented on its website, additionally providing links to some YouTube videos (less than an hour in total) where the authors videocapture and comment on major workflows.*
>
> *In short, the manuscript describes a valuable resource for the community of tomography practitioners.*
>
> *Strengths:*
>
> *A public repository of segmentation models; easiness of working with several models and comparing/merging the results.*
>
> *Weaknesses:*
>
> *A certain lack of concretion when describing the overall features of the software that differentiate it from others.*

We thank the reviewer for their kind and constructive feedback. Following the suggestion to use the Pix2pix results to illustrate the utility of Ais for analyzing results, we have added a new supplementary figure (Fig. S6) and brief discussion, showing the use of Ais in iteratively improving segmentation results. We have also expanded the online documentation and included a note in the supplementary information about how models are saved/loaded (Supplemetary note 1)

> ***Recommendations:***
>
> *I would like to ask the authors about some concerns about the Ais project as a whole:*
>
> *(1) The website that accompanies the paper (aiscryoet.org), albeit functional, seems to be in its first steps. Is it planned to extend it? In particular, one of the major contributions of the paper (the maintenance of an open repository of models) could use better documentation describing the expected formats to submit models. This could even be discussed in the supplementary material of the manuscript, as this feature is possibly the most distinctive one of the paper. Engaging third-party users would require giving them an easier entry point, and the superficial mention of this aspect in the online documentation could be much more generous.*

We have added a new page to the online documentation, titled ‘Sharing models’ where we include an explanation of the structure of model files and demonstrate the upload page. We also added a note to the Supplementary Information that explains the file format for models, and how they are loaded/saved (i.e., that these standard keras model obects).

To make it easier to interface Ais with other tools, we have now also made some of the core functionality available (e.g. training models, batch segmentation) via the command line interface. Information on how to use this is included in the online documentation. All file formats are common formats used in cryoET, so that using Ais in a workflow with, e.g. AreTomo -> Ais -> Relion should now be more straightforward.

> *(2) A different major line advanced by the authors to underpin the novelty of the software, is its claimed flexibility and modularity. In particular, the restrictions of other*

> *packages in terms of visualization and user interaction are mentioned. Although in the manuscript it is also mentioned that most of the functionalities in Ais are already available in major established packages, as a reader I am left confused about what exactly makes the offer of Ais different from others in terms of operation and interaction: is it just the two aspects developed in the manuscript (possibility of using different models and tools to operate model interaction)? If so, it should probably be stated; but if the authors want to pinpoint other aspects of the capacity of Ais to drive smoothly the interactions, they should be listed and described, instead of leaving it as an unspecific comment. As a potential user of Ais, I would suggest the authors add (maybe in the supplementary material) a listing of such features. Figure 1 does indeed carry the name "overview of (...) functionalities", but it is not clear to me which functionalities I can expect to be absent or differently solved on the other tools they mention.*

We have rewritten the part of the introduction where we previously listed the features as below. We think it should now be clearer for the reader to know what features to expect, as well as how Ais can interface with other software (i.e. what the inputs and outputs are). We have also edited the caption for Figure 1 to make it explicit that panels A to C represent the annotation, model preparation, and rendering steps of the Ais workflow and that the images are screenshots from the software.

“In this report we present Ais, an open-source tool that is designed to enable any cryoET user – whether experienced with software and segmentation or a novice – to quickly and accurately segment their cryoET data in a streamlined and largely automated fashion. Ais comprises a comprehensive and accessible user interface within which all steps of segmentation can be performed, including: the annotation of tomograms and compiling datasets for the training of convolutional neural networks (CNNs), training and monitoring performance of CNNs for automated segmentation, 3D visualization of segmentations, and exporting particle coordinates or meshes for use in downstream processes. To help generate accurate segmentations, the software contains a library of various neural network architectures and implements a system of configurable interactions between different models. Overall, the software thus aims to enable a streamlined workflow where users can interactively test, improve, and employ CNNs for automated segmentation. To ensure compatibility with other popular cryoET data processing suites, Ais employs file formats that are common in the field, using .mrc files for volumes, tab-separated .txt or .star files for particle datasets, and the .obj file format for exporting 3D meshes.”

“Figure 1 – an overview of the user interface and functionalities. The various panels represent sequential stages in the Ais processing workflow, including annotation (a), testing CNNs (b), visualizing segmentation (c). These images (a-c) are unedited screenshots of the software. a) […]”

> *(3) Table 1 could have the names of the three last columns. The table has enough empty space in the other columns to accommodate this.*

Done.

> *(4) The comment about Pix2pix needing a larger number of training epochs (being a larger model than the other ones considered) is interesting. It also lends itself for the authors to illustrate the ability of their software to precisely do this: allow the users to flexibly analyze results and test hypothesis*

Please see the response to Reviewer 1 comment #3. We agree that this is a useful example of the ability to iterate between annotation and training, and have added an explicit mention of this in the text:

"Moreover, since Pix2pix is a relatively large network, it might also be improved further by increasing the number of training epochs. In a second iteration of annotation and training, we thus increased the size of the antibody platform training dataset (from 58 to 170 positive samples) and generated an improved Pix2pix model for use in the following analyses."

> ***Reviewer 3 (Public Review):***
>
> *We appreciate the reviewer's extensive and very helpful feedback and are glad to read that they consider Ais potentially quite useful for the users. To address the reviewer's comments, we have made various edits to the text, figures, and documentation, that we think have helped improve the clarity of our work. We list all edits below.*
>
> *Summary*
>
> *In this manuscript, Last and colleagues describe Ais, an open-source software package for the semi-automated segmentation of cryo-electron tomography (cryo-ET) maps. Specifically, Ais provides a graphical user interface (GUI) for the manual segmentation and annotation of specific features of interest. These manual annotations are then used as input ground-truth data for training a convolutional neural network (CNN) model, which can then be used for automatic segmentation. Ais provides the option of several CNNs so that users can compare their performance on their structures of interest in order to determine the CNN that best suits their needs. Additionally, pre-trained models can be uploaded and shared to an online database.*
>
> *Algorithms are also provided to characterize "model interactions" which allows users to define heuristic rules on how the different segmentations interact. For instance, a membrane-adjacent protein can have rules where it must colocalize a certain distance away from a membrane segmentation. Such rules can help reduce false positives; as in the case above, false negatives predicted away from membranes are eliminated.*
>
> *The authors then show how Ais can be used for particle picking and subsequent subtomogram averaging and for the segmentation of cellular tomograms for visual analysis. For subtomogram averaging, they used a previously published dataset and compared the averages of their automated picking with the published manual picking. Analysis of cellular tomogram segmentation was primarily visual.*
>
> *Strengths:*
>
> *CNN-based segmentation of cryo-ET data is a rapidly developing area of research, as it promises substantially faster results than manual segmentation as well as the possibility for higher accuracy. However, this field is still very much in the development and the overall performance of these approaches, even across different algorithms, still leaves much to be desired. In this context, I think Ais is an interesting package, as it aims to provide both new and experienced users with streamlined approaches for manual annotation, access to a number of CNNs, and methods to refine the outputs of CNN models against each other. I think this can be quite useful for users, particularly as these methods develop.*
>
> *Weaknesses:*
>
> *Whilst overall I am enthusiastic about this manuscript, I still have a number of comments:*
>
> *(1) On page 5, paragraph 1, there is a discussion on human judgement of these results. I think a more detailed discussion is required here, as from looking at the figures, I don't know that I agree with the authors' statement that Pix2pix is better. I acknowledge that*

> *this is extremely subjective, which is the problem. I think that a manual segmentation should also be shown in a figure so that the reader has a better way to gauge the performance of the automated segmentation.*

Please see the answer to Reviewer 1's comment #3.

> *(2) On page 7, the authors mention terms such as "emit" and "absorb" but never properly define them, such that I feel like I'm guessing at their meaning. Precise definitions of these terms should be provided.*

We have expanded this paragraph to include some more detail:

"Besides these specific interactions between two models, the software also enables pitching multiple models against one another in what we call 'model competition'. Models can be set to 'emit' and/or 'absorb' competition from other models. Here, to emit competition means that a model's prediction value is included in a list of competing models. To absorb competition means that a model's prediction value will be compared to all values in that list, and that this model's prediction value for any pixel will be set to zero if any of the competing models' prediction value is higher. On a pixel-by-pixel basis, all models that absorb competition are thus suppressed whenever their prediction value for a pixel is lower than that of any of the emitting models."

> *(3) For Figure 3, it's unclear if the parent models shown (particularly the carbon model) are binary or not.*
>
> *The figure looks to be grey values, which would imply that it's the visualization of some prediction score. If so, how is this thresholded? This can also be made clearer in the text.*

The figures show the grayscale output of the parent model, but this grayscale output is thresholded to produce a binary mask that is used in an interaction. We have edited the text to include a mention of thresholding at a user-specified threshold value:

"These interactions are implemented as follows: first, a binary mask is generated by thresholding the parent model's predictions using a user-specified threshold value. Next, the mask is then dilated using a circular kernel with a radius R, a parameter that we call the interaction radius. Finally, the child model's prediction values are multiplied with this mask."

To avoid confusion, we have also edited the figure to show the binary masks rather than the grayscale segmentations.

> *(4) Figure 3D was produced in ChimeraX using the hide dust function. I think some discussion on the nature of this "dust" is in order, e.g. how much is there and how large does it need to be to be considered dust? Given that these segmentations can be used for particle picking, this seems like it may be a major contributor to false positives.*

'Dust' in segmentations is essentially unavoidable; it would require a perfect model that does not produce any false positives. However, when models are sufficiently accurate, the volume of false positives is typically smaller than that of the structures that were intended to be segmented. In these cases, discarding particles based on size is a practical way of filtering the segmentation results. Since it is difficult to generalize when to consider something 'dust' we decided to include this additional text in the Method's section rather than in the main text:

"… with the use of the 'hide dust' function (the same settings were used for each panel, different settings used for each feature).

This ‘dust’ corresponds to small (in comparison to the segmented structures of interest) volumes of false positive segmentations, which are present in the data due to imperfections in the used models. The rate and volume of false positives can be reduced either by improving the models (typically by including more examples of the images of what would be false negatives or positives in the training data) or, if the dust particles are indeed smaller than the structures of interest, they can simply be discarded by filtering particles based on their volume, as applied here. In particle picking a ‘minimum particle volume’ is specified – particles with a smaller volume are considered ‘dust’.

In combination with the newly included text about the method of converting volumes into lists of coordinates (see Reviewer 1’s comment #6).

“Third, a watershed transform is applied to the resulting volume, so that the sets of pixels closest to any local maximum in the distance transformed volume are assigned to one group. Fourth, groups that are smaller than a user-specified minimum volume are discarded…”

We think it should now be clearer that (some form of) discarding ‘dust’ is a step that is typically included in the particle picking process.

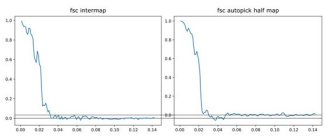
> *(5) Page 9 contains the following sentence: "After selecting these values, we then launched a batch particle picking process to determine lists of particle coordinates based on the segmented volumes." Given how important this is, I feel like this requires significant description, e.g. how are densities thresholded, how are centers determined, and what if there are overlapping segmentations?*

Please see the response to Reviewer 1’s comment #6.

> *(6) The FSC shown in Figure S6 for the auto-picked maps is concerning. First, a horizontal line at FSC = 0 should be added. It seems that starting at a frequency of ~0.045, the FSC of the autopicked map increases above zero and stays there. Since this is not present in the FSC of the manually picked averages, this suggests the automatic approach is also finding some sort of consistent features. This needs to be discussed.*

Thank you for pointing this out. Awkwardly, this was due to a mistake made while formatting the figure. In the two separate original plots, the Y axes had slightly different ranges, but this was missed when they were combined to prepare the joint supplementary figure. As a result, the FSC values for the autopicked half maps are displayed incorrectly. The original separate plots are shown below to illustrate the discrepancy:

Author response image 1.



The corrected figure is Figure S9 in the manuscript. The values of 44 Å and 46 Å were not determined from the graph and remain unchanged.

> *(7) Page 11 contains the statement "the segmented volumes found no immediately apparent false positive predictions of these pores". This is quite subjective and I don't know that I agree with this assessment. Unless the authors decide to quantify this through subtomogram classification, I don't think this statement is appropriate.*

We originally included this statement and the supplementary figure because we wanted to show another example of automated picking, this time in the more crowded environment of the cell. We do agree that it requires better substantiation, but also think that the demonstration of automated picking of the antibody platforms and IgG3-C1 complexes for subtomogram averaging suffices to demonstrate Ais' picking capabilities. Since the supplementary information includes an example of picked coordinates rendered in the Ais 3D viewer (Figure S7) that also used the pore dataset, we still include the supplementary figure (S10) but have edited the statement to read:

"Moreover, we could identify the molecular pores within the DMV, and pick sets of particles that might be suitable for use in subtomogram averaging (see Fig. S11)."

We have also expanded the text that accompanies the supplementary figure to emphasize that results from automated picking are likely to require further curation, e.g. by classification in subtomogram averaging, and that the selection of particles is highly dependent on the thresholds used in the conversion from volumes to lists of coordinates.

> *(8) In the methods, the authors note that particle picking is explained in detail in the online documentation. Given that this is a key feature of this software, such an explanation should be in the manuscript.*

Please see the response to Reviewer 1's comment #6.

> ***Recommendations:***
>
> *(9) The word "model" seems to be used quite ambiguously. Sometimes it seems to refer to the manual segmentations, the CNN architectures, the trained models, or the output predictions. More precision in this language would greatly improve the readability of the manuscript.*

This was indeed quite ambiguous, especially in the introduction. We have edited the text to be clearer on these differences. The word 'model' is now only used to refer to trained CNNs that segment a particular feature (as in 'membrane model' or 'model interactions'). Where we used terms such as '3D models' to describe scenes rendered in 3D, we now use '3D visualizations' or similar terms. Where we previously used the term 'models' to refer to CNN architectures, we now use terms such as 'neural network architectures' or 'architecture'. Some examples:

… with which one can automatically segment the same or any other dataset …

Moreover, since Pix2pix is a relatively large network, …

… to generate a 3D visualization of ten distinct cellular …

… with the use of the same training datasets for all network architectures …

> *In Figure 1, the text in panels D and E is illegible.*

We have edited the figure to show the text more clearly (the previous images were unedited screenshots of the website).

> *(10) Prior to the section on model interactions, I was under the impression that all annotations were performed simultaneously. I think it could be clarified that models are generated per annotation type.*

Multiple different features can be annotated (i.e. drawn by hand by the user) at the same time, but each trained CNN only segments one feature. CNNs that output segmentations for multiple features can be implemented straightforwardly, but this introduces the need to provide training data where for every grayscale image, every feature is annotated. This can make preparing the training data much more cumbersome. Reusability of the models is also hampered. We now mention the separateness of the networks explicitly in the introduction:

“Multiple features, such as membranes, microtubules, ribosomes, and phosphate crystals, can be segmented and edited at the same time across multiple datasets (even hundreds). These annotations are then extracted and used as ground truth labels upon which to condition multiple separate neural networks, …”

> *(11) On page 6, there is the text "some features are assigned a high segmentation value by multiple of the networks, leading to ambiguity in the results". Do they mean some false features?*

To avoid ambiguity of the word ‘features’, we have edited the sentence to read:

“… some parts of the image are assigned a high segmentation value by multiple of the networks, leading to false classifications and ambiguity in the results.”

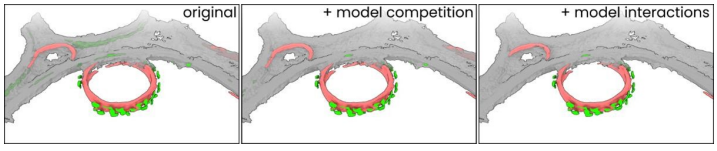
> *(12) Figures 2 and 3 would be easier to follow if they had consistent coloring.*

We have changed the colouring in Figure 2 to match that of Figure 3 better:

> *(13) For Figure 3D, I'm confused as to why the authors showed results from the tomogram in Figure 2B. It seems like the tomogram in Figure 3C would be a more obvious choice, as we would be able to see how the 2D slices look in 3D. This would also make it easier to see the effect of interactions on false negatives. Also, since the orientation of the tomogram in 2B is quite different than that shown in 3D, it's a bit difficult to relate the two.*

We chose to show this dataset because it exemplifies the effects of both model competition and model interactions better than the tomogram in Figure 3C. See Figure 3D and Author response image 2 for a comparison:

Author response image 2.



> *(14) I'm confused as to why the tomographic data shown in Figures 4D, E, and F are black on white while all other cryo-ET data is shown as white on black.*

The images in Figure 4DEF are now inverted.

> *(15) For Figure 5, there needs to be better visual cueing to emphasize which tomographic slices are related to the segmentations in Panels A and B.*

We have edited the figure to show more clearly which grayscale image corresponds to which segmentation:

> *(16) I don't understand what I should be taking away from Figures S1 and S2. There are a lot of boxes around membrane areas and I don't know what these boxes mean.*

We have added a more descriptive text to these figures. The boxes are placed by the user to select areas of the image that will be sampled when saving training datasets.

https://doi.org/10.7554/eLife.98552.2.sa0