Supporting Information for

High-throughput cryo-ET structural pattern mining by unsupervised deep iterative subtomogram clustering

Xiangrui Zeng, Anson Kahng, Liang Xue, Julia Mahamid, Yi-Wei Chang, and Min Xu

Min Xu.
E-mail: mxu1@cs.cmu.edu

This PDF file includes:

Supporting Information Text

Fast visualization of cluster centers through a decoder

To validate that the learned features encode essential structural information of the input subtomograms, we trained a decoder using the *Rattus* neuron cryo-ET dataset (1) as an example. The input to the decoder is the learned features from DISCA and the output is the reconstruction of the input 3D subtomograms. Similar to (2), we then decoded the cluster centers, arithmetic averages of all feature vectors in a cluster, into reconstructed 3D images. Alternatively, instead of cluster center, features closest to each cluster center can also be decoded, which yields similar results. As shown in **Fig. S6**, center decodings of identifiable clusters resemble the type of structures contained, which validates the essential structural information effectively learned by the extracted features. Center decodings of non-identifiable clusters mostly resemble a tiny globular structure, which likely to indicates that most subtomograms contained in these clusters are either noises or structures too small. Therefore, DISCA can be used to efficiently filter out false-positive particles picked by template-free particle picking methods.

In addition, it is very useful to quickly identify interesting clusters for downstream analysis before doing the computationally intensive subtomogram averaging step. The training of the decoder from scratch on this dataset of 36,377 subtomograms took less than 10 minutes. Therefore, the decoding of cluster centers can be used for such identification purposes, especially for structural clusters that can be easily recognized such as the ribosome, surface patterns, and fiducial markers.

We note here that, since the relevant features are already extracted using DISCA, we directly used a decoder to decode the cluster centers to provide fast guidance on the structural content of each cluster. Previously, we have designed an autoencoder approach (2) to extract relevant features for coarse clustering purposes. The autoencoder serves as a baseline comparison in **Table 1**. The performance of the autoencoder is much worse than DISCA. This is mainly because DISCA is a significantly more sophisticated method that involves iterative feature learning and modeling in order to recognize the fine structure differences between different types of macromolecules. Studies (3, 4) have shown that vanilla autoencoders only learn representative features to reconstruct the input images, and do not learn discriminative features between different semantic classes.

Visualization of subtomogram averages from the *Rattus* neuron dataset

We visualized the 19 (automatically determined K) subtomogram cluster averages by DISCA sorting and *Relion 3.0* single-class averaging in **Fig. S7**.

Visual comparison and FSC curve of subtomogram averages

The gold-standard Fourier Shell Correlation (FSC) curve of subtomogram average of detected macromolecular structures in the five experimental datasets are produced by the *Postprocess* program in *Relion 3.0* using default parameters. The black, green, blue, and red curves stand for "rln Fourier Shell Correlation Corrected", "rln Fourier Shell Correlation Unmasked Maps", "rln Fourier Shell Correlation Masked Maps", and "rln Fourier Shell Correlation Phase Randomized Masked Maps", respectively. In the left side of each figure, we visually compare the subtomogram average with an existing structure from the Protein DataBank (5). The isosurface representation of each structural template is filtered to the estimated resolution of the subtomogram average for better visual comparison.

Distortion-based Davies-Bouldin Index. We mathematically formulate the proposed distortion-based DBI (DDBI) as:

$$D = \frac{1}{K}\sum_{i=1}^{K}\max_{j\neq i}\frac{t_i + t_j}{d_{ij} + d_{ji}}\; j \in 1, 2, \ldots, K, \qquad [1]$$

where t_i measures the tightness of ith cluster (same for t_j) and d_{ij} measures the separation between cluster i and j:

$$t_i = \frac{1}{|C_i|}\sum_{x_n \in C_i}(x_n - c_i)^T \Sigma_i^{-1}(x_n - c_i), \qquad [2]$$

$$d_{ij} = (c_i - c_j)^T \Sigma_i^{-1}(c_i - c_j), \qquad [3]$$

where C_i denotes the subtomograms x_n in the ith cluster and c_i denotes its centroid.

Automatic estimation of the number of structurally homogeneous subsets. Because we operate in an unsupervised learning setting, the number of structurally homogeneous subsets K is unknown to us. Furthermore, the automatic estimation of the number of clusters in a feature space is a classic yet highly challenging and largely unsolved problem, which means that, practically, most studies just set an arbitrary K or test multiple candidate values of K and manually compare the results. Nevertheless, in our statistical modeling, it is beneficial to choose K properly. When the chosen K is too small, a subset may contain mixed structures. In contrast, when the chosen K is too large, a structurally homogeneous subset may be over-partitioned to multiple subsets. Over-partitioning likely results in some subsets containing too few subtomograms to

recover the structure. Both situations may lead to poorly recovered structures by subtomogram averaging. For this reason, it is helpful to automatically determine K.

Automatic estimation of K relies on observing the extracted feature vectors. Most recent and popular methods for estimating K are either prediction-based or stability-based, and require running the given clustering algorithm repeatedly on bootstrapped samples. These methods are not suitable for our study because they are too slow to process large-scale datasets. Other methods for estimating K compute a summary index measuring cluster tightness. For example, the silhouette coefficient compares the average distance of a data point to all the other data points in its own cluster and in its nearest cluster. However, computing the silhouette coefficient involves comparing all pairs of data points (time complexity: $O(N^2)$), which is still poor in scalability.

To overcome these shortcomings, we take an alternative approach from a statistical model selection perspective. The number of model parameters increases along with K, which may result in increased likelihood, but also runs the risk of overfitting. When modeling the structurally homogeneous subsets in the feature space, a good statistical model would ideally have a higher likelihood with relatively few parameters. To balance the likelihood and number of parameters among a set of models with different Ks, we use the Bayesian Information Criterion (BIC) (6) to select among a set of fitted models M, where the BIC is defined as:

$$\begin{aligned} BIC(M_k) &= P(M_k)\ln(N) - 2\ln(\hat{L}(M_k)) \\ &= (K(P^2+P)/2)\ln(N) - 2\sum_{n=1}^{N}\ln(\sum_{k=1}^{K}\phi_k g(x_n;\mu_k,\Sigma_k)), \end{aligned} \quad [4]$$

where M_k is the fitted model with K structurally homogeneous subsets, $P(M_k)$ denotes the number of parameters in model M_k and $\hat{L}(M_k)$ denotes the maximized value of the likelihood function of M_k. For each candidate K, one model is fitted. The model with the lowest BIC is selected. We also tested Akaike information criterion (AIC) (7), CH index (8), KL index (9), and Jump statistic (10), our preliminary results showed that BIC achieved superior performance.

Matching clustering solutions. From our experience, the estimated K stays the same in most iterations. In such cases, instead of replacing the last classification layer, we directly match the current clustering solution with the one in the previous iteration. When there are multiple clustering solutions from the same samples, the label of a specific cluster is not necessarily the same between different solutions. For example, the same group of samples may be labeled as '1' by one clustering solution and '2' by another even if they result from the same clustering algorithm with exactly the same parameters. The inconsistency will cause strong instability during training **Fig. S5**. Therefore, matching clustering solutions is necessary.

We formulate the problem of matching two clustering solutions as a maximum weighted bipartite matching problem. First, we define a bipartite graph that consists of two disjoint and independent sets. In our case, the two sets are the two clustering solutions from consecutive iterations. Then, we define a cluster as a graph vertex and the number of overlapping samples in two vertices (one in each of the two clustering partitions) as the graph edge weight. Maximum weighted bipartite matching finds a subset of the edges where no two edges share a common vertex and maximizes the sum of edge weights. In our case, the two sets have the same number of vertices (K) and each vertex has precisely one edge in the optimal matching.

Let B be a Boolean matrix to represent the matching where $B_{i,j} = 1$ if cluster i in a is matched to cluster j in b. The optimal matching is formulated by maximizing the objective function:

$$\begin{aligned} &\max \sum_i \sum_j A_{i,j}B_{i,j},\ i,j \in 1,2,...,K, \\ &A_{i,j} = \sum_n \mathbb{1}\{a_n = i \cap b_n = j\},\ n \in 1,2,...,N, \end{aligned} \quad [5]$$

where A is the matching matrix (a.k.a. confusion matrix in supervised learning) between the two solutions a and b, and $\mathbb{1}\{\}$ is the indicator function.

In each iteration of DISCA, the estimated labels are assigned on model fitting solutions to the Gaussian mixture models. Due to the reasons mentioned above, in DISCA, the clustering solution from one iteration needs to be matched with the previous clustering solution to stabilize the training. We apply the Hungarian algorithm (11) to optimize the objective function (Eq. 5), which is guaranteed to find a global optimum in polynomial time. Then, the current labels are permuted according to the matching to achieve the highest consistency with the labels in the previous iteration.

Missing wedge effect. A major cryo-ET limitation, the missing wedge effect, must be considered when designing analysis methods (12). In cryo-ET imaging, cell samples are imaged through a series of tilt projections. The tilt projections are subsequently fed into a reconstruction algorithm to produce a 3D tomographic reconstruction. Because of the increasing effective sample thickness during tilting, to prevent excessive electron beam damage to the cell sample, the tilt angle range is limited typically to $\pm 60^\circ$ with a 1° step size. This results in a double V-shaped missing value region of Fourier coefficients of

the reconstructed tomogram in Fourier space. The missing wedge effect also produces image distortion in the spatial domain; for instance, it may elongate features along the direction of the missing wedge axis.

DISCA tackles the missing wedge effect from two aspects. First, in our previous work (2), we have empirically demonstrated the robustness of CNN feature extraction to image distortions caused by the missing wedge effect. Moreover, the robustness of YOPO feature extraction to image noise and distortion is further improved by the Gaussian dropout layer. Second and most importantly, during the self-supervision step, when a subtomogram is rotated, the direction of the image distortion caused by the missing wedge effect rotates correspondingly. By enforcing the rotated copy to have the same label and thus similar extracted feature vectors during YOPO training, we explicitly increase the robustness of YOPO feature extraction to the missing wedge effect from various angles. In the results section, we showed that DISCA can still perform well on simulated datasets of large missing wedge (tilt-angle range $\pm 40^\circ$) and various SNR, thus demonstrating the robustness of DISCA to the missing wedge effect.

Since the missing wedge effect is also affecting other data processing steps, we note that it can also be treated in those steps despite that it is out of the scope of DISCA. Before feeding into DISCA, the tomograms can be reconstructed by algorithms compensating for the missing wedge effect such as Weighted BackProjection for better particle picking. In the postprocessing step, subtomogram averaging using *Relion* (13) involves missing wedge compensation from model estimation, whereas structural pattern re-embedding by Gum-Net (14) uses a spectral data imputation technique to reduce the missing wedge effect on subtomogram alignment. Other subtomogram alignment methods that consider the missing wedge effect can also be applied.

We conducted an experiment to show the effectiveness of missing wedge compensation techniques for pre-processing the tomograms. As observed in **Fig. 5B**, the membrane structure parallel to the x-axis is affected by the missing wedge effect and is presented with weaker signals. It is likely that the DoG picker did not select the membrane feature of that region which resulted in the missing detection by DISCA. We used the most recent missing wedge compensation method *IsoNet* (15) to pre-process the reconstructed tomograms in the *Synechosystis* dataset and performed DISCA again. The resulting detection of membrane features was re-embedded, Gaussian smoothed, and visualized in **Fig. S8**. The missing wedge compensation pre-processing step reduced the missing wedge effect and improved the detection of membrane features in affected regions.

Preferred orientations. Some of the ribosome subtomogram averages, especially from the *Cercopithecus aethiops* kidney cell dataset, are of lower quality than others. We then investigate whether the detected ribosomes exhibit preferred orientations. In **Fig. S17A**, we visualize the orientation of ribosomes from *Relion* subtomogram averaging output. The orientation of ribosome in each subtomogram is transformed as a 3D unit vector. If there is no preferred orientation, the vectors should distribute randomly on the unit sphere. Preferred orientation is not obvious on any datasets except clearly on the *Cercopithecus aethiops* kidney cell dataset. The preferred orientation on the *Cercopithecus aethiops* kidney cell dataset is likely to cause its low averaging quality. Preferred orientation can be caused either in the DISCA detection step or in the post-processing subtomogram averaging step. We use the pose normalization technique described in (2) to estimate the orientations directly using PCA and plotted the results in **Fig. S17B**, which suggests that the preferred orientation on the *Cercopithecus aethiops* kidney cell dataset is likely to be caused by the post-processing step due to the low quantity of ribosomes and low SNR of the dataset. From a methodology perspective, DISCA detection should be robust to different orientations as the self supervision step enforce the same features to be extracted from the same structure of different orientations.

Time cost and complexity analysis. Currently, there are more than 100 TB of cryo-ET data in public repositories such as EMDB (16), ETDB (17), and EMPIAR (18). With the fast accumulation of cryo-ET data, it is necessary to have high-throughput analysis algorithms. We now show theoretically that DISCA can achieve an overall time complexity of $O(N)$, and therefore our framework scales well to large datasets. This leads to the following theorem.

Theorem 1. *When m, the number of iterations, K, the number of clusters, and P, the dimension of the feature space, are held constant and are relatively small compared to N, the number of entries in the dataset, the time complexity of DISCA is $O(N)$.*

Proof. In each of m iterations, the algorithm performs feature extraction by YOPO, estimates the number of components, fits mixed multivariate Gaussian distributions to the extracted features, matches clustering solutions, validates clustering solutions, and trains the YOPO network using current estimated labels. The deep learning process to extract features takes time $O(N)$. Estimating K using BIC takes time $O(K)$. Statistical model fitting takes time $O(NKP^2)$ using the FIGMN algorithm (19). In the matching stage, the Hungarian algorithm takes time $O(K^3)$ (11). Finally, when validating clustering solutions, calculating the distortion-based DBI takes time $O(N)$.

Therefore, the total time complexity is $O(m(N + K + NKP^2 + K^3 + N))$, but because m, K, and P are constant, the overall computational complexity of DISCA is $O(N)$. □

In terms of sample complexity, we leverage work by (20) that has shown that $\tilde{\Theta}(KP^2/\epsilon^2)$ samples are both necessary and sufficient for learning mixed multivariate Gaussian distributions with K components in a P-dimensional feature space with up to ϵ error in total variation distance. This result implies that learning reasonably accurate models that achieve a low,

constant error ϵ requires relatively few samples in practice, as K and P are assumed to be small compared to N in large-scale datasets.

Practically, on our computer with 4 GPUs and 48 CPU cores, the pre-processing template-free particle picking step takes less than 20 minutes to pick 100,000 to 200,000 subtomograms from a dataset of more than 10 tomograms. Training DISCA from scratch to sort these subtomograms takes less than 10 hours. When our clustering model is properly trained, the prediction on new data is very fast, which takes less than an hour to process millions of subtomograms. Since data parallelism is used for training on multiple GPUs, with limited computational resources, such as one GPU instance, the computing time would approximately be 4 times longer. The memory storage for training neural networks can be effectively adjusted by changing the batch size.

Before the subtomogram averaging step, the cluster centers of extracted features can optionally be decoded to select interesting clusters for thorough downstream analysis. The post-processing subtomogram averaging step using *Relion* (13) takes less than two days to achieve resolution better than 40 Å. Here, we use 'subtomogram averaging' to refer to the averaging process to recover a single class and 'subtomogram classification' to refer to averaging and classification process to recover multiple classes which are more time-consuming. By comparison, the template matching approach on the same computer equipment would take roughly one to two months to complete, which requires visual inspection by experts, computational template matching, and subtomogram classification.

Implementation details

The neural network model YOPO was implemented in platform Keras (21) with Tensorflow backend (22). No external pre-trained models or additional supervision were used. Orthogonal kernel initializer and zero bias initializer were used. All models were trained on a computer with 4 NVIDIA GeForce Titan X Pascal GPU instances and 48 CPU cores. In terms of memory cost, the RAM can be monitored by varying the batch size during neural network training. It is not necessary to have multiple GPU instances and CPU cores in order to run DISCA. The statistical model fitting used functions in Python package *numpy* and *sklearn*. The implementation of the Hungarian algorithm used functions in Python package *scipy*. The data augmentation used random 3D rotation functions implemented in *AITom* (23). During the YOPO model training, the label smoothing factor gradually decreases by a factor of 0.9 in each iteration as we expect the amount of mislabeled data to decrease over time, and therefore YOPO becomes more certain about its prediction over time. During the Gaussian mixture model fitting, the extracted features are dimension reduced by PCA to a length of 16 as an optional step for faster clustering. To measure the convergence of DISCA, a generalized EM framework, we set two stopping criteria: (1) the estimated K and the vast majority (99%) of the estimated labels stay the same for three consecutive iterations, or (2) the maximum number of iterations has been reached. The template matching baseline on experimental datasets were performed using *PyTom* (24).

Preprocessing. For template-free particle picking, we applied the 3D Difference of Gaussians (DoG) (25) volume transform algorithm implemented in *AITom*. 3D DoG first computes a map I_{DoG} by subtracting two Gaussian blurred versions of the input tomogram v using the Gaussian function I with different standard deviations σ_1 and σ_2, where, without loss of generality, $\sigma_1 > \sigma_2$. The 3D DoG map is computed on tomogram v as $I_{DoG} = I_v(\sigma_1) - I_v(\sigma_2)$.

Local maxima are detected to extract a set of subtomograms S from v as:

$$S = \left\{ s \in v \;\middle|\; \frac{dI_{DoG}(s)}{ds} = 0,\ \frac{d^2 I_{DoG}(s)}{ds^2} < 0,\ I_{DoG}(s) > C \right\}, \qquad [6]$$

where s is a 3D location in I_{DoG} and C is a threshold applied for selecting local peaks. In our implementation, we ensured a minimum distance of 15 voxels between two peaks by filtering out peaks with low values. We note that the minimum distance should be adjusted for tomograms with larger voxel spacing or crowded structures. The input to the DoG particle picking step is a set of reconstructed 3D tomograms. Optionally, denoising or missing wedge compensation algorithms (26**?** –28) can be applied to the tomograms before performing particle picking and DISCA sorting. The *Mycoplasma pneumoniae* (29) dataset was denoised using *Warp* (30) whereas other simulated and experimental datasets were not denoised, which showed that DISCA is relatively invariant to the denoising preprocessing step.

Postprocessing. For subtomograms in each structurally homogeneous subset obtained from DISCA, iterative 3D averaging was performed using *Relion 3.0* (31). As a template-and-label-free framework, we did not use any external structural templates in the averaging process. The initial averages were obtained by our unsupervised deep learning based subtomogram alignment method Gum-Net (implemented in AITom) (14, 23). After the 3D averaging process, the subtomogram averages were re-embedded into the original tomogram by Gum-Net for visualization purposes. The resolution of the subtomogram averages was estimated using *Relion 3.0* function.

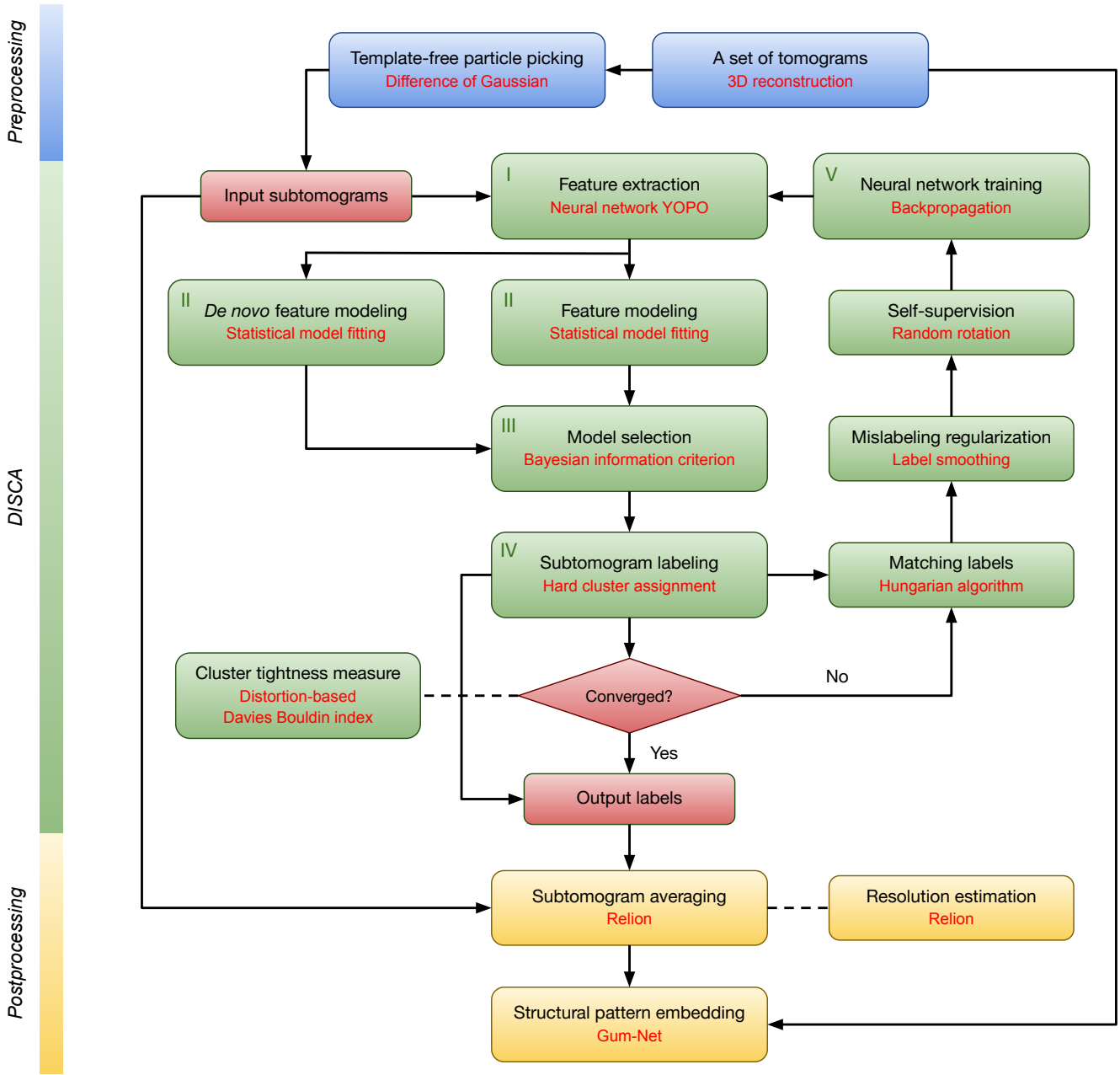

Fig. S1. The DISCA workflow for cryo-ET structural pattern mining. Key steps are numbered. The preprocessing and postprocessing steps are included here for an overview of the processing pipeline. They are not part of the proposed method DISCA.

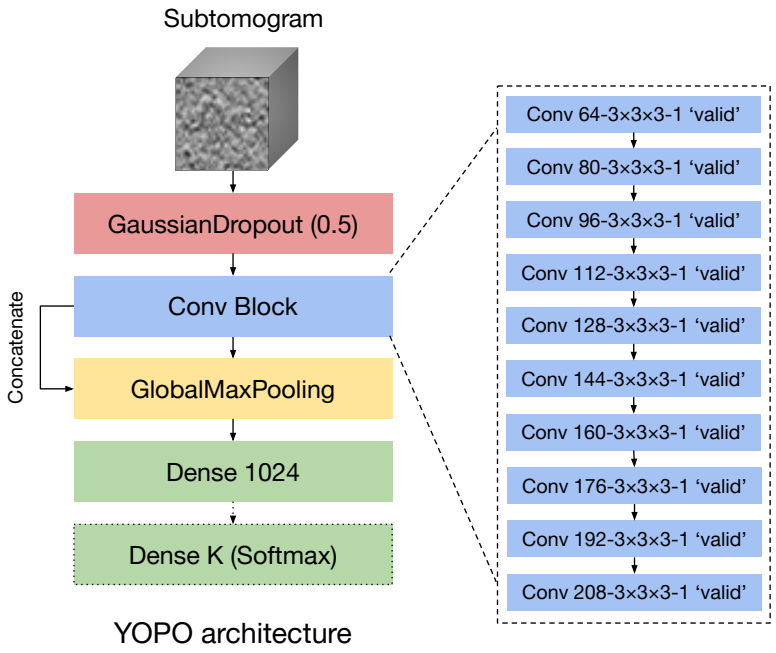

Fig. S2. The architecture of YOPO (You Only Pool Once) model. Each colored box denotes one layer in the neural network. 'GaussianDropout (0.5)' denotes a dropout layer with a dropout rate of 0.5 and multiplicative 1-centered Gaussian noise. 'Conv 64-3x3x3-1 'valid" denotes a convolutional layer with 64 channels, kernel size $3 \times 3 \times 3$, strides of size 1, and valid padding (no padding). Each convolutional layer is equipped with an exponential linear unit activation function and batch normalization. 'Concatenate' denotes concatenated feature outputs. 'Dense K (Softmax)' denotes a fully connected layer with K neurons. As a feature extraction network, the last classification layer of YOPO is only used during model training. The extracted features are the output from the 'Dense 1024' layer.

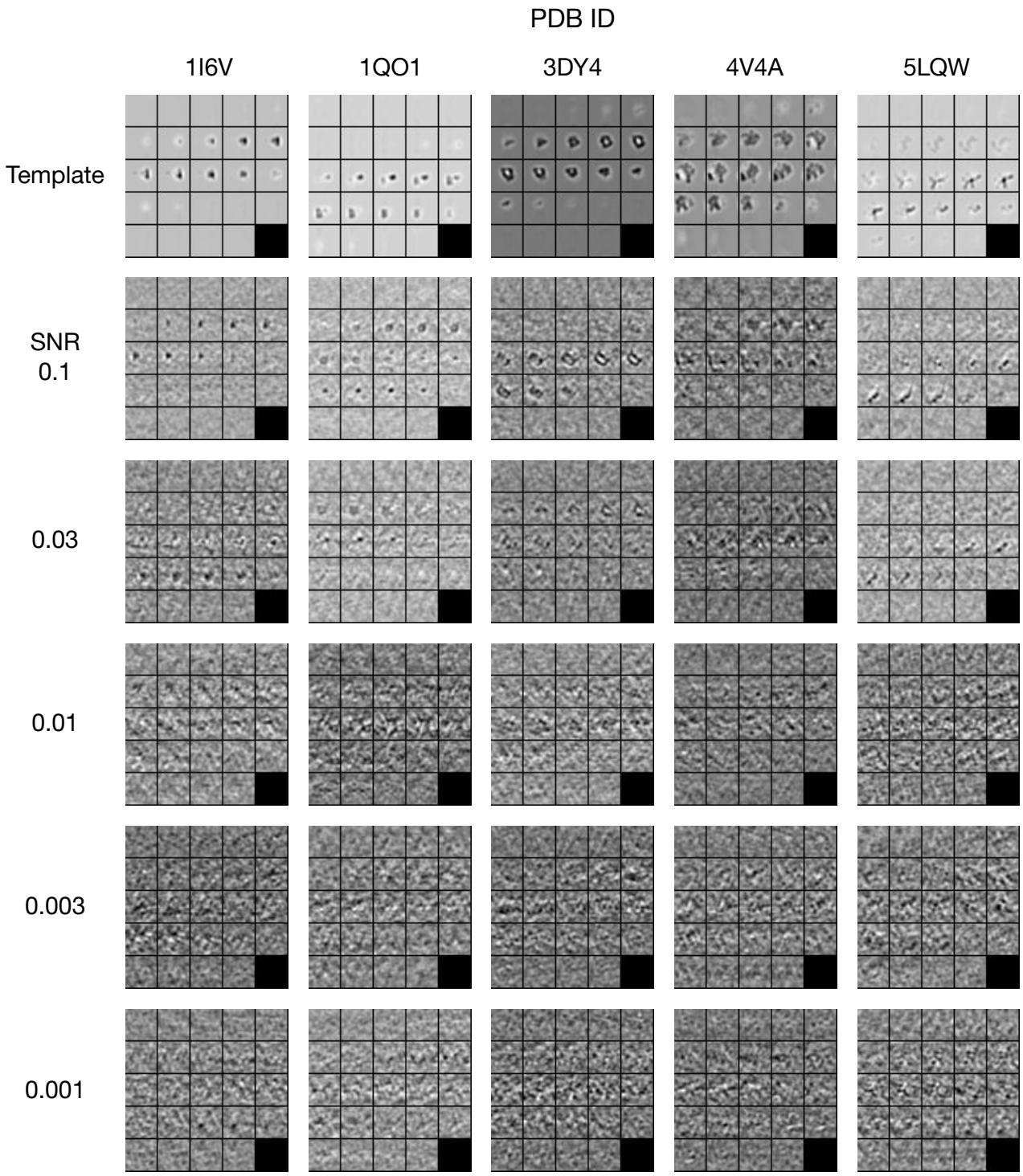

Fig. S3. 2D slice visualization of the template and example subtomograms in the simulated datasets in Table 1 with 30° missing wedge: 1I6V (RNA polymerase, 0.3 MDa), 1QO1 (rotary motor in ATP synthase, 0.4 MDa), 3DY4 (proteasome, 0.7 MDa), 4V4A (ribosome 2.1 MDa), 5LQW (spliceosome, 2.3 MDa).

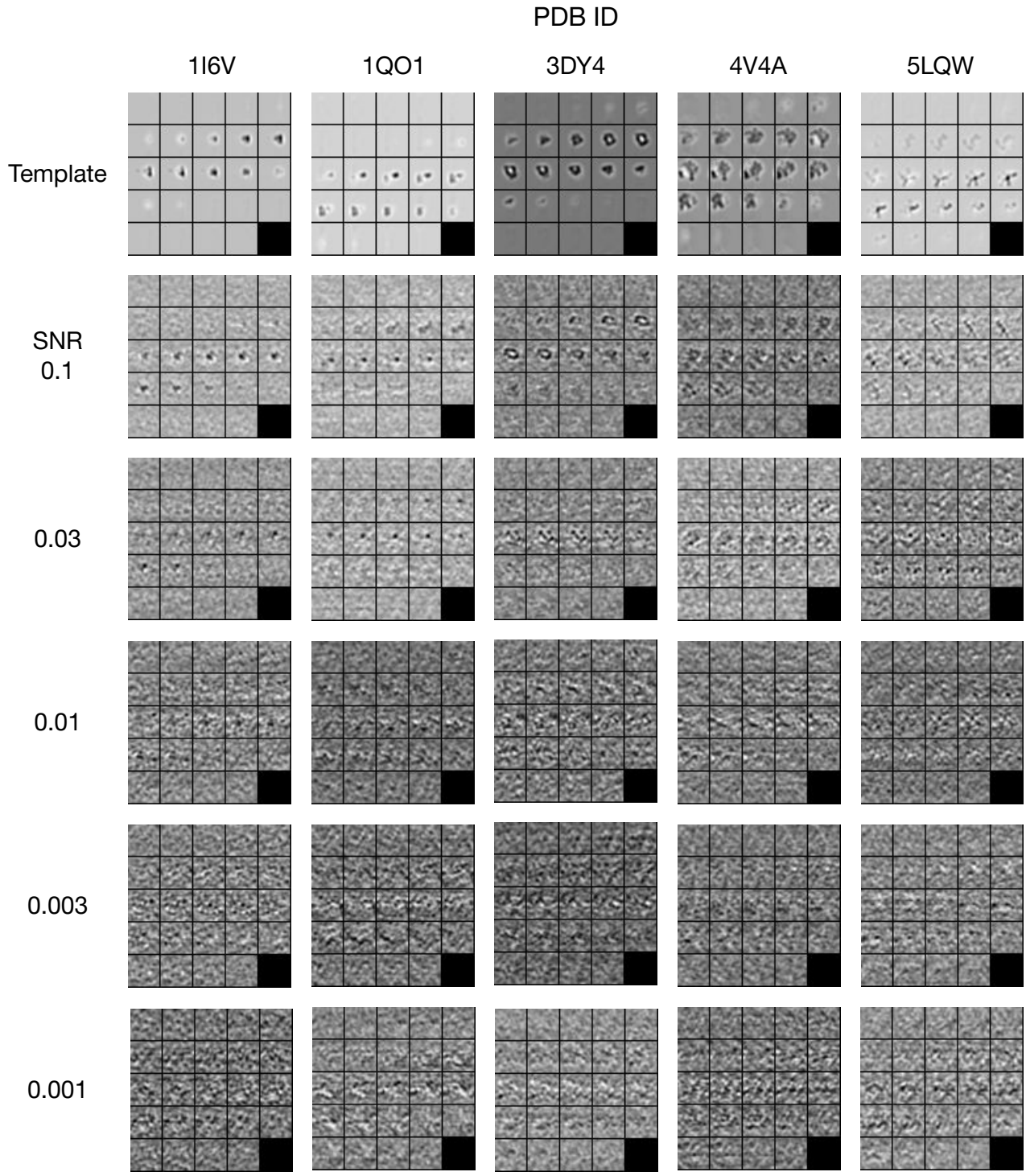

Fig. S4. 2D slice visualization of the template and example subtomograms in the simulated datasets in Table 1 with 50° missing wedge: : 1I6V (RNA polymerase, 0.3 MDa), 1QO1 (rotary motor in ATP synthase, 0.4 MDa), 3DY4 (proteasome, 0.7 MDa), 4V4A (ribosome 2.1 MDa), 5LQW (spliceosome, 2.3 MDa).

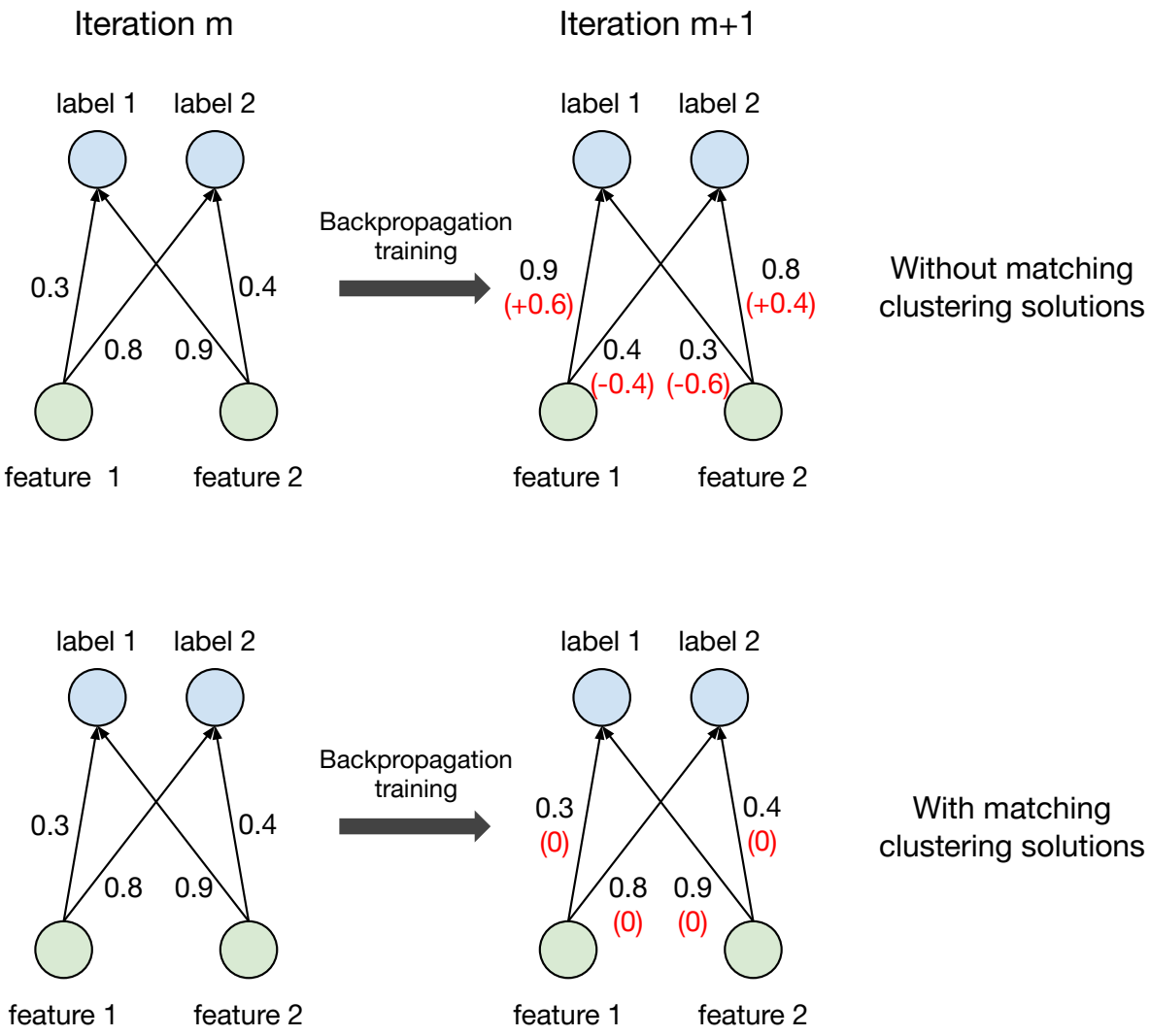

Fig. S5. We assume the last fully connected layer for classification has two input feature nodes and two output label nodes. And we assume the clustering solution has two clusters 1 and 2 with labels flipped from iteration m to iteration m+1. Without matching clustering solutions, the backpropagation training needs to re-learn (large changes in weights) the already optimized weights to correctly output the flipped labels. This will cause strong instability during training. However, with matching clustering solutions, the already optimized weights no longer need to be re-learned (no change in weights).

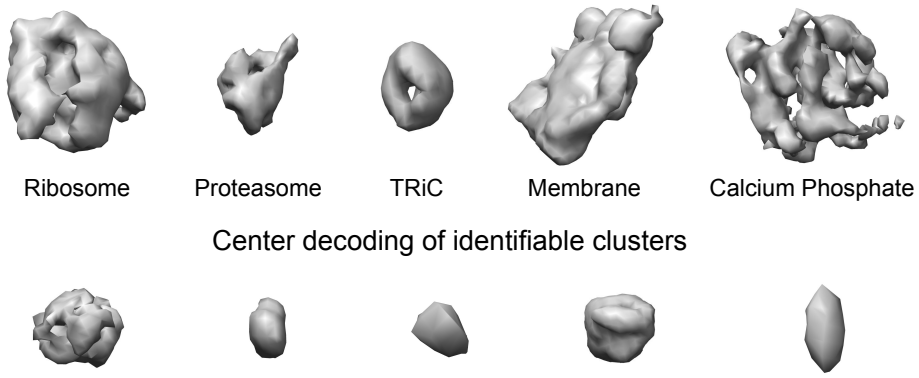

Fig. S6. Example decodings of cluster centers from the *Rattus* neuron dataset.

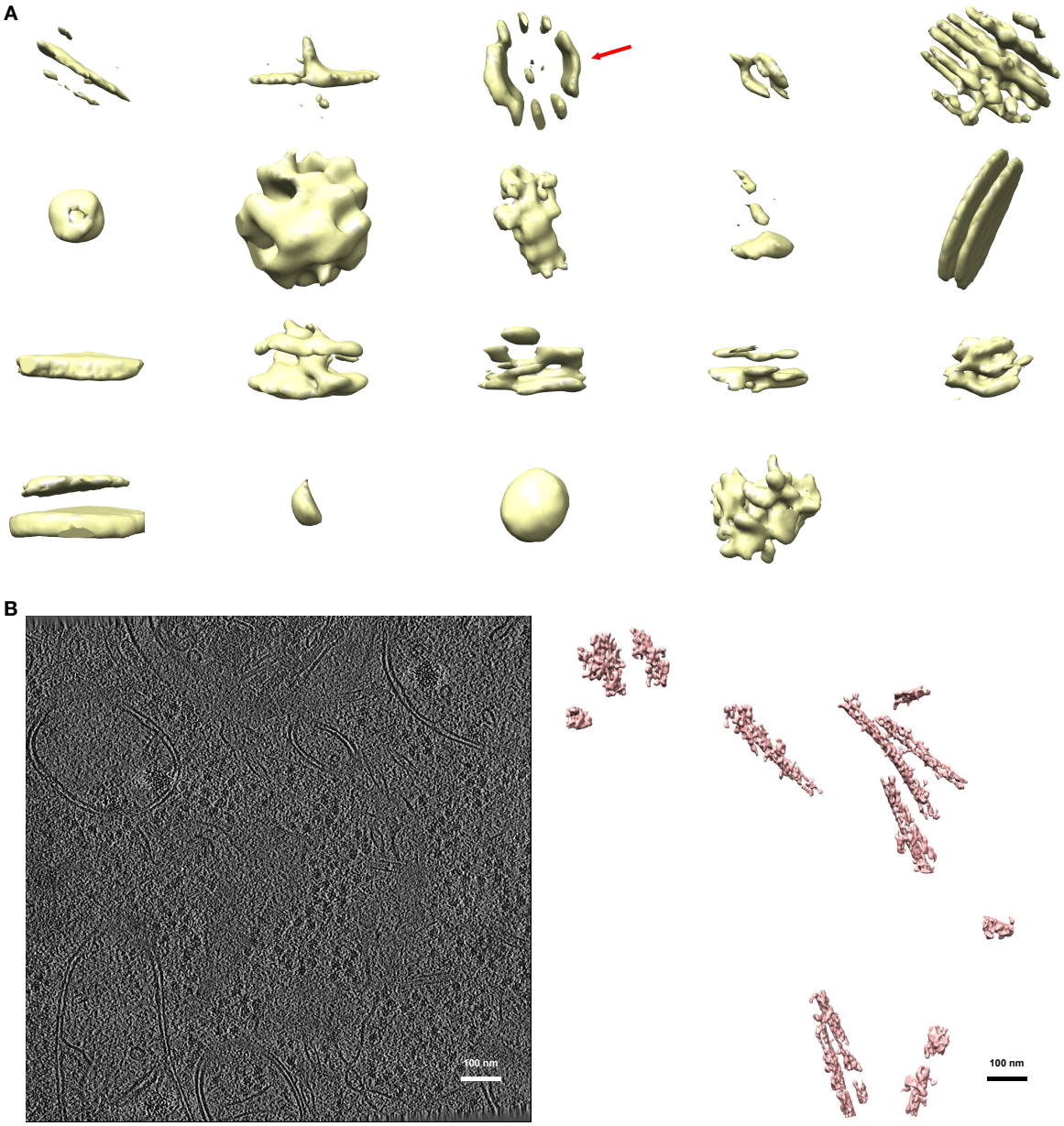

Fig. S7. A. Isosurface representation of subtomogram averages by DISCA sorting and *Relion 3.0* single-class averaging on the *Rattus* neuron dataset. In addition, DISCA may have detected a microtubule-resembling structure as pointed out by the red arrow. **B.** We have re-embedded, Gaussian smoothed, and visualized the cluster corresponding to the microtubule-like structures for more information. The detected structures in the middle are likely to be true-positive microtubules whereas the structures in the top left are likely to be false positives.

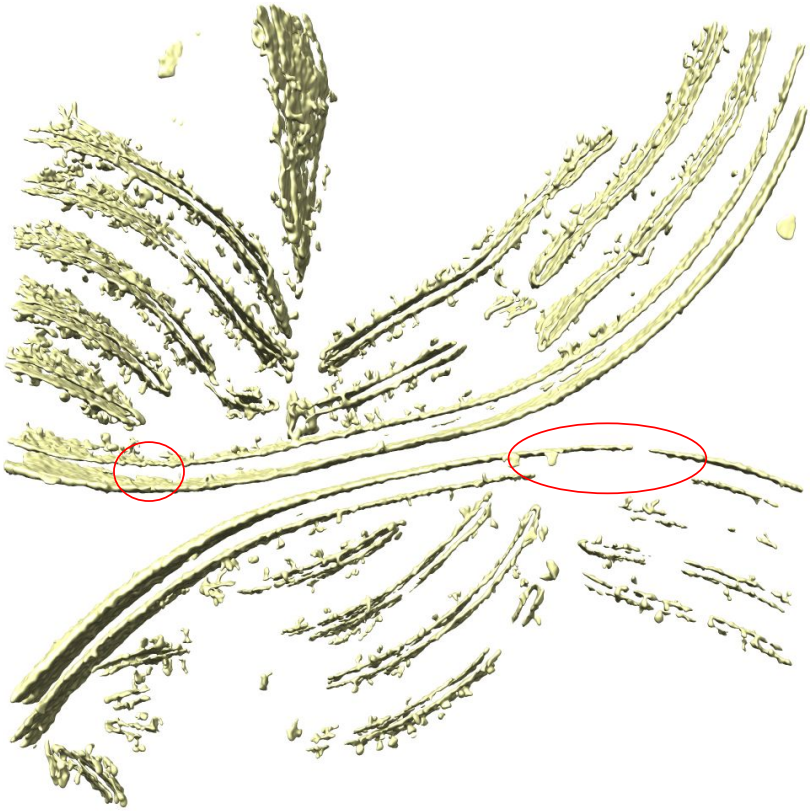

Fig. S8. Detection of membrane features in the *Synechocystis* dataset after *IsoNet* (15) pre-processing. The missing membrane features in **Fig. 5B** parallel to the x-axis are now detected and highlighted by the red circles.

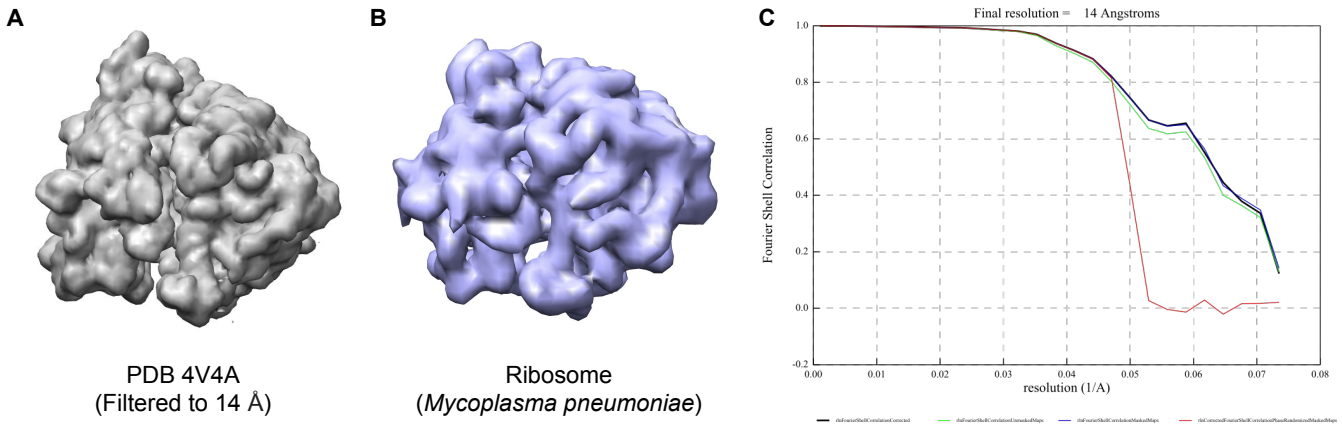

Fig. S9. Ribosome subtomogram average from the *Mycoplasma pneumoniae* dataset. The Pearson's correlation coefficient between the existing structure from PDB and the subtomogram average is 0.91.

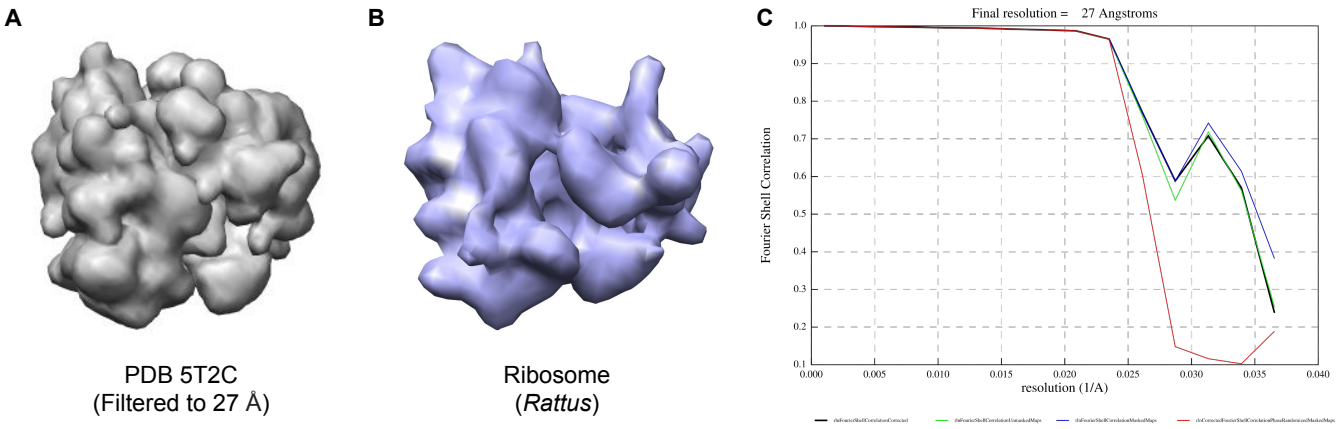

Fig. S10. Ribosome subtomogram average from the *Rattus* neuron dataset. The Pearson's correlation coefficient between the existing structure from PDB and the subtomogram average is 0.83.

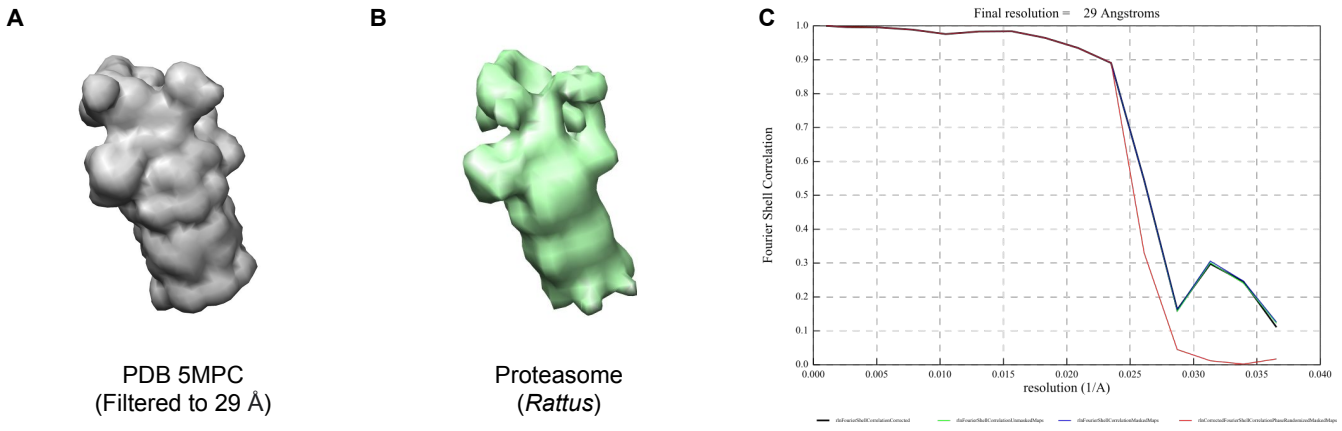

Fig. S11. 26S proteasome subtomogram average from the *Rattus* neuron dataset. The Pearson's correlation coefficient between the existing structure from PDB and the subtomogram average is 0.90.

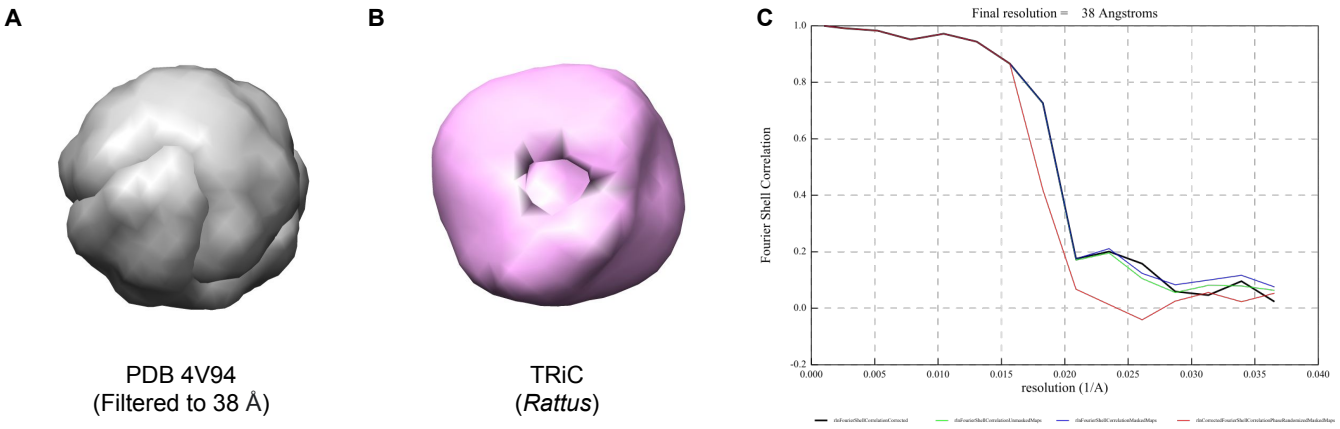

Fig. S12. TRiC subtomogram average from the *Rattus* neuron dataset. The Pearson's correlation coefficient between the existing structure from PDB and the subtomogram average is 0.93.

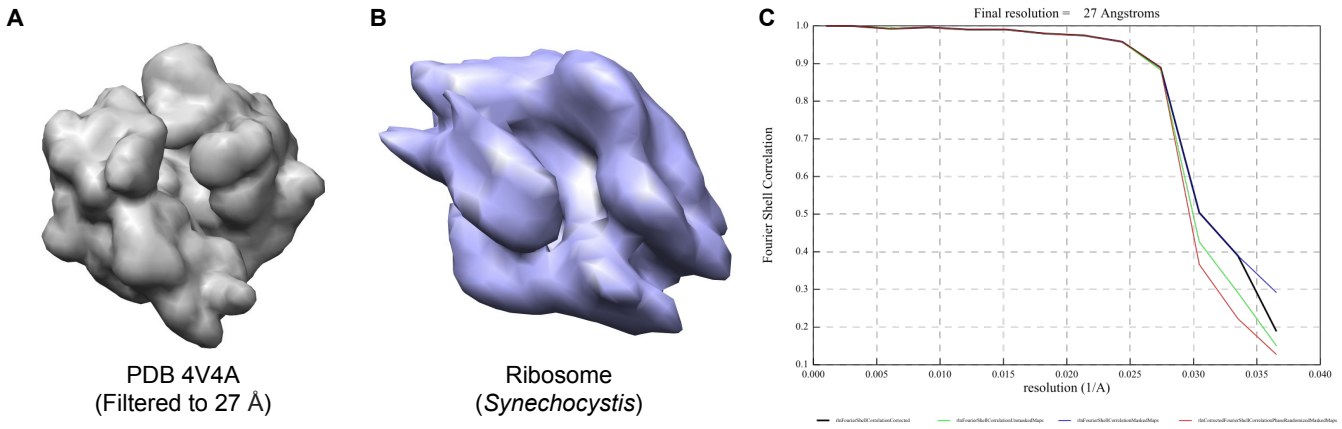

Fig. S13. Ribosome subtomogram average from the *Synechosystis* dataset. The Pearson's correlation coefficient between the existing structure from PDB and the subtomogram average is 0.78.

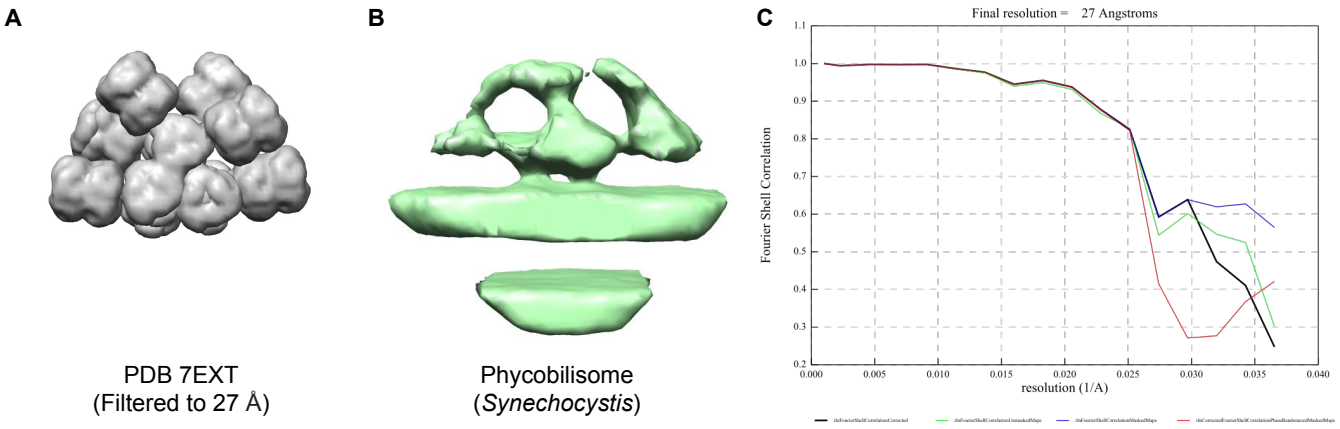

Fig. S14. Phycobilisome array (membrane-bounded) subtomogram average from the *Synechosystis* dataset. The Pearson's correlation coefficient between the existing structure from PDB and the subtomogram average is 0.75 (excluding bounded membrane).

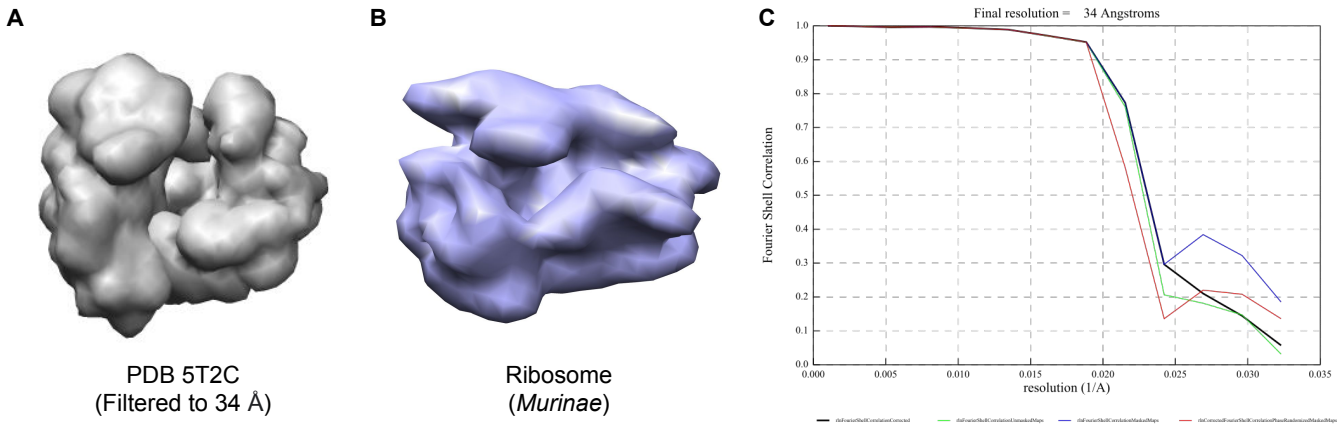

Fig. S15. Ribosome subtomogram average from the *Murinae* embryonic fibroblast dataset. The Pearson's correlation coefficient between the existing structure from PDB and the subtomogram average is 0.96.

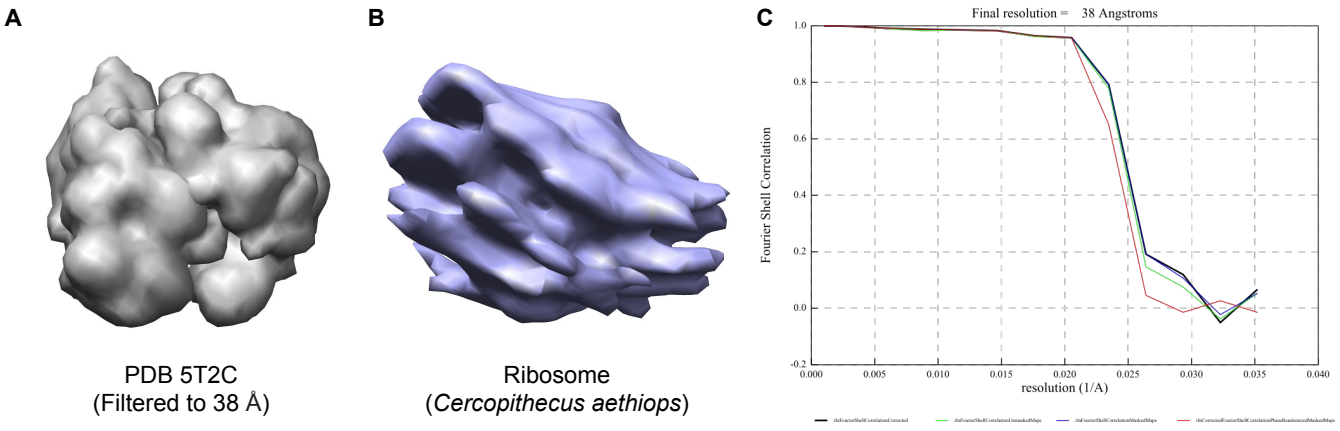

Fig. S16. Ribosome subtomogram average from the *Cercopithecus aethiops* kidney cell dataset. The Pearson's correlation coefficient between the existing structure from PDB and the subtomogram average is 0.77.

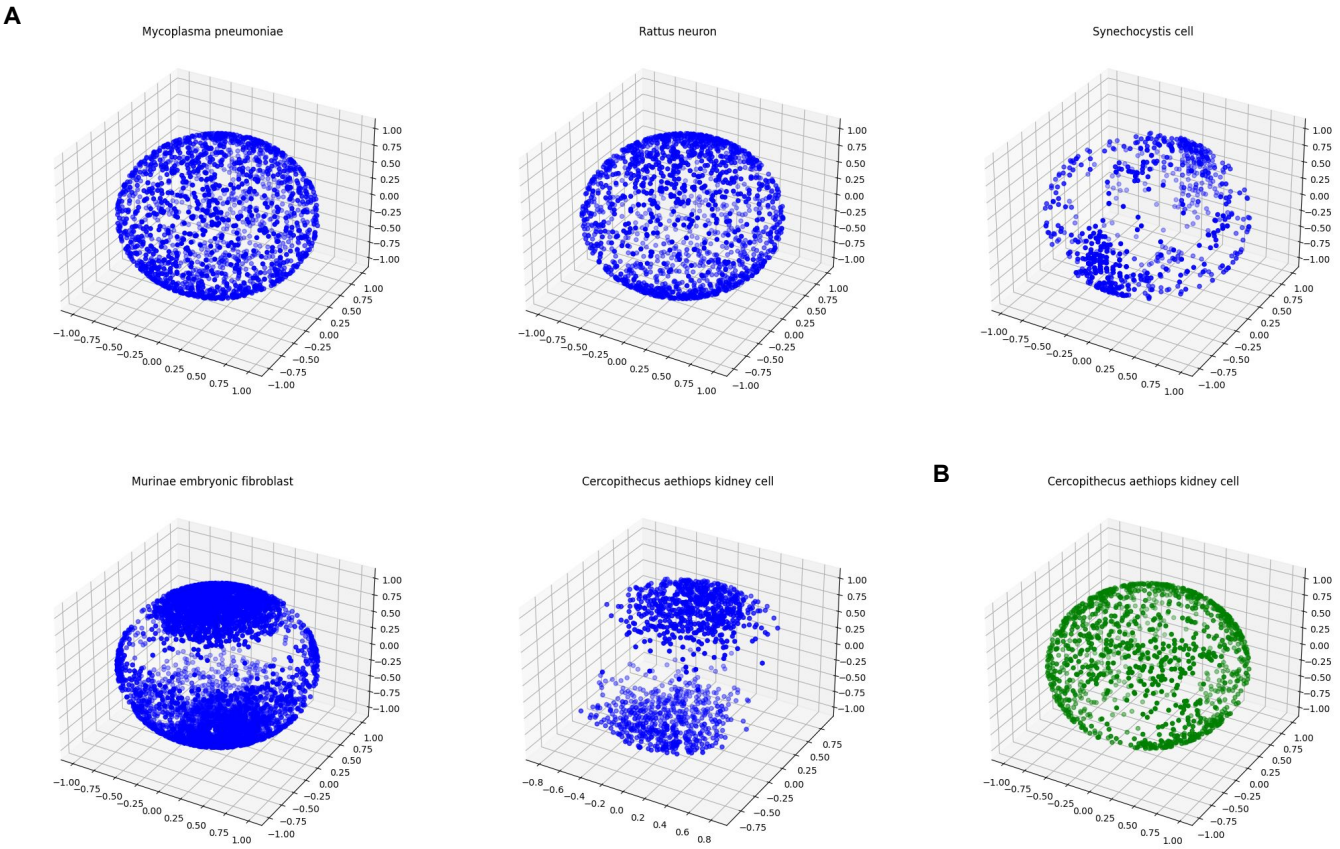

Fig. S17. A. Orientation of ribosomes in subtomogram averaging. Each dot denotes a ribosome detected by DISCA for the corresponding dataset. The orientation is visualized as the position of the orientation vector on the unit sphere. **B.** Ribosome orientation estimation on the *Cercopithecus aethiops* kidney cell dataset using pose normalization (2).

References

1. Q Guo, et al., In situ structure of neuronal c9orf72 poly-ga aggregates reveals proteasome recruitment. *Cell* **172**, 696–705 (2018).
2. X Zeng, MR Leung, T Zeev-Ben-Mordehai, M Xu, A convolutional autoencoder approach for mining features in cellular electron cryo-tomograms and weakly supervised coarse segmentation. *J. structural biology* **202**, 150–160 (2018).
3. D Bank, N Koenigstein, R Giryes, Autoencoders. *arXiv preprint arXiv:2003.05991* (2020).
4. Y Ren, et al., Deep clustering: A comprehensive survey. *arXiv preprint arXiv:2210.04142* (2022).
5. HM Berman, et al., The protein data bank. *Nucleic acids research* **28**, 235–242 (2000).
6. G Schwarz, , et al., Estimating the dimension of a model. *The annals statistics* **6**, 461–464 (1978).
7. H Akaike, A new look at the statistical model identification. *IEEE transactions on automatic control* **19**, 716–723 (1974).
8. T Caliński, J Harabasz, A dendrite method for cluster analysis. *Commun. Stat. Methods* **3**, 1–27 (1974).
9. WJ Krzanowski, Y Lai, A criterion for determining the number of groups in a data set using sum-of-squares clustering. *Biometrics* pp. 23–34 (1988).
10. GM James, CA Sugar, Clustering for sparsely sampled functional data. *J. Am. Stat. Assoc.* **98**, 397–408 (2003).
11. HW Kuhn, The hungarian method for the assignment problem. *Nav. research logistics quarterly* **2**, 83–97 (1955).
12. A Bartesaghi, et al., Classification and 3d averaging with missing wedge correction in biological electron tomography. *J. structural biology* **162**, 436–450 (2008).
13. SH Scheres, Relion: implementation of a bayesian approach to cryo-em structure determination. *J. structural biology* **180**, 519–530 (2012).
14. X Zeng, M Xu, Gum-net: Unsupervised geometric matching for fast and accurate 3d subtomogram image alignment and averaging in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. pp. 4073–4084 (2020).
15. YT Liu, et al., Isotropic reconstruction for electron tomography with deep learning. *Nat. Commun.* **13**, 1–17 (2022).
16. CL Lawson, et al., Emdatabank. org: unified data resource for cryoem. *Nucleic acids research* **39**, D456–D464 (2010).
17. DR Ortega, et al., Etdb-caltech: A blockchain-based distributed public database for electron tomography. *PloS one* **14**, e0215531 (2019).
18. A Iudin, PK Korir, J Salavert-Torres, GJ Kleywegt, A Patwardhan, Empiar: a public archive for raw electron microscopy image data. *Nat. methods* **13**, 387–388 (2016).
19. RC Pinto, PM Engel, A fast incremental gaussian mixture model. *PLOS ONE* **10**, 1–12 (2015).
20. H Ashtiani, et al., Nearly tight sample complexity bounds for learning mixtures of gaussians via sample compression schemes in *Advances in Neural Information Processing Systems*. pp. 3412–3421 (2018).
21. F Chollet, , et al., Keras: The python deep learning library. *Astrophys. Source Code Libr.* pp. ascl–1806 (2018).
22. M Abadi, et al., Tensorflow: A system for large-scale machine learning in *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*. pp. 265–283 (2016).
23. X Zeng, M Xu, Aitom: Open-source ai platform for cryo-electron tomography data analysis. *arXiv preprint arXiv:1911.03044* (2019).
24. T Hrabe, et al., Pytom: a python-based toolbox for localization of macromolecules in cryo-electron tomograms and subtomogram analysis. *J. structural biology* **178**, 177–188 (2012).
25. D Woolford, B Hankamer, G Ericksson, The laplacian of gaussian and arbitrary z-crossings approach applied to automated single particle reconstruction. *J. structural biology* **159**, 122–134 (2007).
26. Z Yang, F Zhang, R Han, Self-supervised cryo-electron tomography volumetric image restoration from single noisy volume with sparsity constraint in *Proceedings of the IEEE/CVF International Conference on Computer Vision*. pp. 4056–4065 (2021).
27. E Moebel, C Kervrann, A monte carlo framework for missing wedge restoration and noise removal in cryo-electron tomography. *J. Struct. Biol. X* **4**, 100013 (2020).
28. TO Buchholz, M Jordan, G Pigino, F Jug, Cryo-care: content-aware image restoration for cryo-transmission electron microscopy data in *2019 IEEE 16th International Symposium on Biomedical Imaging (ISBI 2019)*. (IEEE), pp. 502–506 (2019).
29. FJ O'Reilly, et al., In-cell architecture of an actively transcribing-translating expressome. *Science* **369**, 554–557 (2020).
30. D Tegunov, P Cramer, Real-time cryo-electron microscopy data preprocessing with warp. *Nat. methods* **16**, 1146–1152 (2019).
31. J Zivanov, et al., New tools for automated high-resolution cryo-em structure determination in relion-3. *elife* **7**, e42166 (2018).