RobPicker: A Meta Learning Framework for Robust Identification of Macromolecules in Cryo-ET

Anonymous Author(s)
Affiliation
Address
email

Abstract

Accurate particle picking of macromolecules of all kinds, shapes, and sizes in cryogenic electron tomograms (cryo-ET) is critical for understanding the molecular architecture of biological systems in their native state. Current deep learning methods have shown potential in identifying macromolecules from tomograms, but they are vulnerable to issues of noise in the training dataset obtained through human or automatic labeling and imbalanced distribution of macromolecule species. To address these limitations, we developed RobPicker, a meta-learning framework that effectively mitigates these issues by automatically learning deep neural networks to correct label errors and give greater emphasis to underrepresented macromolecule species. In evaluations across diverse cryo-ET datasets with noisy labels and imbalanced species distributions, RobPicker substantially outperforms state-of-the-art methods, particularly in identifying small and rare macromolecules. The efficiency and robustness of RobPicker can also be used for rapid fine-tuning of the tilt-series alignment, leading to improved tomogram reconstruction and enabling high-resolution cellular structural biology analysis.

1 Introduction

Cryo-electron tomography (cryo-ET) has revolutionized structural biology by enabling the high-resolution visualization and structure determination of macromolecules in their near-native states within the complex environment of cells in situ [1–8]. Accurately identifying and locating macromolecules in cryo-ET is crucial for understanding their functions, interactions, and dynamics within the cellular context [9]. By training neural networks on annotated tomograms, recent deep learning methods have shown promise in automatically picking macromolecule complexes (referred to as particles) from cryo-ET data [10–17]. In particular, macromolecule segmentation networks, in particular 3D U-Net [18], have been successfully applied to segment and pick particles such as ribosomes [14, 15]. Yet, these networks often struggle with robustness, particularly facing two key challenges: noise in the training labels brought in by human or automatic annotations and imbalanced distribution of different species [16, 19].

The low signal-to-noise ratio (SNR) of cryo-ET raw data, combined the 'missing wedge' artifact [20], presents significant challenges in detecting and classifying particles in cryo-ET, especially those of smaller size [21]. The challenge is further compounded by the noise in training labels brought in by particle annotations. Automated annotation often results in false positives [22]. While manual annotation is more accurate, it is extremely labor intensive [23]. In summary, particle annotation in cryo-ET is inherently difficult (especially for small particles) and time-consuming. When the noisy labels are used for deep learning model training, the resulting models cannot accurately identify particles when applied to new tomograms (an illustration of the labeling noise is shown in Fig. 3a in the appendix).

Training macromolecule segmentation networks is further impeded by the significant imbalance of the occurrence of different macromolecular species within the cellular environment [24]. Larger, more abundant species occupy more voxels in the tomograms and tend to have a clearer boundary against the background, while smaller or less abundant species are less represented and occupy fewer voxels. As training macromolecule segmentation networks mostly relies on voxel-wise loss functions—such as Dice loss [25, 26]—to update the model parameters, larger and more abundant species have a larger influence on the model update [27]. This imbalance leads to a skewed performance where models are proficient at identifying large, well-delineated species but struggle with picking smaller, rarer ones (Fig. 3b). The inability to consistently pick small particles in cryo-ET data has hindered our understanding of smaller macromolecules and their critical roles in biological systems.

To address these limitations, we introduce RobPicker, a novel meta-learning [28] framework that improves the robustness of macromolecule segmentation networks for cryo-ET. Unlike traditional supervised deep learning methods, which rely heavily on vast quantities of well-labeled data, RobPicker is designed to overcome challenges posed by labeling noise and imbalanced species distribution. It is based on a bi-level meta learning framework [29] by automatically learning a label correction network that rectifies labeling errors and a data reweighting network that gives greater emphasis to underrepresented macromolecule species. These two networks are updated by evaluating the performance of the macromolecule segmentation network on a small validation set with cleaner labels and a more balanced species distribution. These corrected and reweighted data are then used to learn a more robust macromolecule segmentation network (Fig. 1).

Through comprehensive evaluations of both experimental cryo-ET data and benchmarks, RobPicker demonstrates significantly improved robustness in identifying particle species in various cell types even when the labels are noisy and the species distribution is imbalanced. Moreover, we demonstrated the effectiveness of RobPicker by utilizing it to pick a set of initial ribosome particles efficiently and robustly, which were further used for quick multiple-particle refinement [30], enabling a fast fine-tuning of the alignment of tilt-series.

2 Methods

RobPicker is composed of three deep neural networks: a macromolecule segmentation network, a label correction network, and a data reweighting network. The macromolecule segmentation network takes a cryo-electron tomogram as input and outputs the detection result in the form of a segmentation map, which indicates the likelihood of macromolecule species at each voxel in the tomogram. The segmentation map is used to derive the class of species and the center of particles in post processing (Appendix G). While we employed a 3D U-Net [18] as the macromolecule segmentation network—following prior deep learning pickers [14], our method can benefit from using more advanced segmentation network architectures. The label correction network is designed to correct noisy labels in the training data. It is a small 3D U-Net that takes a tomogram and a noisy particle segmentation map as inputs and produces a corrected label (Fig. 1 and Appendix E). The parameters of the label correction network are updated to minimize the segmentation loss function on a separate, smaller validation set with cleaner—compared to the training set—labels [31] (Stage II in Fig. 1). Intuitively, a bad label correction network will result in corrupted supervision signal for the segmentation network to learn in the training stage (Stage I in Fig. 1), which will result in low segmentation performance in the validation stage—thus a high validation loss—since the validation labels are cleaner. Therefore, minimizing the validation loss can update the label correction network towards the direction such that it outputs better supervision signal, i.e., cleaner labels. In parallel to the label correction network, the data reweighting network [32] predicts a weight for each tomogram to increase the influence of smaller or less abundant particle species on the update of the segmentation network. Specifically, the reweighting network takes as input the prediction loss of a tomogram and outputs a scalar weight between 0 and 1, which is multiplied by the prediction loss so that the weight controls the influence—higher weights lead to higher influence—of the tomogram on model parameter update (Stage I in Fig. 1). The reweighting network is implemented as a multi-layer perceptron (MLP) with one hidden layer. Similar to the update of the label correction network, the update of the reweighting network is also guided by minimizing the validation loss on the validation set where balanced data sampling is used to increase the frequency of less abundant species (Appendix F). The mathematical framework of RobPicker and the bi-level optimization is detailed in Appendices C and D.

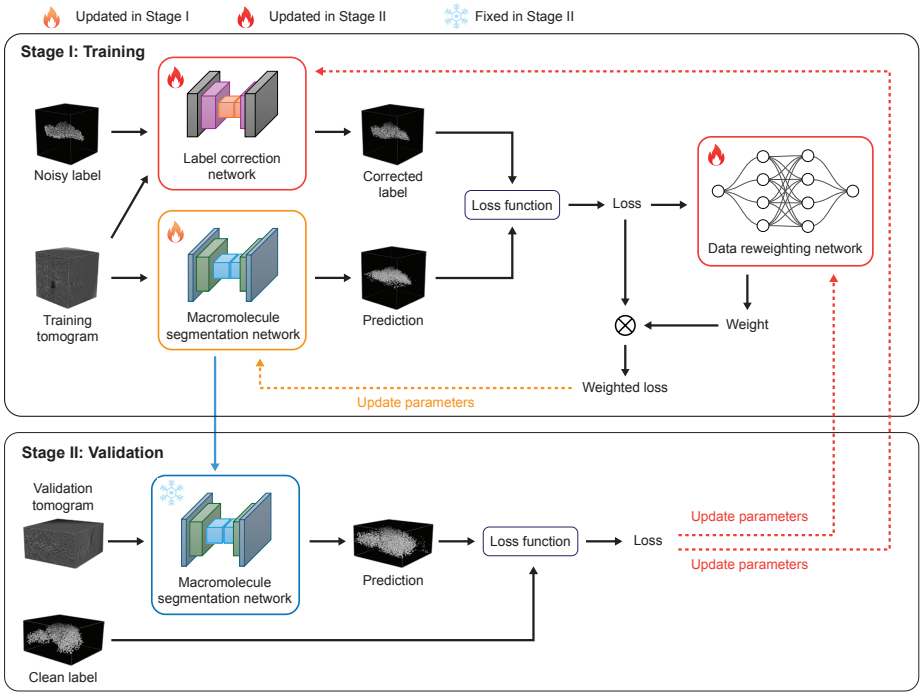


Figure 1: The bi-level optimization framework for training RobPicker.

3 Experiments

3.1 RobPicker tolerates noise in cryo-ET particle picking training data

Typical cryo-ET particle picking datasets contain annotations of particle locations in the form of their 3D coordinates. Following DeepFinder's data preprocessing approach, we applied two methods—a sphere-based and a shape-based—to derive label maps from the location annotations [14]. The shape-based method uses subtomogram averaging [33] to create a mask that mimics the shape of each particle species, and then the mask is placed at each particle location to represent the particles in the label map. The sphere-based method simply uses a sphere mask to represent each particle in the label map, which is more computationally efficient than the shape-based method but resulting in noisier labels. Following the widely used protocol for model training and evaluation [14], we divided our cryo-ET datasets into three subsets: a training set, a validation set, and a test set. We aim to experiment with using the sphere-based method to efficiently generate labels for the majority of the tomograms and examine if RobPicker can tolerate the noise brought in by the sphere-based labels. To this end, we used the sphere-based method to create the label maps for the tomograms in the training set (which has the majority of the tomograms), while we used the shape-based method to create the label maps for the remaining small amount of tomograms and put them in the validation and test sets. The data preprocessing is efficient since subtomogram averaging is only done for the validation and test tomograms.

We conducted experiments on four cryo-ET real datasets (denoted by D1–D4) comprising tomograms with annotations for ribosomes from different cell types. The first dataset (D1) contains annotations for cytosolic ribosomes (ct-ribos) and membrane-bound 80S ribosomes (mb-ribos) in *C. reinhardtii* cells [14] (detailed statistics can be found in Table 1 in the Appendix H). D2 was annotated for 80S ribosomes in *S. cerevisiae* (yeast) cells [34]. D3 contains annotations for 50S large subunits and fully assembled 70S ribosomes of *E. coli* cells. D4 also contains annotations for ribosomes in yeast cells. The details of data annotation can be found in the Appendix H. We compared RobPicker against a state-of-the-art supervised deep learning method DeepFinder [14], which uses a 3D U-Net [18] for multi-class semantic segmentation to pick particles in tomograms. We used the same 3D U-Net as the segmentation network in RobPicker as DeepFinder to demonstrate that

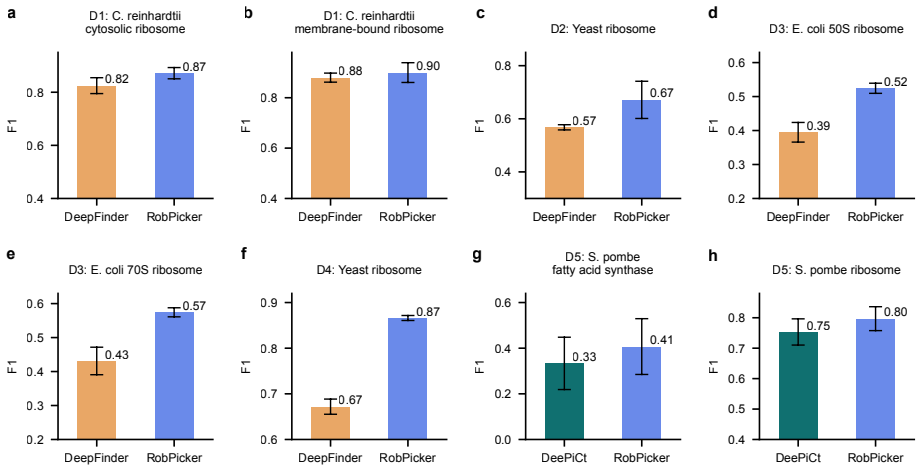


Figure 2: RobPicker achieved higher mean picking F1-scores than DeepFinder and DeePiCt in various cryo-ET real datasets. The mean and standard deviation are shown.

the robustness of RobPicker results from the training framework (Fig. 1) instead of more advanced network architectures. Therefore, the major distinction between RobPicker and DeepFinder is that RobPicker leveraged a data reweighting network and a label correction network to reweight and correct the training data. To report model performance, we calculated the picking F1 scores as the evaluation metric as in DeepFinder. In the evaluation, a particle prediction is considered as a true positive only when the prediction and the ground truth have sufficient overlap (Appendix I). The experiments were repeated three times, and the mean and standard deviation are reported in Fig. 2a–f. The performance comparison on the four real cryo-ET datasets show that RobPicker demonstrated substantially enhanced performance over DeepFinder in picking particles across diverse datasets. The experiments underscore that RobPicker tolerated the label noise in the training set robustly, enabling efficient data preprocessing for the training data, as only a smaller amount of clean labels is needed to calculate the validation loss of RobPicker. In contrast, the supervised learning method DeepFinder [14] achieved lower performance when the training data contains noisy labels created from sphere-based particle masks. The robustness of RobPicker can be attributed to the label correction mechanism in RobPicker, where the sphere-based particle labels in the training set are rectified by the label correction network to better represent the particle shapes, thus resulting in a more accurate macromolecule segmentation model.

3.2 RobPicker robustly picks small and rare macromolecular species

We experimented with a cryo-ET real dataset (denoted by D5) obtained from DeePiCt [15], which contains 10 tomograms acquired from wild-type *S. pombe* using a Volta potential phase plate (VPP). It contains annotations of 731 fatty acid synthases (FAS) and 25,311 ribosomes, and high-quality label masks of the macromolecules were generated for model training and evaluation [15]. We trained DeePiCt and RobPicker to localize the *S. pombe* FAS and ribosomes. FAS segmentation is particularly challenging due to its sporadic presence in cells [15]. We followed the original work [15] to use five-fold cross-validation, splitting the tomograms into five equal-sized subsets. In each fold, one subset was reserved for testing, while the remaining four were used for training. For RobPicker, the training set (8 tomograms) was further divided into a new training set and a validation set in a 7:1 ratio, used for Stage I and Stage II optimizations, respectively (Fig. 1). We adhered to the same pre-processing and post-processing configurations as DeePiCt to ensure consistency. The mean and standard deviation of the picking F1 scores are shown in Fig. 2g-h.

RobPicker achieved a mean F1 score of 0.41 for FAS picking, marking a 24.2% relative improvement over DeePiCt's mean F1 score of 0.33 ($P = 0.181$). For ribosome picking, RobPicker's mean F1 score of 0.80 also surpasses DeePiCt's mean F1 score of 0.75 ($P = 0.029$) with a 6.67% relative improvement. These results highlight RobPicker's higher performance in particle picking compared to DeePiCt. Since only 731 FAS particles are present in the dataset, which is much fewer than the

abundant ribosomes with a total of 25,311 particles, the species distribution is extremely imbalanced. Moreover, the FAS particles are much smaller than the ribosome particles. These factors make picking FAS much more challenging than picking ribosomes. Nevertheless, the relative improvement of the F1 scores of RobPicker over DeePiCt is much more significant on the FAS than on the ribosome (24.2% vs 6.67%), underscoring RobPicker's robustness on picking smaller and less abundant particles. RobPicker's robustness to imbalanced species distribution is attributed to the inclusion of the data reweighting network, which dynamically up-weight the minority species in training. This strategy is in contrast to traditional supervised learning methods like DeePiCt, which directly optimizes the segmentation loss function without reweighting the data samples, causing the segmentation network biases towards the major species in the segmentation.

3.3 RobPicker enables efficient, generalizable particle picking for macromolecular structure determination

The output of particle picking from cryo-ET data is often used as input for structure solving analysis. We showed the utilization of RobPicker in combination of subtomogram averaging tool RELION [35] and multi-particle refinement tool M [30] in structure determination pipelines, which demonstrated RobPicker as a fast, first-round picking method without any supervision to get a high-quality 3D reconstruction (also referred to as 3D map or map). We first trained RobPicker to pick ribosomes on just four yeast tomograms in D4, where RobPicker achieved a much higher F1 score than DeepFinder (0.87 vs 0.67) (Fig. 2f). As illustrated in Fig. 4a, RobPicker produced more accurate particle probability maps compared to DeepFinder, with significantly fewer false positives, reflecting its strong specificity. Furthermore, we applied RobPicker on another eight held-out tomograms of yeast, which was efficiently done in ~15 minutes on an A100 GPU. We performed a quick 3D refinement followed by 3D classification on the picked particles in RELION [35], resulting in a good and representative yeast ribosome map (resolution ~19 Å) shown in Fig. 4b, further underscoring RobPicker's efficiency, accuracy, and specificity.

To evaluate RobPicker's generalization capacity, the same RobPicker model trained on the four yeast tomograms was used to pick ribosomes in 14 in-cell tomograms at a highly coarse pixel size (apix) of another organism (bacterial; *M. pneumoniae* [30]). Subsequently, we aimed to create a high-resolution bacterial ribosome map using a pipeline subjected to Fig.4c. Specifically, we performed a 3D refinement with the picked particles in RELION [35], which was quick due to the coarse apix. The 3D classification resulted in a small subset of high-quality particles. These particles were re-extracted at finer apix for further refinement. To further improve and fine-tune tilt-series alignment and its resulting reconstruction quality, we used the multi-particle refinement tool M [30], which jointly optimizes tilt-series alignment and contrast transfer function (CTF) parameters across the full tomograms. Application of M to the refined bacterial ribosome particles resulted in a marked resolution improvement from ~15 Å to 7 Å at the 0.143 Fourier shell correlation (FSC) criterion (Fig.4c). This whole process that led to a significant improvement in the tilt-series alignment and resolution of the ribosome average can be performed quickly within a few hours. This highlights the utility of RobPicker for fast, accurate particle detection and its synergy with modern refinement tools in enhancing tilt-series alignment and structural resolution.

4 Conclusion

We introduced RobPicker—a meta learning framework for robust particle picking in cryo-ET data. In this framework, a data reweighting network and a label correction network are trained to minimize the validation loss on a smaller, cleaner validation set. Then, these two networks reweight the importance of data samples and correct the noisy labels in the training set, enabling better supervision signal to train the macromolecule segmentation network. The two learning stages are performed iteratively in a bi-level optimization framework for continual refinement of these networks. Experiments on cryo-ET real datasets show that RobPicker outperforms two strong particle pickers DeepFinder and DeePiCt, especially on small and rare particle species. Moreover, RobPicker trained on yeast tomograms can pick particles on bacterial tomograms, showing its strong generalizablility across organisms. Finally, the synergy of RobPicker and existing cryo-ET processing tools demonstrates its utilization in structure determination pipelines.

References

[1] Martin Beck and Wolfgang Baumeister. Cryo-electron tomography: can it reveal the molecular sociology of cells in atomic detail? *Trends in cell biology*, 26(11):825–837, 2016.

[2] Julia Mahamid, Stefan Pfeffer, Miroslava Schaffer, Elizabeth Villa, Radostin Danev, Luis Kuhn Cuellar, Friedrich Förster, Anthony A Hyman, Jürgen M Plitzko, and Wolfgang Baumeister. Visualizing the molecular sociology at the hela cell nuclear periphery. *Science*, 351(6276): 969–972, 2016.

[3] Fabian Eisenstein, Haruaki Yanagisawa, Hiroka Kashihara, Masahide Kikkawa, Sachiko Tsukita, and Radostin Danev. Parallel cryo electron tomography on in situ lamellae. *Nature Methods*, 20 (1):131–138, 2023.

[4] Caitlyn L McCafferty, Sven Klumpe, Rommie E Amaro, Wanda Kukulski, Lucy Collinson, and Benjamin D Engel. Integrating cellular electron microscopy with multimodal data to explore biology across space and time. *Cell*, 187(3):563–584, 2024.

[5] Eva Nogales and Julia Mahamid. Bridging structural and cell biology with cryo-electron microscopy. *Nature*, 628(8006):47–56, 2024.

[6] Shujun Cai, Chen Chen, Zhi Yang Tan, Yinyi Huang, Jian Shi, and Lu Gan. Cryo-et reveals the macromolecular reorganization of s. pombe mitotic chromosomes in vivo. *Proceedings of the National Academy of Sciences*, 115(43):10977–10982, 2018.

[7] Cheng-Yu Huang, Piotr Draczkowski, Yong-Sheng Wang, Chia-Yu Chang, Yu-Chun Chien, Yun-Han Cheng, Yi-Min Wu, Chun-Hsiung Wang, Yuan-Chih Chang, Yen-Chen Chang, et al. In situ structure and dynamics of an alphacoronavirus spike protein by cryo-et and cryo-em. *Nature communications*, 13(1):4877, 2022.

[8] Euan Pyle, Elizabeth A Miller, and Giulia Zanetti. Cryo-electron tomography reveals how copii assembles on cargo-containing membranes. *Nature Structural & Molecular Biology*, 32(3): 513–519, 2025.

[9] Vladan Lučić, Alexander Rigort, and Wolfgang Baumeister. Cryo-electron tomography: the challenge of doing structural biology in situ. *Journal of Cell Biology*, 202(3):407–419, 2013.

[10] Shan Gao, Renmin Han, Xiangrui Zeng, Zhiyong Liu, Min Xu, and Fa Zhang. Macromolecules structural classification with a 3d dilated dense network in cryo-electron tomography. *IEEE/ACM transactions on computational biology and bioinformatics*, 19(1):209–219, 2021.

[11] Yuchen Zeng, Gregory Howe, Kai Yi, Xiangrui Zeng, Jing Zhang, Yi-Wei Chang, and Min Xu. Unsupervised domain alignment based open set structural recognition of macromolecules captured by cryo-electron tomography. In *2021 IEEE International Conference on Image Processing (ICIP)*, pages 106–110. IEEE, 2021.

[12] Lorenz Lamm, Ricardo D Righetto, Wojciech Wietrzynski, Matthias Pöge, Antonio Martinez-Sanchez, Tingying Peng, and Benjamin D Engel. Membrain: A deep learning-aided pipeline for detection of membrane proteins in cryo-electron tomograms. *Computer methods and programs in biomedicine*, 224:106990, 2022.

[13] Xiangrui Zeng, Anson Kahng, Liang Xue, Julia Mahamid, Yi-Wei Chang, and Min Xu. High-throughput cryo-et structural pattern mining by unsupervised deep iterative subtomogram clustering. *Proceedings of the National Academy of Sciences*, 120(15):e2213149120, 2023.

[14] Emmanuel Moebel, Antonio Martinez-Sanchez, Lorenz Lamm, Ricardo D Righetto, Wojciech Wietrzynski, Sahradha Albert, Damien Larivière, Eric Fourmentin, Stefan Pfeffer, Julio Ortiz, et al. Deep learning improves macromolecule identification in 3d cellular cryo-electron tomograms. *Nature methods*, 18(11):1386–1394, 2021.

[15] Irene de Teresa-Trueba, Sara K Goetz, Alexander Mattausch, Frosina Stojanovska, Christian E Zimmerli, Mauricio Toro-Nahuelpan, Dorothy WC Cheng, Fergus Tollervey, Constantin Pape, Martin Beck, et al. Convolutional networks for supervised mining of molecular patterns within cellular context. *Nature Methods*, 20(2):284–294, 2023.

[16] Gavin Rice, Thorsten Wagner, Markus Stabrin, Oleg Sitsel, Daniel Prumbaum, and Stefan Raunser. Tomotwin: generalized 3d localization of macromolecules in cryo-electron tomograms with structural data mining. *Nature methods*, 20(6):871–880, 2023.

[17] Guole Liu, Tongxin Niu, Mengxuan Qiu, Yun Zhu, Fei Sun, and Ge Yang. Deepetpicker: Fast and accurate 3d particle picking for cryo-electron tomography using weakly supervised deep learning. *Nature Communications*, 15(1):2090, 2024.

[18] Özgün Çiçek, Ahmed Abdulkadir, Soeren S Lienkamp, Thomas Brox, and Olaf Ronneberger. 3d u-net: learning dense volumetric segmentation from sparse annotation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2016: 19th International Conference, Athens, Greece, October 17-21, 2016, Proceedings, Part II 19*, pages 424–432. Springer, 2016.

[19] Emmanuel Moebel and Charles Kervrann. Towards unsupervised classification of macromolecular complexes in cryo electron tomography: Challenges and opportunities. *Computer Methods and Programs in Biomedicine*, 225:107017, 2022.

[20] Rui Yan, Singanallur V Venkatakrishnan, Jun Liu, Charles A Bouman, and Wen Jiang. Mbir: A cryo-et 3d reconstruction method that effectively minimizes missing wedge artifacts and restores missing information. *Journal of structural biology*, 206(2):183–192, 2019.

[21] Martin Turk and Wolfgang Baumeister. The promise and the challenges of cryo-electron tomography. *FEBS letters*, 594(20):3243–3261, 2020.

[22] Sergio Cruz-León, Tomáš Majtner, Patrick Hoffmann, Jan P Kreysing, Maarten W Tuijtel, Stefan L Schaefer, Katharina Geißler, Martin Beck, Beata Turoňová, and Gerhard Hummer. High-confidence 3d template matching for cryo-electron tomography. *Biophysical Journal*, 123 (3):183a, 2024.

[23] Muyuan Chen, Wei Dai, Stella Y Sun, Darius Jonasch, Cynthia Y He, Michael F Schmid, Wah Chiu, and Steven J Ludtke. Convolutional neural networks for automated annotation of cellular cryo-electron tomograms. *Nature methods*, 14(10):983–985, 2017.

[24] Ziqian Luo, Xiangrui Zeng, Zhipeng Bao, and Min Xu. Deep learning-based strategy for macromolecules classification with imbalanced data from cellular electron cryotomography. In *2019 International joint conference on neural networks (IJCNN)*, pages 1–8. IEEE, 2019.

[25] Rongjian Zhao, Buyue Qian, Xianli Zhang, Yang Li, Rong Wei, Yang Liu, and Yinggang Pan. Rethinking dice loss for medical image segmentation. In *2020 IEEE international conference on data mining (ICDM)*, pages 851–860. IEEE, 2020.

[26] Michael Yeung, Evis Sala, Carola-Bibiane Schönlieb, and Leonardo Rundo. Unified focal loss: Generalising dice and cross entropy-based losses to handle class imbalanced medical image segmentation. *Computerized Medical Imaging and Graphics*, 95:102026, 2022.

[27] Zeju Li, Konstantinos Kamnitsas, and Ben Glocker. Analyzing overfitting under class imbalance in neural networks for image segmentation. *IEEE transactions on medical imaging*, 40(3): 1065–1077, 2020.

[28] Timothy Hospedales, Antreas Antoniou, Paul Micaelli, and Amos Storkey. Meta-learning in neural networks: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 44 (9):5149–5169, 2021.

[29] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning*, pages 1126–1135. PMLR, 2017.

[30] Dimitry Tegunov, Liang Xue, Christian Dienemann, Patrick Cramer, and Julia Mahamid. Multi-particle cryo-em refinement with m visualizes ribosome-antibiotic complex at 3.5 å in cells. *Nature methods*, 18(2):186–193, 2021.

[31] Guoqing Zheng, Ahmed Hassan Awadallah, and Susan Dumais. Meta label correction for noisy label learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(12): 11053–11061, 2021.

[32] Jun Shu, Qi Xie, Lixuan Yi, Qian Zhao, Sanping Zhou, Zongben Xu, and Deyu Meng. Meta-weight-net: Learning an explicit mapping for sample weighting. *Advances in neural information processing systems*, 32, 2019.

[33] Muyuan Chen, James M Bell, Xiaodong Shi, Stella Y Sun, Zhao Wang, and Steven J Ludtke. A complete data processing workflow for cryo-et and subtomogram averaging. *Nature methods*, 16(11):1161–1168, 2019.

[34] Ya-Ting Chang, Benjamin A Barad, Juliette Hamid, Hamidreza Rahmani, Brian M Zid, and Danielle A Grotjahn. Cytoplasmic ribosomes on mitochondria alter the local membrane environment for protein import. *Journal of Cell Biology*, 224(4):e202407110, 2025.

[35] Jasenko Zivanov, Takanori Nakane, Björn O Forsberg, Dari Kimanius, Wim JH Hagen, Erik Lindahl, and Sjors HW Scheres. New tools for automated high-resolution cryo-em structure determination in relion-3. *elife*, 7:e42166, 2018.

[36] Ilja Gubins, Gijs van der Schot, Remco C. Veltkamp, Friedrich Förster, Xuefeng Du, Xiangrui Zeng, Zhenxi Zhu, Lufan Chang, Min Xu, Emmanuel Moebel, Antonio Martinez-Sanchez, Charles Kervrann, Tuan M. Lai, Xusi Han, Genki Terashi, Daisuke Kihara, Benjamin A. Himes, Xiaohua Wan, Jingrong Zhang, Shan Gao, Yu Hao, Zhilong Lv, Xiaohua Wan, Zhidong Yang, Zijun Ding, Xuefeng Cui, and Fa Zhang. Classification in Cryo-Electron Tomograms. *Eurographics Workshop on 3D Object Retrieval*, 2019. ISSN 1997-0471. doi: 10.2312/3dor.20191061.

[37] Ilja Gubins, Marten L Chaillet, Gijs van Der Schot, Remco C Veltkamp, Friedrich Förster, Yu Hao, Xiaohua Wan, Xuefeng Cui, Fa Zhang, Emmanuel Moebel, et al. Shrec 2020: Classification in cryo-electron tomograms. *Computers & Graphics*, 91:279–289, 2020.

[38] Thomas Hrabe, Yuxiang Chen, Stefan Pfeffer, Luis Kuhn Cuellar, Ann-Victoria Mangold, and Friedrich Förster. Pytom: a python-based toolbox for localization of macromolecules in cryo-electron tomograms and subtomogram analysis. *Journal of structural biology*, 178(2): 177–188, 2012.

[39] Marten L Chaillet, Gijs van der Schot, Ilja Gubins, Sander Roet, Remco C Veltkamp, and Friedrich Förster. Extensive angular sampling enables the sensitive localization of macromolecules in electron tomograms. *International Journal of Molecular Sciences*, 24(17):13375, 2023.

[40] Erik Genthe, Sean Miletic, Indira Tekkali, Rory Hennell James, Thomas C Marlovits, and Philipp Heuser. Pickyolo: Fast deep learning particle detector for annotation of cryo electron tomograms. *Journal of Structural Biology*, 215(3):107990, 2023.

[41] Qinwen Huang, Ye Zhou, and Alberto Bartesaghi. Milopyp: self-supervised molecular pattern mining and particle localization in situ. *Nature Methods*, 21(10):1863–1872, 2024.

[42] Yuanpeng Tu, Boshen Zhang, Yuxi Li, Liang Liu, Jian Li, Yabiao Wang, Chengjie Wang, and Cai Rong Zhao. Learning from noisy labels with decoupled meta label purifier. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19934–19943, 2023.

[43] Görkem Algan and Ilkay Ulusoy. Meta soft label generation for noisy labels. *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 7142–7148, 2021.

[44] Mengye Ren, Wenyuan Zeng, Bin Yang, and Raquel Urtasun. Learning to reweight examples for robust deep learning. In *International conference on machine learning*, pages 4334–4343. PMLR, 2018.

[45] Lee R Dice. Measures of the amount of ecologic association between species. *Ecology*, 26(3): 297–302, 1945.

[46] Hanxiao Liu, Karen Simonyan, and Yiming Yang. DARTS: Differentiable architecture search. In *International Conference on Learning Representations*, 2019. URL `https://openreview.net/forum?id=S1eYHoC5FX`.

[47] Sang Keun Choe, Willie Neiswanger, Pengtao Xie, and Eric Xing. Betty: An automatic differentiation library for multilevel optimization. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=LV_MeMS38Q9.

[48] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

[49] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. pmlr, 2015.

[50] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814, 2010.

[51] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *KDD*, volume 96, pages 226–231, 1996.

[52] James R Kremer, David N Mastronarde, and J Richard McIntosh. Computer visualization of three-dimensional image data using imod. *Journal of structural biology*, 116(1):71–76, 1996.

[53] Eric F Pettersen, Thomas D Goddard, Conrad C Huang, Gregory S Couch, Daniel M Greenblatt, Elaine C Meng, and Thomas E Ferrin. Ucsf chimera—a visualization system for exploratory research and analysis. *Journal of computational chemistry*, 25(13):1605–1612, 2004.

[54] Thomas D Goddard, Conrad C Huang, Elaine C Meng, Eric F Pettersen, Gregory S Couch, John H Morris, and Thomas E Ferrin. Ucsf chimerax: Meeting modern challenges in visualization and analysis. *Protein Science*, 27(1):14–25, 2018.

[55] Dimitry Tegunov and Patrick Cramer. Real-time cryo-electron microscopy data preprocessing with warp. *Nature methods*, 16(11):1146–1152, 2019.

[56] Yun-Tao Liu, Heng Zhang, Hui Wang, Chang-Lu Tao, Guo-Qiang Bi, and Z Hong Zhou. Isotropic reconstruction for electron tomography with deep learning. *Nature communications*, 13(1):6482, 2022.

[57] Felix R Wagner, Reika Watanabe, Ruud Schampers, Digvijay Singh, Hans Persoon, Miroslava Schaffer, Peter Fruhstorfer, Jürgen Plitzko, and Elizabeth Villa. Preparing samples from whole cells using focused-ion-beam milling for cryo-electron tomography. *Nature protocols*, 15(6): 2041–2070, 2020.

[58] Guang Tang, Liwei Peng, Philip R Baldwin, Deepinder S Mann, Wen Jiang, Ian Rees, and Steven J Ludtke. Eman2: an extensible image processing suite for electron microscopy. *Journal of structural biology*, 157(1):38–46, 2007.

[59] Shun-ichi Amari. Backpropagation and stochastic gradient descent method. *Neurocomputing*, 5 (4-5):185–196, 1993.

[60] Nikhil Ketkar and Nikhil Ketkar. Stochastic gradient descent. *Deep learning with Python: A hands-on introduction*, pages 113–132, 2017.

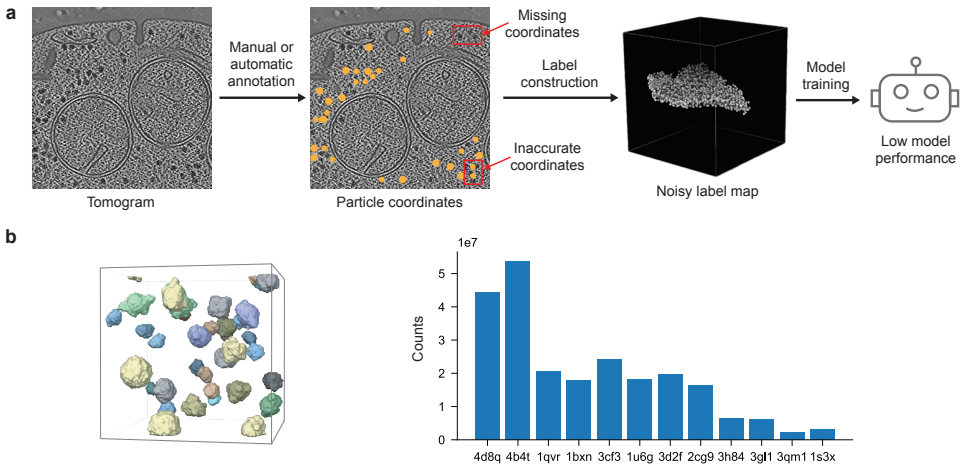


Figure 3: (a) An illustrations of how noisy labels are generated. (b) The distribution of classes are imbalanced in an exemplar cryo-ET dataset SHREC 2019.

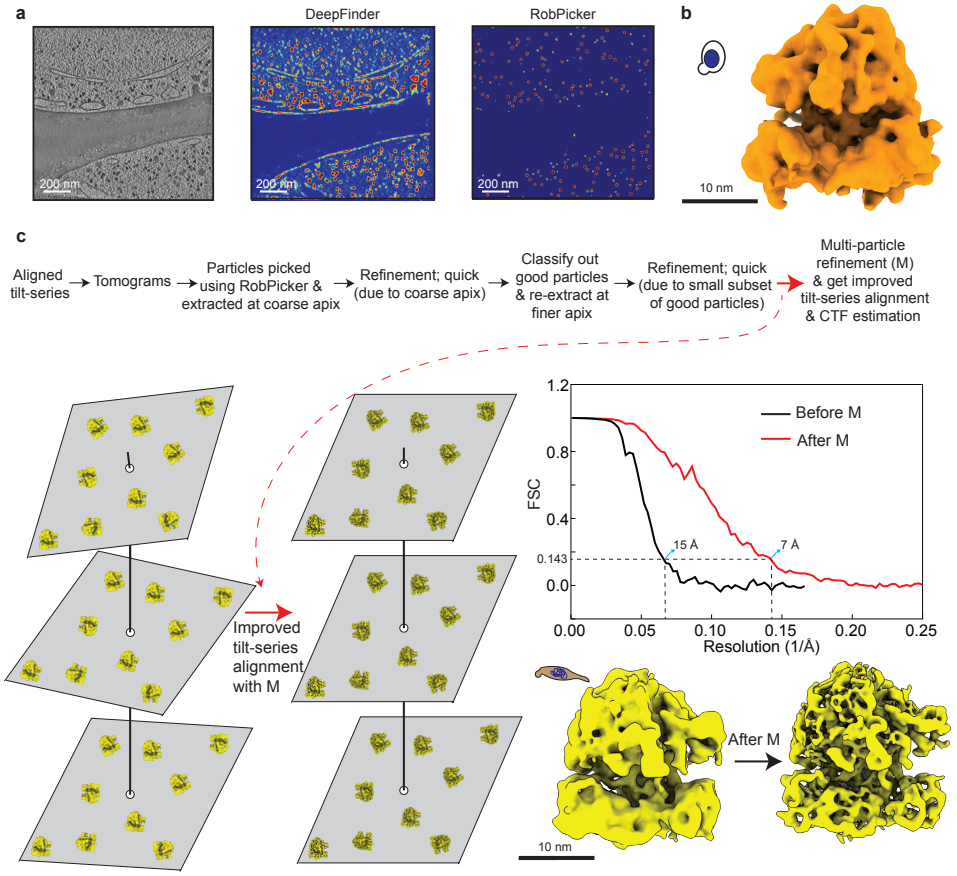


Figure 4: (a) Comparison of the ribosome probability maps from DeepFinder and RobPicker. (b) Yeast ribosome 3D map resulted from particles picked by RobPicker. (c) RobPicker is used in combination of standard cryo-ET tools, RELION and M, to improve the resolution of ribosome 3D map.

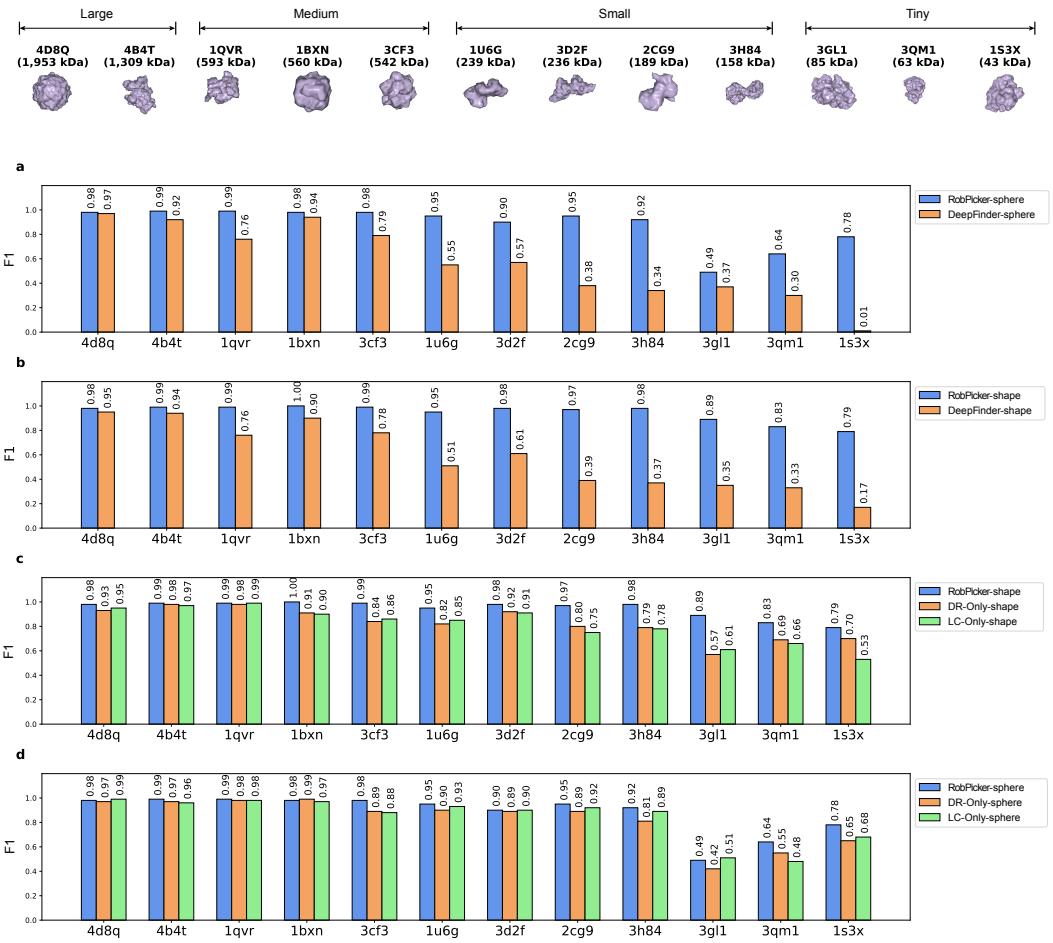


Figure 5: (a) and (b) RobPicker outperformed DeepFinder across 12 distinct classes encompassing particles of different sizes. (a) The F1 scores for models with training labels being sphere-based. (b) The F1 scores for models with training labels being shape-based. (c) and (d) RobPicker integrated with both data reweighting network (DRN) and label correction network (LCN) outperformed the baselines of RobPicker with only DRN or LCN, in both shape- and sphere-label experiments.

A More experiments

A.1 RobPicker robustly picks particles of diverse sizes in noisy, imbalanced data

We conducted experiments on a benchmark dataset from the SHREC 2019 Cryo-ET Challenge, which consists of ten synthetic tomograms with ground truth labels for particles across 12 distinct species, covering a wide range of sizes [36]. These particles were categorized by size into four groups: large, medium, small, and tiny (Fig. 5). The dataset was divided into a training set, a validation set, and a test set. The label maps for the validation and test sets were created using the shape-based method, while the label maps for the training set were created based on two settings, one with the shape-based method and the other with the sphere-based method—to demonstrate the robustness of RobPicker facing data with different noise levels.

We compared RobPicker's performance with DeepFinder's [14] in the benchmark. RobPicker consistently outperformed DeepFinder in terms of picking F1-scores across all particle size categories, including large, medium, small, and tiny in the sphere-based experiments (Fig. 5a). Notably, RobPicker demonstrated a substantial improvement over DeepFinder in detecting small and tiny particle species.

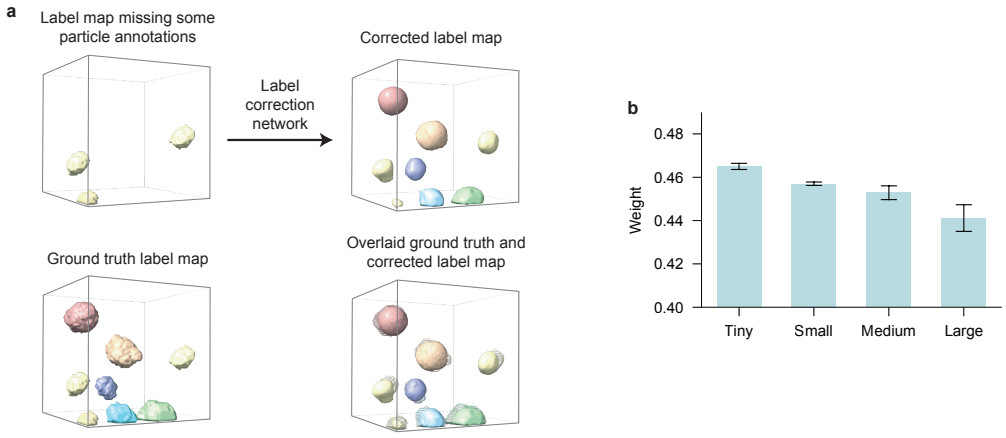


Figure 6: RobPicker's label correction network and data reweighting network effectively correct missing labels and assign higher weights to minority species in the SHREC 2019 dataset. (a) Based on a tomogram and a noisy label map with some missing particle annotations, RobPicker's label correction network predicted a new label map that recovered the missing particle labels. The overlaid ground truth (shown in mesh) and corrected label (shown in solid objects) show high alignment. Different colors denote different species. (b) RobPicker's data reweighting network assigned larger weights—the mean and standard deviation are shown—to smaller macromolecule species, accounting for the data imbalance. The x-axis denotes particle sizes.

RobPicker also outperformed DeepFinder in the shape-based experiments (Fig. 5b) where the training labels were cleaner than the sphere-based experiments. Moreover, RobPicker demonstrated a more pronounced improvement over DeepFinder for smaller particles compared to larger ones.

A.2 Ablation studies

To evaluate the effectiveness of the data reweighting (DR) network and label correction (LC) network, we tested two variants of RobPicker: DR-Only, which performs data reweighting without label correction, and LC-Only, which applies label correction without data reweighting. RobPicker with both DR an LC networks significantly outperformed the two variants (Fig. 5c,d), especially for picking particles of tiny species. These results highlight the importance of combining data reweighting and label correction to achieve robust performance in RobPicker.

A.3 Qualitative results of label correction network and data reweighting network

Our analysis revealed that RobPicker's label correction network recovered missing labels. Specifically, we had a noisy label map of a tomogram from the SHREC 2019 cryo-ET dataset [37], where the labels for some particles were missing (Fig. 6a). The noisy label map and the subtomogram were inputted into RobPicker's label correction network, which outputted a new label map that showed multiple particles' occupancy (Fig. 6b). We found that the clean label map (Fig. 6c), which showed the actual particle occupancy, and the label map predicted by the label correction network highly overlap with each other (Fig. 6d). Therefore, the missing labels were recovered by the label correction network, which provided improved, cleaner labels to train the macromolecule segmentation network for robust identification of particles. Moreover, we found that the data reweighting network assigned larger weights to underrepresented macromolecule species. The SHREC 2019 dataset categories the particles into four categories, including large, medium, small, and tiny categories, by the particle mass (Fig. 5). For each subtomogram in the training set, it was classified as one of the large, medium, small, and tiny categories if the majority of particles were from that category in the subtomogram. Each subtomogram was assigned a weight by the data reweighting network during training. We recorded the weights of the subtomograms, and plotted the mean and standard deviation of the weights for the large, medium, small, and tiny categories. As shown in Fig. 6b, the data reweighting network assigned larger weights to the subtomograms with smaller particles, which indicated that the network learned to give more emphasis to underrepresented particles in training the macromolecule network.

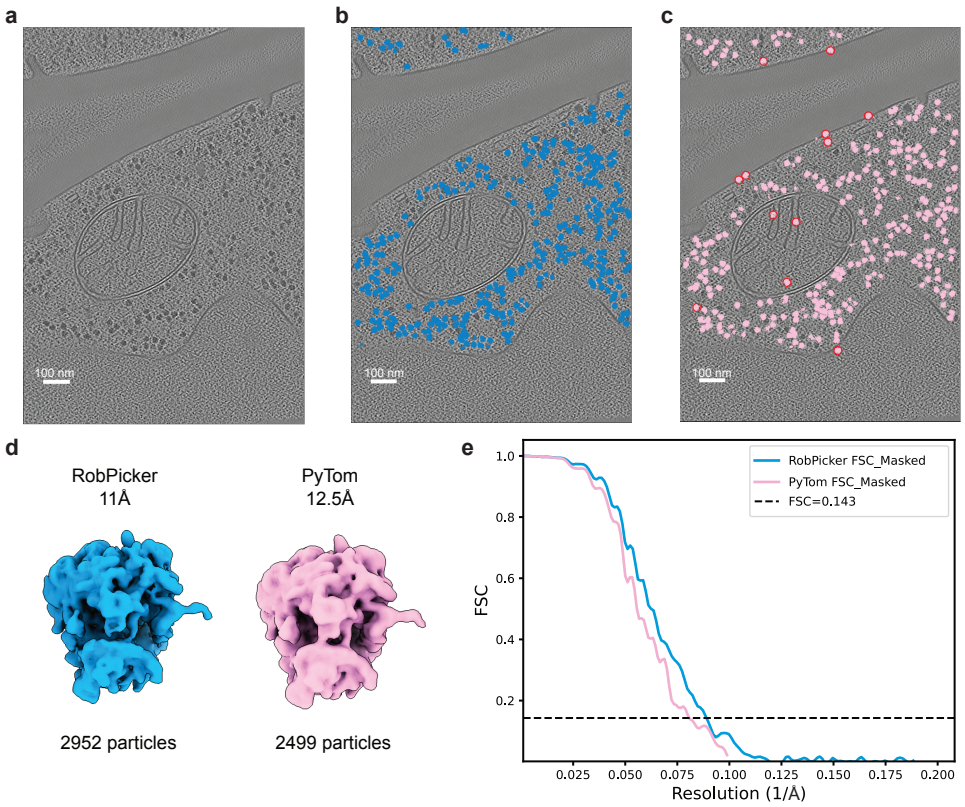
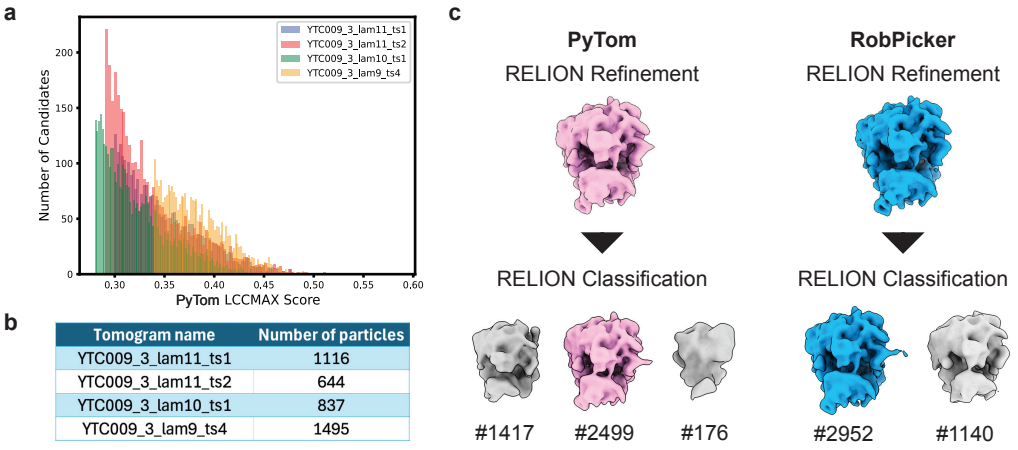


Figure 7: RobPicker provided supervision-free particle picks, compared with PyTom. (a) Tomographic slice of a representative tomogram from yeast. (b) Tomographic slice with ribosome picks by RobPicker highlighted. A total of 1,116 particles were found on this tomogram. (c) The same slice with the top 1,116 candidates from PyTom template-matching highlighted. Typical false positive picks are marked in red circles. (d) RELION refinement and classification as well as M refinement were used on both populations of particles to maximize the resolution. RobPicker produced a better reconstruction with more high-quality particles. (e) FSC curves of the reconstructions shown in (d).

This strategy effectively mitigated the issue of imbalanced species and improved the particle picking performance on small species (Fig. 5).

A.4 Comparison with PyTom on structure determination

We performed an experiment on another yeast tomogram dataset to compare traditional template matching methods and RobPicker in solving ribosome structures. The lack of supervision and manual curation required to clean up the picks is a major advantage of RobPicker. Such manual curation is a significant bottleneck for traditional template-matching methods like PyTom [38, 39] for particle picking. The manual curation is required in such template matching methods due to prevalence of false positives in the picked particles. But even this manual curation is ill-defined. Typically, a user-defined threshold (on quality score of template matching picks) has to be applied to pick high quality particles (presumably representing true positives). In contrast, RobPicker's detection of ribosomes requires no manual input and can generalize well to other organisms without changing any hyper-parameters (as shown in the bacterial ribosome experiments above). After training RobPicker on five yeast tomograms in D2, we used another four held-out yeast tomograms as the test dataset for a comparison between PyTom and RobPicker. While RobPicker identified 4,092 particles in the four tomograms, the automatic thresholding in PyTom returned very few particles. This can be explained by the fact that there is no visible second peak in the histogram of the local cross-correlation (LCC) scores when we extract the top 10,000 candidates (Fig. 8a). To compare the two particle-picking methods, we extracted the same number of particles as the RobPicker output from PyTom candidates (breakdown in Fig. 8b). The comparison between the RobPicker picks and PyTom picks, shown in



Tomogram name	Number of particles
YTC009_3_lam11_ts1	1116
YTC009_3_lam11_ts2	644
YTC009_3_lam10_ts1	837
YTC009_3_lam9_ts4	1495

Figure 8: (a) Histogram of the LCC scores from PyTom for the top 10,000 particles does not show the desired distribution with peaks for automatic thresholding. (b) Number of ribosomes for each tomogram picked by RobPicker which does not need any manual input. (c) Top PyTom particles from the 4 tomograms are extracted and compared before and after classification. The best classes then are refined in RELION and M to produce final structures shown in Fig. 7d.

Fig. 7b–c, demonstrates the higher specificity of the RobPicker picks. We performed several rounds of refinement and classification in RELION [35] and further multi-particle refinement in M [30], for both particle sets to select a subset of particles that showed consistent high-resolution features during iterative refinement. This process resulted in an 11 Å reconstruction from the RobPicker-identified dataset, and a 12.5 Å reconstruction from the PyTom-identified dataset. Visual inspection of the reconstructions (Fig. 7d) shows a marked increase in detail in the map from RobPicker-identified picks compared to those identified in PyTom, emphasizing the improved ability of RobPicker to identify more particles that contribute to high-resolution information in the resulting reconstruction. The ability to use RobPicker as a pre-trained model offers the potential of improving and speeding this process up even further for initial reconstruction of 80S ribosomes that can be used to improve our tomograms in packages like RELION and M, similar to the pipeline described in Fig. 4c.

B Related works

Deep learning for cryo-ET particle picking Deep learning has rapidly displaced classical template matching for particle picking in cryo-ET. Supervised 3D U-Net–style approaches established that neural networks can localize multiple macromolecular species directly in tomograms (DeepFinder [14]) and mine cellular context to improve picking (DeePiCt [15]). Subsequent architectures targeted speed and scalability, e.g., one-stage 3D detectors like PickYOLO [40], and generalizable feature learning [16]. More recently, self-supervised contrastive pipelines like MiLoPYP [41] use unsupervised pattern mining in particle segmentation model training.

Meta learning for robust machine learning Meta-learning has emerged as a powerful paradigm for building models that adapt quickly to new tasks or noisy supervision by leveraging higher-order optimization. The seminal Model-Agnostic Meta-Learning (MAML) framework demonstrated that a single initialization can be optimized across tasks to enable rapid adaptation with only a few gradient steps, inspiring a wave of follow-up work on robust learning [29]. In the context of noisy labels, meta-learning has been widely applied to dynamically refine supervision quality. Meta label correction methods use auxiliary validation sets to guide the correction of mislabeled data, while extensions such as the decoupled Meta Label Purifier [42] and Meta Soft Label Generation [43] propose strategies for disentangling clean and noisy signals or generating reliable soft labels. Complementary to label correction, another line of work has focused on adaptive reweighting. The method of learning to reweight examples introduced the idea of meta-learned weighting of training samples to mitigate the effect of noise [44], later extended by Meta-Weight-Net [32], which parameterizes an explicit weighting function optimized via meta-gradients. Collectively, these approaches highlight how

meta-learning enables data-driven strategies for both correcting supervision and reweighting training signals, offering a robust framework for learning under noisy or imperfect data.

C The RobPicker framework

RobPicker is formulated as a meta learning problem, with a bi-level optimization (BLO) formulation. A BLO problem has two levels of optimization problems: a lower level and a upper level. In RobPicker, the training of the macromolecule segmentation network is at the lower level, while the training of the DR network and LC network is at the upper level. The two levels are nested and mutually dependent on each other: the optimal parameters of the segmentation network from the lower level are used to define the objective function at the upper level; the non-optimal parameters of the DR and LC networks from the upper level are used to define the objective function at the lower level. Due to this mutual dependency, the model parameters of the two levels are iteratively updated, with the detailed process described below.

Given a training dataset $D_{\text{tr}} = \{(t_i, a_i)\}_{i=1}^{N_{\text{tr}}}$, where t_i represents an input tomogram, a_i denotes the corresponding macromolecule label, and N_{tr} is the number of training samples, we input t_i into the macromolecule segmentation network $f(t_i; I)$ with I representing the network's weight parameters, which outputs a prediction. Simultaneously, a_i is fed into the LC network $g(a_i; C)$, where C represents the weight parameters for this network, producing a corrected label. A loss function $\ell(f(t_i; I), g(a_i; C))$, which is the Dice loss [45], is computed to quantify the discrepancy between the detection result and the corrected label. The loss value is fed into the DR network to output a scalar weight $h(\ell(f(t_i; I), g(a_i; C)); R)$, where the DR network is parameterized by R. The loss is then multiplied by this weight. Let $L(D_{\text{tr}}, I, C, R)$ represent the sum of the reweighted losses over the entire training dataset:

$$\begin{aligned} &L(D_{\text{tr}}, I, C, R) = \\ &\textstyle\sum_{i=1}^{N_{\text{tr}}} h(\ell(f(t_i; I), g(a_i; C)); R)\ell(f(t_i; I), g(a_i; C)). \end{aligned} \tag{1}$$

We train the segmentation network by updating its parameters I to minimize this loss $L(D_{\text{tr}}, I, C, R)$ while keeping the parameters of the LC and DR networks fixed. This leads to the following optimization problem:

$$I^*(C, R) = \text{argmin}_I \; L(D_{\text{tr}}, I, C, R). \tag{2}$$

Here, $I^*(C, R)$ indicates that the optimal trained parameter I^* is a function of the parameters of the LC and DR network C and R, as I^* is determined by the loss function, which itself depends on C and R. Note that C and R are not optimized in this step, as doing so could result in a trivial solution where the weight $h(\ell(f(t_i; I), g(a_i; C)); R)$ becomes zero for every i.

Next, we evaluate the trained segmentation network $I^*(C, R)$ on a clean validation dataset $D_{\text{val}} = \{(t_j, a_j)\}_{j=1}^{N_{\text{val}}}$, where N_{val} represents the number of validation examples. For each validation tomogram t_j, the segmentation network is used to detect macromolecules, and the resulting prediction $f(t_j; I^*(C, R))$ is compared to the corresponding ground truth label a_j, yielding a loss $\ell(f(t_j; I^*(C, R)), a_j)$. Let $L(D_{\text{val}}, I^*(C, R))$ represent the total loss over the entire validation dataset:

$$L(D_{\text{val}}, I^*(C, R)) = \sum_{j=1}^{N_{\text{val}}} \ell(f(t_j; I^*(C, R)), a_j). \tag{3}$$

We optimize the parameters of the CL and DR networks C and R by minimizing the validation loss, which leads to solving the following optimization problem:

$$\text{min}_{C,R} \; L(D_{\text{val}}, I^*(C, R)) \tag{4}$$

Combining Equations 2 and 4, we obtain the following bi-level optimization problem:

$$\begin{aligned} &\text{min}_{C,R} \; L(D_{\text{val}}, I^*(C, R)) \\ &s.t. \qquad I^*(C, R) = \text{argmin}_I \; L(D_{\text{tr}}, I, C, R) \end{aligned} \tag{5}$$

The two levels of optimization problems are interdependent. The solution from the lower level, $I^*(C, R)$, is used to define the loss function at the upper level, while the optimization variables C and R from the upper level are also involved in defining the loss function at the lower level.

D Optimization algorithm

We address the optimization problem in Equation 5 using a hypergradient-based approach. Specifically, we approximate the optimal solution $I^*(C, R)$ through a single-step gradient descent update as follows:

$$I^*(C, R) \approx I'(C, R) = I - \eta_I \nabla_I L(D_{\text{tr}}, I, C, R). \tag{6}$$

where η_I denotes the learning rate.

We substitute the approximation $I^*(C, R) \approx I'(C, R)$ into the upper-level loss function, yielding an approximate loss $L(D_{\text{val}}, I'(C, R))$. We then update C and R using gradient descent with respect to the approximate loss:

$$C \leftarrow C - \eta_c \nabla_C L(D_{\text{val}}, I'(C, R)), \tag{7}$$

$$R \leftarrow R - \eta_r \nabla_R L(D_{\text{val}}, I'(C, R)), \tag{8}$$

where η_c and η_r denote the respective learning rates. The gradient $\nabla_C L(D_{\text{val}}, I'(C, R))$ can be computed using the chain rule:

$$\nabla_C L(D_{\text{val}}, I'(C, R)) = \frac{\partial I'}{\partial C} \frac{\partial L(D_{\text{val}}, I'(C, R))}{\partial I'}, \tag{9}$$

where

$$\frac{\partial I'}{\partial C} = -\eta_I \nabla^2_{C,I} L(D_{\text{tr}}, I, C, R). \tag{10}$$

Similarly, the gradient $\nabla_R L(D_{\text{val}}, I'(C, R))$ is computed as:

$$-\eta_R \nabla^2_{R,I} L(D_{\text{tr}}, I, C, R) \frac{\partial L(D_{\text{val}}, I'(C, R))}{\partial I'}. \tag{11}$$

The updates in Equations 6, 7, and 8 are performed iteratively until convergence. The steps of the algorithm are outlined in Algorithm 1.

In Equation 11, computing the matrix $\frac{\partial I'}{\partial C} = -\eta_I \nabla^2_{C,I} L(D_{\text{tr}}, I, C, R)$ and performing multiplication with the vector $\frac{\partial L(D_{\text{val}}, I'(C,R))}{\partial I'} = \nabla_{I'} L(D_{\text{val}}, I'(C, R))$ incur noticeable computational cost. To mitigate this, we adopted the finite difference approximation method [46]. This method is a numerical technique for estimating derivatives when their analytical forms are either complex or infeasible to compute. By evaluating the function at perturbed points, finite difference approximation provides an efficient way to approximate derivatives. Let I^+ and I^- denote $I + \epsilon \nabla_{I'} L(D_{\text{val}}, I'(C, R))$ and $I - \epsilon \nabla_{I'} L(D_{\text{val}}, I'(C, R))$, respectively, where ϵ is a small scalar. Using this approach, the matrix-vector multiplication can be approximated as:

$$\nabla^2_{C,I} L(D_{\text{tr}}, I, C, R) \nabla_{I'} L(D_{\text{val}}, I'(C, R)) \approx \frac{\nabla_C L(D_{\text{tr}}, I^+, C, R) - \nabla_C L(D_{\text{tr}}, I^-, C, R)}{2\epsilon} \tag{12}$$

Similarly, $\nabla^2_{R,I} L(D_{\text{tr}}, I, C, R) \nabla_{I'} L(D_{\text{val}}, I'(C, R))$ can be approximated as:

$$\nabla^2_{R,I} L(D_{\text{tr}}, I, C, R) \nabla_{I'} L(D_{\text{val}}, I'(C, R)) \approx \frac{\nabla_R L(D_{\text{tr}}, I^+, C, R) - \nabla_C L(D_{\text{tr}}, I^-, C, R)}{2\epsilon}. \tag{13}$$

This bi-level optimization algorithm has been implemented in our Python library Betty [47], based on which RobPicker was implemented.

E Model architecture

We used PyTorch [48] to implement the macromolecule segmentation network. For the experiments in comparison with DeepFinder, we used the 3D U-Net [18] following DeepFinder [14]. The input of the macromolecule segmentation network is a 3D tensor (e.g., $h \times w \times d$), representing a 3D tomogram. The output of the macromolecule segmentation network is a 4D tensor ($h \times w \times d \times c$) with c channels, where c matches the number of classes (number of species plus one for the background). The model has four down-blocks, where each down-block has four 3D convolutional layers (kernel size: 3x3x3, stride: 1x1x1, padding: 1x1x1). Each convolutional layer is followed by a 3D batch

Algorithm 1: Bi-level optimization for RobPicker

Input: Training set D_{tr}, validation set D_{val};
initial parameters for segmentation network $I^{(0)}$, label correction network $C^{(0)}$, data reweighting network $R^{(0)}$;
learning rates η_I, η_C, η_R; maximum iterations T
Output: Learned parameters I, C, R
for $t \leftarrow 1$ **to** T **do**
  `// Approximate inner optimum` $I^*(C, R)$ `by a single-step gradient descent`
  $I' \leftarrow I^{(t-1)} - \eta_I \nabla_I L\big(D_{\text{tr}}, I^{(t-1)}, C^{(t-1)}, R^{(t-1)}\big)$;
  `// Update outer parameters` C `and` R `using hypergradient`
  `// The hypergradient is approximated by Equations 12 and 13`
  $C^{(t)} \leftarrow C^{(t-1)} - \eta_C \nabla_C L\big(D_{\text{val}}, I'\big)$;
  $R^{(t)} \leftarrow R^{(t-1)} - \eta_R \nabla_R L\big(D_{\text{val}}, I'\big)$;
  $I^{(t)} \leftarrow I'$
return $I^{(T)}$, $C^{(T)}$, $R^{(T)}$

normalization layer [49] and a ReLU activation function [50]. There is a 3D max-pooling (kernel size: 2, stride: 2, padding: 1, dilation: 1) between the second and the third convolutional layer in each down-block. The model essentially predicts the probabilities of each voxel belonging to each class. If we perform the $\arg\max$ operation along the channel dimension, we can get the predicted label of each voxel belonging to one of the classes (see post-processing below). The down-blocks are followed by a bottleneck block with two 3D convolutional layers, which is followed by four up-blocks. Each up-block has a transposed 3D convolutional layer (kernel size: 2x2x2, stride: 2x2x2) and two 3D convolutional layers (kernel size: 3x3x3, stride: 1x1x1, padding: 1x1x1). The final layer of this segmentation network is 3D convolutional layer (kernel size: 1x1x1, stride: 1x1x1). For the experiment on D5 where RobPicker was compared with DeePiCt, we used the same 3D U-Net as DeePiCt [15], which has a similar architecture as DeepFinder and it has 41 million parameters.

The data reweighting network has a linear layer that expands the input dimension of 1 (i.e., the loss value) to a hidden dimension of 500, followed by a ReLU activation [50], and then a linear layer that shrinks the hidden dimension of 500 to the output dimension of 1, followed by a sigmoid function that converts the output to a weight between 0 and 1. The data reweighting network has only 1.5 thousand parameters. The label correction network takes as input the noisy label, denoted by $\mathbf{L}$ (4D tensor with c channels, where the channel dimension is one-hot), and the input tomogram (3D tensor). The label correction network starts with a sub-network that processes the input tomogram and outputs a feature map of the tomogram, denoted by $\mathbf{M}$. The sub-network shares the same architecture and parameters as the macromolecule segmentation network for efficiency consideration. A 3D convolution layer maps the input channel size of $\mathbf{M}$ to the output channel size of the number of classes (with kernel size being 1, padding 1, and stride 1), followed by a softmax operation. The result of the softmax is denoted by $\mathbf{S}$. Then, $\mathbf{S}$ is concatenated with the noisy label. The result feature map has a channel size of two times the number of classes. Then, a 3D convolution layer (with kernel size being 1, padding 1, and stride 1) shrink the channel size to the number of classes, followed by a ReLU activation, another 3D convolution layer (with kernel size being 1, padding 1, and stride 1), and a sigmoid operation. The result, denoted by α, acts as a mixing coefficient of the noisy label $\mathbf{S}$ and the softmax result $\mathbf{S}$ to compute the corrected label $\mathbf{C}$:

$$\mathbf{C} = \alpha \cdot \mathbf{S} + (1 - \alpha) \cdot \mathbf{L}. \tag{14}$$

Due to the sharing of parameters between the label correction network and the macromolecule segmentation network, the label correction network only has around 1.3 thousand parameters. The small amount of parameters of the label correction and data reweighting networks has significantly reduced the computational complexity of the bi-level optimization in RobPicker.

F Data preprocessing

Data structure A tomogram is represented as a 3D tensor with shape $h \times w \times d$, while its label map is represented as a 4D tensor with shape $h \times w \times d \times c$, where c is the number of classes for segmentation. For a label map denoted by M, and for a specific 3D coordinate (x, y, z), we have a ground-truth class, denoted by $e \in \{1, 2, \ldots, c\}$. Then, the label map is constructed so that $M_{x,y,z,e} = 1$ and $M_{x,y,z,j} = 0$ for $j \neq e$.

Tomogram patching In the training phase of deep learning models for tomographic data of large sizes (e.g., $500 \times 500 \times 500$), GPU memory capacity often becomes a bottleneck, often causing out of memory error. To circumvent this limitation, we divided a whole tomogram into distinct 3D subtomograms, or "patches". Following DeepFinder [14], we used a patch size of $64 \times 64 \times 64$. We also applied random rotation and random shift to the input for data augmentation during training. The labels were extracted using the same patch size accordingly. Therefore, the segmentation network only needs to take a 3D subtomogram as input and predict the label on the subtomogram. The advantage of this strategy lies in its memory efficiency—only the batch currently under scrutiny is loaded into the GPU memory. This ensures that our methodology remains adaptable across a spectrum of GPUs. During inference, the test tomogram is also divided into subtomogram to input to the segmentation network. However, the prediction on the boundary of the subtomograms can be less accurate due to the lack of sufficient tomographic context. Therefore, we extrated overlapping subtomograms (with an overlap of 20 voxels) and averaged the overlapping regions of the predicted segmentation maps.

Resampling To make the particle classes (i.e., macromolecule species) distributed balanced in the validation data, we used resampling when extracting the subtomograms. Specifically, we first uniformly sample a particle class from {1,2,. . . , c} and then sample a particle from the chosen particle class. Then, the subtomogram with the sampled particle is extracted for training.

G Data post-processing

Similar to the methodology adopted by DeepFinder [14], RobPicker's post-processing operation is a multi-step procedure involving classification and clustering. Applying the macromolecule segmentation network on the input tomogram results in a segmentation map, which is denoted by M. Then, the first step is classification of every voxel within the tomogram—ensuring each voxel is attributed to a specific macromolecule species—by selecting the class with the highest predicted probability. Mathematically, we compute $C_{x,y,z} = \arg\max_i M_{x,y,z,i}$ to obtain a 3D tensor C, which effectively becomes the label map predicted by the model. Building on the voxel classification, RobPicker's subsequent focus is clustering. In this phase, a robust clustering algorithm, called DBSCAN [51], groups the classified voxels into distinct units that represent individual particles. For the configuration of the DBSCAN algorithm, we set the radius of a neighborhood to 1 voxel and minimum number of voxels in a cluster to 5. After clustering, the exact location of each particle is revealed within the tomogram volume by calculating the gravity center of the voxel clusters. For the 3D visualization of the particles, we use IMOD [52], Chimera [53], and ChimeraX [54].

H Datasets

D1 was obtained from DeepFinder [14]. It encapsulates 57 tomograms of *C. reinhardtii* cells. Experts manually annotated the 3D coordinates of 8,792 mb-ribos using a combination of template matching, subtomogram classification, and visual inspection. Ct-ribos were annotated by semi-automatic tools without expert supervision.

D2 was generated to study co-translating 80S ribosomes in yeast cells [34]. It includes 6 tomograms from yeast lamellae, where PyTom [38] template-matching and manual expert annotations were used to annotate 80S ribosomes. Tomograms were collected at the pixel size of 2.63 Å and reconstructed at the pixel size of 10 Å in Warp [55]. All tomograms were denoised in IsoNet [56] to compensate for the missing wedge effects.

Dataset	Object	Training	Validation	Test
D1	tomogram	48	1	8
	ct-ribos	6,687	254	2,594
	mt-ribos	6,834	222	1,736
D2	tomogram	4	1	1
	ribosome	2,986	707	843
D3	tomogram	26	1	1
	50S	6,013	443	238
	70S	6,318	512	285
D4	tomogram	3	1	1
	ribosome	6,467	1,238	1,496

Table 1: **Dataset statistics.** We report the number of tomograms, and number of annotated particles in the training, validation, and test sets.

For D3, we collected 28 tomograms from *E. coli* cells. These tomograms were collected at pixel size of 1.66 Å and reconstructed and binned by 6 voxels at 9.98 Å using Warp and were not denoised any further. The annotation for this dataset was done through template-matching and classification in RELION [35] to separate the 70S and 50S classes.

For D4, we collected 5 tomograms using the following procedure. Yeast cells in the logarithmic growth phase were vitrified and prepared for cryo-ET using cryo-FIB milling [57]. A ribosome particle picker was trained on this dataset and subsequently applied to eight additional tomograms of yeast (also prepared using cryo-FIB) without any supervision, yielding ~23,000 ribosome coordinates. These particles were subjected to refinement in RELION 3 [35]. The workflow included an initial refinement of all particles, followed by 3D classification. Particles belonging to the best-resolved class (~5,500 particles) were selected for subsequent refinements, resulting in the yeast ribosome map shown in Fig. 4b.

The *M. pneumoniae* dataset that was used to demonstrate the pipeline in Fig. 4c was obtained from M [30]. The same RobPicker trained on D4 was used to pick ~5,300 particles from 14 tomograms of *M. pneumoniae*. These picks were subjected to initial refinement and classification in RELION 3 [35]. From these, ~2,000 particles belonging to the highest-quality class were selected for multi-particle refinement in M [30]. This refinement, coupled with tilt-series alignment fine-tuning, improved the ribosome map resolution from ~15 Å to 7 Å (Fig. 4c). All reported resolutions were estimated using the gold-standard Fourier shell correlation (FSC) at the 0.143 criterion.

D5 was obtained from DeePiCt [15]. It includes 10 VPP tomograms of *S. pombe* cells. An iterative workflow was used to localize ribosome and FAS in 4×-binned tomograms (13.48 Å voxel size). Manually curated template matching was performed for ribosomes, and non-exhaustive manual picking was performed for FAS (step 1). The resulting annotations were used to train the 3D CNNs of DeePiCt (step 2). For ribosomes, step 2 was repeated three times (always trained on combined predictions of step 1 and the preceding round). Cumulative predictions were manually revised in tom_chooser in PyTom (for ribosomes) and in EMAN2 [58] (for FAS). This comprehensive pipeline ensured the labels were high-quality.

The SHREC 2019 challenge [36] utilized 12 Protein Data Bank (PDB) identifiers (1bxn, 1qvr, 1s3x, 1u6g, 2cg9, 3cf3, 3d2f, 3gl1, 3h84, 3qm1, 4b4t, and 4d8q) to generate tomogram density maps. Then, the density maps were subsequently transformed into a tilt series of projection images, which were then deliberately degraded with noise and contrast adjustments. The reconstructed tomograms from tilt series have a dimension of $512 \times 512 \times 512$ with a 1 nm voxel resolution. Each tomogram contains approximately 200 particles per species on average. Nine tomograms were provided for training, while the tenth tomogram was reserved for testing. For RobPicker, we partitioned the nine training tomograms into two subsets, assigning eight tomograms for training and one for validation. These subsets were employed in Stage I and Stage II—respectively—of the training process shown in Fig. 1c.

I Experimental Settings

Loss function We used the Dice loss function following DeepFinder [14] and DeePiCt [15]. The Dice loss function is computed for each channel (i.e., class) and the final loss value is the averaged loss values of all the channels. Specifically, for a ground truth label map A (4D tensor of shape $h \times w \times d \times c$) and a predicted label map B, whose values are in the interval $[0, 1]$, the Dice loss is computed as:

$$\text{Dice}(A, B) = \frac{1}{c} \sum_{j=1}^{c} \left(1 - \frac{2|A_j \cap B_j|}{|A_j| + |B_j|} \right) \tag{15}$$

where j is the index for the last dimension, and the intersection $|A_j \cap B_j| = \sum_{x,y,z} A_{x,y,z,j} B_{x,y,z,j}$, and the union $|A_j| = \sum_{x,y,z} A_{x,y,z,j}$ and $|B_j| = \sum_{x,y,z} B_{x,y,z,j}$.

Evaluation metrics We used the picking F1 score to assess particle picking performance. The F1-score is the harmonic mean of precision and recall:

$$\text{F1} = 2 \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \tag{16}$$

$$\text{precision} = \frac{\text{tp}}{\text{pred}} \tag{17}$$

$$\text{recall} = \frac{\text{tp}}{\text{gt}} \tag{18}$$

where $\text{tp}, \text{pred}, \text{gt}$ denote the numbers of true positives, predicted particles, and ground truth particles, respectively. A predicted particle was considered to be a true positive if the centroid of the predicted particle was located within the boundary of a ground truth particle with the same class label. For the evaluation of model performance on D5 from DeePiCt [15], we followed DeePiCt's evaluation metric to define the true positives as those predicted particles whose coordinates overlap with a ground truth particle within a tolerance radius (10 voxels, 135 Å).

Hyperparameters and Optimization The stochastic gradient descent (SGD) algorithm [59, 60] was used to update model parameters. SGD encompasses a learning rate of `lr=0.0001`, momentum parameterized at `momentum=0.9`, and a weight decay set to `weight_decay=0.0005`. Efficient training was facilitated with batches of size `batch_size=25`. The model was subjected to 10,000 training iterations, as captured by `train_iters=10000`, with validation step performed at every 500 iterations, `valid_iters=500`. The meta-learning facets of our architecture utilized parameters such as `meta_alpha=0.0`, and `lamda=1.0`, among other configurations which are available in the documentation of our code. For a dataset with 10 tomograms, the training process typically takes a few hours on an Nvidia Tesla A100 (80 GB) GPU.