ProPicker: Promptable Segmentation for Particle Picking in Cryogenic Electron Tomography

Simon Wiedemann[1], Zalan Fabian[2], Mahdi Soltanolkotabi[2], Reinhard Heckel[1]
[1] Department of Computer Engineering, Technical University of Munich
[2] Department of Electrical and Computer Engineering, University of Southern California

Abstract

Cryogenic electron tomography (cryo-ET) can produce detailed 3D images, called tomograms, of cellular environments. An essential step of cryo-ET data analysis is to find all instances of a particle in a set of tomograms. This particle picking task is a challenging 3D object detection problem due to strong noise and artifacts in the tomograms, as well as the diverse, crowded cellular environment. To enable a fast, flexible, and data-efficient workflow for particle picking, we propose ProPicker, a pretrained, promptable 3D segmentation model. By specifying a prompt, the model is conditioned to detect a specific particle of interest. The prompted model can be used for particle picking as is or can be fine-tuned to a particle-specific picker. Through experiments on simulated and real-world tomograms, we demonstrate that based on a single prompt, ProPicker can accurately detect a range of particles; in some cases even such that are not part of the training set. ProPicker can achieve performance close to or on par with state-of-the-art prompt-based pickers, while being up to an order of magnitude faster. We also show that fine-tuning ProPicker outperforms state-of-the-art particle-specific pickers if limited training data is available.

1 Introduction

Cryo-electron tomography (cryo-ET) is gaining popularity due to its unique ability to image biological macromolecules in their native environments [TB20; HS21]. An ambitious goal of cryo-ET is to obtain an atlas of the cell with all of its constituent macromolecules mapped in their native environment. This would revolutionize our understanding of essential protein interactions and has the potential to provide breakthroughs in modern medicine spanning cell biology to drug discovery [Bod+23].

In this paper, we focus on particle picking, which is the task of finding all instances of a particle of interest in 3D volumes, called tomograms, obtained with cryo-ET. Particle picking is an essential step and often a bottleneck in important cryo-ET analysis pipelines [Gen+23].

Particle picking is a 3D object detection problem and is particularly challenging for various reasons. Due to the fundamental limitations of data acquisition in cryo-ET, tomograms have a very low signal-to-noise ratio and exhibit strong artifacts. Moreover, tomograms are often large ($200 \times 1000 \times 1000$ voxels and higher), and cryo-ET datasets can consist of hundreds of tomograms, making their analysis computationally demanding [Gen+23; Zen+23]. Finally, due to the significant diversity in protein types within the cell, there is a vast array of unique object classes to be detected, many of which only differ subtly, rendering differentiation challenging. For instance, the human body alone is estimated to contain more than 20,000 unique proteins [LB21].

A particle picking method should be fast and flexible, i.e., should be able to accurately pick any particle of interest using no or little additional data for, e.g., model training or fine-tuning. Existing methods for particle picking are either slow or not flexible. Most state-of-the-art methods are based on deep learning

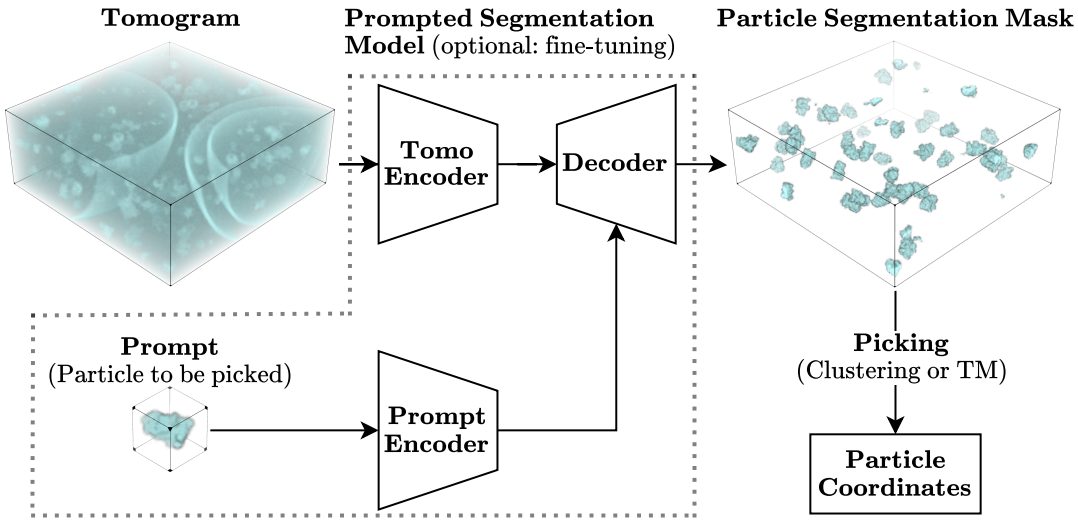

Figure 1: Overview of ProPicker: First, ProPicker extracts features of the particle to be picked in the tomogram with a prompt encoder. Conditioned on the features, ProPicker segments the tomogram with a pre-trained segmentation model. Finally, the the particle coordinates are extracted from the output segmentation mask. The segmentation model can be fine-tuned on annotated data. The 3D objects in this figure have been rendered from data from the SHREC 2021 dataset [Gub+20].

models which can detect only a small, fixed set of particles and require training on large amounts of labeled data, which is particularly difficult to obtain in the cryo-ET domain.

In this work, we propose ProPicker, a **Pro**mptable particle **Picker** that can target any particle of interest via a versatile prompting mechanism. ProPicker is trained on a large and diverse synthetic dataset and leverages an efficient 3D segmentation network to segment particles of interest in tomograms and to accurately locate their positions. The promptable design enables the user to select which single particle class is targeted by the segmentation model (prompt-based picking). Particle-specific fine-tuning can be used to improve over ProPicker's out-of-the-box performance. The promptable design and overall workflow of ProPicker is inspired by methods in the general domain like the Segment Anything Model (SAM) [Rav+24; Kir+23] and CLIPSeg [LE22].

ProPicker is able to pick new particles unseen during training in synthetic tomograms with high F1 scores based on a single prompt (see Section 4.1.1). On two real-world datasets, DS-10440 [Pec+24] and EMPIAR-10988 [DTT+23], we obtain decent results for prompt-based picking of large particles and those that produce strong contrast, e.g., ribosomes and apoferritin.

In both synthetic and real-world settings, we also encountered situations where prompt-based picking with ProPicker and other flexible baselines, i.e., TomoTwin [Ric+22] and CryoSAM [Zha+24] do not give satisfactory results. In Section 4.1.2 and Section 4.2, we discuss such scenarios and demonstrate that fine-tuning ProPicker on little data (even $\leq 25\%$ of a tomogram) can improve the F1 score by factors of up to 4, depending on the particle. Furthermore, we show that fine-tuning ProPicker requires less data than training the state-of-the-art particle-specific method, DeepETPicker [Liu+24], to achieve comparable picking performance. Finally, ProPicker is the fastest among flexible pickers capable of detecting particles based on a single prompt or reference, being up to an order of magnitude faster than the state-of-the-art method, TomoTwin [Ric+23] (see Section 4.1.1 and Appendix E).

Our findings illustrate the promising capabilities of ProPicker, while highlighting the need for large, diverse, and extensively annotated real-world training datasets to unlock its full potential (see Section 5).

2 Background & Related Work

Deep learning methods have already revolutionized particle picking from 2D micrographs in the context of single particle cryo-EM [Wan+16; Bep+19; Wag+19]. They are also on the rise for picking particles in 3D tomograms produced with cryo-ET. However, especially in cryo-ET, traditional methods like template matching still play an important role.

Particle Picking with Template Matching. Template Matching is the most widely used method for particle picking in cryo-ET [Boh+00; CL+24]. Template-matching-based methods compare a template of the particle to be picked to candidate sub-tomograms extracted by a 3D sliding window. This approach is highly flexible, as it can target any particle as long as a template is available. However, template matching is computationally demanding (up to several hours per tomogram), as the stride of the sliding window needs to be small for accurate picking [Gen+23; MSK24].

Building upon classical template-based approaches, the recent TomoTwin method [Ric+23] utilizes a learned convolutional encoder to map both template and sub-tomogram into a structured latent space, where similarity is evaluated. The training of the encoder follows a deep metric learning approach such that latent representations of particles of the same class have high cosine similarity, while those of different classes have low cosine similarity. TomoTwin can outperform classical template matching in terms of speed and performance, and is more convenient to use [Ric+23].

Particle Picking with Deep Learning-Based Object Detection. Particle pickers using deep learning-based object detection often outperform template matching in terms of performance and picking speed [Gub+20; Gen+23]. Many such pickers use a convolutional network to segment particles of interest belonging to one or more classes and produce candidate particle locations by clustering the predicted segmentation masks. Examples for such methods include DeepFinder [Moe+21], DeePiCt [DTT+23], and DeepETPicker [Liu+24]. Deep learning-based object detection approaches for particle picking are typically significantly faster than template-matching-like methods [Gub+20]. However, they are trained on datasets containing a fixed set of particles of interest and are therefore limited to picking particles seen during training.

As a first step towards an entirely training-free segmentation-based picker, Zhao et al. [Zha+24] recently proposed CryoSAM. CryoSAM takes user-specified center coordinates of target proteins as input. Leveraging DINOv2 [Oqu+23], it extracts features around these points and generates new prompts via feature matching with other parts of the tomogram. Based on the newly generated prompts, the 2D segmentation model SAM [Kir+23] is then used to segment particles.

Relation of ProPicker to Existing Particle Pickers. ProPicker shares the setup of picking particles based on a single observation (without any training or fine-tuning) with TomoTwin and CryoSAM. In contrast to TomoTwin, ProPicker does not rely on slow template matching, but is based on fast 3D segmentation. CryoSAM is segmentation-based, but the technique used differs significantly from ProPicker: CryoSAM does not involve any training on cryo-ET specific data, but relies on 2D models trained in the general domain on mostly natural images. ProPicker on the other hand uses 3D segmentation and has been trained on domain-specific cryo-ET data.

The second application of ProPicker, i.e., fine-tuning ProPicker on annotated data to pick a specific particle, assumes the same problem setup as deep-learning-based pickers which involve training a particle-specific segmentation network, e.g., DeepETPicker [Liu+24].

3 ProPicker: Promptable Segmentation for Particle Picking

After formally stating the particle picking problem, we describe ProPicker's promptable segmentation model, and detail how to use the trained model for prompt-based particle picking and fine-tuning.

Problem statement. ProPicker takes a tomogram $\boldsymbol{x} \in \mathbb{R}^{n \times n \times n}$, and a prompt $\boldsymbol{p} \in \mathbb{R}^{m \times m \times m}$ representing the particle of interest as input. The output is a finite set $\mathcal{C} \subset \{1, ..., n\}^3$ that contains predicted coordinates of the voxels of the center of the particles.

Promptable segmentation model. The ProPicker model shown in Figure 1 consists of a prompt encoder and a conditional segmentation model. The prompt encoder $\varepsilon : \mathbb{R}^{m \times m \times m} \to \mathbb{R}^d$ extracts a feature vector $\varepsilon(\boldsymbol{p}) \in \mathbb{R}^d$ that encodes information required for efficiently detecting the particle in tomograms.

Given an input volume $\boldsymbol{x} \in \mathbb{R}^{n \times n \times n}$ and prompt $\boldsymbol{p}$, our promptable segmentation model $\mathcal{S} : \mathbb{R}^{n \times n \times n} \times \mathbb{R}^d \to \mathbb{R}^{n \times n \times n}$ can be conditioned on the input prompt in order to steer the output map $\boldsymbol{y} \in \{0, 1\}^{n \times n \times n}$ to the desired particle class, that is $\boldsymbol{y} = \mathcal{S}\big(\boldsymbol{x}; \varepsilon(\boldsymbol{p})\big)$. The model output $\boldsymbol{y}$ is the voxel-wise prediction of the model with respect to the absence/presence of the particle described in the input prompt. We detail the architecture of the conditional segmentation model in Appendix B.

We train the promptable segmentation model on a synthetic dataset of tomograms containing a large variety of diverse particles and corresponding ground-truth segmentation masks (see Section 4).

Prompt-based particle picking with ProPicker. First, we choose a prompt for the particle of interest. Next, we embed the prompt and segment the tomogram using ProPicker. As tomograms are typically very large, we segment the volume using a strided 3D sliding window approach. Specifically, we slide a moderately sized window across the tomogram to extract sub-tomograms, and segment each sub-tomogram individually. We obtain a full-sized segmentation mask for the tomogram by combining the sub-tomogram level masks, averaging overlapping regions. Finally, we extract predicted particle center coordinates $\mathcal{C}$ from the segmentation mask $\boldsymbol{y}$. For this, we propose two strategies:

- **Clustering-based picking (ProPicker-C):** We detect clusters in the segmentation mask by finding connected components. The centroid of each cluster is a predicted particle center. The precision of this approach can be improved by leveraging prior information about the target particle size by excluding clusters that are too small or too large.
- **Template-matching-based picking (ProPicker-TM):** We apply a template-matching-based picker (in this work, we use TomoTwin [Ric+23]) to the input tomogram over regions where our segmentation mask predicts the presence of a particle.

Our default variant is ProPicker-C. ProPicker-TM is particularly useful to speed up an existing, well-functioning template matching pipeline. Another use-case is picking in densely packed environments where particle clusters are not easily separable, or eliminating false positives as shown in Section 4.1.2.

Particle-specific fine-tuning of ProPicker. Fine-tuning requires (parts of) one or more tomograms with corresponding binary segmentation masks of the particle of interest, as well as a single prompt for the particle of interest. We only fine-tune the segmentation model and keep the prompt and the prompt encoder fixed. Picking with the fine-tuned segmentation model works as in the prompt-based case, i.e. either via clustering or template matching. Note that the out-of-the-box ProPicker model can help during the annotation of tomograms for fine-tuning, e.g., we can manually select true positives from the picks produced with prompt-based picking.

4 Experiments

We start with describing the setup of ProPicker used throughout the experiments and the training and evaluation setup, and then present our experiments on prompt-based picking (Section 4.1) and fine-tuning (Section 4.2).

ProPicker Architecture. As prompt encoder, we use the TomoTwin encoder [Ric+23] which maps a $37 \times 37 \times 37$ sub-tomogram, that has been extracted from one of the tomograms and shows the particle of interest, to a 32-dimensional vector. Vectors produced by the TomoTwin encoder are well-suited representations of the particle of interest as they have been shown to be useful for particle classification based on cosine similarity (for more details, see Section 2).

We use a larger and slightly modified version of the 3D U-Net used in DeepETPicker [Liu+24] as the encoder-decoder segmentation model. Details on the U-Net and how we condition it on the output of the TomoTwin prompt encoder are in Appendix B.

Training Dataset of ProPicker. We train ProPicker on realistically simulated tomograms from Rice et al. [Ric+23] and Gubins et al. [Gub+20], which have also been used to train TomoTwin. Our training set is a subset of TomoTwin's training data and consists of 85 tomograms containing a total of 121 unique protein types, as well as gold fiducial markers, vesicles and filaments. Each tomogram contains around 1500 protein instances, each belonging to a set of $10-13$ unique protein types. We train on sub-tomograms of size $64 \times 64 \times 64$ extracted from all tomograms with a 3D sliding window with stride 32. Details on model training are in Appendix C.

Evaluation Metrics. We measure picking performance with F1 score, and report best-case performance with hyperparameters like thresholds optimized on test data for all methods following common practice [Ric+23]. Details are in Appendix D.

4.1 Prompt-Based Picking

In this Section, we discuss experiments on prompt-based picking, which requires only a single observation of the particle as prompt, and is, therefore, highly data efficient. We also discuss cases where prompt-based picking fails (Section 4.1.2).

Baselines. As baselines, we consider pickers that match our setup, i.e., that are able to pick particles based on a single observation or prompt. State-of-the-art baselines that fall into this category are TomoTwin [Ric+22] and CryoSAM [Zha+24].

4.1.1 Prompt-Based Picking of Unseen Particles in a Synthetic Tomogram

We first demonstrate that, based on a single prompt, ProPicker can accurately pick particles it has not seen during training in a synthetic tomogram generated with the same simulator as the training data.

Dataset & Setup. We test prompt-based picking with ProPicker on a regenerated version of the generalization tomogram from Rice et al. [Ric+23], a tomogram with dimensions $200 \times 512 \times 512$ voxels containing 7 unique particles that are not part of our training set. We applied TomoTwin and ProPicker directly to the tomogram using the same prompts. Since we were unable to produce any picks with CryoSAM when applying it to the noisy tomogram, we applied it to the clean ground truth tomogram that underlies the simulation of the noisy tomogram to probe the best-case performance.

Method / PDB	1avo	1e9r	1fpy	1fzg	1jz8	1oao	2df7	**Mean**	**Runtime** (s)
CryoSAM	0.21	0.30	0.43	0.22	0.50	0.45	0.70	0.40	262
TomoTwin (stride=2)	0.59	0.72	0.86	0.48	0.73	0.70	0.92	0.71	2620
TomoTwin (stride=4)	0.55	0.70	0.81	0.49	0.69	0.63	0.92	0.68	362
TomoTwin (stride=8)	0.07	0.30	0.27	0.33	0.14	0.12	0.88	0.30	80
ProPicker-TM	0.60	0.72	0.84	0.58	**0.78**	0.65	0.92	0.73	154
ProPicker-C	**0.72**	**0.82**	**0.88**	**0.68**	0.69	**0.78**	**0.98**	**0.79**	77

Table 1: Best-case F1 picking scores and runtime (measured on a single NVIDIA L40 GPU) for CryoSAM, TomoTwin, and ProPicker-C on eight particles not seen during training. The F1 scores for CryoSAM have been obtained on the clean, ground-truth tomogram, as we could not produce good results on the noisy tomogram used as input for TomoTwin and ProPicker.

Results. As seen in Table 1, ProPicker-C performs best on most particles. Its mean performance is better than TomoTwin, which is the state-of-the-art method. CryoSAM achieves lower F1 scores than TomoTwin and ProPicker for all particles even though we applied it to clean ground truth tomograms. Upon inspection, CryoSAM's segmentation masks confuse particles. It is likely that both DINOv2 [Oqu+23] and SAM [Kir+23], which are both used in CryoSAM without any fine-tuning on cryo-ET data, fail to discriminate between the particles even in the absence of noise, highlighting the need for domain-specific training for accurate particle picking.

We also compare the runtimes of all methods. In Table 1, we report the time it takes to pick a single particle, not accounting for speedups achievable via caching, e.g., for TomoTwin, the embeddings of sub-tomograms only have to be calculated once if one wants to pick multiple particles within a single tomogram.

TomoTwin is the slowest method as it relies on template matching and processes the tomogram with a finely-strided (the default stride is 2 voxels) sliding window to achieve high picking performance, which comes at the cost of a high runtime. ProPicker, too, employs a sliding window approach but due to its 3D segmentation network, we can choose significantly larger strides (our default is 32 voxels). TomoTwin can be sped up by choosing larger strides: Increasing the stride from 2 to 4 voxels results in a much shorter runtime and slightly lower performance, but further increasing the stride results in a large performance drop. ProPicker is significantly less sensitive to the sliding window stride and can be sped up even further while losing only little performance. A more extensive analysis of the speed vs. performance trade-off of TomoTwin and ProPicker can be found in Appendix E.

Our experiment shows that ProPicker is able to accurately pick unseen particles in a context that is familiar from the training set.

4.1.2 Prompt-Based Picking in a Real-World Tomogram

To further showcase the generalizability of ProPicker, we demonstrate that ProPicker can generalize to real-world tomograms, even though we have trained exclusively on synthetic data. In this Section, we only compare to TomoTwin which we find to be the stronger baseline, and as we were unable to produce good results for CryoSAM.

Dataset & Setup. We consider the DS-10440 dataset generated by Peck et al. [Pec+24], which contains 7 annotated tomograms and is available through the Cryo ET Data Portal [Erm+24]. Among other particles, the dataset contains expert annotations for cytosolic ribosomes and apoferritin. Apoferritin has a molecular weight of 450 kilodalton which is 10.47% of that of the ribosomes in the dataset (4.3 megadalton), and is

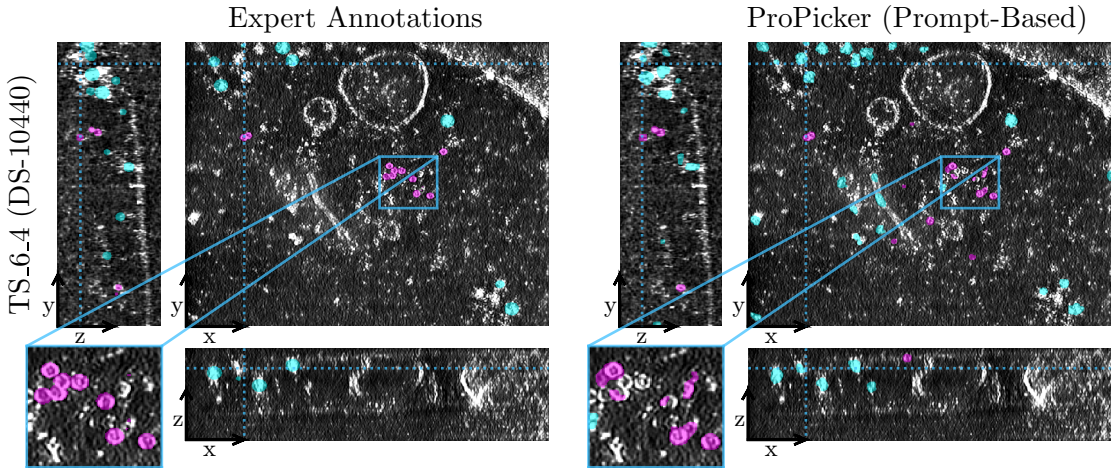

Figure 2: Slices through a single real-world tomogram from the DS-10440 dataset taken from the Cryo ET Data Portal. Ground-truth annotations and ProPicker segmentation masks for apoferritin and cytosolic ribosomes are shown in pink and blue respectively. The tomogram underlying the segmentation masks has been denoised by Peck et al. [Pec+24]. The ProPicker masks have been obtained from the raw, noisy tomogram (not shown). The images are high resolution, and we recommend readers to zoom in.

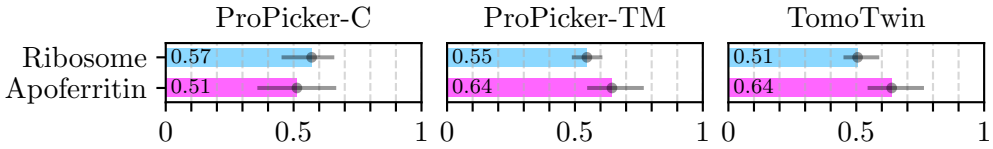

Figure 3: Best-case F1 scores for apoferritin and cytosolic ribosomes picked in the 7 raw real-world tomograms from the DS-10440 dataset. For ProPicker-TM, we use TomoTwin as template matching picker. The error bars indicate the minimum and maximum F1 score across the 7 tomograms in the dataset.

also smaller in size than the ribosomes [Pec+24]. Our goal is to pick all instances of the ribosomes and the apoferritin in all 7 tomograms based on a single prompt each.

Results. As can be seen in Figure 2 and Figure 3, ProPicker and TomoTwin perform decently at picking apoferritin and the ribosomes. ProPicker-TM performs best and ProPicker-C worst. ProPicker-C performs worst due to false positives in the predicted segmentation mask and crowded areas (see, e.g. the zoomed-in region in Figure 2), where several instances of the same particle appear close together, leading to inseparable clusters. This makes cluster-based picking challenging.

ProPicker-TM and TomoTwin (both leveraging template matching) eliminates false positives and separates overlapping clusters which yields higher picking performances. Overall, ProPicker-TM performs on-par (apoferritin) or better (ribosomes) than TomoTwin on its own and at the same time offers large speedups: On average, the ProPicker segmentation masks used for ProPicker-TM masked out 99.1% of all voxels for apoferritin and 95.9% of all voxels for ribosomes, resulting in relative speedups over TomoTwin which are similar to those shown in Table 1.

We could not produce good picks for the other particles contained in the tomogram, which are virus-like particles, beta amylase, beta galactosidase, and thyroglobulin, with neither TomoTwin, nor ProPicker. Beta amylase, beta galactosidase, and thyroglobulin are very small, produce little contrast and are therefore considered very hard to pick [Pec+24]. The virus-like particles in contrast are large and clearly visible even in the noisy tomograms. However, the training dataset of ProPicker and TomoTwin contains mainly

proteins and no virus-like particles which might explain the low performance and highlights the importance of training on more diverse datasets.

Technical Remarks. For this experiment, we smoothened the segmentation masks predicted by ProPicker for ribosomes and apoferritin by applying a 2×2 mean filter $5\times$, followed by 2×2 max pooling. This optional smoothing follows [Liu+24], and can slightly reduce false positives, and enlarge cluster size and connectivity in the segmentation mask. Therefore, it mainly improves the performance of ProPicker-C. However, we have observed that the smoothing can give worse results in crowded tomograms because it can create overlap of segmentation masks for individual particles, rendering the not separable with clustering. For results for ProPicker-C and ProPicker-TM without smoothing, see Appendix F.

4.2 Improving Over Prompt-Based Picking with Fine-Tuning

In some cases, ProPicker is able to accurately pick particles based on a single prompt, but there are particles which neither ProPicker nor TomoTwin are able to locate, as discussed in Section 4.1. Here, we show that in such cases ProPicker's performance can be improved through fine-tuning on little data.

Baselines. Aside from TomoTwin, we compare to particle-specific deep-learning-based pickers, specifically to the recent DeepETPicker method [Liu+24]. DeepETPicker is segmentation-based and uses a 3D U-Net-like convolutional architecture. It is a suitable baseline as it gives state-of-the-art performance [Liu+24], and as we chose ProPicker-C's segmentation model as a scaled-up version of DeepETPicker's U-Net. We also tested DeepFinder [Moe+21], which is conceptually similar to DeepETPicker, but found that its performance and data efficiency are lower than those of DeepETPicker, which is in line with the comparison of the two methods by Liu et al. [Liu+24].

We consider two variants of DeepETPicker. For the first variant, "DeepETPicker (1 Class)", we train one DeepETPicker model for each particle separately. This is the same setup as when we fine-tune ProPicker. For the second variant, "DeepETPicker (8 Classes)", we train one DeepETPicker model to pick all 8 particles *simultaneously*. Moebel et al. [Moe+21] observed that multi-class training can yield substantial improvements in performance for difficult-to-pick particles. In contrast to the single class setup, the multi-class setup requires annotations for *all 8 particles.*

Finally, recall that ProPicker and DeepETPicker use a very similar 3D U-Net architecture, but ProPicker's model has around $42\times$ more parameters. To ensure that the difference in model size does not distort our experiments, we re-ran parts of the experiments discussed below with a variant of DeepETPicker in which we scaled its U-Net to the same size as ProPicker's U-Net (100 million parameters). The larger U-Net performed on-par or slightly worse (see Appendix G).

4.2.1 Prompt-Based Picking and Fine-Tuning for Unseen Particles in Synthetic Tomograms

We first fine-tune ProPicker on a set of synthetic tomograms.

Dataset & Setup. We consider a set of 8 tomograms from TomoTwin's training set, each of which contains instances of 8 unique particles. For our fine-tuning experiment, we first re-trained both TomoTwin and ProPicker, exlcuding the 8 tomograms containing the 8 particles from the training set. As, by design, TomoTwin's original training set contains particles whose structures are all very different from one another (see [Ric+23] for details), all 8 particles are hard, unseen examples for ProPicker and TomoTwin but the context is similar to that during training as the tomograms come from the same simulator.

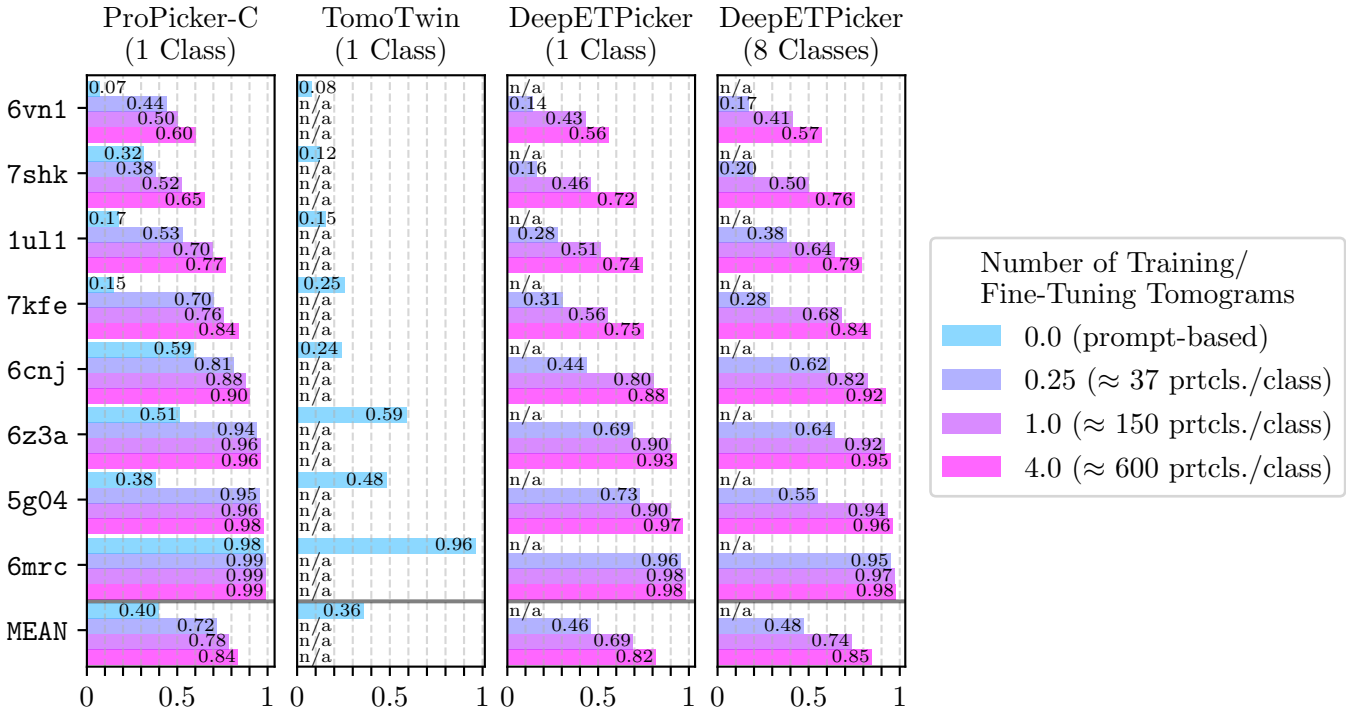

Figure 4: Best-case F1 scores of ProPicker-C, TomoTwin and DeepETPicker versus amount of fine-tuning data for unseen particles. Out of the 8 available tomograms, we used up to 4 for fine-tuning, one for validation the remaining 3 for testing. All tomograms consist of $200 \times 512 \times 512$ voxels.

Results. Figure 4 shows the results of our fine-tuning experiment. First note that both TomoTwin, which we re-trained without the 8 particles from the test set, and ProPicker obtain decent (`6g04, 6cnj, 6z3a`) or even near-perfect (`6mrc`) picking F1 scores for some particles, while struggling to pick others.

Fine-tuning ProPicker-C significantly boosts picking performance: Depending on the particle, fine-tuning on as little as 25% of a single tomogram improves the performance up to more than 4× (`7kfe`). The performance of the fine-tuned models saturates quickly as more data becomes available.

If little data *or* only annotations for the particle of interest are available, fine-tuning ProPicker-C yields superior performance compared to both variants of DeepETPicker: When training/fine-tuning on up to one single tomogram, the fine-tuned ProPicker-C outperforms DeepETPicker (1 Class) and DeepETPicker (8 Classes). The performance gap between the methods narrows as more data becomes available.

DeepETPicker (8 Classes) which is trained on all 8 classes simutaneously performs on par or slightly better than the best ProPicker-C pickers if enough training data is available. The improved performance over the single-class version comes at the cost of having to annotate *all* particles in the tomogram even if one is only interested in a single one.

Further improving the data efficiency of fine-tuning ProPicker with techniques for data-efficient training, e.g. positive un-labeled learning as proposed by Huang et al. [Hua+22] and Uddin et al. [Udd+24] in the context of learning from sparse/incomplete annotations, is a promising direction for future work.

4.2.2 Prompt-Based Picking and Fine-Tuning for Ribosomes in *In-Situ* Tomograms

Our final experiment is on a set of crowded *in-situ* tomograms. *In-situ* data is among the most interesting, yet most challenging types of data that is studied with cryo-ET.

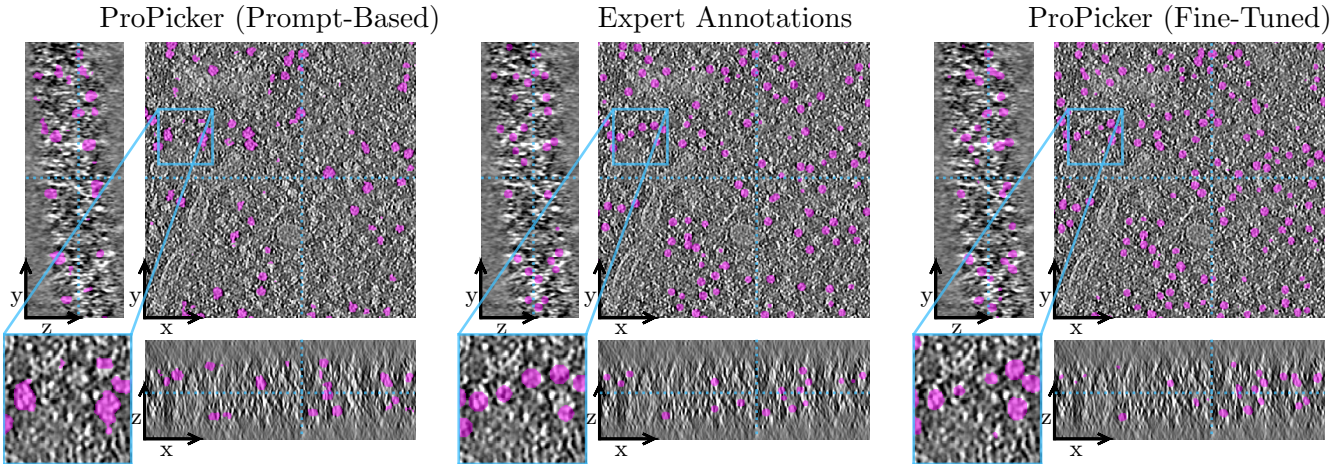

Figure 5: Slices through a single real-world tomogram (TS_30) from EMPIAR-10988 and expert annotations in the center panel by De Teresa-Trueba et al. [DTT+23]. Fine-tuning ProPicker (right panel) significantly increases the number of picked particles compared to prompt-based picking (left panel). The images are high resolution, and we recommend readers to zoom in.

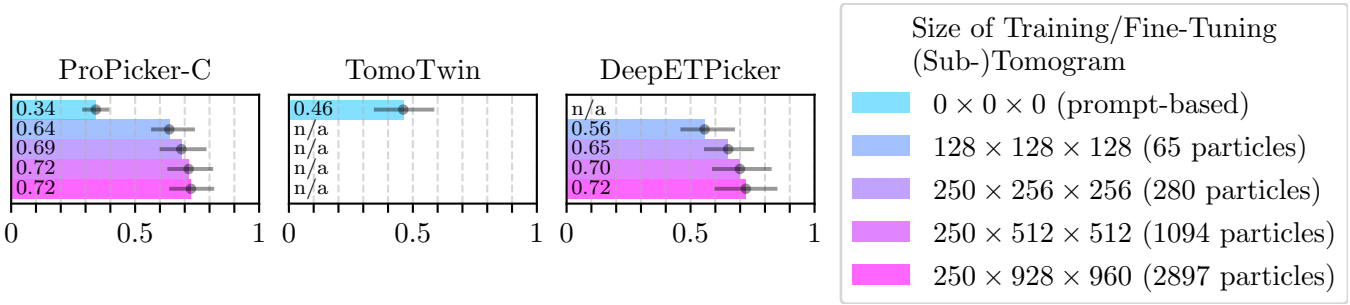

Figure 6: Best-case F1 scores of ProPicker-C, TomoTwin and DeepETPicker versus the amount of training/fine-tuning data for the ribosomes in the EMPIAR-10988 tomograms. The bars show the average scores across 9 (defocus-only) tomograms, excluding tomogram TS_29, which we used for training.

Dataset & Setup. We consider the EMPIAR-10988 dataset by De Teresa-Trueba et al. [DTT+23]. The tomograms in the EMPIAR-10988 dataset mainly show ribosomes within *S. pombe* cells.

Our goal is to pick all ribosomes in 9 (defocus-only) tomograms. We fine-tune ProPicker-C on center-crops of different sizes which we extract from a single tomogram (TS_29). We use center-crops from another tomogram (TS_30) as validation data. As reference, we train DeepETPicker models with the same setup.

In addition to annotations for ribosomes, the EMPIAR-10988 dataset also contains annotations for fatty acid synthases (FAS). We were unable to produce prompt-based picking results for FAS with neither ProPicker nor TomoTwin, therefore we do not include them in this fine-tuning experiment. The poor performance is not surprising, as FAS is hard to pick and even pickers that have been explicitly trained on EMPIAR-10988 for picking FAS obtain low F1 scores [DTT+23].

Results. In Figure 6, we see that TomoTwin outperforms ProPicker-C in prompt-based picking, but fine-tuning even on the smallest sub-tomogram which contains only 65 particles results in a large improvement over TomoTwin. Due to its pre-training, the fine-tuned ProPicker-C outperforms DeepETPicker in the low data regime, but the difference in performance diminishes as more data becomes available.

Technical Remarks. The F1 scores shown in Figure 6 are lower than the ones reported by De Teresa-Trueba et al. [DTT+23] for DeepFinder, even though DeepETPicker generally outperforms DeepFinder [Liu+24]. This is because we use a stricter true-positive criterion in order to be consistent with the other experiments in the paper and because we only used (parts of) a single tomogram for training.

We obtained all results for training/fine-tuning DeepETPicker and ProPicker discussed in this section using the binary cross entropy loss. Binary cross entropy is ProPicker's default training and fine-tuning loss, whereas DeepETPicker uses the Dice loss [Sud+17] as default. When using the Dice loss, we observed overlapping segmentation masks for individual particles due to the highly crowded nature of the *S. pombe* cells. Such overlapping parts cannot be easily separated via clustering and, therefore, yield lower picking performance: The F1 score when training DeepETPicker on the full tomogram with the Dice loss was around 0.1 lower.

5 Discussion & Conclusion

In this work, we have proposed ProPicker, a particle picking method for cryo-ET that leverages a promptable 3D segmentation model trained on a large and diverse synthetic dataset.

ProPicker can target any particle of interest by specifying a single example (prompting), and therefore belongs to the class of flexible pickers like TomoTwin [Ric+23] and CryoSAM [Zha+24]. In Section 4.1 and Section 4.2.2, we have seen that such flexible pickers can, in some cases, achieve good performance on particles they have not seen during training and on tomograms that significantly differ from their training data.

However, in Section 4.2, we have seen that the performance of prompt picking with TomoTwin and ProPicker falls short compared to methods that involve training dataset- and particle-specific models, such as DeepETPicker [Liu+24]. We have also seen that we can close this performance gap for ProPicker by fine-tuning, and that fine-tuning ProPicker is more data-efficient compared to DeepETPicker.

We believe that pre-trained flexible particle pickers like TomoTwin and ProPicker will play an important role in cryo-ET particle picking in the future, as they can greatly reduce human annotation effort.

A promising direction for future work is to make such flexible pickers more robust in order to improve the performance achievable through prompt-based picking and fine-tuning. It is widely accepted that training on larger and more diverse datasets improves the robustness of deep learning models [Rad+21; Fan+22; LH24]. Therefore, we believe that a good next step towards a robust universal picker is to collect large and diverse datasets of real-world tomograms with ground-truth particle annotations for a variety of particles for training. Large-scale efforts to do so have recently been initiated [Ish+23; Erm+24].

We expect the robustness of methods like TomoTwin and ProPicker to improve with more and better training data. However, given the high diversity of biological specimen and macromolecules studied with cryo-ET, we believe that developing a flexible picker that can robustly generalize to arbitrary particles and contexts may not be possible. Therefore, particle-specific fine-tuning, as demonstrated for ProPicker in section 4.2, will most likely continue to play an important role in achieving the best picking performance for individual datasets.

Code Availability

Code for prompt-based picking, as well as training and fine-tuning ProPicker is on GitHub: `https://github.com/MLI-lab/ProPicker`.

References

[Bep+19] T. Bepler, A. Morin, M. Rapp, J. Brasch, L. Shapiro, A. J. Noble, and B. Berger. "Positive-unlabeled convolutional neural networks for particle picking in cryo-electron micrographs". In: *Nature Methods* 16.11 (2019), 1153–1160.

[Bod+23] S. Bodakuntla, C. C. Kuhn, C. Biertümpfel, and N. Mizuno. "Cryo-electron microscopy in the fight against COVID-19—mechanism of virus entry". In: *Frontiers in Molecular Biosciences* 10 (2023), 1252529.

[Boh+00] J. Bohm, A. S. Frangakis, R. Hegerl, S. Nickell, D. Typke, and W. Baumeister. "Toward detecting and identifying macromolecules in a cellular context: template matching applied to electron tomograms". In: *Proceedings of the National Academy of Sciences* 97 (2000), 14245–14250.

[CL+24] S. Cruz-León et al. "High-confidence 3D template matching for cryo-electron tomography". In: *Nature Communications* 15.1 (2024), 3992.

[DTT+23] I. De Teresa-Trueba et al. "Convolutional networks for supervised mining of molecular patterns within cellular context". In: *Nature Methods* 20.2 (2023), 284–294.

[Erm+24] U. Ermel, A. Cheng, J. X. Ni, J. Gadling, M. Venkatakrishnan, K. Evans, J. Asuncion, A. Sweet, J. Pourroy, Z. S. Wang, et al. "A data portal for providing standardized annotations for cryo-electron tomography". In: *Nature Methods* 21.12 (2024), 1–3.

[Fan+22] A. Fang, G. Ilharco, M. Wortsman, Y. Wan, V. Shankar, A. Dave, and L. Schmidt. "Data determines distributional robustness in contrastive language image pre-training (clip)". In: *International Conference on Machine Learning*. (2022).

[Gen+23] E. Genthe, S. Miletic, I. Tekkali, R. Hennell J., T. C. Marlovits, and P. Heuser. "PickYOLO: Fast deep learning particle detector for annotation of cryo electron tomograms". In: *Journal of Structural Biology* 215.3 (2023), 107990.

[Gub+20] I. Gubins, M. L. Chaillet, G. van Der Schot, R. C. Veltkamp, F. Förster, Y. Hao, X. Wan, X. Cui, F. Zhang, E. Moebel, et al. "SHREC 2020: Classification in cryo-electron tomograms". In: *Computers & Graphics* 91 (2020), 279–289.

[Hua+22] Q. Huang, Y. Zhou, H.-F. Liu, and A. Bartesaghi. "Accurate detection of proteins in cryo-electron tomograms from sparse labels". In: *European Conference on Computer Vision*. (2022).

[HS21] R. K. Hylton and M. T. Swulius. "Challenges and triumphs in cryo-electron tomography". In: *iScience* 24.9 (2021), 102959.

[IS15] S. Ioffe and C. Szegedy. "Batch normalization: accelerating deep network training by reducing internal covariate shift". In: *International Conference on Machine Learning*. (2015).

[Ish+23] A. Ishemgulova, A. J. Noble, T. Bepler, and A. D. Marco. *Preparation of labeled cryo-ET datasets for training and evaluation of machine learning models*. 2023.

[KB15] D. P. Kingma and J. Ba. "Adam: a method for stochastic optimization." In: *International Conference on Learning Representations*. (2015).

[Kir+23] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, et al. "Segment anything". In: *IEEE/CVF International Conference on Computer Vision*. (2023).

[LB21] Z.-L. Li and M. Buck. "Beyond history and "on a roll": The list of the most well-studied human protein structures and overall trends in the protein data bank". In: *Protein Science* 30.4 (2021), 745–760.

[LH24] K. Lin and R. Heckel. "Robustness of deep learning for accelerated MRI: benefits of diverse training data". In: *International Conference on Machine Learning.* (2024).

[Liu+24] G. Liu, T. Niu, M. Qiu, Y. Zhu, F. Sun, and G. Yang. "DeepETPicker: Fast and accurate 3D particle picking for cryo-electron tomography using weakly supervised deep learning". In: *Nature Communications* 15.1 (2024), 2090.

[LE22] T. Lüddecke and A. Ecker. "Image segmentation using text and image prompts". In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition.* (2022).

[MSK24] V. J. Maurer, M. Siggel, and J. Kosinski. "PyTME (Python Template Matching Engine): A fast, flexible, and multi-purpose template matching library for cryogenic electron microscopy data". In: *SoftwareX* 25 (2024), 101636.

[Moe+21] E. Moebel, A. Martinez-Sanchez, L. Lamm, R. D. Righetto, W. Wietrzynski, S. Albert, D. Larivière, E. Fourmentin, S. Pfeffer, J. Ortiz, et al. "Deep learning improves macromolecule identification in 3D cellular cryo-electron tomograms". In: *Nature Methods* 18.11 (2021), 1386–1394.

[Oqu+23] M. Oquab et al. "DINOv2: learning robust visual features without supervision". In: *arXiv:2304.07193* (2023).

[Pec+24] A. Peck, Y. Yu, J. Schwartz, A. Cheng, U. H. Ermel, S. Kandel, D. Kimanius, E. Montabana, D. Serwas, H. Siems, et al. "Annotating cryoET volumes: a machine learning challenge". In: *bioRxiv* (2024).

[Per+18] E. Perez, F. Strub, H. d. Vries, V. Dumoulin, and A. Courville. "FiLM: visual reasoning with a general conditioning layer". In: *AAAI Conference on Artificial Intelligence* 32.11 (2018).

[Rad+21] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al. "Learning transferable visual models from natural language supervision". In: *International Conference on Machine Learning.* (2021).

[Rav+24] N. Ravi, V. Gabeur, Y.-T. Hu, R. Hu, C. Ryali, T. Ma, H. Khedr, R. Rädle, C. Rolland, L. Gustafson, et al. "SAM 2: Segment anything in images and videos". In: *arXiv:2408.00714* (2024).

[Ric+22] G. Rice, T. Wagner, M. Stabrin, and S. Raunser. *TomoTwin demonstration dataset (1.0).* 2022. URL: `https://doi.org/10.5281/zenodo.7186070`.

[Ric+23] G. Rice, T. Wagner, M. Stabrin, O. Sitsel, D. Prumbaum, and S. Raunser. "TomoTwin: generalized 3D localization of macromolecules in cryo-electron tomograms with structural data mining". In: *Nature Methods* 20.66 (2023), 871–880.

[RFB15] O. Ronneberger, P. Fischer, and T. Brox. "U-Net: convolutional networks for biomedical image segmentation". In: *Medical Image Computing and Computer-Assisted Intervention.* (2015).

[Sud+17] C. H. Sudre, W. Li, T. Vercauteren, S. Ourselin, and M Jorge C. "Generalised dice overlap as a deep learning loss function for highly unbalanced segmentations". In: *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support.* (2017).

[TB20] M. Turk and W. Baumeister. "The promise and the challenges of cryo-electron tomography". In: *FEBS Letters* 594.20 (2020), 3243–3261.

[Udd+24] M. R. Uddin, A. Y. Ahmed, M. T. Tahmid, M. Z. Ul Alam, Z. Freyberg, and M. Xu. "TomoPicker: annotation-efficient particle picking in cryo-electron tomograms". In: *bioRxiv* (2024).

[UVL16] D. Ulyanov, A. Vedaldi, and V. Lempitsky. "Instance normalization: the missing ingredient for fast stylization". In: *arXiv:1607.08022* (2016).

[Wag+19] T. Wagner, F. Merino, M. Stabrin, T. Moriya, C. Antoni, A. Apelbaum, P. Hagel, O. Sitsel, T. Raisch, D. Prumbaum, et al. "SPHIRE-crYOLO is a fast and accurate fully automated particle picker for cryo-EM". In: *Communications Biology* 2.1 (2019), 218.

[WB16] W. Wan and J. Briggs. "Cryo-electron tomography and subtomogram averaging". In: *Methods in Enzymology* 579 (2016), 329–367.

[Wan+16] F. Wang, H. Gong, G. Liu, M. Li, C. Yan, T. Xia, X Li, and J. Zeng. "DeepPicker: a deep learning approach for fully automated particle picking in cryo-EM". In: *Journal of Structural Biology* 195.3 (2016), 325–336.

[Zen+23] X. Zeng, A. Kahng, L. Xue, J. Mahamid, Y.-W. Chang, and M. Xu. "High-throughput cryo-ET structural pattern mining by unsupervised deep iterative subtomogram clustering". In: *Proceedings of the National Academy of Sciences* 120.15 (2023), e2213149120.

[Zha+24] Y. Zhao, H. Bian, M. Mu, M. R. Uddin, Z. Li, X. Li, T. Wang, and M. Xu. "Cryosam: training-free cryoet tomogram segmentation with foundation models". In: *International Conference on Medical Image Computing and Computer-Assisted Intervention.* (2024).

Appendix

A Sub-Tomogram Averaging

Sub-tomogram averaging is a central application of cryo-ET and one of the main motivations for particle picking. The goal of sub-tomogram averaging is to get high-resolution models of particles like proteins based on data from tomograms. The idea is to align and average sub-tomograms that show the same particle, which suppresses the strong noise and artifacts present throughout the tomograms and can yield high-resolution models of the particle [WB16]. For this procedure, particle picking is an essential component and bottleneck in terms of both time demands and accuracy.

B Details on the Promptable Segmentation Model Architecture

Here, we provide details on our concrete choice for the segmentation model and how we condition it on a prompt.

B.1 Segmentation Model Architecture

We use a well-established convolutional 3D U-Net [RFB15], which is an encoder-decoder architecture (see Figure 1), as our segmentation model. The U-Net's architecture follows the one used in DeepETPicker [Liu+24], but we scaled it up for better performance on the large pre-training dataset: The encoder consists of 5 convolution-based spatial downsampling layers, and the corresponding decoder has 5 convolution-based spatial upsampling layers. The layers process 3D volumes with 108, 216, 432, and 864 channels, respectively, which amounts to 100 million trainable parameters. For comparison: the default U-Net in DeepETPicker, too, uses 5 up- and downsampling layers but with 24, 48, 96 and 192 channels, which results in 2.4 million trainable parameters.

Another difference between the U-Net we use in ProPicker and the one used in DeepETPicker is that we use instance normalization [UVL16] instead of batch normalization [IS15] as the former generalizes better and allows smaller batches during training, which is favorable for large models like ours.

B.2 Prompt Conditioning Technique

We condition each of the decoder's 5 spatial upsampling layers on the encoded prompt $\varepsilon(\boldsymbol{p}) \in \mathbb{R}^d$ with the FiLM [Per+18] approach, which works as follows: Let c be the number of channels (features) of an intermediate 3D feature map after upsampling. First, we multiply the encoded prompt $\varepsilon(\boldsymbol{p}) \in \mathbb{R}^d$ with two (learnable) matrices $\boldsymbol{A}, \boldsymbol{B} \in \mathbb{R}^{c \times d}$. We then map each channel $k \in \{1, ..., c\}$, with an affine transformation with slope $\big(\boldsymbol{A}\varepsilon(\boldsymbol{p})\big)_k \in \mathbb{R}$ and intercept $\big(\boldsymbol{B}\varepsilon(\boldsymbol{p})\big)_k \in \mathbb{R}$, which gives the conditioned feature map. We use a separate pair of learnable matrices $(\boldsymbol{A}, \boldsymbol{B})$ for each of the 5 up-sampling layers.

C Training ProPicker

To reduce computational cost, we keep the TomoTwin prompt encoder frozen during training. We train the segmentation model using the Adam optimizer [KB15] with a fixed learning rate of 0.01.

Forch each gradient step with Adam, we first randomly sample a batch of 8 sub-tomograms. Each sub-tomogram can contain particles that belong to up to 13 unique classes (see above). Next, we randomly sample 8 prompts. Each prompt corresponds to one particle class of which instances may be contained in the sub-tomogram. We pass each sub-tomogram and its corresponding 8 prompts through the conditional segmentation model. This yields a total of $64 = 8 \cdot 8$ predicted segmentation masks. Finally, we compute

the average voxel-wise binary cross entropy between the model outputs and the 64 single-class target masks as loss for the calculation of gradients.

D Evaluating (Best-Case) Picking Performance

We evaluate the picking performance of all methods using the F1 score, which is defined as the harmonic mean of precision and recall. Computing the F1 score requires a criterion based on which we consider a predicted particle position (given by 3 coordinates) to be a true-positive.

D.1 True-Positive Criterion

As our main baseline in this work is TomoTwin, we adopt the criterion used by Rice et al. [Ric+23], who consider a predicted position a true-positive if the intersection-over-union (IoU) between a cubic box centered at the predicted position and a cubic box of the same size centered at a ground-truth particle position is at least 0.6. The side length of the cubic box is chosen according to the size of the particle of interest.

D.2 Picking Hyperparameters and Computing Best-Case F1 Scores

All methods we consider in this work do not directly predict particle positions. Instead, they produce 3D segmentation masks (ProPicker, CryoSAM, DeepETPicker) or heatmaps (TomoTwin) from which particle locations are derived using clustering (ProPicker-C, CryoSAM, DeepETPicker) or peak-detection (ProPicker-TM, TomoTwin).

Both clustering and peak-detection use particle-specific hyper-parameters that have to be determined by the user, usually using a small calibration dataset. To facilitate the comparison of all methods and to avoid distortions due to insufficient calibration, we report best-case F1 scores for all methods. This entails tuning hyper-parameters on the test data itself. For each method, we tune the following hyper-parameters:

- **TomoTwin:** TomoTwin's peak-detection mainly relies on four thresholds: A minimum value for when a candidate point is considered a peak, a threshold for flood-filling which is used to enlarge the peak and two thresholds for the minimum and maximum number of voxels the flood-filled peak must consist of. The TomoTwin software package has an option to tune the peak threshold and the two size thresholds, which we used in all experiments. For all experiments on synthetic data and on DS-10440 [Pec+24], we found the default flood-filling threshold of 0.2 to yield good performance. For the experiments involving the EMPIAR 10988 dataset (Section 4.2.2), the default threshold resulted in poor performance, so we had to manually tune it, and found 0.018 to be optimal.

- **ProPicker-C:** The output of ProPicker's segmentation model is a sigmoided 3D volume of the same shape as the input tomogram. To produce particle positions via clustering, we first binarize the sigmoided volume based on a threshold, then cluster the binary volume based on a connected component analysis (which is inspired by from DeePiCt [DTT+23]), and, finally, remove clusters that are too lare or too small based on a threshold. For the best-case F1 score, we optimized the binarization threshold and the two size thresholds using a grid search. In practice, the size thresholds can be chosen according to the particle size.

- **ProPicker-TM:** ProPicker-TM applies template matching, we use TomoTwin in this work, to a segmentation mask produced with ProPicker. This requires first binarizing the sigmoided ProPicker output. In all experiments, we used 0.1 as threshold. Then we applied TomoTwin, ignoring all locations where the mask is 0, with its hyperparameters tuned as discussed above.

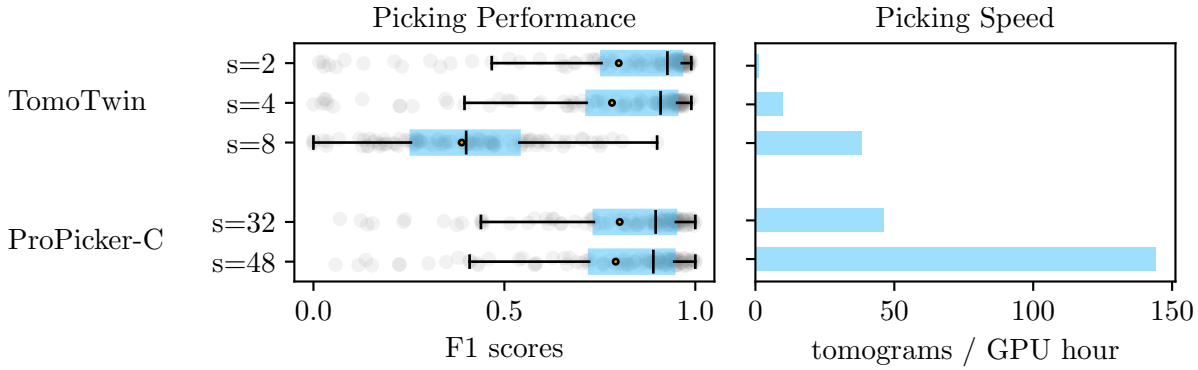

Figure A1: Picking performance and speed for our method with ProPicker-C and TomoTwin (TT) for 100 unique particles seen during training. Vertical markers are medians, circles are means.

- **CryoSAM:** CryoSAM outputs a binary segmentation mask. The implementation by Zhao et al. [Zha+24] does not contain a method to produce particle locations based on this mask, so we applied the same clustering as for ProPicker-C. As CryoSAM's output is already binary, we only optimized the cluster size thresholds.
- **DeepETPicker:** The implementation of DeepETPicker provided by Liu et al. [Liu+24] comes with a cluster-based picking routine. However, we found that the clustering method used for ProPicker-C yields better results for DeepETPicker on synthetic data. Therefore, and because it makes the comparison more principled, we used the same clustering routine as for ProPicker-C instead of DeepETPicker's and optimized the same hyperparameters.

E Picking Speed

Both TomoTwin and ProPicker process the tomogram in a 3D sliding window fashion. Therefore, inference time is cubically related to the window stride. However, large strides (small overlap) often results in low detection performance. Here, we explore this trade-off. To quantify speed, we report the throughput in tomograms per GPU hour measured on a single NVIDIA L40 GPU for picking a single particle of interest in a tomogram of size $200 \times 512 \times 512$.

As the speed at which a particle can be reliably picked depends on, e.g., the particle's size (especially for template matching methods like TomoTwin), we measure picking speed on a set of 11 tomograms which contain 108 particles in total. Both TomoTwin ProPicker and have seen all of these particles during training, but within in different tomograms, i.e., in different tomograms. We evaluate performance on unseen particles in Section 4.1.1.

As can be seen in Figure A1, TomoTwin with $s = 2$ performs best but is the slowest (< 2 tomograms / GPU hour). Increasing the stride to $s = 4$ results in a small drop in performance (-0.03 mean and median F1 score) and large relative speedup of around $8\times$. ProPicker-C with $s = 32$ performs as well as TomoTwin for $s = 4$, but is $4.5\times$ faster.

Achieving a picking speed similar to that of ProPicker-C with TomoTwin by increasing the stride to $s = 8$ comes at the cost of a large drop in performance. ProPicker-C on the other hand can be sped up further: Increasing the stride from $s = 32$ to $s = 48$ increases the speed by a factor 3.3 ($15\times$ speedup compared to TomoTwin with $s = 4$) while achieving almost the same performance.

ProPicker's speed advantage over TomoTwin, and template matching in general, is due to its use of a segmentation network for picking rather than the template matching approach of comparing candidate

particles to a template on fixed grid. The reason is that the "resolution" at which template matching can pick particles depends directly on the granularity of the grid. In general, small strides (fine grids) are required to obtain high accuracy. In contrast, segmentation-based picking does not rely on grid resolution in the same way. Here, smaller strides (increased sub-tomogram overlap) instead improve the consistency of the full tomogram segmentation mask during inference with the sliding window approach.

Other picking methods that use convolutional models for segmentation like DeepFinder [Moe+21], DePiCt [DTT+23] or DeepETPicker [Liu+24] offer favorable speed-performance trade-offs similar to ProPicker, but are strictly limited to picking particles seen during training.

F Effect of Smoothing ProPicker Segmentation Masks Before Cluster-Based Picking

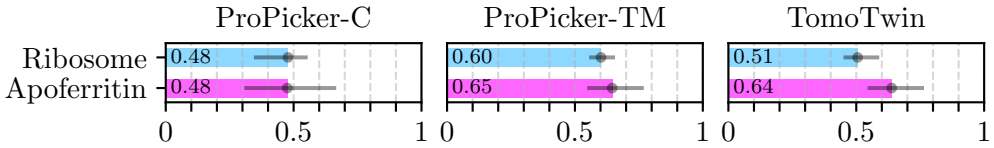

Figure A2: Results for the experiment discussed in Section 4.1, whithout smooting applied to the ProPicker segmentation masks. The higher F1 scores come mainly from increased precision, as the smoothing can eliminate false positive voxels in the segmentation masks.

G Influence of the Model Size on the Performance of DeepETPicker

The U-Net used in ProPicker has approximately 100 million trainable parameters and is, therefore, around 41× larger than the DeepETPicker U-Net. To ensure a fair comparsion between DeepETPicker and ProPicker, we compared the performance of the original DeepETPicker U-Net with 2.4 million parameters to a scaled-up version with 100 million parameters, which is very similar to the ProPicker U-Net.

We re-ran some trainings from the experiment discussed in Section 4.2.1, where we trained DeepETPicker for individual particles on synthetic tomograms. Specifically, we trained the 100 million parameter DeepETPicker U-Net to pick `1ul1` and `7shk` particles on 1 and 4 tomograms.

For `1ul1`, we obtained a mean best-case F1 score across 3 test tomograms of 0.52 when training on 1 tomogram and 0.73 when training on 4 tomograms. The corresponding scores for `7shk` are 0.41 when training on 1 tomogram and 0.60 when training on 4 tomograms. For `1ul1`, the scores obtained with the 100-million-parameter model differ by less than 0.01 from those shown in Figure 4, whereas the scores for `7shk` are 0.05 and 0.12 worse.

H Dependence of Picking Performance on the Prompt

For prompt-based picking with ProPicker, the user has to manually extract the prompt from a tomogram at inference. As the tomogram typically contains many instances of the particle of interest, a natural question is how the performance of ProPicker depends on the concrete choice of the prompt. We investigate this prompt-dependence on three synthetic tomograms, each containing 10 unique particles which are all contained in the training set. As the ground truth location of all particles is known for the synthetic tomograms, we randomly sampled 10 prompts for each particle and evaluated the best-case picking F1 score.

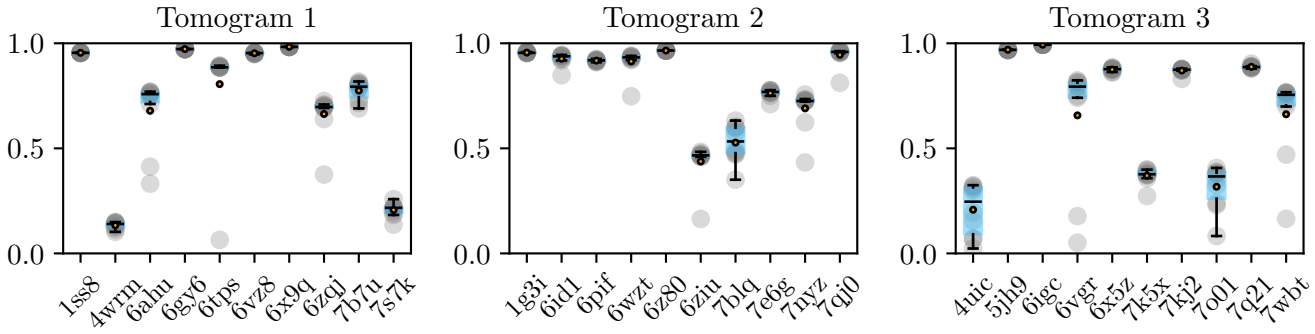

Figure A3: Dependence of the picking F1 score of ProPicker-C on the choice of the prompt for two synthetic tomograms containing particles which were seen during training. The particles are labeled according to their accession code in the protein data bank (PDB).

As can be seen in Figure A3, for all but few particles, the picking performance of ProPicker-C has only few lower outliers as the prompt varies. In general, we recommend to try a few prompts for each particle.