BioSegment: Active Learning segmentation for 3D electron microscopy imaging

Benjamin Rombaut[1,2][0000−0002−4022−715X], Joris Roels[2,3][0000−0002−2058−8134], and Yvan Saeys[1,2][0000−0002−0415−1506]

[1] Department of Applied Mathematics, Computer Science and Statistics, Faculty of Science, Ghent University, Ghent, Belgium
{benjamin.rombaut, yvan.saeys}@ugent.be
[2] Data Mining and Modelling for Biomedicine, VIB-UGent Center for Inflammation Research, Ghent, Belgium
[3] VIB Bioimaging Core, VIB-UGent Center for Inflammation Research, Ghent, Belgium

Abstract. Large 3D electron microscopy images require labor-intensive segmentation for further quantitative analysis. Recent deep learning segmentation methods automate this computer vision task, but require large amounts of labeled training data. We present **BioSegment**, a turnkey platform for experts to automatically process their imaging data and fine-tune segmentation models. It provides a user-friendly annotation experience, integration with familiar microscopy annotation software and a job queue for remote GPU acceleration. Various active learning sampling strategies are incorporated, with maximum entropy selection being the default. For mitochondrial segmentation, these strategies can improve segmentation quality by 10 to 15% in terms of intersection-over-union score compared to random sampling. Additionally, a segmentation of similar quality can be achieved using 25% of the total annotation budget required for random sampling. By comparing the state-of-the-art in human-in-the-loop annotation frameworks, we show that **BioSegment** is currently the only framework capable of employing deep learning and active learning for 3D electron microscopy data.

Keywords: Active learning · Electron microscopy · Computer vision.

1 Introduction

Volume electron microscopy (vEM or 3D EM) describes a set of high-resolution imaging techniques used in biomedical research to reveal the 3D structure of cells, tissues and small model organisms at nanometer resolution. EM techniques have emerged over the past 20 years, largely in response to the demands of the connectomics field in neuroscience, and vEM is expected to be adopted into mainstream biological imaging [23]. Generally, vEM data processing can be divided into four consecutive steps: preprocessing, segmentation, post-processing and downstream analysis.

For the imaging data to be used by deep learning networks, some additional *preprocessing* transformations include normalization and data augmentation. An imaging experiment often includes metadata of the multiple samples, which need to be compared against each other in downstream analysis. This is documented using a folder structure or a data table. Some preprocessing steps to improve imaging data include denoising [29,38], histogram equalization [39] and artifact removal. Usually, the imaging data is downsampled or *binned* in order to reduce data size and to speed up expert and model annotation, while still retaining enough resolution to allow correct segmentation.

Next is *segmentation*, the detection and delineation of structures of interest. Segmentation is required for extraction of quantitative information from rich vEM data sets. Non-discriminant contrast, diversity of appearance of structures and large image volumes turn vEM segmentation into a highly non-trivial problem, where cutting-edge methods relying on state-of-the-art computer vision techniques are still far from reaching human parity in segmentation accuracy [23]. Here, we only consider segmentation of mitochondria, but other cellular components or tissue regions can also be of interest. Pretrained models can be applied to a small sample in order to evaluate segmentation quality. If no model of sufficient quality is available, a new model is created by using some training data annotated by an expert (microscopist or biologist). Machine learning methods can be trained to produce different flavors of segmentation, labelling the pixels either by semantics (for example, label all mitochondria pixels as 1 and the rest as 0) or by the objects they belong to (for example, label all pixels of the first mitochondrion as 1, of the second mitochondrion as 2, of the nth mitochondrion as n, with non-mitochondrion pixels as 0).

There are various *post-processing* steps to transform a semantic segmentation to an object instance segmentation, such as connected components and watershed transform. To further clean up the segmentation, there is usually some filtering based on instance size.

After processing all samples of the experiment, a research question is answered in a *downstream analysis*. Statistics of interest are calculated such as number of mitochondria, mitochondria surface and volume. These statistics are summarized in a data table and combined with the experiment metadata to quantify effects.

Although significant progress has been made in recent years, largely owing to the introduction of deep learning-based methods, there is not yet a single reliable and easy-to-use solution for fully automated segmentation of vEM images. Imaging experts must choose between (or combine) manual, semi-automated and fully automated solutions based on the difficulty of the segmentation problem, the data size and the computational expertise and resources of their team or institution. Furthermore, almost all automated solutions rely on machine learning and may require large amounts of example segmentations to train a model, although in some cases models trained for the same task on similar data sets are available and can be applied directly [23].

Machine learning-based segmentation models can be divided into two categories: feature-based learning and deep learning. Feature-based learning methods use a set of predefined features (usually linear and non-linear image filters) as input to a non-linear classifier such as a support vector machine or a random forest that outputs the (semantic) segmentation. They need few examples and are available via user-friendly tools. Methods using deep learning do not rely on pre-computed features but, instead, learn features and segmentation jointly. They can solve more difficult segmentation problems, but their superior accuracy requires much larger amounts of examples, and the training must be performed on graphics processing units (GPUs). Efficient training and post-processing procedures for deep learning methods in vEM constitute an active area of research [23].

For successful application, the deep learning model needs to be trained on data very similar to the data at hand, but annotated vEM training data is time-consuming to create. Various approaches try to alleviate this problem: increasing annotator efficiency using professional annotation software (*i.e.* MIB or Imaris), sparse labeling [36] or refining model predictions using only points [10]. Additionally, model performance can increase through self-supervised learning on large unlabeled and heterogeneous data sets [14], generalizability-enhancing tricks such as data augmentation or domain adaptation [27]. In any case, additional fine-tuning on some labeled domain-specific data will improve segmentation performance and may be even required [11]. When fine-tuning, model performance can be further increased by choosing the most interesting samples to annotation using active learning [24].

Active learning (AL) is a subdomain of machine learning that aims to minimize label effort without sacrificing model performance. This is achieved by iteratively querying a batch of samples to a label providing oracle, adding them to the train set and retraining the predictor. The challenge is to come up with a smart selection criterion to query samples and maximize the steepness of the training curve [33]. In the setting of vEM segmentation, the oracle is a human imaging expert, such as a microscopist or biologist. This makes our application human or expert-in-the-loop, as the expert will be queried to provide labels through an annotation interface. We consider the total volume of EM data as an offline pool of unlabeled 2D training patches. A general overview of a human-in-the-loop annotation workflow using AL for semantic segmentation is given in Figure 1.

To our knowledge, segmentation of vEM data in an AL setting is not an established practice, *i.e.* the recent Empanada napari plugin [11] for vEM only supports random sampling. In other fields, various tools employ AL to great effect: Label Studio [35] is a flexible data annotation tool that supports semantic segmentation, AL and prediction refinement. MONAI Label [12] is an open source image labeling and learning tool that helps researchers and clinicians to collaborate, create annotated datasets, and build AI models. It features 3D segmentation refinement using 3D Slicer and AL sample selection. Kaibu [21] is a web application for visualizing and annotating multidimensional images, featuring deep learning powered interactive segmentation. Ilastik [7] is an easy-to-use

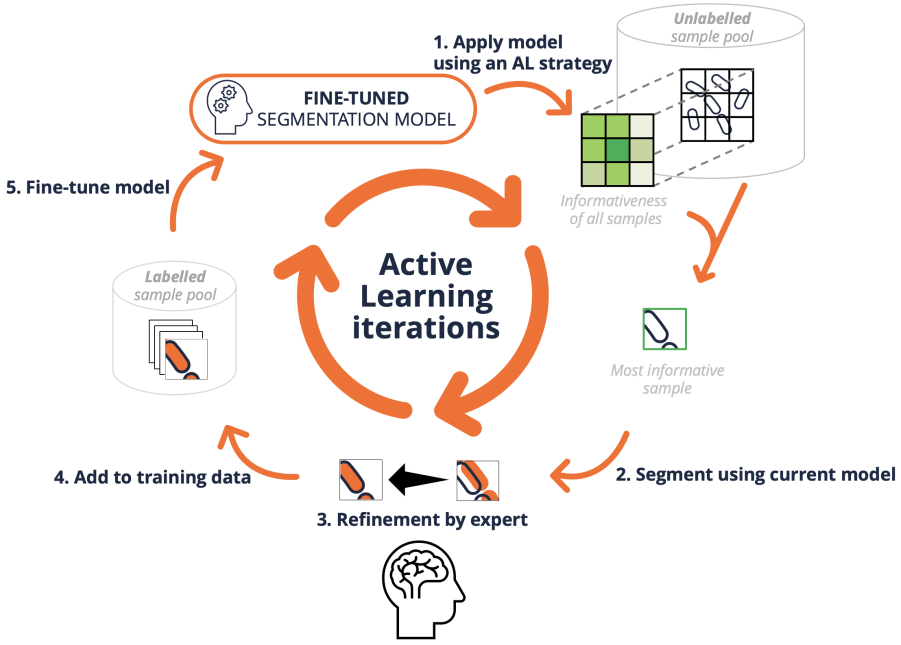


Fig. 1: Overview of active learning for image segmentation. A human imaging expert starts an AL iteration and, using the existing segmentation model and an active learning sampling strategy, ranks the unlabeled samples for labeling. Batches of the most informative samples are annotated by the expert and added to the labeled data pool. After enough new training data is created, the model is fine-tuned on the labeled pool and model performance is expected to improve. The expert can run subsequent AL iterations with the updated model on the remaining unlabeled data, or stop the iterations when model performance is sufficient or the annotation budget is spent.

interactive tool that brings machine-learning-based (bio)image analysis to end users without substantial computational expertise. It contains pre-defined workflows for image segmentation, object classification, counting and tracking.

In this paper, we propose three new contributions:

1. A comparison of five AL strategies for semantic segmentation on three vEM datasets, on which we previously reported in our preprint [28].
2. A feature comparison between current state-of-the-art software frameworks for human-in-the-loop active learning using deep learning segmentation models.
3. **BioSegment**, an integrated platform for imaging experts to process vEM datasets using AL strategies.

First, we describe the software architecture of an AL semantic segmentation framework in Section 2.1, the deep learning models in Section 2.2. We continue with used AL strategies in Section 2.3 and validation datasets in Section 2.4. Our three contributions are presented and discussed in Section 3. Lastly, we envision future work in Section 4 and conclude in Section 5.

2 Methods

2.1 Software Architecture

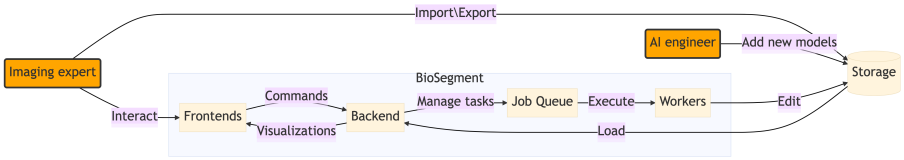


Fig. 2: Flowchart of the **BioSegment** software stack. Users interact with a frontend using their browser. They can visualize a dataset, edit annotations and create segmentations using AI models. The **BioSegment** backend handles the tasks given by the frontend and fetches the datasets from disk storage. For long-running tasks like conversion, active learning, segmentation and fine-tuning, separate workers are used.

We give an overview of the **BioSegment** software architecture in Figure 2. A central database is managed by a backend, implemented using FastAPI. It features a documented REST API, database schemas for all modelled objects and a job queue using Celery and Redis. For long-running tasks like conversion and fine-tuning, separate workers are used, communicating via the messaging bus of the job queue. For data conversion and viewing AICSImageIO [3] and BioFormats [18] are used. The only communication requirement for the workers is access to the Redis server port and the data storage. They can run on a

different machine with GPU acceleration or a network with access to secure and confidential imaging data. Segmentation models and tasks are implemented in PyTorch, and models are serialized to disk. Tensorboard is used to visualize training progression and predicted segmentation performance on selected image samples.

The **BioSegment** software stack is reproducible using *conda* environments and Docker containers. Staging and production deployments are managed using *Docker Swarm*. Restrictive enterprise firewalls can be overcome through the *Traefik* reverse-proxy, which also provides security with automated HTTPS certificate management. Admin interfaces for network, user, database and job queue management are also implemented. Clients can communicate with the backend REST API to add imaging data, manage jobs and visualize results. Using a code generation tool like OpenAPI Generator, the documented REST API from the backend can automatically generate the client code library. This automated step improves maintainability of multiple client interfaces and annotation software plugins. A JavaScript frontend implements most of the backend API and provides management of all data objects like users, datasets, segmentation, annotations and models. A Dash dashboard provides an interface for sparse semantic labelling. Datasets are accessed using file system paths in the backend and workers. These paths resolve to a local mount of the remote disk storage. The mount point is set up using *sshfs*.

2.2 Deep learning methods

We build on the PyTorch Lightning framework, which allows high-level but advanced training loops without the boilerplate code. It supports different accelerator architectures and allows for reproducible and maintainable code. It also features fine-tuning strategies, automated learning rate, batch size finders and support for multiple GPUs and mixed integer training. Various segmentation models are available: our own advanced U-Net implementations in the published *neuralnets* [26] package and *torchvision* [5] which features pretrained model weights.

2.3 Active learning strategies

Five AL strategies were implemented by us and are explained here. We consider the task of semantic segmentation, *i.e.* given an image $\boldsymbol{x} \in X \subset \mathbb{R}^N$ with a total amount of N pixels, we aim to compute a pixel-level labeling $\boldsymbol{y} \in Y$, where $Y = \{0, \ldots, C-1\}^N$ is the label space and C is the number of classes. In particular, we focus on the case of binary segmentation, *i.e.* $C = 2$. Let $\boldsymbol{p}_j(\boldsymbol{x}) = [\boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{x})]_j$ be the probability class distribution of pixel j of a parameterized segmentation algorithm $\boldsymbol{f}_{\boldsymbol{\theta}}$ (*i.e.* an encoder-decoder network, such as U-Net[30]).
Consider a large pool of n i.i.d. sampled data points over the space $Z = X \times Y$ as $\{\boldsymbol{x}_i, \boldsymbol{y}_i\}_{i \in [n]}$, where $[n] = \{1, \ldots, n\}$, and an initial pool of m randomly chosen distinct data points indexed by $S_0 = \{i_j | i_j \in [n]\}_{j \in [m]}$. An active learning

algorithm initially only has access to $\{\boldsymbol{x}_i\}_{i\in[n]}$ and $\{\boldsymbol{y}_i\}_{i\in S_0}$ and iteratively extends the currently labeled pool S_t by querying k samples from the unlabeled set $\{\boldsymbol{x}_i\}_{i\in[n]\setminus S_t}$ to an oracle. After iteration t, the predictor is retrained with the available samples $\{\boldsymbol{x}_i\}_{i\in[n]}$ and labels $\{\boldsymbol{y}_i\}_{i\in S_t}$, thereby improving the segmentation quality. Note that, without loss of generalization, the active learning approaches below are described for $k = 1$. We can also query $k > 1$ samples for k iterations, without retraining, to achieve a batch of samples. The complete active learning workflow is shown in Figure 1.

Maximum entropy sampling [16,17] Maximum entropy is a straightforward selection criterion that aims to select samples for which the predictions are uncertain. Formally speaking, we adjust the selection criterion to a pixel-wise entropy calculation as follows:

$$\boldsymbol{x}^*_{t+1} = \arg\max_{\boldsymbol{x}\in[n]\setminus S_t} -\sum_{j=0}^{N-1}\sum_{c=0}^{C-1} \left[\boldsymbol{p}_j(\boldsymbol{x})\right]_c \log\left[\boldsymbol{p}_j(\boldsymbol{x})\right]_c. \tag{1}$$

In other words, the entropy is calculated for each pixel and summed up. Note that a high entropy will be obtained when $\boldsymbol{p}_j(\boldsymbol{x}) = \frac{1}{C}$, this is exactly when there is no real consensus on the predicted class (*i.e.* high uncertainty).

Least confidence selection [9] Similar to maximum entropy sampling, the least confidence criterion selects samples for which the predictions are uncertain:

$$\boldsymbol{x}^*_{t+1} = \arg\min_{\boldsymbol{x}\in[n]\setminus S_t} \sum_{j=0}^{N-1} \max_{c=0,\ldots,C-1} \left[\boldsymbol{p}_j(\boldsymbol{x})\right]_c. \tag{2}$$

As the name suggests, the least confidence criterion selects the probability that corresponds to the predicted class. Whenever this probability is small, the predictor is not confident about its decision. For image segmentation, we sum up the maximum probabilities in order to select the least confident samples.

Bayesian active learning disagreement [13] The Bayesian active learning disagreement (BALD) approach is specifically designed for convolutional neural networks (CNNs). It makes use of Bayesian CNNs in order to cope with the small amounts of training data that are usually available in active learning workflows. A Bayesian CNN assumes a prior probability distribution placed over the model parameters $\boldsymbol{\theta} \sim p(\boldsymbol{\theta})$. The uncertainty in the weights induces prediction uncertainty by marginalizing over the approximate posterior:

$$\left[\boldsymbol{p}_j(\boldsymbol{x})\right]_c \approx \frac{1}{T}\sum_{t=0}^{T-1} \left[\boldsymbol{p}_j(\boldsymbol{x};\hat{\boldsymbol{\theta}}_t)\right]_c, \tag{3}$$

where $\hat{\boldsymbol{\theta}}_t \sim q(\boldsymbol{\theta})$ is the dropout distribution, which approximates the prior probability distribution p. In other words, a CNN is trained with dropout and inference is obtained by leaving dropout on. This causes uncertainty in

the outcome that can be used in existing criteria such as maximum entropy (Equation (1)).

K-means sampling [8] Uncertainty-based approaches typically sample close to the decision boundary of the classifier. This introduces an implicit bias that does not allow for data exploration. Most explorative approaches that aim to solve this problem transform the input $\boldsymbol{x}$ to a more compact and efficient representation $\boldsymbol{z} = \boldsymbol{g}(\boldsymbol{x})$ (*e.g.* the feature representation before the fully connected stage in a classification CNN). The representation that we used in our segmentation approach was the middle bottleneck representation in the U-Net, which is the learned encoded embedding of the model. The k-means sampling approach in particular then finds k clusters in this embedding using k-means clustering. The selected samples are then the k samples in the different clusters that are closest to the k centroids.

Core set active learning [32] The core set approach is an active learning approach for CNNs that is not based on uncertainty or exploratory sampling. Similar to k-means, samples are selected from an embedding $\boldsymbol{z} = \boldsymbol{g}(\boldsymbol{x})$ in such a way that a model trained on the selection of samples would be competitive for the remaining samples. Similar as before, the representation that we used in our segmentation approach was the bottleneck representation in the U-Net. In order to obtain such competitive samples, this approach aims to minimize the so-called core set loss. This is the difference between the average empirical loss over the set of labeled samples (*i.e.* S_t) and the average empirical loss over the entire dataset that includes the unlabeled points (*i.e.* $[n]$).

2.4 Validation datasets

Three public EM datasets were used to validate our approach:

- The EPFL dataset[4] represents a $5 \times 5 \times 5$ μm^3 section taken from the CA1 hippocampus region of the brain, corresponding to a $2048 \times 1536 \times 1065$ volume. Two $1048 \times 786 \times 165$ subvolumes were manually labelled by experts for mitochondria. The data was acquired by a focused ion-beam scanning EM, and the resolution of each voxel is approximately $5 \times 5 \times 5$ nm^3.
- The VNC dataset[5] represents two $4.7 \times 4.7 \times 1$ μm^3 sections taken from the *Drosophila melanogaster* third instar larva ventral nerve cord, corresponding to a $1024 \times 1024 \times 20$ volume. One stack was manually labelled by experts for mitochondria. The data was acquired by a transmission EM and the resolution of each voxel is approximately $4.6 \times 4.6 \times 45$ nm^3.
- The MiRA dataset[6][37] represents a $17 \times 17 \times 1.6$ μm^3 section taken from the mouse cortex, corresponding to a $8624 \times 8416 \times 31$ volume. The complete volume was manually labelled by experts for mitochondria. The data was

[4] Data available at https://cvlab.epfl.ch/data/data-em/
[5] Data available at https://github.com/unidesigner/groundtruth-drosophila-vnc/
[6] Data available at http://95.163.198.142/MiRA/mitochondria31/

acquired by an automated tape-collecting ultramicrotome scanning EM, and the resolution of each voxel is approximately $2 \times 2 \times 50$ nm^3.

In order to properly validate the discussed approaches, we split the available labeled data in a training and testing set. In the cases of a single labeled volume (VNC and MiRA), we split these datasets halfway along the y axis. A smaller U-Net (with 4 times less feature maps) was initially trained on $m = 20$ randomly selected 128×128 samples in the training volume (learning rate of $1e^{-3}$ for 500 epochs). Next, we consider a pool of $n = 2000$ samples in the training data to be queried. Each iteration, $k = 20$ samples are selected from this pool based on one of the discussed selection criteria, and added to the labeled set S_t, after which the segmentation network is fine-tuned (learning rate of $5e^{-4}$ for 200 epochs). This procedure is repeated for $T = 25$ iterations, leading to a maximum training set size of 500 samples. We validate the segmentation performance using the intersection-over-union (IoU) metric, also known as the Jaccard score:

$$J(\boldsymbol{y}, \hat{\boldsymbol{y}}) = \frac{\sum_i [\boldsymbol{y} \cdot \hat{\boldsymbol{y}}]_i}{\sum_i [\boldsymbol{y}]_i + \sum_i [\hat{\boldsymbol{y}}]_i - \sum_i [\boldsymbol{y} \cdot \hat{\boldsymbol{y}}]_i} \tag{4}$$

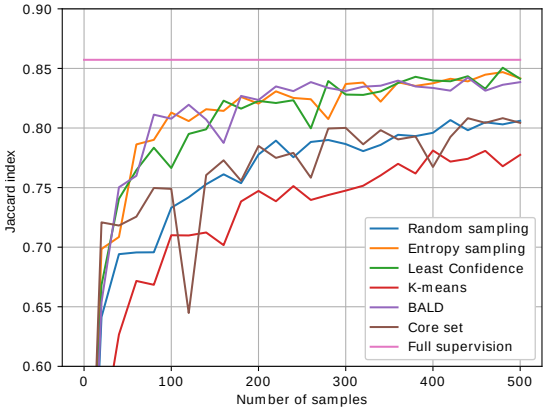
3 Results

3.1 Active Learning validation

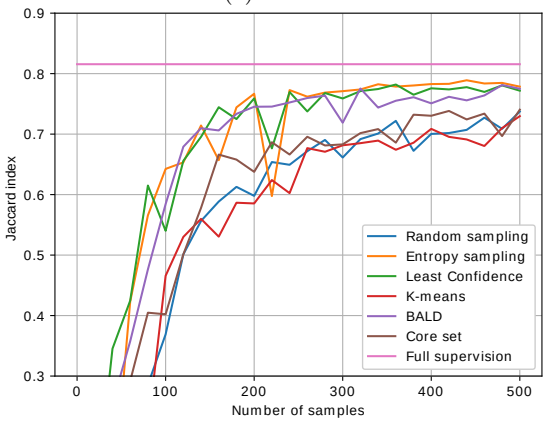
We validated five AL strategies on three public EM datasets. The resulting learning curves of the discussed approaches on the three datasets are shown in Figure 3. We additionally show the performance obtained by full supervision (*i.e.* all labels are available during training), which is the maximum achievable segmentation performance. There is an indication that maximum entropy sampling, least confidence selection and BALD outperform the random sampling baseline. These methods obtain about 10 to 15% performance increase for the same amount of available labels for all datasets. Additionally, a segmentation of similar quality can be achieved using 25% of the total annotation budget required for random sampling. The core set approach performs similar to slightly better than the baseline. We expect that this method can be improved by considering alternative embeddings. Lastly, we see that k-means performs significantly worse than random sampling. Even though this could also be an embedding problem such as with the core set approach, we think that exploratory sampling alone will not allow the predictor to learn from challenging samples, which are usually outliers. We expect that a hybrid approach based on both exploration and uncertainty might lead to better results, and consider this future work.

Figure 4 shows qualitative segmentation results on the EPFL dataset. In particular, we show results of the random, k-means and maximum entropy sampling methods using 120 samples, and compare this to the fully supervised approach. The maximum entropy sampling technique is able to improve the others by a large margin and closes the gap towards fully supervised learning significantly.

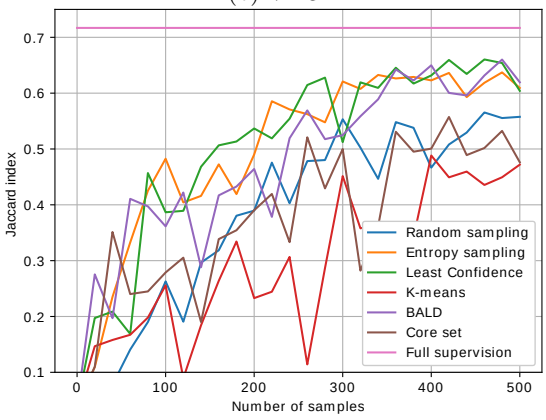
Lastly, we are interested in what type of samples the active learning approaches select for training. Figure 5 shows 4 samples of the VNC dataset that



(a) EPFL

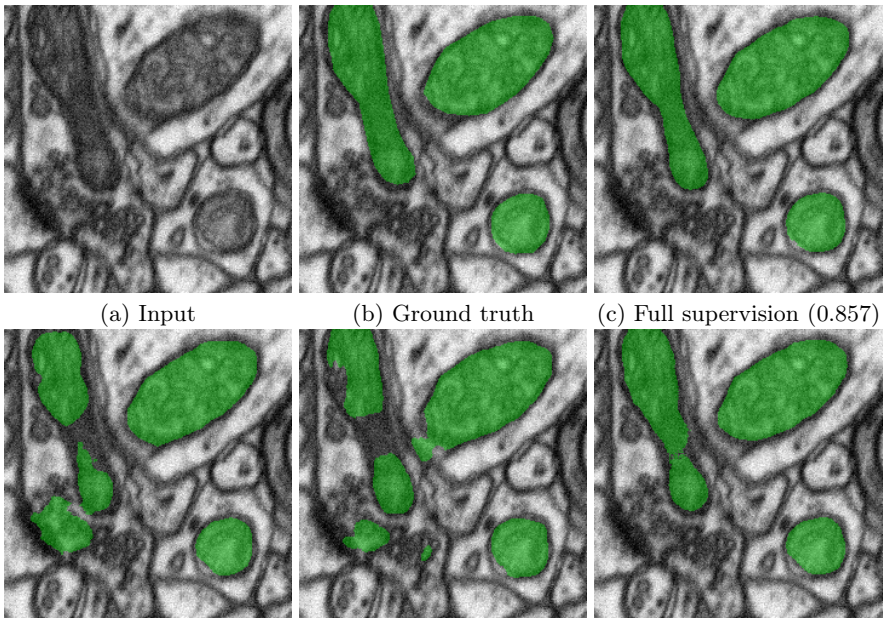


(b) VNC



(c) MiRA

Fig. 3: Learning curves for the five discussed active learning approaches, random sampling and full supervision for the three different datasets. Entropy sampling performs well across the datasets. Note that for entropy and random sampling for EPFL, the difference in model performance for the same number of samples (difference in y-axis) is 15% and difference in number of samples needed for the same model performance (difference in x-axis) is 25%.



(a) Input (b) Ground truth (c) Full supervision (0.857)

(d) Random (0.733) (e) k-means (0.710) (f) Maximum entropy (0.813)

Fig. 4: Segmentation results obtained from an actively learned U-Net with 120 samples of the EPFL dataset based on random, k-means and maximum entropy sampling, and a comparison to the fully supervised approach. Jaccard scores are indicated between brackets.

correspond to the highest prioritized samples, according to the least confidence criterion, that were selected in the first 4 iterations. The top row illustrates the probability predictions of the network at that point in time, whereas the bottom row shows the pixel-wise uncertainty of the sample (*i.e.* the maximum in Equation (2)). Note that the initial predictions at $t = 1$ are of poor quality, as the network was only trained on 20 samples. Moreover, the uncertainty is high in regions where the network is uncertain, but it is low in regions where the network is wrong. The latter is a common issue in active learning and related to the exploration vs. uncertainty trade-off. However, over time, we see that the network performance improves, and more challenging samples are being queried to the oracle.

3.2 Feature comparison

We define five software features of interest for an AL software framework for vEM data:

Interactive fine-tuning The expert should be able to fine-tune a segmentation model with their own newly annotated data. For deep learning models, this

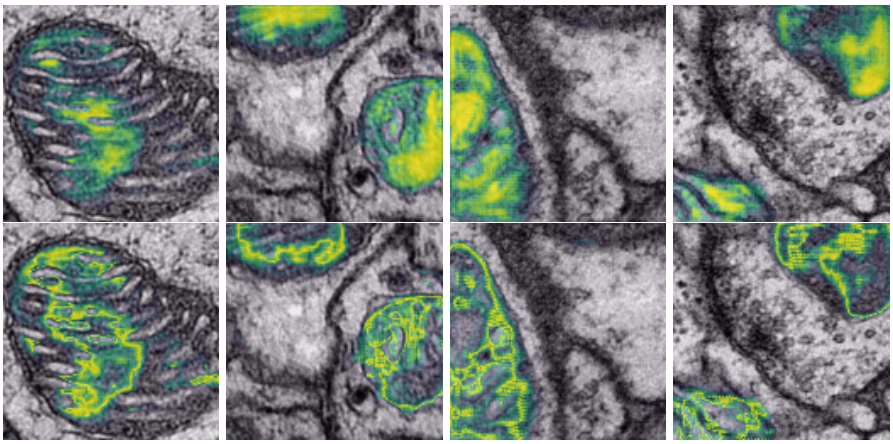


Fig. 5: Illustration of the selected samples in the VNC dataset over time in the active learning process. The top row shows the pixel-wise prediction of the selected samples at iterations 1 through 4. The bottom row show the pixel-wise least confidence score on the corresponding images.

	Software features				
Software frameworks	interactive fine-tuning	active learning	large datasets	3D support	remote resources
Label Studio [35]	x	x			x
Kaibu [21]	x				x
napari-empanada [11]	x		x	x	
ilastik [7]	x	x	x	x	x
MONAI Label [12]	x	x	x	x	x
BioSegment	x	x	x	x	x

Table 1: Comparison of open-source software frameworks for human-in-the-loop active learning using segmentation models.

involves optional GPU acceleration and reporting on training status and accuracy. All considered frameworks have this feature.

Active learning The framework should support sampling the unlabeled data using an AL strategy. Some frameworks have only proposed this feature for future work and only implemented a random sampling strategy.

Large datasets The expert should be able to apply existing and newly trained models on their whole dataset, no matter the size. This feature is the most lacking, as it requires support for tiled inference and long-running jobs.

3D support The supported annotation interfaces of the framework should allow the expert to freely browse consecutive slices or volumes in 3D.

Remote resources In order to process large datasets, large storage and computational resources such as workstations and GPU's are needed. This usually

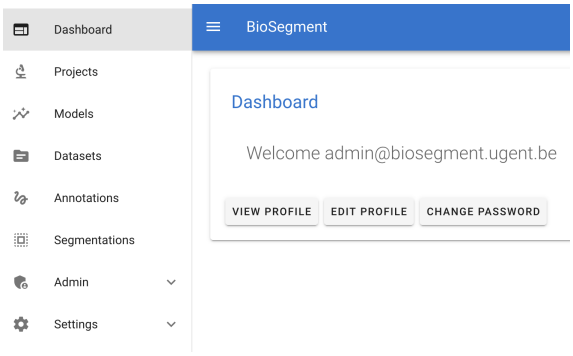
requires a flexible software architecture and communication over a network interface or software worker queue.

BioSegment combines the desirable software features needed for analyzing vEM data in one framework and is the only AL framework currently used as such. Ilastik is an established interactive annotation tool with support for standard ML segmentation. Recently, it has added beta support for a remote GPU task server (*tiktorch* [2]) and an active learning ML segmentation workflow [1] using SLIC features and supervoxels for vEM. All this functionality is however still in beta, sparsely documented and not yet applied for deep learning models or mitochondria segmentation. napari-empanada is the most recent development in vEM segmentation, but has no support for AL. The lack of support for remote resources could however be solved by running napari remotely using VirtualGL [22] or using a remote Dask cluster or data store [25]. Lastly, the recently developed feature set of MONAI Label is exciting. However, it is little over a year old, has no reported usage by the EM community, and mostly targets radiology and pathology use cases. Nevertheless, it can be adapted for EM and integrated in our **BioSegment** workflow, as shown in 6c. We note that a remote GPU-accelerated model execution is a hallmark of most frameworks: the worker queue in **BioSegment** and MONAI Label, the Label Studio ML backend and the ilastik tiktorch server.

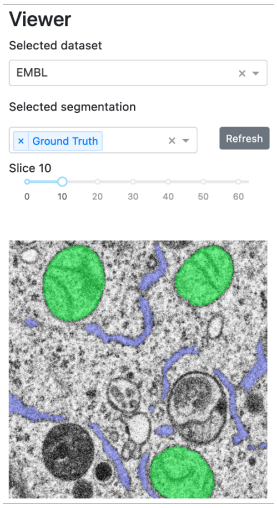
3.3 BioSegment workflow

After image capturing and storing the raw microscopy data on disk, experts start the **BioSegment** workflow. Through a dedicated dashboard (Figure 6a), the expert can create a new dataset holder and import the imaging data directly by providing the folder path. This starts a new annotation workflow. The expert can start preprocessing and segmentation jobs for the whole dataset and visualize the result (Figure 6b).
If no existing model has the desired quality, experts can choose a model to fine-tune. A batch of sampled images from the unlabeled dataset is chosen for annotation. An interface for sparse semantic labelling is provided, and the subset can be exported to different bioimaging annotation software like 3D Slicer (Figure 6c), Amira (ThermoFisher Scientific), Imaris (Oxford Instruments), Fiji [31] or napari [34]. The chosen model can be fine-tuned on the created training data and model performance can again be evaluated.

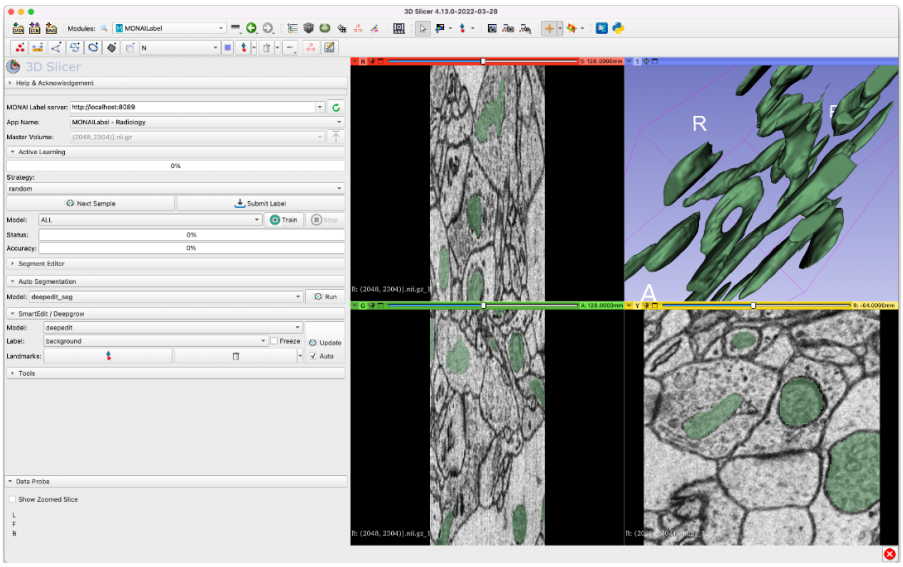
The annotation workflow can be augmented using active learning loops: the subset of images to be sampled can be selected by one of the five implemented active learning strategies, informed by the chosen model. After annotation by the expert, this model will be fine-tuned and again be used for selecting the following batch of images, creating an active learning loop and immediately incorporating the expert feedback in the sampling process. By empowering imaging experts with a dashboard to run by themselves multiple active learning iterations and segmentation jobs on their datasets, active learning can be incorporated into their normal annotation workflow. The expert can stop the iterations when they



(a) **BioSegment** Dashboard



(b) **BioSegment** model viewer



(c) External viewer (3D Slicer)

Fig. 6: Three example **BioSegment** interfaces. 6a: The dashboard where users can manage all settings. 6b: Models can be viewed and fine-tuned with training data using the viewer interface data. 6c: Results can be exported and used in external programs such as 3D Slicer and MONAI Label.

are satisfied with the segmentation quality in the preview or their annotation budget is depleted. The number of iterations is usually three or higher, but this highly depends on the dataset and on the computer vision task.
When a segmentation model of high enough quality is achieved, it can be applied to the whole dataset like the other pre-existing models. The labelled data can be added to a pool of general training data in order to train better performing models for future fine-tuning tasks. Experts can download the segmented dataset for further downstream analysis.
The **BioSegment** software stack is deployed at biosegment.ugent.be and used internally at the Flemish Institute for Biotechnology (VIB) for annotating new vEM datasets. It automates the previous manual active learning loops between imaging experts at a partnering imaging facility and deep learning scientists in our computational lab. The code is available at GitHub and features a documentation site.

4 Future work

Computer vision is not limited to single class semantic segmentation problems. Mitochondria form 3D shapes and networks, requiring 3D post-processing to achieve accurate instance segmentation. Other cell organelles are of equal interest, and large amounts of existing data are now available through the OpenOrganelle data portal [15]. Multi-class semantic segmentation is currently implemented, but the label map is not standardized. Interfacing with the BioImage Model Zoo [20] would help in this regard. We also plan to further integrate preprocessing steps like denoising, as these are still done with a separate script. Beside image enhancement, volume reconstruction and multimodal registration are two different data processing workflows in EM that would be beneficial to implement.
Recent advances in tooling include napari, an interactive, multidimensional image viewer for Python and the Java-based Paintera [4] for dense labeling of large 3D datasets. Together with cloud-based file formats like NGFF [19] these would facilitate annotating and processing large imaging experiments. Integration with Dask [25], a flexible open-source Python library for parallel computing, would allow immediate preview of complex workflows and scaling for the whole dataset using long-running jobs. These advances allow for new annotation experiences. For example, a region-of-interest free approach where the annotator freely browses the whole dataset and the current model prediction and uncertainty is lazily updated depending on the view-port. By creating multi-resolution maps of the model uncertainty, the expert is informed on the model performance over the whole dataset and is free to choose which regions to annotate.
Complexity of the software stack can be out-sourced to existing free software libraries. Lightning AI further removes boilerplate code in deep learning models by providing App and Flow interfaces. Data management and worker communication in **BioSegment** can be handled by *Girder*, which also utilizes the Celery job queue. By creating or integrating with plugins for already established an-

notation tools, adoption of the **BioSegment** workflow can be improved. Active development in the 3D Slicer and napari communities for chunked and multidimensional file formats, instance segmentation and collaborative annotation proofreading tools will also improve the future **BioSegment** feature set. For AL research, it would be valuable to add instrumentation to these annotation tools in order to better capture the burden of the annotation work by the expert. Currently, number of samples and total annotated pixels can be measured, but actual time and number of clicks would be more accurate metrics. **BioSegment** can be adapted to capture these interesting metrics. Greater model performance can be achieved by including automated hyperparameter optimization such as Optuna [6]. This and other AutoML strategies would further automate model training.

5 Conclusions

We present **BioSegment**, a turnkey solution for Active Learning segmentation of vEM imaging. It provides a user-friendly annotation experience, integration with familiar microscopy annotation software and a job queue for remote GPU acceleration. Expert annotation is augmented using active learning strategies. For mitochondrial segmentation, these strategies can improve segmentation quality by 10 to 15% in terms of intersection-over-union score compared to random sampling. Additionally, a segmentation of similar quality can be achieved using 25% of the total annotation budget required for random sampling. The software stack is maintainable through various automated tests, and the code base is published under an open-source license. By comparing the state-of-the-art in human-in-the-loop annotation frameworks, we show that **BioSegment** is currently the only framework capable of employing deep learning and active learning for 3D electron microscopy data.

Acknowledgements The computational resources and services used in this work were provided by NVIDIA, VIB IRC IT and the VSC (Flemish Supercomputer Center), funded by the Research Foundation – Flanders (FWO) and the Flemish Government. Imaging data and feedback was provided by the VIB BioImaging Core. Funding was provided by the Flanders AI Research Program.

References

1. ilastik - Voxel Segmentation Workflow (beta), `https://www.ilastik.org/documentation/voxelsegmentation/voxelsegmentation`
2. tiktorch (Dec 2021), `https://github.com/ilastik/tiktorch`, original-date: 2017-07-18T10:25:47Z
3. AICSImageIO (Jun 2022), `https://github.com/AllenCellModeling/aicsimageio`, original-date: 2019-06-27T16:43:22Z
4. Paintera (Jun 2022), `https://github.com/saalfeldlab/paintera`, original-date: 2018-04-26T21:55:50Z

5. pytorch/vision (Jun 2022), https://github.com/pytorch/vision, original-date: 2016-11-09T23:11:43Z
6. Akiba, T., Sano, S., Yanase, T., Ohta, T., Koyama, M.: Optuna: A Next-generation Hyperparameter Optimization Framework (Jul 2019). https://doi.org/10.48550/arXiv.1907.10902, http://arxiv.org/abs/1907.10902, number: arXiv:1907.10902 arXiv:1907.10902 [cs, stat]
7. Berg, S., Kutra, D., Kroeger, T., Straehle, C.N., Kausler, B.X., Haubold, C., Schiegg, M., Ales, J., Beier, T., Rudy, M., Eren, K., Cervantes, J.I., Xu, B., Beuttenmueller, F., Wolny, A., Zhang, C., Koethe, U., Hamprecht, F.A., Kreshuk, A.: ilastik: interactive machine learning for (bio)image analysis. Nature Methods **16**(12), 1226–1232 (Dec 2019). https://doi.org/10.1038/s41592-019-0582-9, https://www.nature.com/articles/s41592-019-0582-9, number: 12 Publisher: Nature Publishing Group
8. Bodó, Z., Minier, Z., Csató, L.: Active Learning with Clustering. In: Active Learning and Experimental Design workshop In conjunction with AISTATS 2010. pp. 127–139. JMLR Workshop and Conference Proceedings (Apr 2011), https://proceedings.mlr.press/v16/bodo11a.html, iSSN: 1938-7228
9. Brinker, K.: Incorporating diversity in active learning with support vector machines. In: Proceedings of the Twentieth International Conference on International Conference on Machine Learning. pp. 59–66. ICML'03, AAAI Press, Washington, DC, USA (Aug 2003)
10. Chen, X., Zhao, Z., Zhang, Y., Duan, M., Qi, D., Zhao, H.: FocalClick: Towards Practical Interactive Image Segmentation. arXiv:2204.02574 [cs] (Apr 2022), http://arxiv.org/abs/2204.02574, arXiv: 2204.02574 version: 1
11. Conrad, R.W., Narayan, K.: Instance segmentation of mitochondria in electron microscopy images with a generalist deep learning model. Tech. rep., bioRxiv (May 2022). https://doi.org/10.1101/2022.03.17.484806, https://www.biorxiv.org/content/10.1101/2022.03.17.484806v2, section: New Results Type: article
12. Diaz-Pinto, A., Alle, S., Ihsani, A., Asad, M., Nath, V., Pérez-García, F., Mehta, P., Li, W., Roth, H.R., Vercauteren, T., Xu, D., Dogra, P., Ourselin, S., Feng, A., Cardoso, M.J.: MONAI Label: A framework for AI-assisted Interactive Labeling of 3D Medical Images. arXiv:2203.12362 [cs, eess] (Mar 2022), arXiv: 2203.12362
13. Gal, Y., Islam, R., Ghahramani, Z.: Deep Bayesian Active Learning with Image Data. Tech. Rep. arXiv:1703.02910, arXiv (Mar 2017). https://doi.org/10.48550/arXiv.1703.02910, http://arxiv.org/abs/1703.02910, arXiv:1703.02910 [cs, stat] type: article
14. Han, H., Dmitrieva, M., Sauer, A., Tam, K.H., Rittscher, J.: Self-Supervised Voxel-Level Representation Rediscovers Subcellular Structures in Volume Electron Microscopy. pp. 1874–1883 (2022), https://openaccess.thecvf.com/content/CVPR2022W/CVMI/html/Han_Self-Supervised_Voxel-Level_Representation_Rediscovers_Subcellular_Structures_in_Volume_Electron_Microscopy_CVPRW_2022_paper.html
15. Heinrich, L., Bennett, D., Ackerman, D., Park, W., Bogovic, J., Eckstein, N., Petruncio, A., Clements, J., Xu, C.S., Funke, J., Korff, W., Hess, H.F., Lippincott-Schwartz, J., Saalfeld, S., Weigel, A.V., Team, C.P.: Automatic whole cell organelle segmentation in volumetric electron microscopy (Nov 2020). https://doi.org/10.1101/2020.11.14.382143, https://www.biorxiv.org/content/10.1101/2020.11.14.382143v1, pages: 2020.11.14.382143 Section: New Results

16. Joshi, A.J., Porikli, F., Papanikolopoulos, N.: Multi-class active learning for image classification. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition. pp. 2372–2379 (Jun 2009). https://doi.org/10.1109/CVPR.2009.5206627, iSSN: 1063-6919
17. Li, X., Guo, Y.: Adaptive Active Learning for Image Classification. In: 2013 IEEE Conference on Computer Vision and Pattern Recognition. pp. 859–866 (Jun 2013). https://doi.org/10.1109/CVPR.2013.116, iSSN: 1063-6919
18. Linkert, M., Rueden, C.T., Allan, C., Burel, J.M., Moore, W., Patterson, A., Loranger, B., Moore, J., Neves, C., MacDonald, D., Tarkowska, A., Sticco, C., Hill, E., Rossner, M., Eliceiri, K.W., Swedlow, J.R.: Metadata matters: access to image data in the real world. Journal of Cell Biology **189**(5), 777–782 (May 2010). https://doi.org/10.1083/jcb.201004104, https://doi.org/10.1083/jcb.201004104
19. Moore, J., Allan, C., Besson, S., Burel, J.M., Diel, E., Gault, D., Kozlowski, K., Lindner, D., Linkert, M., Manz, T., Moore, W., Pape, C., Tischer, C., Swedlow, J.R.: OME-NGFF: a next-generation file format for expanding bioimaging data-access strategies. Nature Methods pp. 1–3 (Nov 2021). https://doi.org/10.1038/s41592-021-01326-w, https://www.nature.com/articles/s41592-021-01326-w, bandiera_abtest: a Cc_license_type: cc_by Cg_type: Nature Research Journals Primary_atype: Research Publisher: Nature Publishing Group Subject_term: Computational platforms and environments;Data publication and archiving Subject_term_id: computational-platforms-and-environments;data-publication-and-archiving
20. Ouyang, W., Beuttenmueller, F., Gómez-de Mariscal, E., Pape, C., Burke, T., Garcia-López-de Haro, C., Russell, C., Moya-Sans, L., de-la Torre-Gutiérrez, C., Schmidt, D., Kutra, D., Novikov, M., Weigert, M., Schmidt, U., Bankhead, P., Jacquemet, G., Sage, D., Henriques, R., Muñoz-Barrutia, A., Lundberg, E., Jug, F., Kreshuk, A.: BioImage Model Zoo: A Community-Driven Resource for Accessible Deep Learning in BioImage Analysis (Jun 2022). https://doi.org/10.1101/2022.06.07.495102, https://www.biorxiv.org/content/10.1101/2022.06.07.495102v1, pages: 2022.06.07.495102 Section: New Results
21. Ouyang, W., Le, T., Xu, H., Lundberg, E.: Interactive biomedical segmentation tool powered by deep learning and ImJoy. Tech. Rep. 10:142, F1000Research (Feb 2021). https://doi.org/10.12688/f1000research.50798.1, https://f1000research.com/articles/10-142, type: article
22. Paradis, D.J., Segee, B.: Remote Rendering and Rendering in Virtual Machines. In: 2016 International Conference on Computational Science and Computational Intelligence (CSCI). pp. 218–221 (Dec 2016). https://doi.org/10.1109/CSCI.2016.0048
23. Peddie, C.J., Genoud, C., Kreshuk, A., Meechan, K., Micheva, K.D., Narayan, K., Pape, C., Parton, R.G., Schieber, N.L., Schwab, Y., Titze, B., Verkade, P., Weigel, A., Collinson, L.M.: Volume electron microscopy. Nature Reviews Methods Primers **2**(1), 1–23 (Jul 2022). https://doi.org/10.1038/s43586-022-00131-9, https://www.nature.com/articles/s43586-022-00131-9, number: 1 Publisher: Nature Publishing Group
24. Ren, P., Xiao, Y., Chang, X., Huang, P.Y., Li, Z., Gupta, B.B., Chen, X., Wang, X.: A Survey of Deep Active Learning. ACM Computing Surveys **54**(9), 180:1–180:40 (Oct 2021). https://doi.org/10.1145/3472291, https://doi.org/10.1145/3472291
25. Rocklin, M.: Dask: Parallel Computation with Blocked algorithms and Task Scheduling. Proceedings of the 14th Python in Science Conference pp.

126–132 (2015). https://doi.org/10.25080/Majora-7b98e3ed-013, https://conference.scipy.org/proceedings/scipy2015/matthew_rocklin.html, conference Name: Proceedings of the 14th Python in Science Conference

26. Roels, J.: NeuralNets (May 2022), https://github.com/JorisRoels/neuralnets, original-date: 2019-11-29T09:59:01Z
27. Roels, J., Hennies, J., Saeys, Y., Philips, W., Kreshuk, A.: Domain Adaptive Segmentation In Volume Electron Microscopy Imaging. In: 2019 IEEE 16th International Symposium on Biomedical Imaging (ISBI 2019). pp. 1519–1522. IEEE, Venice, Italy (Apr 2019). https://doi.org/10.1109/ISBI.2019.8759383, https://ieeexplore.ieee.org/document/8759383/
28. Roels, J., Saeys, Y.: Cost-efficient segmentation of electron microscopy images using active learning. arXiv:1911.05548 [cs] (Nov 2019), http://arxiv.org/abs/1911.05548, arXiv: 1911.05548
29. Roels, J., Vernaillen, F., Kremer, A., Gonçalves, A., Aelterman, J., Luong, H.Q., Goossens, B., Philips, W., Lippens, S., Saeys, Y.: An interactive ImageJ plugin for semi-automated image denoising in electron microscopy. Nature Communications **11**(1), 771 (Feb 2020). https://doi.org/10.1038/s41467-020-14529-0, https://www.nature.com/articles/s41467-020-14529-0, number: 1 Publisher: Nature Publishing Group
30. Ronneberger, O., Fischer, P., Brox, T.: U-Net: Convolutional Networks for Biomedical Image Segmentation (May 2015). https://doi.org/10.48550/arXiv.1505.04597, http://arxiv.org/abs/1505.04597, number: arXiv:1505.04597 arXiv:1505.04597 [cs]
31. Schindelin, J., Arganda-Carreras, I., Frise, E., Kaynig, V., Longair, M., Pietzsch, T., Preibisch, S., Rueden, C., Saalfeld, S., Schmid, B., Tinevez, J.Y., White, D.J., Hartenstein, V., Eliceiri, K., Tomancak, P., Cardona, A.: Fiji: an open-source platform for biological-image analysis. Nature Methods **9**(7), 676–682 (Jul 2012). https://doi.org/10.1038/nmeth.2019, https://www.nature.com/articles/nmeth.2019, number: 7 Publisher: Nature Publishing Group
32. Sener, O., Savarese, S.: Active Learning for Convolutional Neural Networks: A Core-Set Approach. arXiv:1708.00489 [cs, stat] (Jun 2018), http://arxiv.org/abs/1708.00489, arXiv: 1708.00489
33. Settles, B.: Active Learning Literature Survey. Technical Report, University of Wisconsin-Madison Department of Computer Sciences (2009), https://minds.wisconsin.edu/handle/1793/60660, accepted: 2012-03-15T17:23:56Z
34. Sofroniew, N., Lambert, T., Evans, K., Nunez-Iglesias, J., Bokota, G., Winston, P., Peña-Castellanos, G., Yamauchi, K., Bussonnier, M., Doncila Pop, D., Can Solak, A., Liu, Z., Wadhwa, P., Burt, A., Buckley, G., Sweet, A., Migas, L., Hilsenstein, V., Gaifas, L., Bragantini, J., Rodríguez-Guerra, J., Muñoz, H., Freeman, J., Boone, P., Lowe, A., Gohlke, C., Royer, L., PIERRÉ, A., Har-Gil, H., McGovern, A.: napari: a multi-dimensional image viewer for Python (May 2022). https://doi.org/10.5281/zenodo.6598542, https://zenodo.org/record/6598542
35. Tkachenko, M., Malyuk, M., Holmanyuk, A., Liubimov, N.: Label Studio: Daata labeling software (2020), https://github.com/heartexlabs/label-studio, original-date: 2019-06-19T02:00:44Z
36. Wolny, A., Yu, Q., Pape, C., Kreshuk, A.: Sparse Object-level Supervision for Instance Segmentation with Pixel Embeddings (Apr 2022). https://doi.org/10.48550/arXiv.2103.14572, http://arxiv.org/abs/2103.14572, number: arXiv:2103.14572 arXiv:2103.14572 [cs]

37. Xiao, C., Chen, X., Li, W., Li, L., Wang, L., Xie, Q., Han, H.: Automatic Mitochondria Segmentation for EM Data Using a 3D Supervised Convolutional Network. Frontiers in Neuroanatomy **12** (2018). https://doi.org/10.3389/fnana.2018.00092, https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6224513/, publisher: Frontiers Media SA
38. Zhang, K., Li, Y., Zuo, W., Zhang, L., Van Gool, L., Timofte, R.: Plug-and-Play Image Restoration with Deep Denoiser Prior (Jul 2021). https://doi.org/10.48550/arXiv.2008.13751, http://arxiv.org/abs/2008.13751, number: arXiv:2008.13751 arXiv:2008.13751 [cs, eess]
39. Zuiderveld, K.: VIII.5. - Contrast Limited Adaptive Histogram Equalization. In: Heckbert, P.S. (ed.) Graphics Gems, pp. 474–485. Academic Press (Jan 1994). https://doi.org/10.1016/B978-0-12-336156-1.50061-6, https://www.sciencedirect.com/science/article/pii/B9780123361561500616