Darea - Handbook

David Kleindienst

July 10, 2020

1 Introduction

Darea (**D**eep learning-**a**ssisted **re**plica image **a**nalysis suite) is a software designed for high-througput analysis of SDS-digest freeze-fracture replica immunogold electron microscopy images. It's features include: Automatic and manual demarcation of regions of interest, automatic and manual annotation of gold particles of different sizes, quantification of gold particle number and density and distances between gold particles as well as Monte-Carlo simulations of randomly distributed gold particles.

Contents

2 Installation

2.1 Hardware Requirements

Most functions of DareIt (including deep-learning prediction of demarcation) should run on any reasonably modern computer. The only exception to this is training neural network for demarcation prediction, for which we highly recommend using a modern NVIDIA GPU (at least 8GB memory). It will also run on CPU, but may take very long time (weeks) to finish.

2.2 Software Prerequisites

- Matlab R2018b or later with Simulink, Deep-learning toolbox and (optional but recommended) Parallel-processing toolbox
- Python 3.6 (Anaconda distribution is highly recommended: `https://www.anaconda.com/distribution/`)
- For GPU: CUDA 9 (see 2.3.6 when wanting to use a different version)

2.3 Installation

The installation guide assumes that python was installed by Anaconda Distribution.

2.3.1 Windows - CPU

(tested on windows 7 and 10)

In anaconda prompt run

```
conda env create -f <path to Darea>/envs/Darea_Windows.yml
```

It is now downloading and installing neccessary packages which may take some time. After it finished, run

```
conda activate Darea
where python
```

the last command will give you a path: now in matlab run:

```
pyversion <path>
```

(replace <path> with the output you got from where python command)

2.3.2 Windows - GPU

(tested on windows 7)

In anaconda prompt run

```
conda env create -f <path to Darea>/envs/Darea_WindowsGPU.yml
```

It is now downloading and installing neccessary packages which may take some time. After it finished, run

```
conda activate Darea
where python
```

the last command will give you a path: now in matlab run:

```
pyversion <path>
```

(replace <path> with the output you got from where python command). Please refer to 2.3.6 to ensure your CUDA and cudatoolkit versions match.

2.3.3 MacOS - CPU

(tested on MacOS Mojave)

In terminal run

```
conda env create -f <path to Darea>/envs/Darea_MacOS.yml
```

It is now downloading and installing neccessary packages which may take some time. After it finished, run

```
conda activate Darea
which python
```

the last command will give you a path: now in matlab run:

```
pyversion <path>
```

(replace <path> with the output you got from where python command)

2.3.4 MacOS - GPU

(not tested)

Follow the instructions for MacOS - CPU. Testing if it works is recommended at this step. Then, type in terminal:

```
conda activate Darea
conda uninstall tensorflow tensorflow-base
conda install tensorflow-GPU=1.12
```

Then follow 2.3.6 to make sure your CUDA and cudatoolkit versions match.

2.3.5 Other operating systems or using other package managers than Anaconda

Unfortunately, due to interplay between Matlab and python, this software is very sensitive to changes in package versions and it is a bit of a gamble if a version change in any of the packages will break the functionality. We recommend having a look at the package version information in any (or all) of the .yml files and trying to install a similar environment. First get it running with CPU, then (if wanted) try to follow the MacOS-GPU guide to get it running on GPU.

2.3.6 Using other versions of CUDA

in terminal run:

```
nvcc --version
```

to get the version number of CUDA then (after installation) in anaconda prompt:

```
conda install cudatoolkit=<version>
```

Replace <version> by major version number returned by nvcc command, e.g. when using CUDA 10, run:

```
conda install cudatoolkit=10
```

3 Typical Workflow

In a typical analysis workflow, first a project is created and images are imported (see 5.1.2). Typically labelings for different proteins are imported into different projects, while different experimental conditions (genotypes, brain regions, ...) are pooled into the same project. If images are provided as 8-bit, they should then be converted to 16-bit (see Fig. 1 (6)). Images are then assigned into different groups (genotypes, brain regions, ...) (see 5.3) and general project settings (e.g. which particle sizes, whether particles should be detected on ROI only) are set (see 5.2). Automated demarcation and particle detection can then be run (see 7). Resulting demarcations can then be revised manually (or performed manually if no automated demarcation was run) (see 6.2) followed by manual revision of particle annotations (see 6.3). After finishing annotation of the data, analysis and Monte-Carlo simulations should be run (see 8.1). From the results, data sheets and figures can be exported (see 8.2) and example images can be created (see 8.3). Annotated data can then be used to further refine automated demarcation (see 9.2) and particle prediction (see 9.1).

4 The Main Menu

To run Darea, start MATLAB. Navigate to the Darea folder and you should see a file named "run.m". Right click on the file and select run from the menu. Now Darea's main menu (Fig. 1) should open.

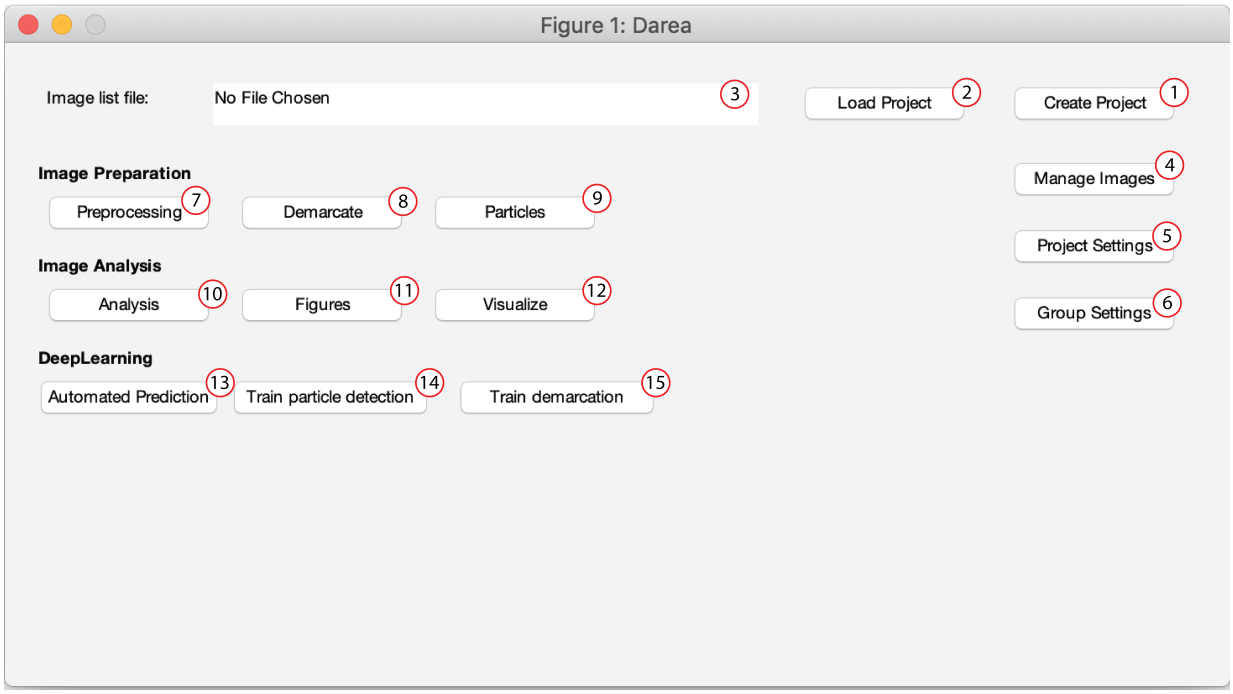

Figure 1: Darea Main Menu

1: Create a new project and import images (see 5.1.2)

2: Load existing project

3: Shows the path to the currently loaded project file

4: Add and remove images and modify scale (see 5.1.3)

5: Set, for example, which particle sizes exist and how far from the demarcation they should be detected (see 5.2)

6: Assign groups to images (see 5.3)

7: Preprocessing of images. Includes converting images to 16 bit, autoadjusting contrast and inverting the images. See 5.4 for details. **Warning: This modifies the original image files!**

8: Manual image demarcation (see 6.2)

9: Manual gold particle annotation (see 6.3)

10: Run analysis and (if desired) Monte-Carlo simulations (see 8.1)

11: Generate data tables and figures (see 8.2)

12: Visualize the results and generate publication ready example images (see 8.3)

13: Automatically annotate demarcation and gold particles (see 7)

14: Train machine learning algorithm for automated gold particle annotation (see 9.1)

15: Train neural network for automated demarcation (see 9.2)

5 Projects and Groups

5.1 Creating projects and importing images

5.1.1 Image requirements and suggested folder structure

Images have to be provided as 8 or 16 bit rgb or grayscale .tif images. When images of different magnifications are used, we recommend saving the magnification (separated by some deliminator) to the filename, as this allows automatic detection of the appropriate scale. Where in the filename the magnification occurs is irrelevant. Examples of usable filenames (we here use 39000x as example magnification, but it can really be anything and does not have to contain an x): 01_39000x.tif ; Hippocampus_39000x_image1.tif ; 39000x_01.tif ; 01-39000x.tif ; 39000x#image1.tif. Examples of filenames where the magnification would not be detected correctly (because the deliminator is lacking): Image39000x.tif ; 39000xImage.tif

For optimal automatic importing of your images we suggest the following structure:

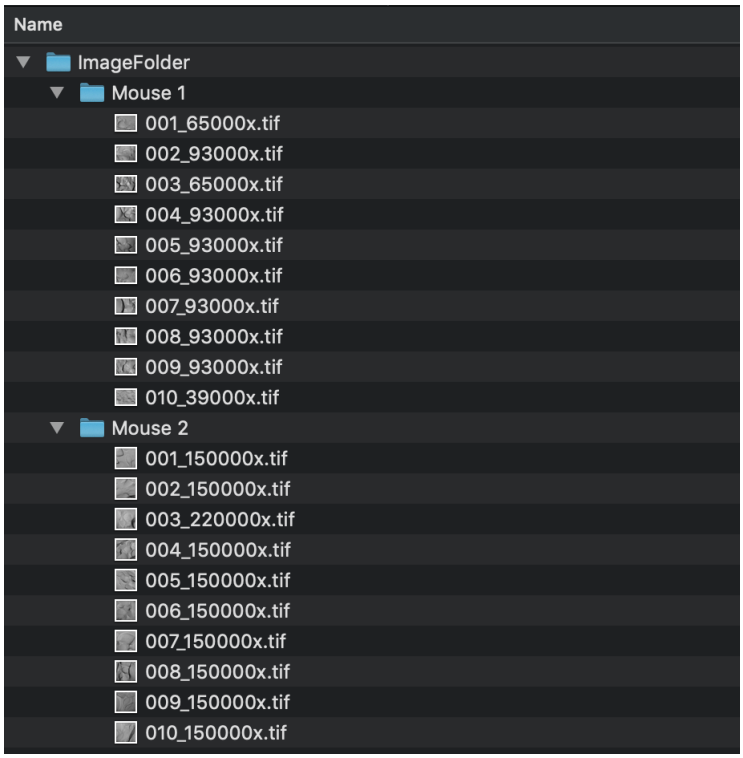

Figure 2: Suggested Folder structure.

In this case "ImageFolder" should be specified for importing, and during auto import the images will be assigned into the groups "Mouse 1" and "Mouse 2". This group assignment can be changed later (see 5.3.

5.1.2 Creating Projects

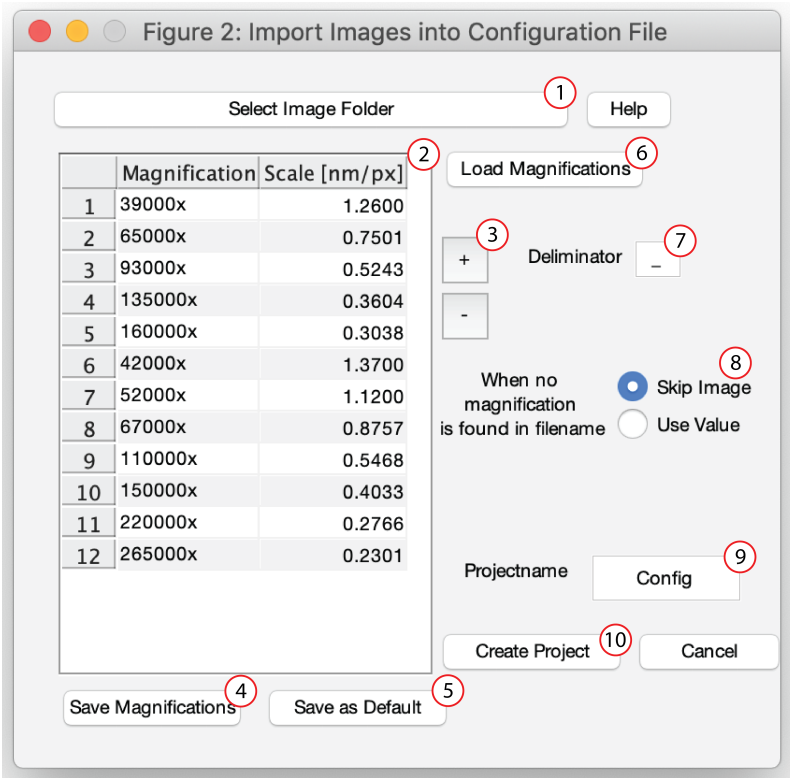

Figure 3: Create Project

1: Select the folder containing your image folders (see 5.1.1).

2: List of magnifications which should be detected in the filename. The left column indicates how the magnification is described in the filename, while the right column indicates the scale (in nm/px). In the displayed case an image called 01_39000x.tif would be assigned a scale of 1.26 nm/px.

3: Add another line to magnification list (+) or remove the currently selected line (-)

4: Save magnification list into a file

5: Save magnification list as future default which will be loaded automatically

6: Load magnification file

7: The delimiter between magnfication and the rest of the filename. If your filename is 01_39000x delimiter should be _ ; if your filename is 01-39000x delimiter should be - .

8: Images that do not contain a valid magnfication (listed in 2) can be either ignored ("skip image") or assigned a specific scale ("use value" and specify the scale in the field that appears). The latter one is especially useful if all your images are the same magnfication and do not contain the magnification in the filename.

9: The name of this project. The project will than be saved in the image folder as *projectname*.dat

10: Create the project

5.1.3 Managing Images

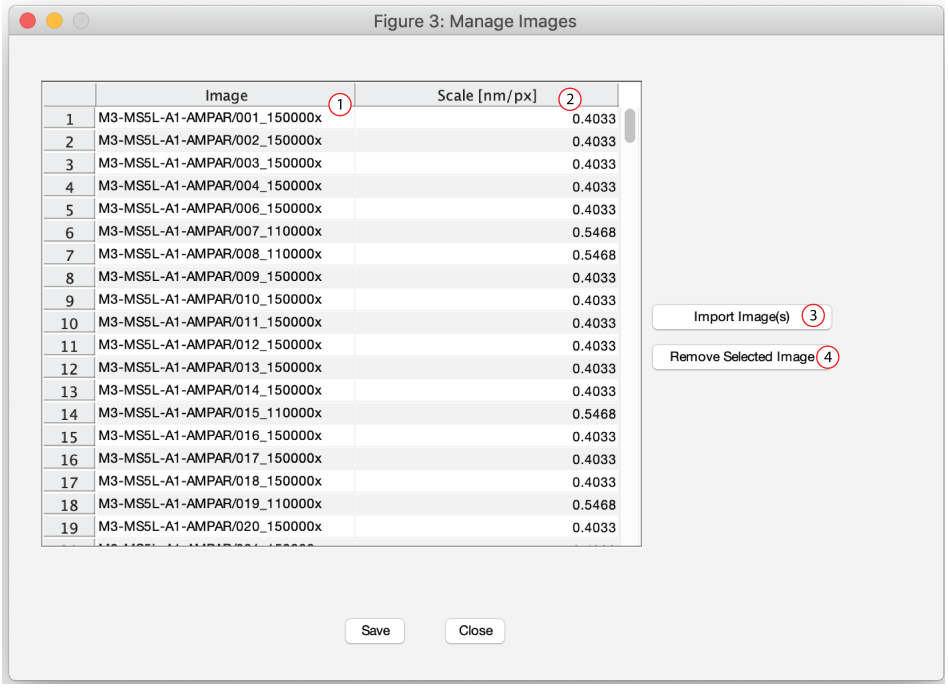

Figure 4: Manage Images

1: List of all the images

2: Scale for each image, click to modify.

3: Select one or multiple images to import into this project. They will then be added to the end of the list

4: Remove the selected image from the project

5.2 Project settings

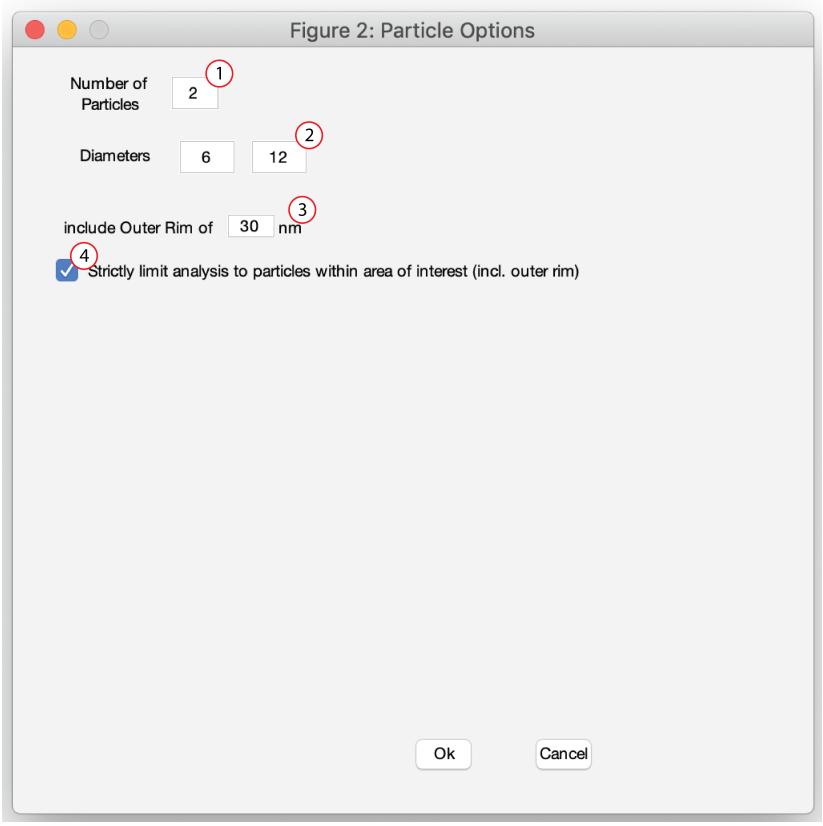

Figure 5: Project settings

1: How many different particle sizes you used

2: The different particle diameters (in nm)

3: Size of the outer rim around the demarcation within which particles should be detected.

4: If ticked, particles outside of both demarcation and outer rim will be ignored (usually that is the expected behavior)

5.3 Groups

Groups can be specified at any time before figures and tables are made.

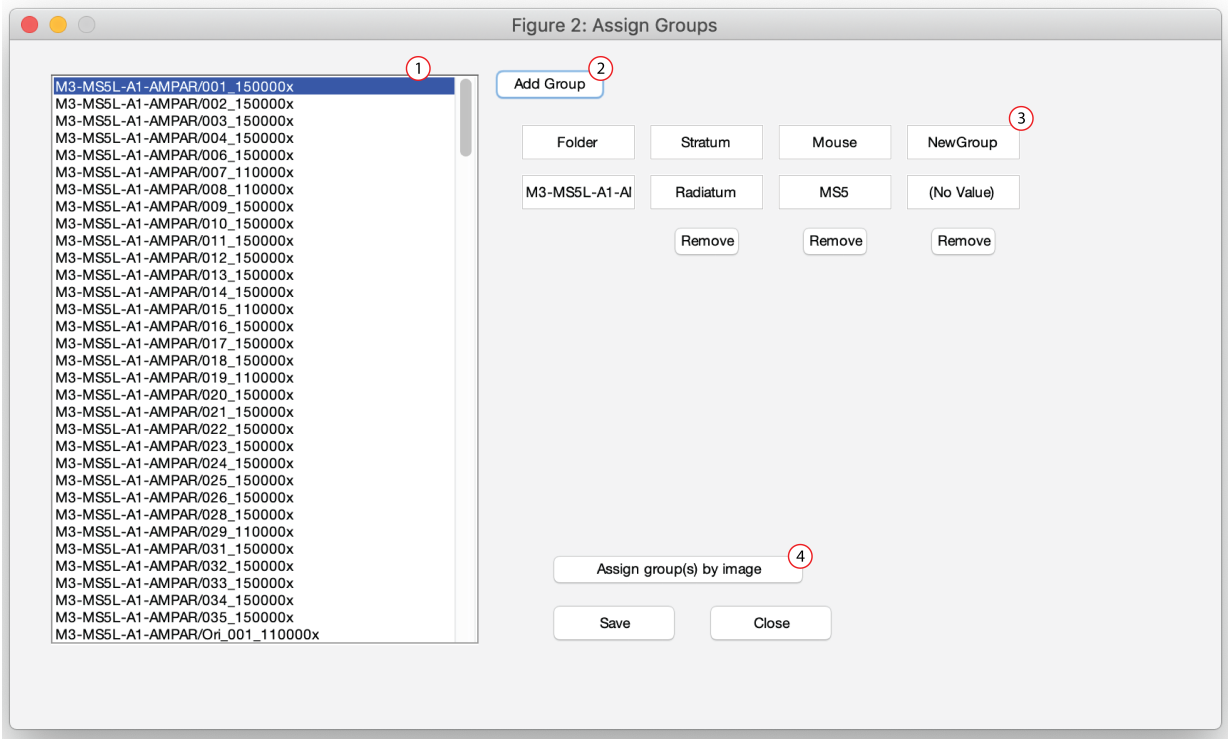

Figure 6: Groups

1: List of all images. Multiple images can be selected using shift or ctrl.

2: Add a new grouping, i.e. a property by which images can be divided into different groups (e.g. brain region, genotype, animal number).

3: Each column contains a grouping. The top row has the names of the groupings (e.g. brain region, genotype), these are the same for all images. The bottom row has the individual groupnames (e.g. hippocampus, wild-type) which may differ between images. Modifying this value will change it for all images selected in (1).

4: Sometimes, you may want to assign groups based on a property of the image (e.g. complete and incomplete profiles or perforated and non-perforated synapses). Assign group by image will show you each image and let you assign a group individually (see 5.3.1).

5.3.1 Assign group by image

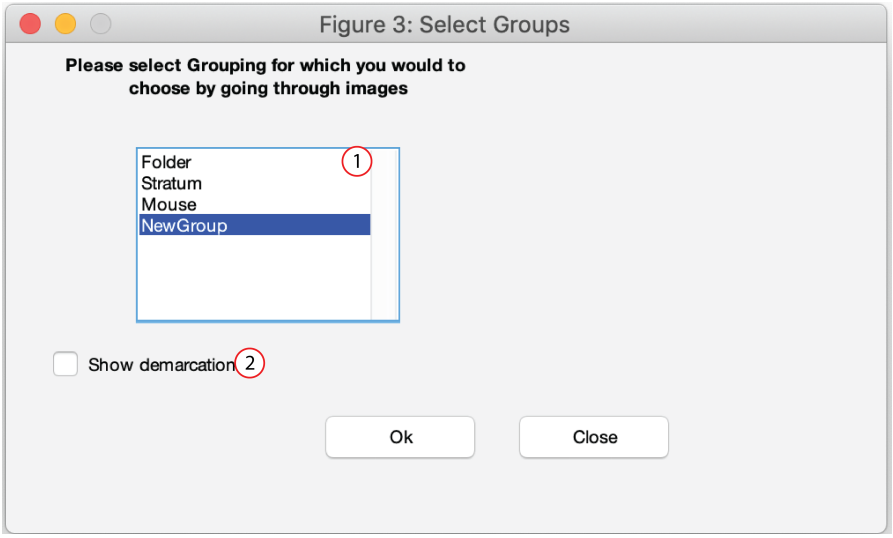

Figure 7: Assign groups by image

1: Select which grouping(s) should be assigned for each image

2: Whether or not the demarcation (if it exists) should be indicated

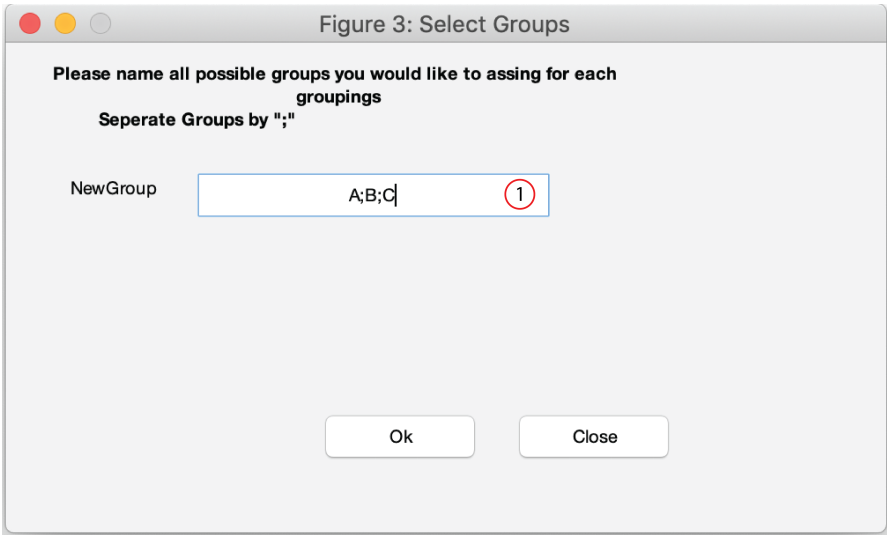

Figure 8: Assign groups by image

1: Specify which are the possible groups, separated by ";"s. In this example each image will be assigned to Group A, B or C.

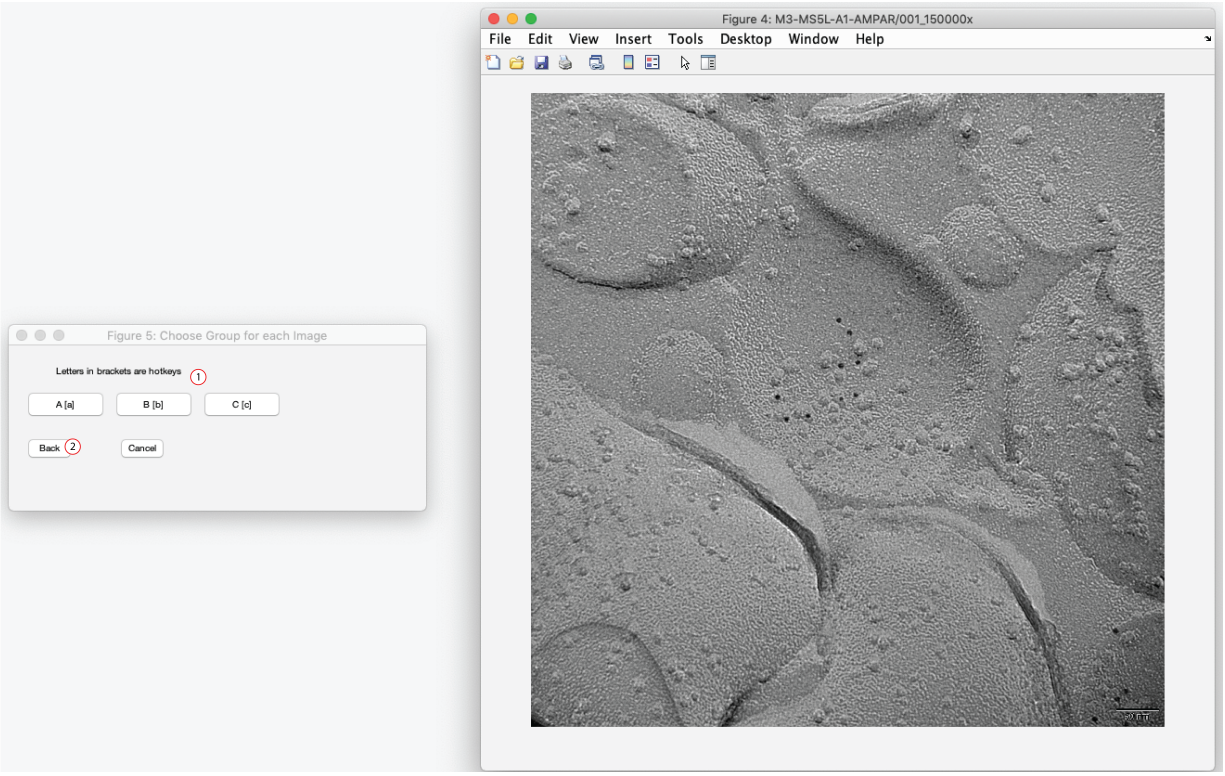

Figure 9: Assign groups by image

1: Click the corresponding button (or hotkey, shown between []) to assign a group to the image. Once you assign a group, the next image will be opened. This will continue until all images were assigned a group. If you cancel before going through all images, your inputs will not be saved!

2: Click back to go to the previous image to correct your choice.

5.4 Preprocessing

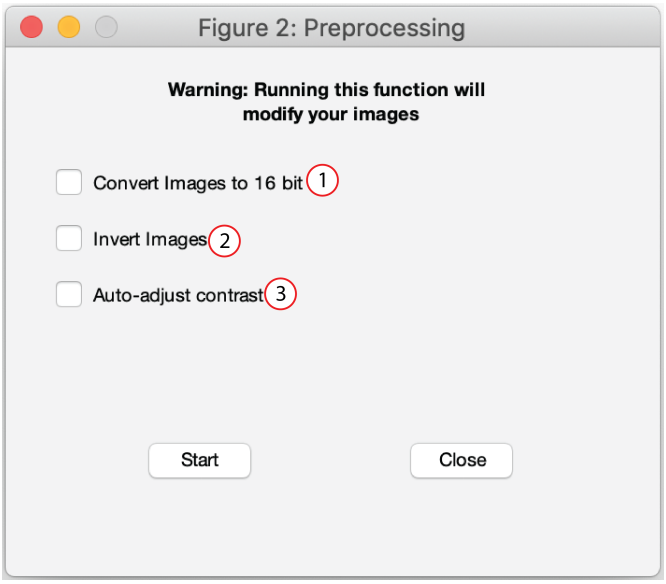

Figure 10: Preprocessing - **Warning this modifies your original image files**

1: Converts all images to 16 bit which is the necessary format for Darea. Running this multiple times will not further change your images

2: Inverts the images. This is useful when working with darkfield images. Running this a second time will restore the orginal image.

3: Automatically adjust contrast. Useful when working with images with bad contrast. Running this multiple times will further increase contrast. *Warning: Running this too often may make your images unusable*

6 Manual Image Annotation

6.1 Opening images

This menu is common for manual demarcation, particle detection and visualizing the results.

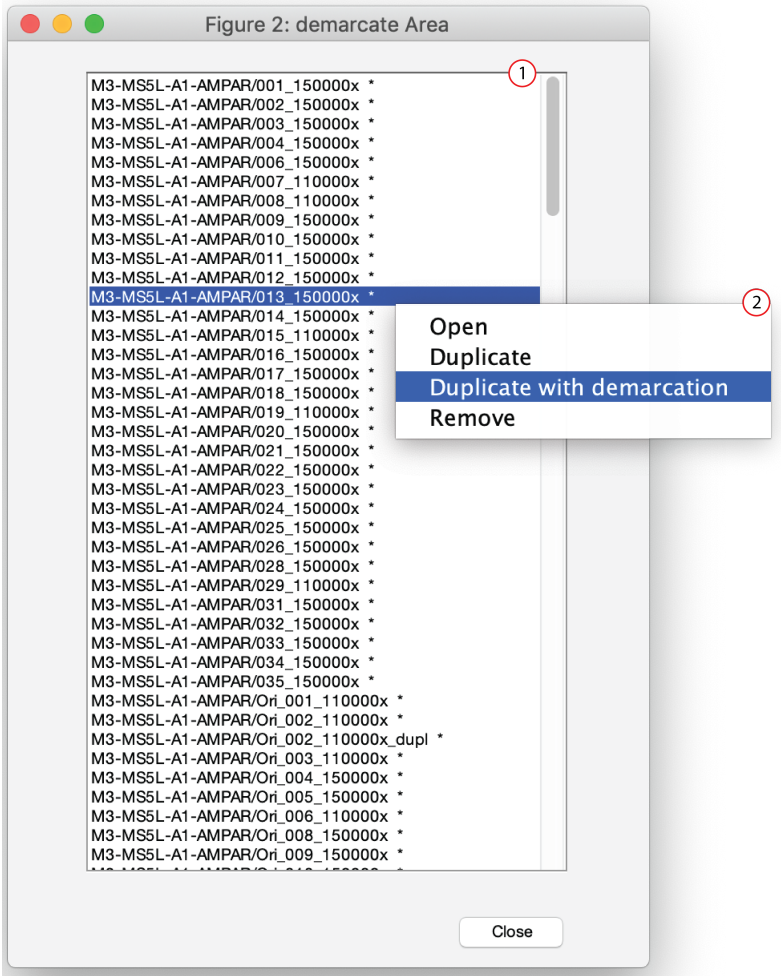

Figure 11: Open image

1: List of all images. Click to open image, or right click to display context menu (2). It is also possible to navigate using arrow keys and open an image by pressing enter.

2: Context Menu: "Open" will open the image (same as left clicking on it), "Duplicate" will duplicate the image. "Duplicate with demarcation" will duplicate image as well as demarcation and annotated particles. "Remove" will remove the image from the project (The image will not be deleted). Duplicating images is useful when more than one region of interest is on the image. If more than one region if interest is annotated, it will be treated as a large discontinuous region of interest. To separate it in two region of interest, duplicate the image and demarcate one region of interest each.

6.2 Manual Demarcation

If more than one region if interest is demarcated, it will be treated as a large discontinuous region of interest. To separate it in two region of interest, duplicate the image and demarcate one region of interest each.

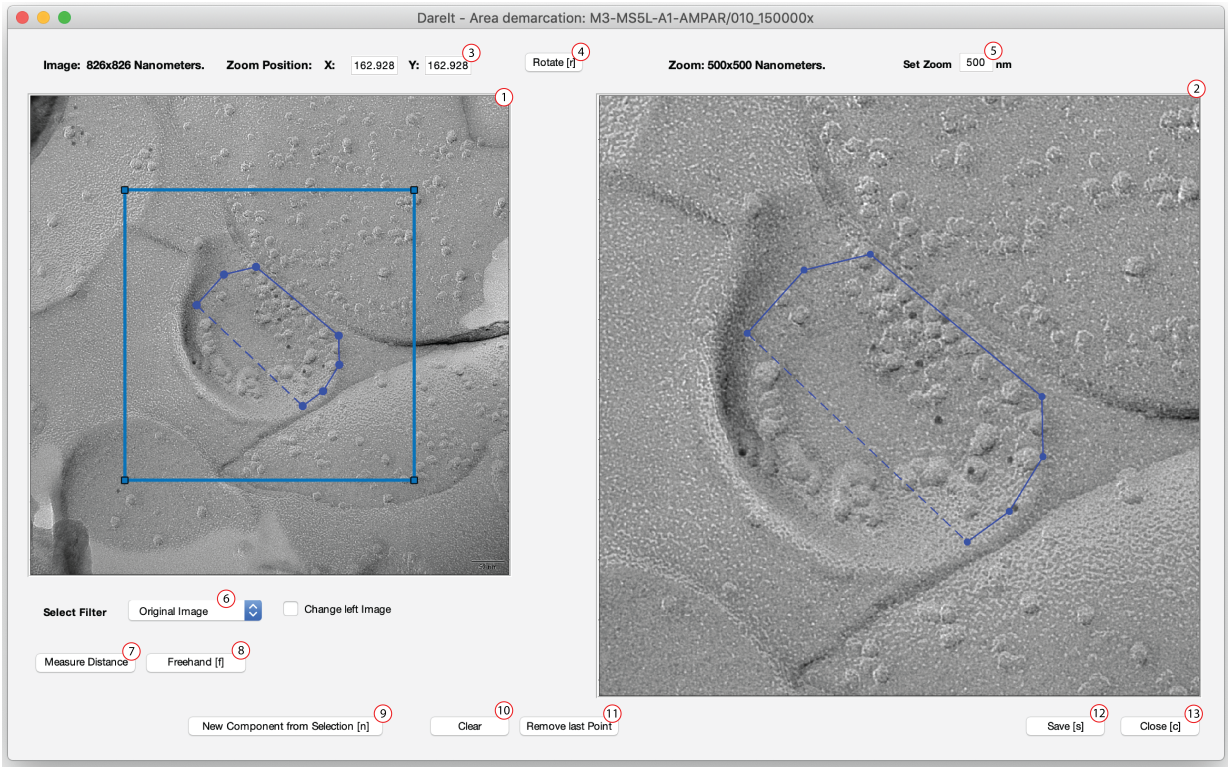

Figure 12: Manual demarcation

1: Overview image. Blue square outlines the regions displayed in (2) and can be moved and scaled.

2: Zoomed region. Clicking into this image allows you to create a polygon around your area of interest

3: Coordinates of the zoomed region. Can be manually edited to move blue rectangle to the described position

4: Rotate image 90°. All demarcations and annotated particle will rotate accordingly.

5: Length of the image shown in (2) in nm. Decrease the value to zoom in or increase to zoom out.

6: Select what should be displayed in the right panel (2). In this case the original image is displayed, which allows you to manually demarcate a region of interest (ROI). If a demarcation has been saved before, you can display it by choosing "saved Component".

7: When pressed click in (2) to measure distance between two points

8: When pressed clicking in (2) will allow to draw a freehand demarcation. After finishing your freehand drawing, click freehand again to convert it to a polygon (neccessary for saving the demarcation).

9: Convert your demarcation into a component that can be further modified (see Fig.13. Clicking this button is equivalent to saving, closing, reopening and selecting "saved Component" in (6).

10: Clears the polygon

11: Removes the last point of the polygon. Can be also done by pressing backspace

12: Save the demarcation

13: Close without saving

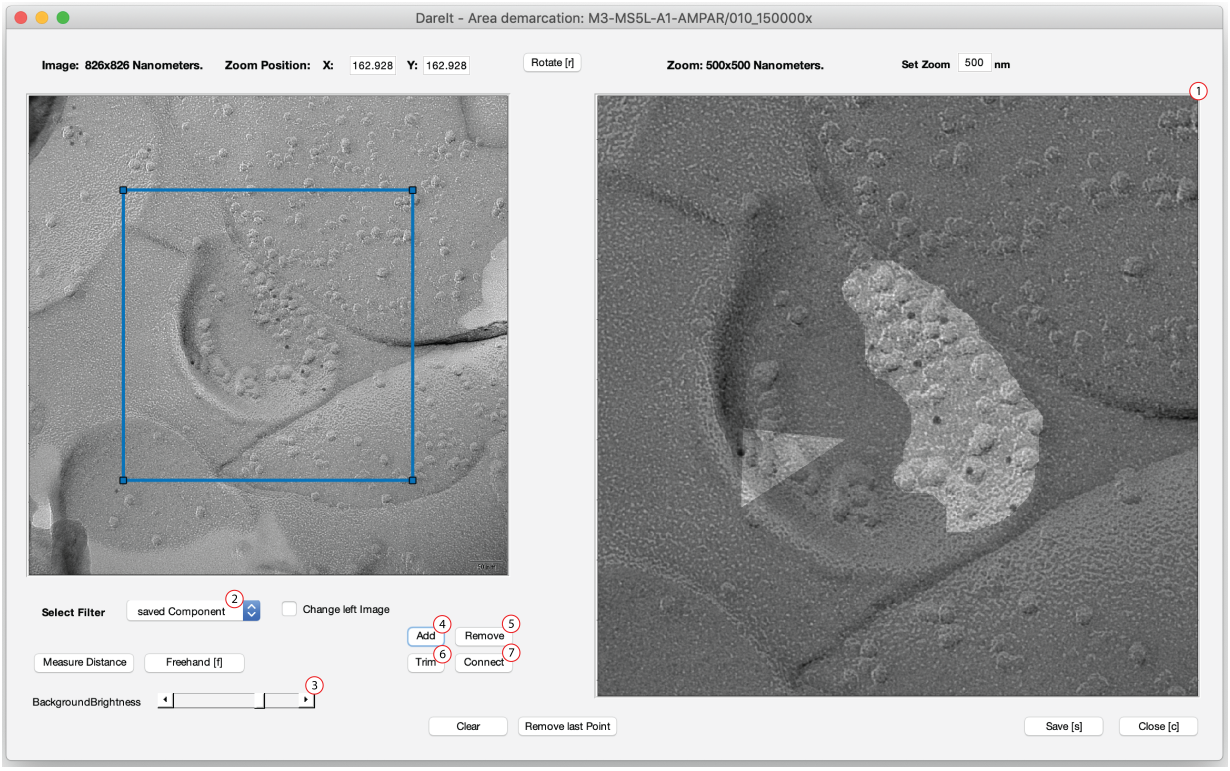

Figure 13: Manual demarcation Please refer to 12 for buttons already described there.

1: Zoomed image showing the components. Clicking on a component here will select this one and discard other components. That component can be then further refined (see Fig.14)

2: Select what should be displayed in the right panel (1). In this case the saved Component is displayed, which allows you to modify it or select a particular connected component by clicking on it.

3: When pressed, clicking on (1) will draw a polygon. Clicking add again will add the selected polygon as a component

4: When pressed, clicking on (1) will draw a polygon. Clicking remove again will remove the selected polygon from the component.

5: When pressed, clicking on (1) will draw a line. Clicking trim again will remove this line from the component. Useful for separating one component into two.

6: When pressed, two components can be selected in (1). They will then be connected by a thin line where the two components are closest.

7: Change the brightness of the background which is not part of a component

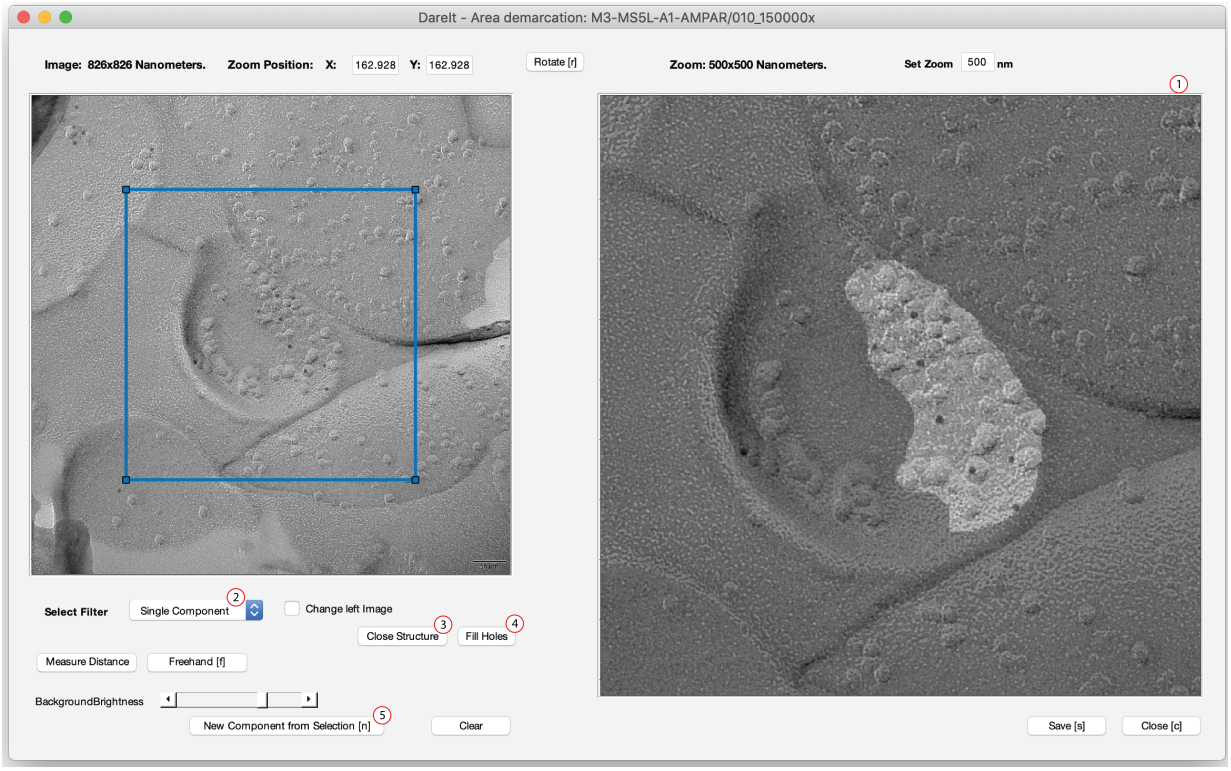

Figure 14: Manual demarcation. Please refer to Fig.12 for buttons already described there.

1: Select what should be displayed in the right panel (1). In this case a single connected component is displayed, which allows you to perform some automated finalizing steps like closing the structure or filling holes. Clicking on (5) will make it a new component which allows further manual modification.

2: This closes the region of interest with a radius of 7.5 nm. Demarcated points closer than 15 nm will be connected. This will lead to the removal of small holes and a smoother outline.

3: Fills any holes within the component

4: Makes the component modifiable again (see Fig.13)

6.3 Manual particle detection

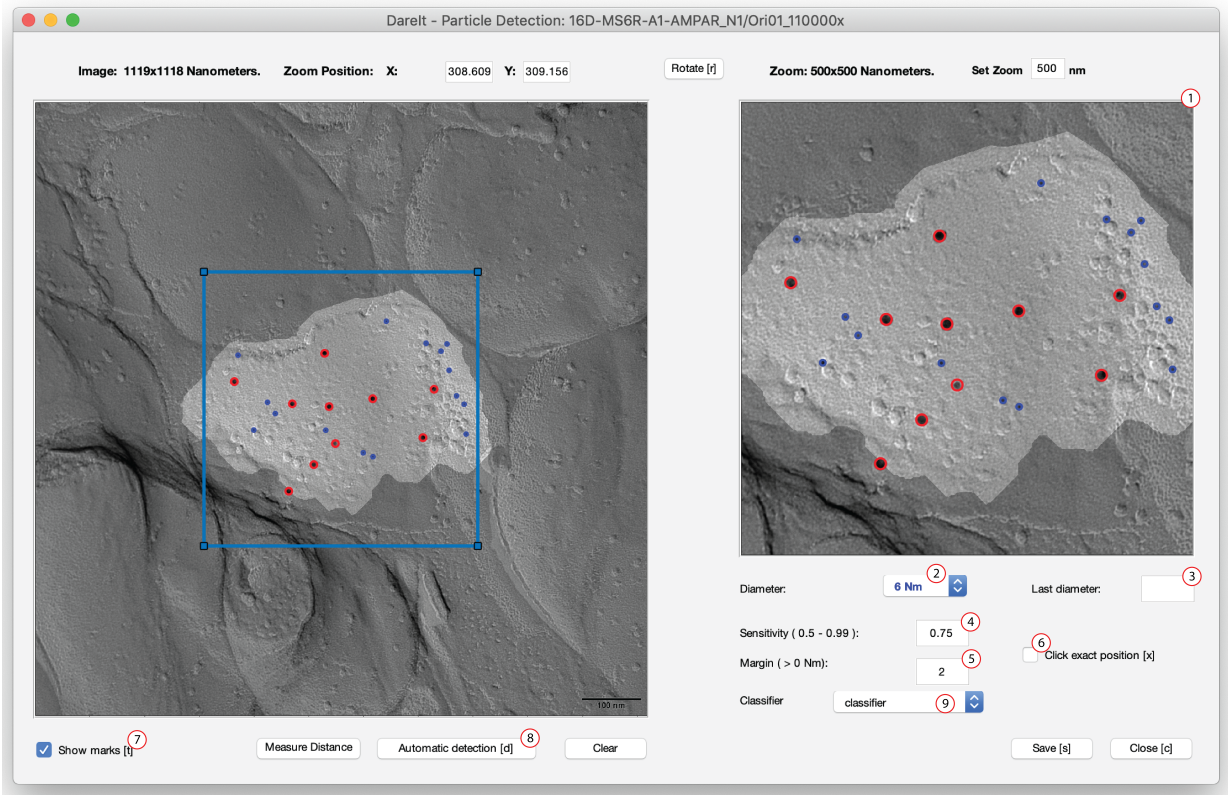

Figure 15: Manual particle labeling. Please refer to Fig.12 for buttons already described there.

1: Click on a particle to annotate it. Automatically, a black circle near where you clicked will be detected and annotated. So it is not necessary to precisely click the center of a particle

2: Select the particle size you are annotating

3: Measured diameter of the last annotated particle (depending on image contrast, this may be more or less accurate)

4: Sensitivity of the circle detection algorithm. Higher values will lead to more false positives, smaller values to more false negatives. If you try to click on a particle, but none is detected, try increasing this value.

5: Margin around the particle size, e.g. if the expected particle diameter is 6 nm and the margin is 2, particles between 4 and 8 nm will be accepted.

6: When selected clicking in (1) will annotate the particle with center exactly where you clicked.

7: Whether or not to show particle labels. When correcting automatic annotation, quickly switching labels on and off facilitates searching for false positives.

8: Automated particle detection of the selected size on this image. Margin and sensitivity are taken into account.

9: The classifier used for automated detection. Please refer to 9.1 for details.

7 Automated demarcation and particle annotation

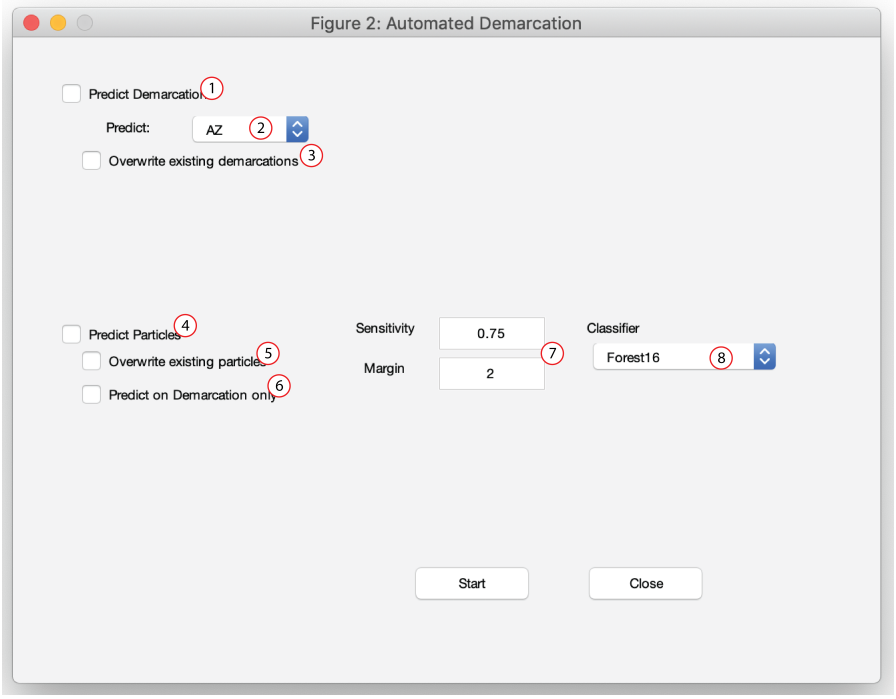

Figure 16: Automated demarcation and particle prediction

1: Whether or not to predict demarcation

2: Which feature should be detected. Refer to 9.2 for details.

3: Should existing demarcations be overwritten?

4: Whether or not particles should be automatically annotated

5: Should existing annotations be overwritten?

6: Should the prediction be limited to the demarcation only (much faster and therefore highly recommended).

7: Sensitivity and size margin of the circle detection algorithm. Please refer to Fig.15 for more details.

8: The classifier used for automated detection. Please refer to 9.1 for details.

8 Analysis

8.1 Running Analysis

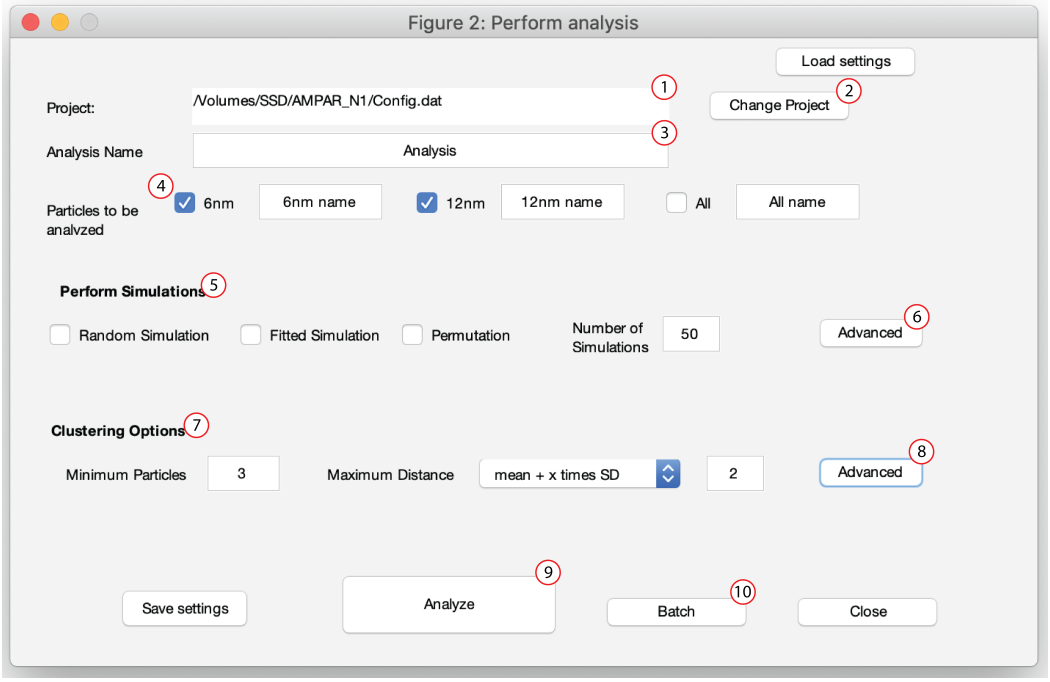

Figure 17: Analysis

1: Path to the project file

2: Change to a different project

3: Name of this analysis. Within the same project, analyses with different parameters (e.g. which simulations should be run) can be performed. They should then have different names to allow for easy distinction

4: Which particle sizes to analyze. You may give each particle size a name to facilitate interpretation of resulting data tables. "All" means all particle sizes are analyzed irrespective of size (in addition to any ticked sizes).

5: Options for Monte-Carlo simulations (see 8.1.1

6: Advanced settings for simulations (see Fig.18)

7: Clustering options (see 8.1.2)

8: Advanced settings for clusterings (see Fig.19)

9: Start analysis

10: Sometimes you may want to run multiple different analyses. Pressing batch designates your current input for analysis, then reopens this window. You may then specify your next analysis to run. After you finished setting up each analysis, press Analyze (9) to run all analyses at once.

8.1.1 Monte-Carlo Simulations

There are 3 kinds of simulations: Random, fitted and permutations.
Random simulation: The same number of particles as in the original image is redistributed in the region of interest. Each pixel as the same probability of becoming the center of a particle. Particles need to be some minimum distance from each other.
Fitted simulation: First a random simulation is performed, than it is checked if the particle distribution is appropriate (see Fig. 18 (2)). If so it is accepted, if not a random particle is randomly redistributed and it is checked if the new simulation is closer to acceptable than the old. If so, it is kept, otherwise it is reverted to the previous situation. This is repeated until the distribution is appropriate.
For both random and fitted simulation, only particles of one size are simulated at the same time (i.e. different simulations will be created for simulated 5nm and simulated 10 nm particles).
Permutation: Particle identities are randomly reassigned (e.g. if there are 2 5nm particles and 3 10nm particles, 2 particles will be randomly picked to become 5nm and the others become 10nm).

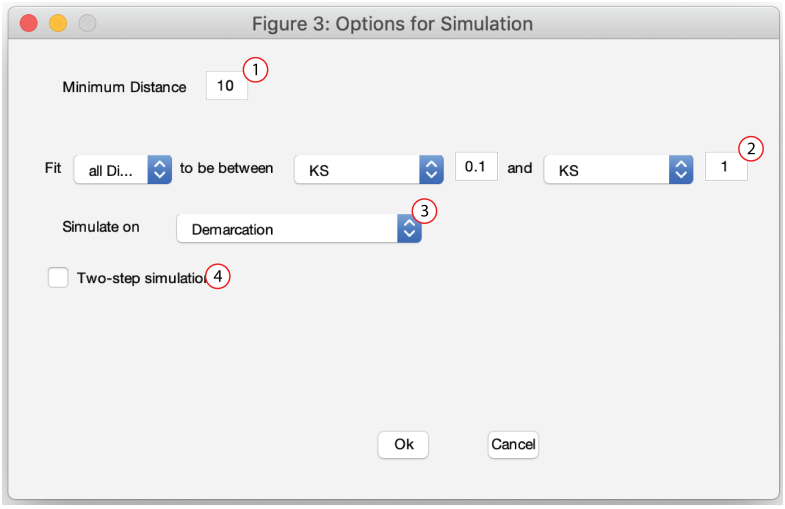

Figure 18: Simulation Options

1: Minimum distance between two particles

2: Describes when a fitted simulation is appropriate. In this case all Distances between particles will be compared between real and simulated using a Kolmogorov-Smirnov (KS) test. If the p-Value is between 0.1 and 1, the simulation is accepted.

3: Whether to simulated particles should be distributed only on demarcation or on demarcation + outer rim

4: If clicked, rather than distributing particles immediately, first epitopes (i.e what you were labeling for) are randomly distributed. In a second step particles are then redistributed a random distance (between 0 and a specified distance) at a random angle away from the epitope.

8.1.2 Clusters

Clusters are composed of a minimum number of particles which are closer to each other than some minimum distance. For these clusters of particles, several parameters will be computed: Cluster Area, number of particle per cluster, number of clusters per image, density of particles in a cluster, distance between clusters, overlap between clusters. The minimum number of particles that can be specified is 3 (because clusters smaller than that would not have an area, as they would just be a line between particles). The maximum distance between particles of the same clusters can be specified either in nm, or can be calculated from average NND. In the example above, this maximum distance is calculated as mean + 2*SD of the NND.

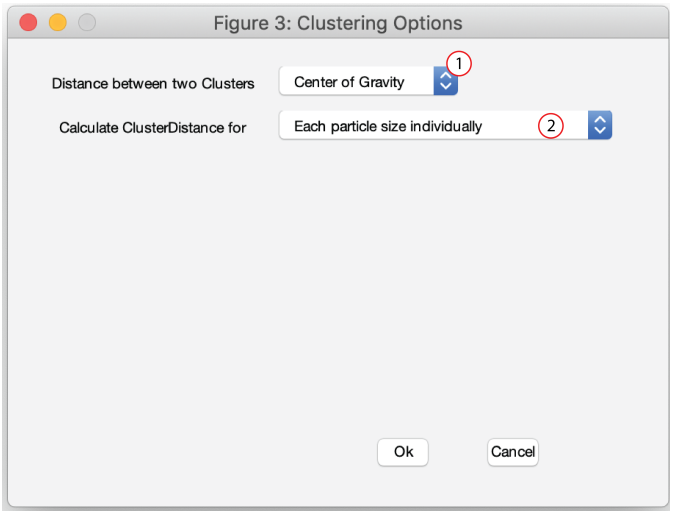

Figure 19: Clustering Options

1: How distance between two clusters should be measured.
The possibilities are: "Center of Gravity" -> Distance between the two cluster's center of gravities.
"Particle to Particle" -> Distance between the two nearest particle of the clusters.
"Outline to Outline" -> Distance between the two nearest points of the cluster outlines (Outline is specified as convex hull of all particle centers of the cluster).

2: If cluster distance is not fixed in nm, but calculated from NND. From what should it be calculated?
The Options are:
"All groups and particle sizes together" -> mean and SD of NND is calculated from pooled data regardless of particle size or group.
"Each particle size individually" -> mean and SD is calculated for each particle size separately. That means different particle sizes will have different maximum distances for clustering.
"Each group individually" -> mean and SD is calculated for each group individually. Different particle sizes are pooled.
"Each group and particle size individually" -> mean and SD is calculated for each group and particle size individually. If groups should calculated individually, a selection will appear where you can select which groupings are relevant.

8.2 Making tables and figures

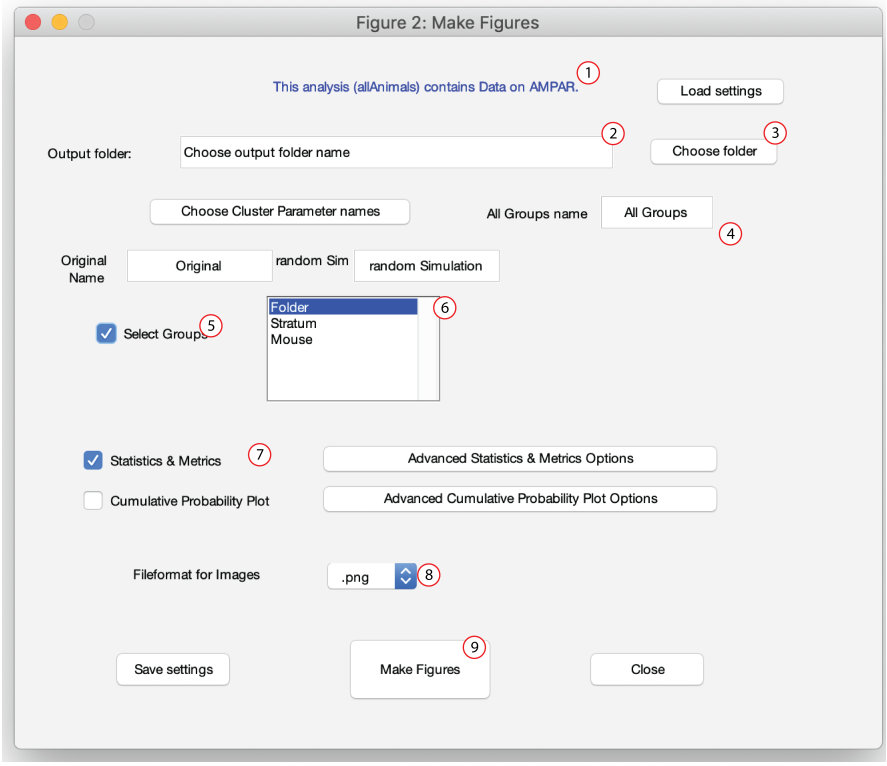

Figure 20: Make Figures.

1: Shows the name of the analysis and the name of the particles analyzed

2: Path to the selected output folder

3: Select an output folder. As there will be a lot of files generated, we suggest selecting an empty folder.

4: How different things should be named in the resulting datasheets and figures. Right now this naming selection is not followed through in all datasheets and we suggest to largely ignore it. It may be removed or modified in the future.

5: If ticked, you may select based on which groupings the data should be separated. Not ticking is equivalent to selecting all groupings in (6).

6: Select the groupings based on which the data should be divided. In this example, if I select Stratum but not Mouse, data will be seperated based on stratum but individual mice will be pooled.

7: Which kinds of data sheets and figures to create. For a standard analysis we recommend ticking statistics and metrics and not changing advanced options.

8: Which file format generated images should have.

9: Make figures and tables

8.2.1 The Output files

When making figures, the following directories and files are created in the selected output folder:

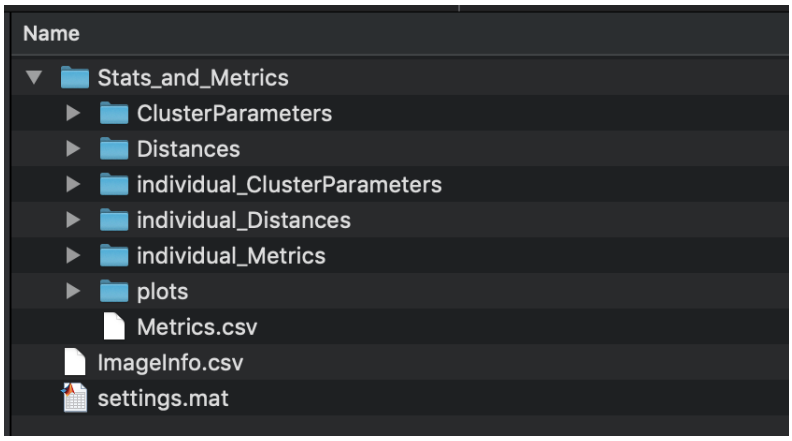

Figure 21: Folderstructure in the output folder

ImageInfo.csv contains a list of all images with their scales and the groups they belong to. settings.mat saves the settings used for making these results.

Within the Stats_and_Metrics folder there are:
Metrics.csv contains descriptive statistics on particle number, demarcated area and particle density for all particle sizes.

The directories "ClusterParameters" and "Distances" contain statistical information about distances and clusterparameters. Files are named <Measured Property>_<type of statistics>.csv.
The <types of statistics> are: "descriptiveStats" - contains mean, standard deviation (STD), standard error (SEM) and median. "analyticStats" - contains pairwise comparisons between each groups. "1by1" - contains an imagewise comparison between real and simulations, it is measured in how many percent of images the measured property is significantly smaller or larger than expected from simulations. "Popmeans" - Tests whether the average of the measured property is different between real and simulation using a paired t-test (pairing is done in an image-wise manner).

The measured properties for ClusterParamaters are: "area" - size of the cluster area. "density" - particle density within the cluster. "excluded Clusters" - number of clusters that were excluded from analysis (Clusters are excluded when all particles are in an exact line, because such clusters have no area. This number should be very close to 0. Large numbers indicate a problem in the analysis). "interDistance_x" - Distance from clusters of x nm particles to clusters of other particle sizes. "intraDistance" - Distance between clusters of the same particle size. "number" - number of clusters per image. "overlap_x" - Overlap between cluster areas of clusters of x nm particles and clusters of other particle sizes. "particles" - number of particles per cluster. "thresholdDist" - maximum distance

between particles which still counts them as belonging to the same cluster (see 8.1.2 for details).

The measured properties for Distances are: "All_Distances" - Distances between each pair of particles. "Distance_from_Center" - Distance from the center of gravity of the ROI (in nm). "Distance_from_Edge" - Distance from the edge of demarcation (in nm). "NND" - Nearest neighbour distance between particles. "normalized_Distance_from_Center" - $\frac{Distance\_from\_Center}{Distance\_from\_Center+Distance\_from\_Edge}$.

The directories starting with "individual" contain the unaggregated raw measurements. For individual_ClusterParameters the files are named <Groupname>_<SimulationType>.csv. For the real data no simulation type is specified. For individual_Distances the files are named <Groupname>_<Distance>.csv. The distances are either of form "Xnm to Ynm" and then contain measurements of distances from Xnm particles to Ynm particles, or of the form "Xnm" and then contain measurements between Xnm particles or from Xnm particles to center or edge. The individual_Metrics folder contains files name <Groupname>.csv. These contain information of number of particles, ROI area and density of particles for each image.

The directory "plots" contains plots of particle number against ROI area and of particle numbers of different sizes against each other.

8.3 Visualizing results

This is useful for creating example images.

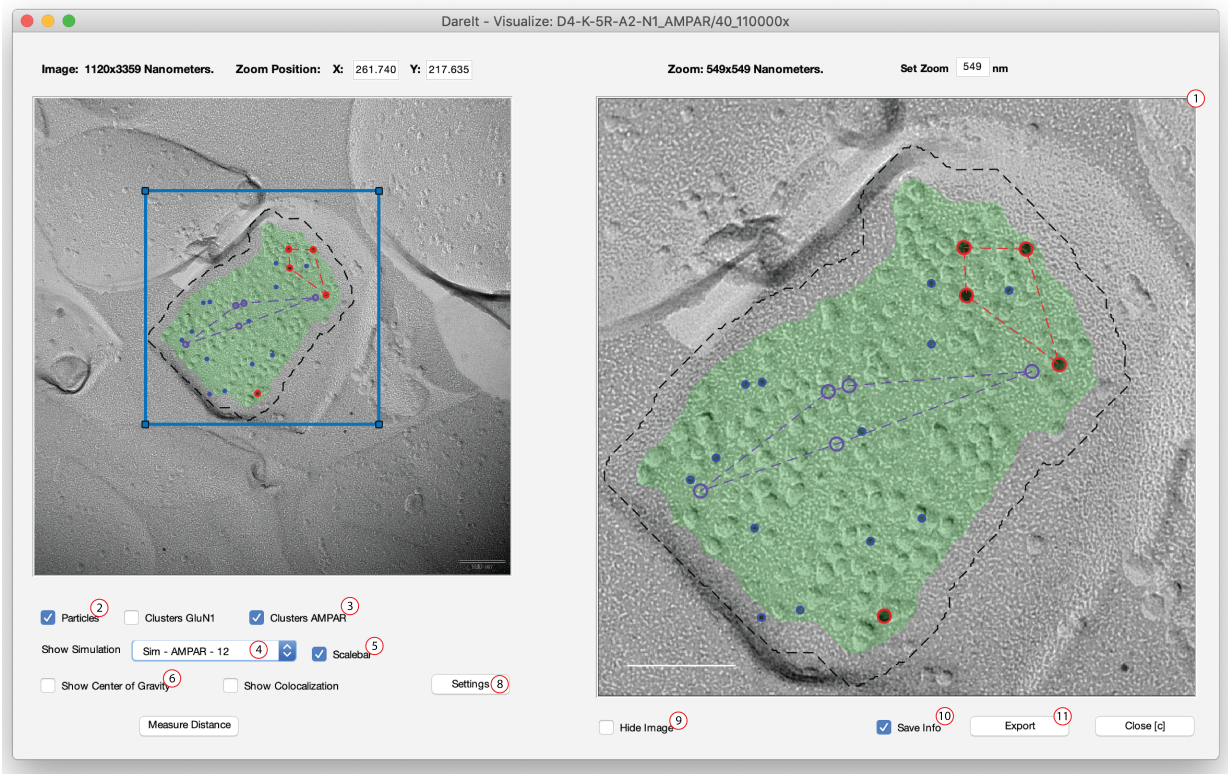

Figure 22: Visualize Results. Please refer to Fig.12 for buttons already described there.

1: When exporting the image, it will look the same as the image shown in this panel

2: Display labels for particles or not

3: Display labels for clusters or not

4: Select simulation from which to show simulated particles

5: Show scalebar or not

6: Displays center of gravity of the demarcation or not

7: Visualization settings, including particle color, how demarcation and outer rim should be displayed. See Fig.23 for details.

8: Hide image and display only labels on a white background

9: When exporting image, additional information (such as zoom size, zoom coordinates, ...) can be saved, so that exact image could be reproduced if minor changes are needed

10: Export image. You may select a path where to save the exported image and specify a fileformat.

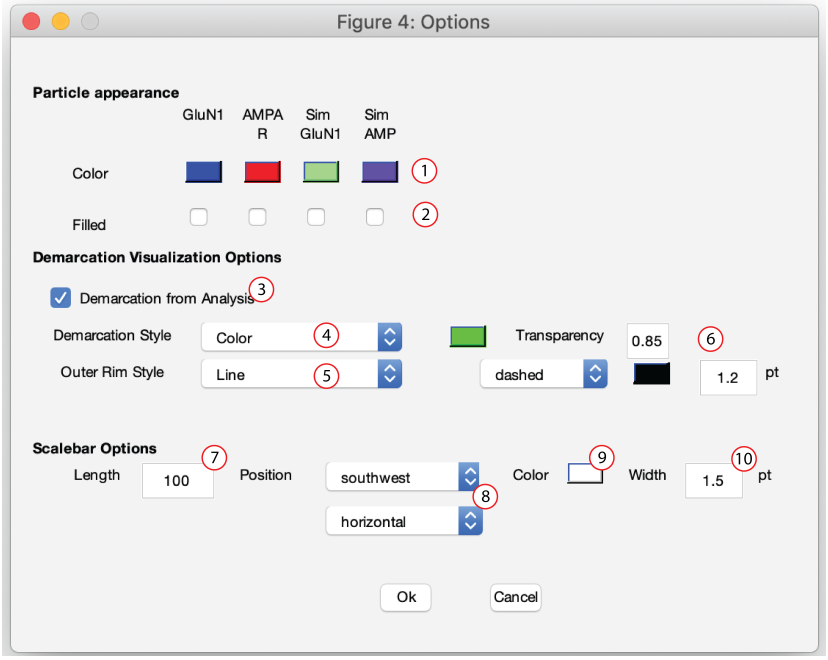

Figure 23: Visualize Options

1: Color of real and simulated particle labels

2: Whether the particle label should be filled (filled circle) or not (ring).

3: The demarcation can be either taken from the analysis or from the current demarcation file. If you changed the demarcation after you ran the analysis, different demarcations will be displayed.

4: How to display the demarcation: "None" -> don't display it. "Color" -> Show a color overlay. "Line" -> show a line at the outline. "Brightness" -> Make background darker.

5: Same as (4) but for outer rim

6: Several Options for displaying demarcation and rim style, depending on the style selected in (4) and (5)

7: Length of the scalebar in nm

8: Position of the scalebar within the image

9: Scalebar color

10: Scalebar line thickness

9 Training automated analysis

Training particle detection can be done on CPU. Preparing dataset for demarcation prediction can be done on CPU. Training demarcation prediction should be done on GPU (will take very very long otherwise).

9.1 Training particle detection

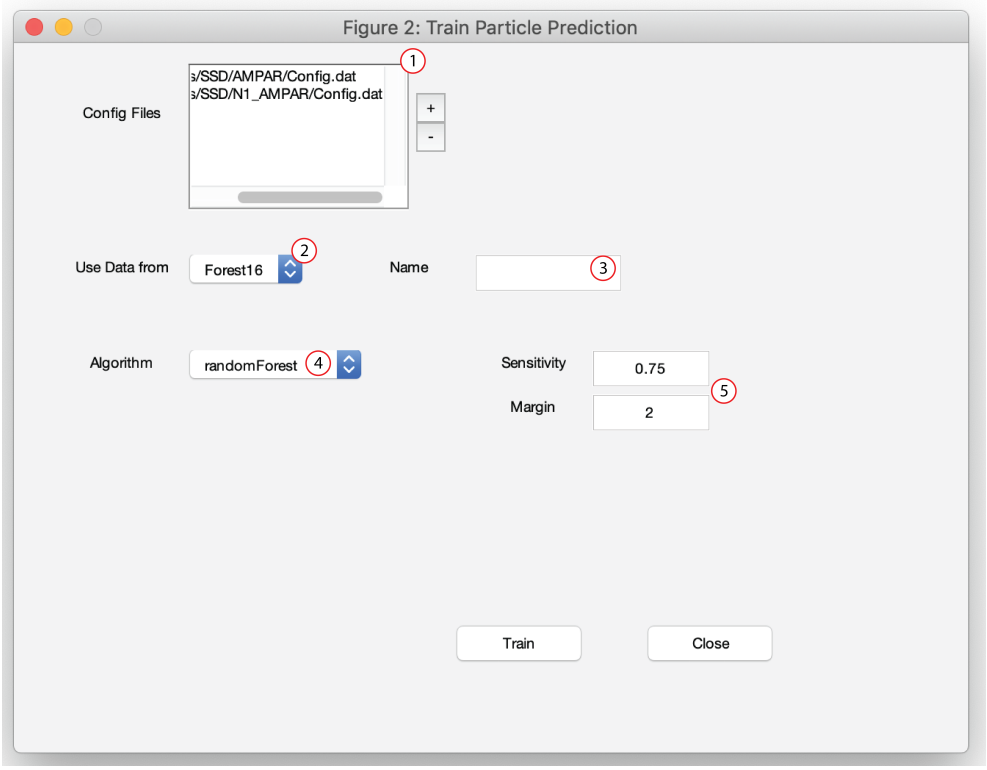

Figure 24: Training particle detection

1: Select which project files to train on. You may use (+) and (-) buttons to add or remove project files. *Important: Particle anntation in these files has to be completed, otherwise trained classifier will learn the wrong thing*

2: Training data used in already trained classifiers can be reused for training the new one (this is very quick). Select from which classifier trained data should be use or select "None" to train a fresh classifier

3: Name of your new classifier. If you specify one of the names shown in (2), it will overwrite this classifier

4: Which algorithm to use for training your classifier

5: Sensitivity and margin to use for circle detection algorithm. See Fig.15 for explanation.

9.2 Training demarcation

For training deep neural networks for automatic demarcation we suggest to create a dataset with the following structure:

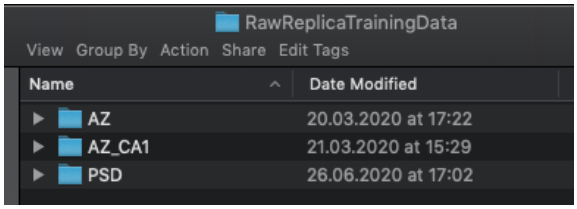

Figure 25: Dataset structure

Each folder corresponds to a feature (e.g. active zone, PSD) you would like to train. Copy your project (<projectname>.dat and all the folders containing the images) into the folder specifying the appropriate feature. From this folder a dataset with ready to train images can be prepared (see below).
If at a later point more fully analyzed projects for the same feature become available, they can simply be copied into the dataset and "prepare dataset" can be rerun. Than the new images will be included for the training. Theoretically, when always adding the latest finished project to the dataset and rerunning training, this should lead to constant improvements of the accuracy of the automated prediction.

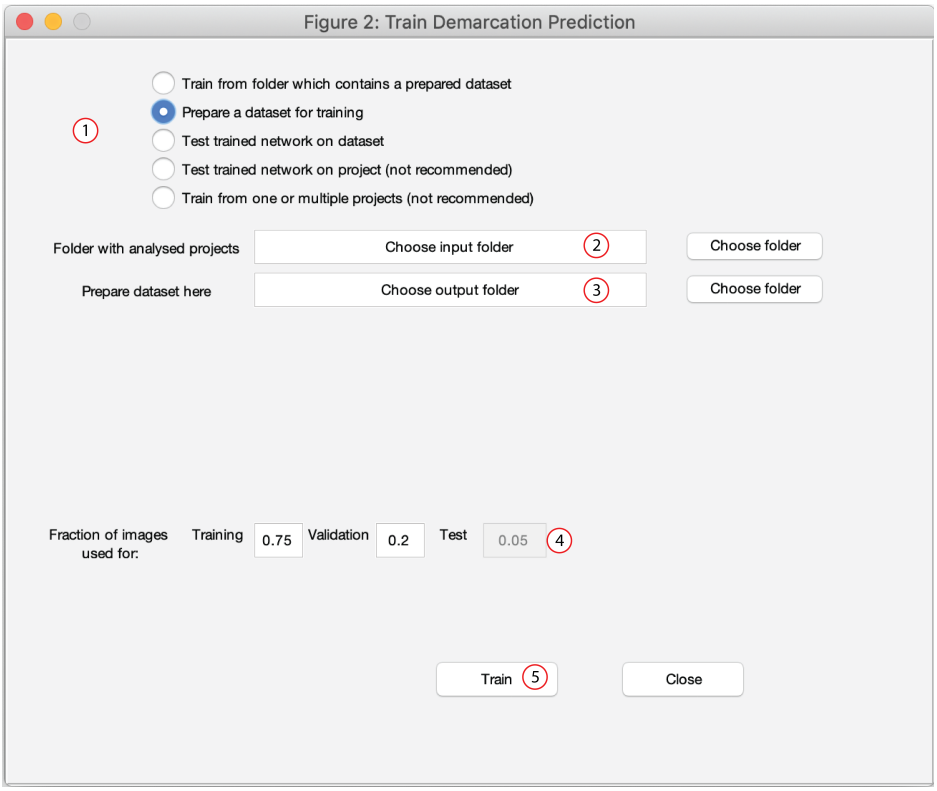

Figure 26: Preparing an image dataset for training

1: Select Prepare dataset

2: Choose the folder with the dataset (see above)

3: Choose the folder which will contain your prepared dataset. (Should be empty or contain a dataset previously prepared from the same folder with annotated projects (more projects could be added since then))

4: Clicking train will now convert your images to make them ready for training. This will take some time.

5: What fraction of images should be used for training, validation and testing (since they need to add up to 1, Test is automatically calculated based on the other two).

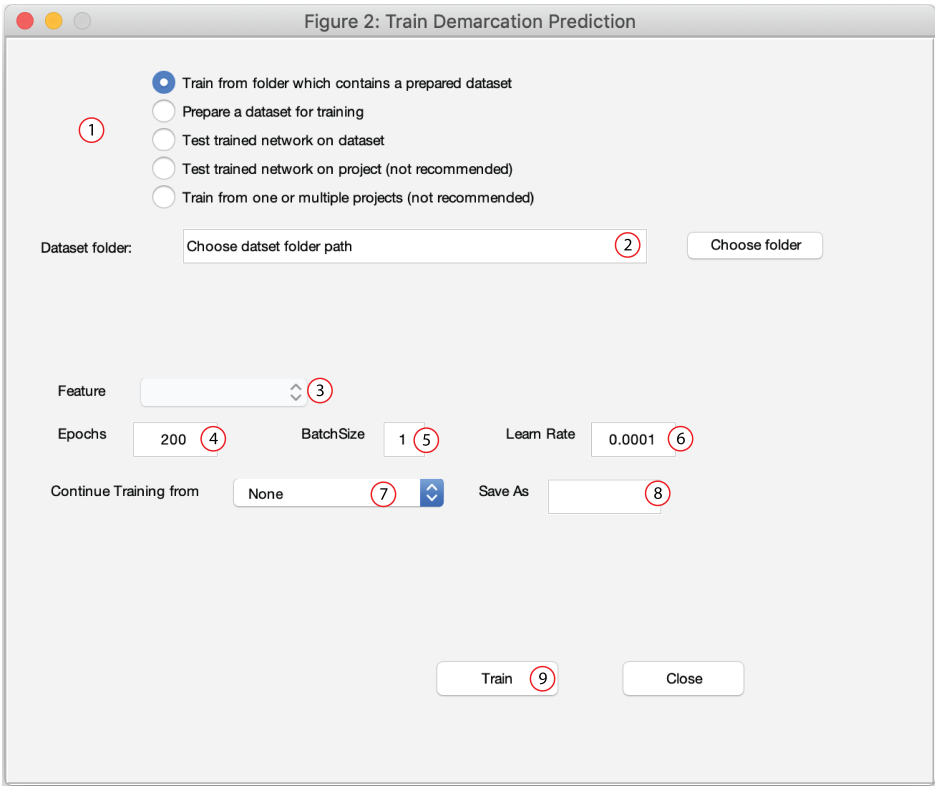

Figure 27: Train deep neural network for demarcation

1: Select train from prepared dataset

2: Choose the folder containing your dataset. This is the same folder as specified in (3) at Fig.26.

3: Choose which feature to train from.

4: Number of epochs. During each epoch, the network will be trained on each image of the training dataset once. The higher this number, the longer it takes but the better the result.

5: How many images should be processed at the same time. Unless your GPU has a very large memory, leave unchanged.

6: How fast the network should learn. Higher rates will speed up the learning but may deteriorate performance.

7: Training can be continued from a previously trained network which will result in much fewer needed epochs for reaching a good performance

8: The name under which the newly trained network should be saved. When you specify a name that already exists (see list in (7)), that one will be overwritten.

9: Click train to start training. Depending on number of training images and epochs this may take up to a few days (on GPU) or multiple weeks (CPU).

Rather than creating a dataset and training from it, directly training from one or more projects is also possible. Except for quick testing of training parameters, this is not recommended for multiple reasons: 1) Images have to be converted into a dataset anyway, but since it is not saved, this has to happen every time leading to (slightly) increased training times. 2) For the same reason, the training/validation images will change every time (as they are randomly picked to reduce bias). This can lead to performance problems

("Overfitting") when retraining the network multiple times on the same project. 3) Sometimes there is more than one region of interest on an image. Usually these are analysed by duplicating the image and selecting one ROI each. If uncorrected, this will cause problems because the network will see the same image, but has to guess which of the ROI would be appropriate, which will reduce performance When preparing a dataset, but not when training from a project, this issue is corrected for.

10 Advanced sections

10.1 How Darea stores data and modifying defaults

10.1.1 Project files

Project files (saved as <projectname>.dat) store the relative paths of the images, their primary group (which corresponds to the folder in which they are in) and their scale (in nm/px). They look like this:

```
GROUP,  ROUTE,  PIXELSIZE
PF_Syn, PF_Syn/01_37984_F1_II,  0.3
PF_Syn, PF_Syn/02_37984_F1,     0.3
```

Columns are seperated by commas and tabs. Images can be manually added to and removed from a project by modifying this file.

10.1.2 Project Settings and modifying defaults

Project settings files (saved as <projectname>_options.dat) store all settings variable differing from the default. All default values (including some which cannot be changed in the GUI) are stored in configDefault_options.dat in the Darea main folder. Here's a small part of it as example:

```
#General
particleTypes   [5,10]  #Particle Diameters
dilate  0       #Size of outer rim in nm
onlyParticlesWithin     true   #Only particles within area of interest
    should be taken into account

BackgroundBrightness    0.7     #How much darker the non demarcated
   area should be

#Analysis
nrsim   50      #Default number of simulations
allName 'All'
SimOptions.mindistance  10
```

Setting names and values are separated by tabs and # can be used for commenting. Changing these values in this file will modify the default behaviour for all projects. Adding any of these settings to the project settings file will change this for that project only. For example, writing

```
nrsim   100
```

into the project settings file (create it yourself if it doesn't exist) will cause the default number of simulations in Fig.17 (5) to be 100.

10.1.3 Groups

Groups are saved as <projecname>_groups.dat which look like this:

```
Image,Folder,Stratum,Mouse
M3-MS5L-A1-AMPAR/001_150000x,M3-MS5L-A1-AMPAR,Radiatum,MS5
M3-MS5L-A1-AMPAR/002_150000x,M3-MS5L-A1-AMPAR,Radiatum,MS5
M3-MS5L-A1-AMPAR/003_150000x,M3-MS5L-A1-AMPAR,Radiatum,MS5
```

The first column contains the path to each image, the other columns show the different groupings. Elements are separated with commas.

10.1.4 Demarcations

For each image the demarcation is stored in a <imagename>_mod.tif file, which is a grayscale image where white denotes background and any other color denotes ROI.

10.1.5 Particle annotations

Particle annotations for each image are stored in a <imagename>dots.csv file, which looks like the following:

```
513.9198, 464.5443, 3.2260, 3.0
625.0329, 500.4324, 3.2260, 3.0
386.2429, 340.1549, 6.4520, 6.0
551.6772, 399.8392, 6.0488, 6.0
```

Each row is one particle. The columns are: X and Y coordinates, measured radius and theoretical radius of the particle (all in nm). Elements are separated with commas.

10.1.6 Analysis results

Analysis results are stored as <projectname>_<analysisname>.mat files.

10.2 Deep learning using the command line

When wanting to train when no graphical user interface (such as when running it on a cluster) is necessary or possible, only the python/SemanticSegmentationSuite folder is necessary. Running the training can be done using the following command:

```
python <path to SemanticSegmentationSuite>/train.py  --learn_rate "<
   learn_rate>" --batch_size "<batch_size>" --num_val_images "-1" --
   num_epochs "<nrEpochs>" --dataset_path "<path to dataset>" --
   dataset "<feature name>" --save_best "1" --h_flip "1" --v_flip "1"
   --brightness "0.5"
```

The terms between <> have to be replaced accordingly. <path to dataset> should be the path to the folder where the prepared dataset is located, same as what you would select in Fig.27 (2). The other terms should be filled the same way as the elements in Fig.27. More options are possible, refer to SemanticSegmentationSuite/train.py for details. Take care to first install all the appropriate python packages (an environment file we use for our cluster is available as Darea_LinuxCluster_GPU.yml) and make sure that the python command links to the proper env having these packages.