MORPHIOUS: A Machine Learning Workflow to Naively Detect the Activation of Microglia and Astrocytes – a Tutorial

Joey Silburt

This readme manual is an accompaniment to the accompanying MORPHIOUS research paper, available at: (Silburt & Aubert, 2020)

Table of Contents

[Brief overview: 1](#_Toc95678640)

[Step 1: Extract features from your images using the provided imageJ macros 1](#_Toc95678641)

[Step 2: Open MORPHIOUS with python. 3](#_Toc95678642)

[Step 3: Load training and test sets into the program: 4](#_Toc95678643)

[Step 4: Select MORPHIOUS Parameters 5](#_Toc95678644)

[Step 5: Select Features. 6](#_Toc95678645)

[Step 6: using cross validation in the training set and finding clusters in the test set. 7](#_Toc95678646)

[Step 7: Save output cluster files. 7](#_Toc95678647)

[Step 8: Using a grid search to find optimal MORPHIOUS parameters 8](#_Toc95678648)

[Step 9: Generating MORPHIOUS generated ROIs 9](#_Toc95678649)

# Brief overview:

MORPHIOUS uses a one-class support vector machine to detect clusters of activated microglia or astrocytes in tissue sections after referencing examples of normal, untreated tissue. MORPHIOUS requires a discrete training set of control images where cells of interest are known to be at baseline or resting states. After training, MORPHIOUS can be applied to a second test set where it can identify the presence of clusters of activated cells.

# Step 1: Extract features from your images using the provided imageJ macros

Four feature extraction macros are provided to extract a variety of features.

1. get\_intensity\_measures.ijm

Evaluates various intensity measures for the image including “Mean”, “IntDen”, “circularity”, etc.

1. get\_fractal\_measures.ijm

Evaluates the fractal dimension.

1. get\_branch\_features.ijm

Evaluates various skeletal metrics, such as branch length, number of branches, number of junctions, etc.

1. count\_microglia.ijm

Identifies the soma body of Iba1+ microglia. Regions of interest corresponding to these somas are saved. Moreover, soma features, such as size and circularity, etc., are evaluated.

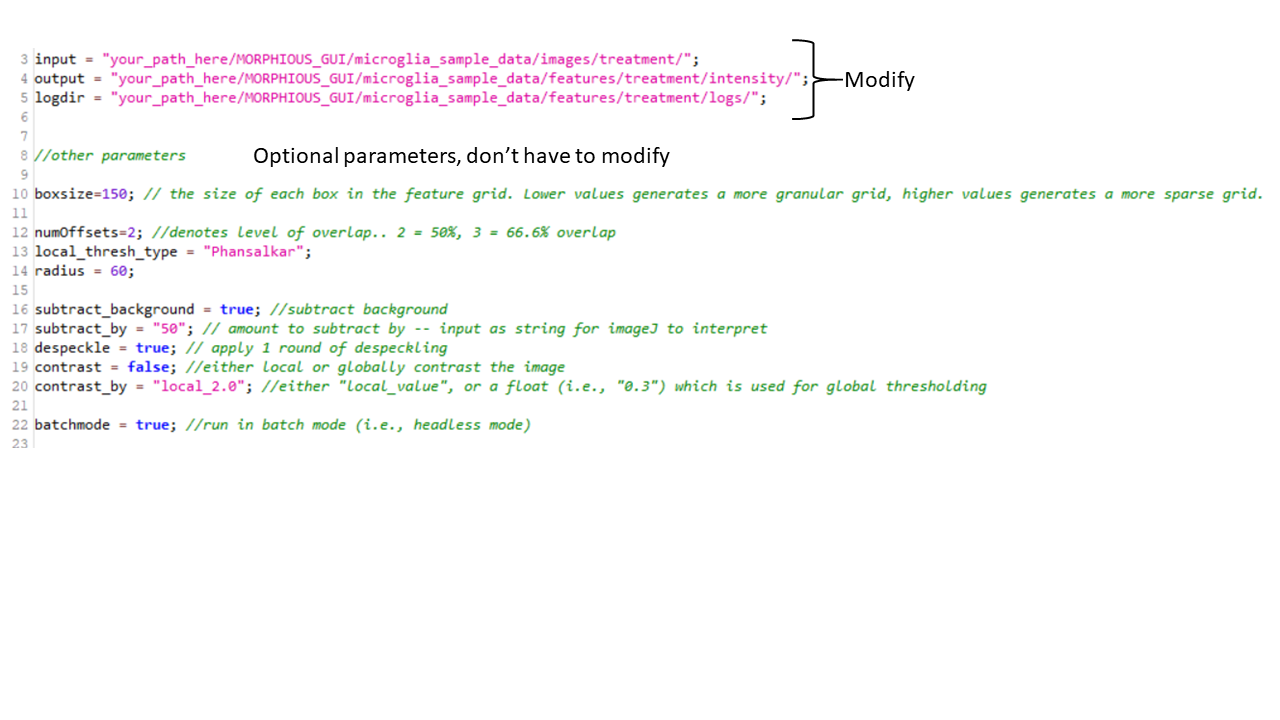
1. count\_astrocytes.ijm

Identifies the soma body of S100β+ astrocytes. Regions of interest corresponding to these somas are saved. Moreover, soma features, such as size and circularity, etc., are evaluated.

In general, to use these macros, you are only required to input the location of your images, and output directories. These should be created in advance. Control and test image datasets should be separated into separate directories. Moreover, extracted features for control and test images should be saved in separate directories. All macros also require the user input the location of a directory for a log file, where all parameters are saved. The get\_fractal\_measures.ijm, get\_branch\_features.ijm, and count\_microglia.ijm, and count\_astrocytes, ask for the input of additional directories for saving binarized images.

Please note: directories should be separated by a forward slash, i.e., “/” not a “\”, which is used as an escape code character in ImageJ’s macro language.

For example, to evaluate intensity measures for the control dataset provided in the sample data, an input scheme would look like this:



For get\_intensity\_measures.ijm, get\_fractal\_measures.ijm, and get\_branch\_features.ijm a number of additional parameters can be altered at the users the discretion. These include:

**Boxsize:** The size of the box created in the grid. Smaller values will generate a more granular feature map, while larger values will generate a more sparse feature map. The boxsize should be sufficiently large to capture meaningful cellular features (e.g., branching complexity). Moreover, boxsizes which are too small may significantly increase processing time.

**numOffsets:** is inversely proportional to the degree a box is slid in the x and y directions. A numOffsets of 2 means that the box is translated by 50% in the x and y directions. A numOffsets value of 3 would result in the box being translated by 33% in the x and y directions. Larger values of numOffsets results in more granular feature maps.

**Local\_thresh\_type:** The autolocalthreshold type used for imageJ, it is recommended to not change this value.

**Radius:** corresponds to the radius parameter in ImageJ autolocalthreshold.

**Subtract\_background:** whether to apply a rolling box background subtraction.

**Subtract\_by:** The radius parameter for the imageJ background subtraction method.

**Despeckle:** applies one round of image despeckling.

**Contrast:** whether to apply either a global or local contrast to the image. If false, the image will not be contrasted regardless of the value of “contrast\_by”.

**Contrast\_by:** defines the contrast method and the degree of contrasting. Values of the form, “local\_x.x” will perform local thresholding with the x.x referring to the maximum parameter value for ImageJ’s Enhance Local Contrast (CLAHE) method. If only a value is provided e.g., “0.3”, a global contrast is applied, with the value provided used as the percent saturation parameter.

**Batchmode:** Whether to run in ImageJ’s batch mode (i.e., headless). Setting Batchmode to true will make the extraction run faster.

The macros, count\_astrocytes.ijm, and count\_microglia.ijm, contain addition parameters which the user is discouraged from modifying, as these have been experimentally optimized to improve soma detection. Nevertheless, it is possible that other parameters will yield better results depending on their specific imaging parameters.

# Step 2: Open MORPHIOUS with python.

Ensure python is installed on your machine. Downloading python via the Anaconda package manager is recommended, as it comes with several important packages pre-installed.

If not already installed, MORPHIOUS depends on the numpy, scipy, pandas, and sklearn libraries.  
To install this with python, simply type:

pip install numpy

pip install scipy

pip install pandas

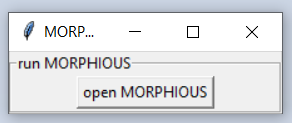
pip install sklearn

Using a computer terminal, locate the MORPHIOUS\_GUI directory. Once in the MORPHIOUS\_GUI directory, type:

python morphious\_main.py

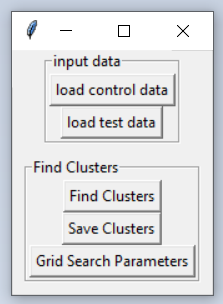


This will produce the main MORPHIOUS frame:

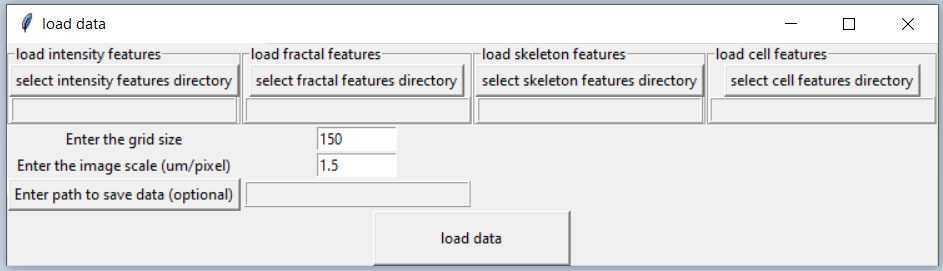


Click open MORPHIOUS to proceed, yielding the main options dialog.

# Step 3: Load training and test sets into the program:



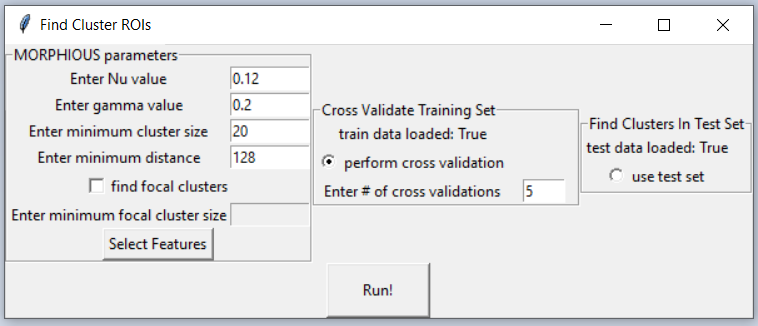
For a typical work flow, the user should load both control and test data sets. Selecting either the “load control data” or “load test data” buttons will yield a panel prompting the user to locate the outputs of the imageJ macros from step 1.



The user is not required to load all of intensity features, fractal features, skeleton features, and cell features, however, data from atleast one of these sources is required. The user is also prompted to enter the boxsize that was chosen during the feature extraction phase in the imageJ macros. Finally, the user should input the scale of the image. This can be found in imageJ via Analyze 🡪 set scale.. on an open image. You can also choose to save the merged feature data frame by entering a path to save.

# Step 4: Select MORPHIOUS Parameters

After data is loaded, users can proceed to the “Find clusters” frame.



The user has several parameters they need to set, however, in general, these parameters are determined experimentally via a grid search (see below).

**Nu:** This parameter governs the ‘misclassification rate’ of the one-class support vector machine. Larger values will yield a greater number of outliers, and therein a greater degree of clustering. Depending on your experimental setup, optimal values may fall somewhere between 0.01 and 0.2.

**Gamma:** This parameter governs the complexity of the radial basis function kernel for the ocSVM. Values should generally fall between 0.1 and 0.25; In our experience, it is not advisable for values to exceed 0.3, as this leads to poor performance.

**Minimum cluster size:** This parameters controls the number of outliers required to be within the minimum distance of a neighbouring outliers to be labelled a cluster. For default feature extraction parameters, this value may fall between 18 and 22. However, reducing the box size, or increasing the number of box offsets will increase the minimum cluster size.

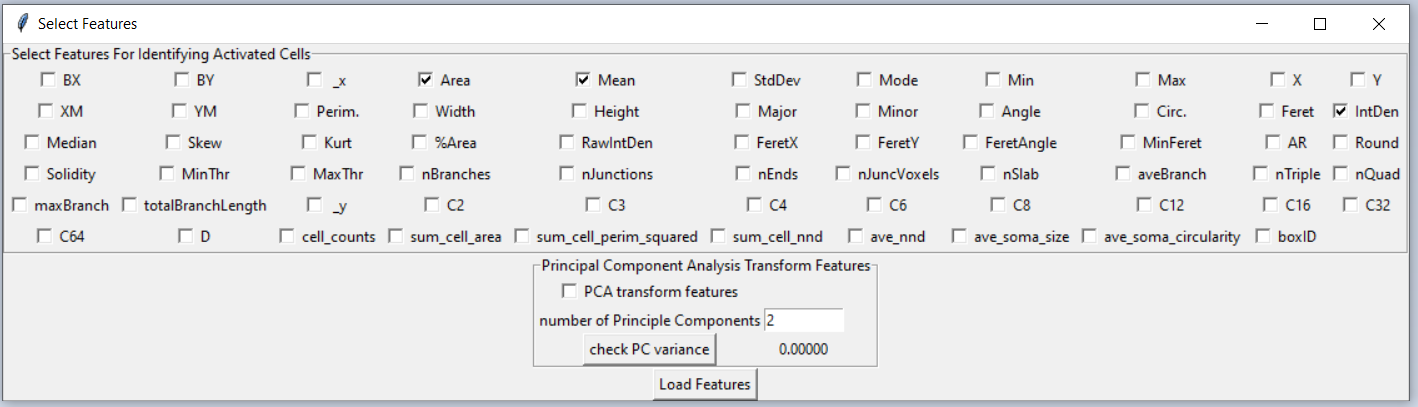
**Enter minimum distance:** The distance between neighbouring clusters to determine if they are part of a cluster. In our experience, a good default setting is to set this distance as the size of the hypotenuse of a boxsize by boxsize triangle.

**Find focal clusters:** if selected, MORPHIOUS will evaluate the presence of focal clusters. Focal clusters represent the most extreme variants of proximal clusters, based upon integrated density values.

**Minimum focal cluster size:** See minimum cluster size, in our experience, a good default setting is 5.

Before proceeding to clustering, the user should select the features they wish to identify activated clusters with.

# Step 5: Select Features.

****

Features extracted via the ImageJ feature extraction macros are available for selection here. The user also has the option to transform the features via principle component analysis. Please be aware, some features available via this dialogue, such as BX/BY contain no meaningful information about the difference between an activated and non-activated cell, and so should be avoided. For more information on the meaning of each feature, read the imageJ documentation.

**PCA transformation:** Our recommendation is to PCA transform features with enough principle components to retain >99% of the feature variance. A button is available to assess the current variance encapsulated by the selected number of principle components.

# Step 6: using cross validation in the training set and finding clusters in the test set.

MORPHIOUS works by learning a set of parameters which result in no activation clusters in control tissue, while maximizing the set of clustering in the test tissue. Cross-validation is used to evaluate whether that selected MORPHIOUS parameters yield clusters of activated cells in the control tissues.

With cross-validation, the training set is randomly divided among an equal number of subsets (as determined by the user). For each round of cross-validation, one of the subsets is used as the test set, while the remainder are used to train the classifier. This is repeated until all data subsets have been evaluated as the test set. In this way, each control set image can be evaluated for the presence of spurious activation clusters. Detecting no clusters in the cross-validation step is needed to validate the integrity of predicted clusters in the test set. Cross-validation only ever searches for “proximal” clusters, therefore selecting the “find focal clusters” option does not impact the cross-validation results.

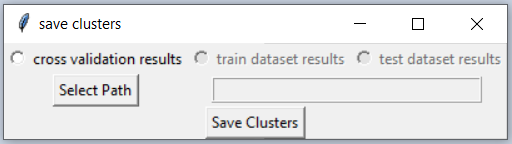
Performing a cross validation is designated by selecting the “perform cross validation” radio button. The number of cross-validation subsets is indicated by “the # of cross validations” parameter. Increasing the number of subsets will improve the model performance, however, may significantly increase computational time.

After a set of MORPHIOUS parameters has been determined, clusters can be detected in the test set. This is accomplished by selecting the “use test set” feature. Both the training set and the test set are needed to use the test set.

For testing purposes, using the provided sample data, using the features=[], the parameters: gamma=0.2, nu=x, minN=x,

# Step 7: Save output cluster files.

After finding clusters, the outputs of these can be saved.

****

Saving results will generate “proximal” clusters and “unclustered” directory outputs. In addition, if the “find focal clusters” option was selected, “focal” and “combined” clusters will also be outputted. Combined clusters represent the sum of both proximal and focal clusters. Whereas proximal clusters likely contain focal clusters as well, “proximal only” clusters can be evaluated by subtracting combined clusters from focal clusters. For each individual image, the xy coordinates of each cluster box are saved. Files in the unclustered directory are blank, but serve as an index to indicate which images had no clustering.

The “make\_cluster\_ROIs.ijm” macro can be used to generate imageJ region of interests (ROIs) from the generated xy cluster files.

To run this macro, a few paths must be clarified:

**Imagedir:** The directory of the images for the rois to be generated for. The names must match what is selected for the clusterdir images.

**Clusterdir:** The path to the directory above where “proximal”, “unclustered”, etc., are located. This directory should be a time stamp.

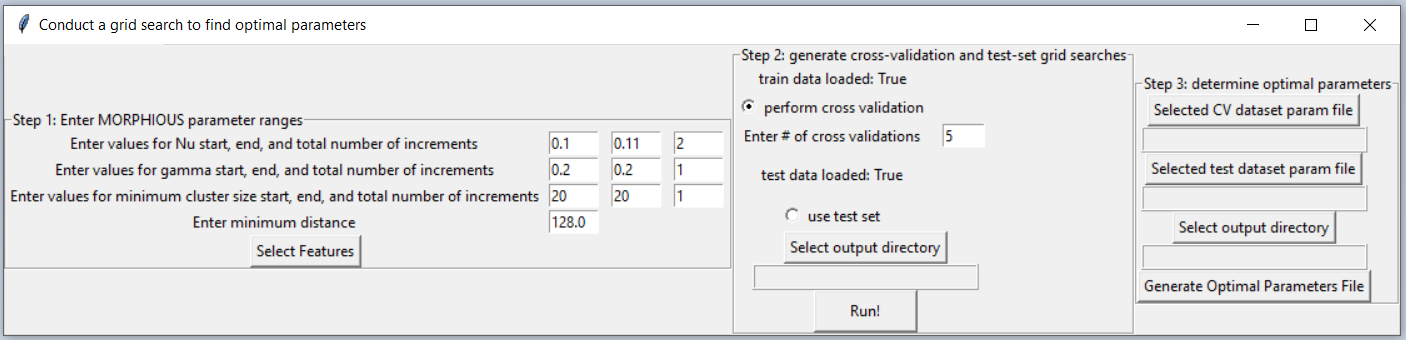
**Outputpath:** The path to where cluster ROIs are to be outputted.

**Clusters:** An array of all the cluster types which for which cluster ROIs are to be generated. These cluster types should correspond exactly to the subdirectories present in the “clusterdir” directory.

# Step 8: Using a grid search to find optimal MORPHIOUS parameters

In most cases a grid search will needed to be conducted to find optimal MORPHIOUS parameters. Briefly, a grid search iterates through a range of possible parameters to identify optimal parameter values.

To conduct a grid search, click the “Grid Search Parameters” button on the options frame.



For step 1, input values for the start, end, and total number of increments for then nu, gamma, and minimum cluster size parameters. The increment defines the number of equally spaced values between the start and end values. Both start and end values are inclusive. Therefore, a nu range of start: 0.1, end: 0.11, number of increments: 2, will yield 0.1, and 0.11 as values. A nu range of start: 0.1, end: 0.2, increments: 6, will yield 0.1, 0.12, 0.14, 0.16, 0.18, and 0.2, as values. For discussion on the minimum distance, refer to the “step 4: selecting MORPHIOUS parameters” section.

After features are selected, the user should proceed to step 2. Grid searches will be needed to be conducted separately for both the train dataset (via cross-validation), and the test dataset. Depending on the size of the dataset, a grid search can take a substantial amount of time. It is recommended to test the timing of a single “Find Clusters” workflow to provide a guide the breadth of grid search selected. Cross validation grid search result files are distinguished by “…X-fold\_CV…” in the resultant file name.

Once both the training grid search and test grid search files are generated, proceed to step 3. In step 3, select the relevant files, select an output directory for the resultant summary file. Afterwards push the “generate optimal parameter file” button. The summary file will indicate all parameters which resulted in no clustering in the training dataset. Parameters will further be sorted in order of which parameters yielded the greatest degree of clustering in the test dataset. Thus, from this list, optimal parameters for nu, minN, and gamma, can be selected.

# Step 9: Generating MORPHIOUS generated ROIs

After you have finished with generating the cluster coordinates, you can use the provided imageJ macro: “make\_cluster\_ROIs.ijm” to create the MORPHIOUS generated clusters.

First, let’s start with opening the “make\_cluster\_roi.ijm” macro.  
A screenshot of a computer

Description automatically generated

This macro has 5 parameters, similar to the feature extraction macros.

**imagedir:** The location of your images, for example, “D:/My\_Test\_Images\_Location/”.

**clusterdir:** The location of the cluster coordinate files saved in step 7. Select the folder that looks like a timestamp. This folder will contain separate subdirectories pertaining to each cluster (e.g., focal, proximal, combined).

**outputpath:** The location of an output directory.

**clusters**: This array corresponds to the directories present within the selected “clusterdir”, the most likely values will be “focal”, “proximal”, “combined”, OR, simply “proximal”.

**batchmode:** if set to true, the macro will run faster and in headless mode, hiding the various image popups etc.