Exercises: Machine Learning for Bio-Image-Analysis

Volker Baecker, Cedric Hassen Khodja, Jean-Bernard Fiche, Francesco Pedaci
Montpellier RIO Imaging

6.6.2019

Contents

List of Figures

Chapter 1

Introduction

In a scratch assay a gap in a tissue is produced with the tip of a pipette. Images of the tissue are taken in regular time intervals. The relative area of the gap for each time–point is then measured to analyze the speed with which the gap is closing.

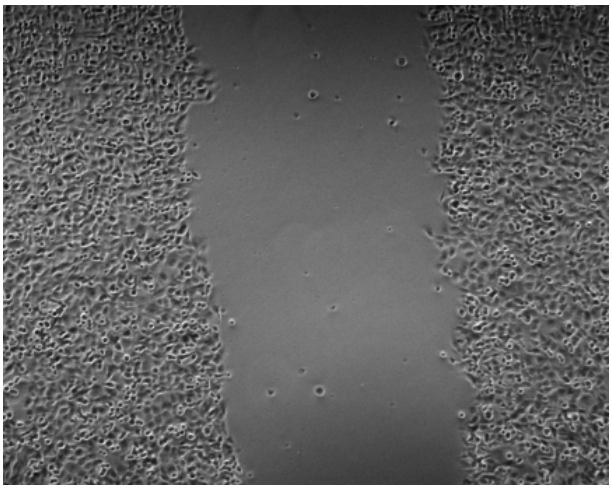

Figure 1.1: An image from sratch assay analysis.

In this exercise you will try to segment the gap first using image analysis with FIJI/ImageJ and the using machine learning with Ilastik.

1.1 Image Analysis, Features and Scalespace

Run `FIJI` and open one of the images from the folder `ex01-scratch-assay`.

Try to segment the gap and measure its area!

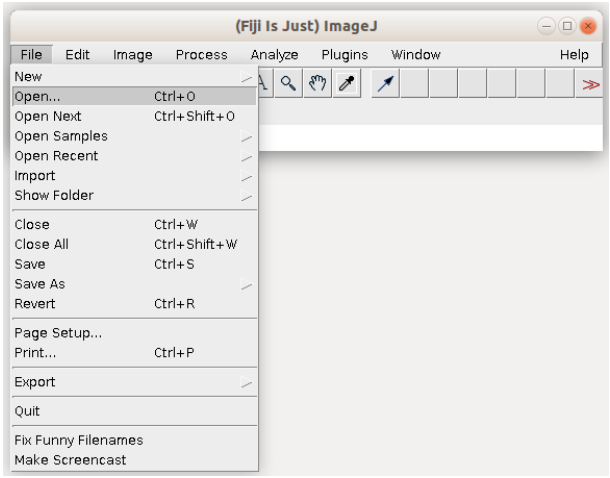

Figure 1.2: Opening an image in FIJI.

1.1.1 ImageJ commands you might need

Image >Adjust >Threshold...
Create a binary mask from the image.

Process >Filters >Variance...
Create a new image in which each pixel is replaced with with the variance in a local neighborhood.

Process >Find Edges (Sobel filter)
An edge detector using a combination of convolution filters.

Process >Filters >Gaussian Blur...
A convolution filter using a Gaussian distribution as kernel. Using a small width of the Gaussian, high frequencies are removed from the image, bigger widths remove lower frequencies. Applying a Gaussian filter with a given width, corresponds to choosing a point in the scale space.

Analyze >Measure
Measure the current image or selection.

The plugin `FeatureJ` provides further features. The selection of a scale, i.e. the application of a Gaussian Blur filter is integrated for the commands of this plugin. The commands of this plugin can be run from a panel that can be opened from `Plugins >FeatureJ >FeatureJ Panel`.

Derivatives
Create an image containing the change (or the change of the change, ...) of the intensities in a given direction.

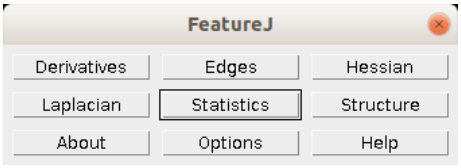

Figure 1.3: The FeatureJ command-panel.

Edges
A canny edge detector.

Hessian
Creates an image based on the second order partial derivatives of the original image. It can be used to distinguish between plate-like, line-like and blob-like structures.

Laplacian
Creates an image based on the sum of the second order partial derivatives of the image. It is often used for edge and for blob detection. The Laplacian can be implemented as a convolution filter.

Statistics
This command does not create a feature but calculates statistics on an image or a selection. The values calculated can be used to classify objects that have been segmented before.

Structure
Creates an image based on the structure tensor which represents the distribution of the gradient directions. It can be used to segment images based on texture.

1.2 Machine Learning

You will now use machine learning to do the segmentation of the gap-area in the scratch assay analysis.

a) Start Ilastik and create a new pixel classification project.

b) Add a new (separate) image from the folder `ex01-scratch-assay` to the project.

c) Adjust the zoom, so that the whole image is visible.

d) Advance to the `Feature Selection` step, open the `Feature Selection Dialog` and select all features on all scales. Also add a new scale of your choice, bigger than the ones initially proposed. We will refine the feature and scale selection later.

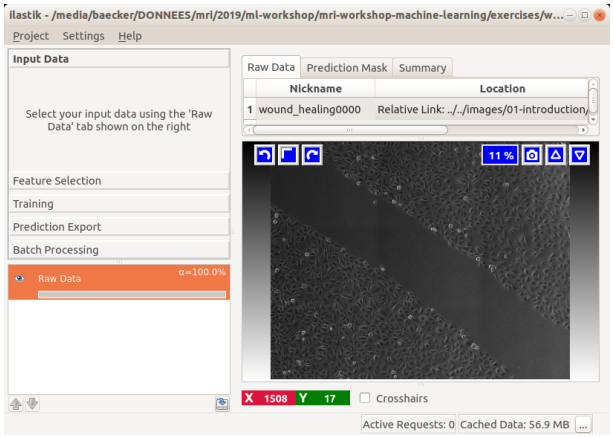

Figure 1.4: Image import into Ilatik.

e) Close the `Feature Selection Dialog` and click on different features at different scales in the features–list. Which features at which scales do you think are the most useful for the segmentation of the gap? Note the 3 most useful features with their scales:

`Feature 1:`

`Scale of feature 1:`

`Feature 2:`

`Scale of feature 2:`

`Feature 3:`

`Scale of feature 3:`

f) Advance to the `Training` step. Rename the classes for the pixel-classifiction from `Label 1` to `tissue` and from `Label 2` to `gap`. Change the colors of the classes from yellow and blue to red and green.

g) Select the `tissue` class and draw 3 or 4 strokes on the tissue using the brush–tool. Then select the `gap` class and draw 3 or 4 strokes in the gap. Press the `Live Update` button to see the result of the segmentation.

h) If necessary add more strokes for the two classes with the brush–tool or

remove strokes using the eraser–tool. The segmentation will be updated immediately as long as the `Live Update` is active.

i) Deactivate the `Live Update` and advance to the `Prediction Export`. Export the segmentation into a `.tiff`–file. You need to select the right `Source` and the right image file-format from the `Choose Export Image Settings` dialog. Then press the `Export All`–button.

j) Open the exported segmentation–image with `Fiji`. The values in the image correspond to the indexes of the classes. In this case they are 1 and 2. Use the threshold-adjuster to create a mask. Run the `fill-holes` command. Use the particle–analyzer to add a selection of the gap to the `roi-manager` and measure the surface of the gap.

`Name of the image:`

`area of the gap [pixel]:`

k) Go back to `Ilastik` and advance to the `Batch Processing` step. Select some of the images not yet processed, from the folder `ex01-scratch-assay`. Run the segmentation on the input files and control the results with `Fiji`.

l) Open a segmentation–result, create a `ROI` from the selection and display it on the original input image. How good is the segmentation? Does it correctly reflect the borders of the gap/tissue?

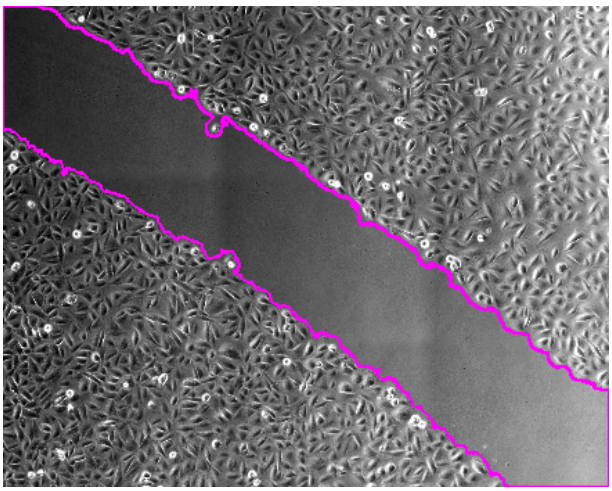

Figure 1.5: The ROI of the segmentation displayed on the input image.

Chapter 2

Clustering

In the exercises in this chapter you will first use a visualization of the k-means clustering that helps to understand how it works. You will then apply the k-means clustering as a machine learning method to segment plants based on their color.

2.1 K-means clustering

Open the file `Visualizing K-Means Clustering.html` from the folder `01-k-means-visualization`. The applet allows you to see the k-means clustering at work. You can select different strategies for the initial mean-values and different data–sets. You can follow the k-means algorithm step by step.

Run the k-means algorithm with different initialization strategies and on different data–sets and answer the following questions.

a) What influence do you think the selection of the initial mean values has on the clustering result ?

b) Does the k-means algorithm always converge, i.e. does it always come to a point where further iterations do not change the result anymore?

c) Is the result of the clustering always the same, no matter what the choice of the initial means is?

d) Does the k-means clustering work as expected on the `Smiley Face` data–set? What is the reason it does or does not work as expected?

2.2 Using K-means clustering for color segmentation

Images of Arabidopsis thaliana are taken in regular intervals. The aim is to measure the speed with which the area of the rosette of leaves augments under different conditions and treatments. You will use k-means clustering to segment the rosettes of the plants. The tool use the `Color clustering` tool from `FIJI` which comes with the `Weka` machine learning software.

Figure 2.1: An arabidopsis plant at 3 different stages.

a) Run `FIJI` and open one of the images from the folder `03 - plants`. Open the `Color clustering` tool from the menu `Plugins>Segmentation>Color Clustering`.

b) How many clusters do you think are appropriate?

`Number of clusters:`

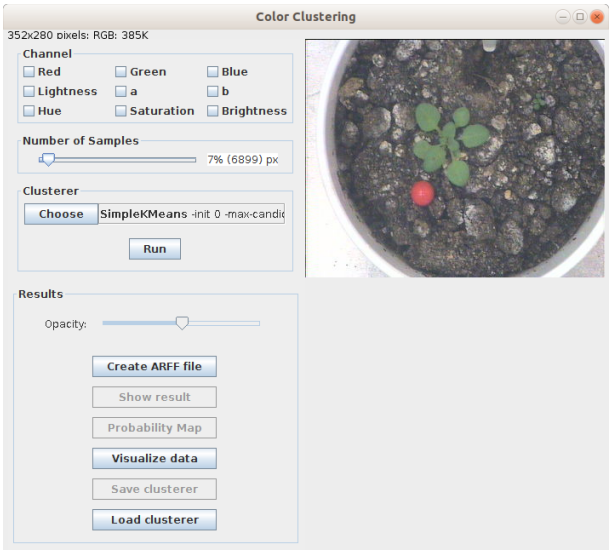

Figure 2.2: The color clustering tool.

Configure the k-means algorithm to use the number of clusters you have decided to use. Click on the field `SimpleKMeans` to open the options-dialog of the algorithm. Set the number of clusters in the field `numClusters`. You can also change the initialization strategy.

c) Reduce the `Number of Samples`, select one or more features and run the k-means clustering. Note that you can mix features from different color spaces. Experiment with different settings until you obtain a reasonable segmentation of the plant.

d) Display the result image and the probability map. The value of a pixel in the result image corresponds to the cluster to which the color of the pixel belongs. The `Probability Map` shows a stack of images, one slice for each cluster.

What is the class of the color values belonging to the plant?

`Class of plant:`

What is the class of the color values belonging to the small red reference object in the form of a ball?

`Class of reference object:`

e) Create a selection (roi) from the result of the pixel classification and display it on the original input image. How good is the selection? Measure the area of the surface of the plant.

`Area of the plant:`

f) Check the log–window. The clustering algorithm has written information about the clustering into the log–window. Redo the clustering using the features `a` and `b` of the CIELab color space. Find out the number of the cluster corresponding to the plant and find the following information in the log:

```
Starting point of the cluster:
Final cluster mean a:
Final cluster mean b:
Size of the training data-set:
```

g) Visualize the data! The first three fields allow to select the data displayed on the x– and y–axis and coded a color. Display the a–feature on the X and the b–feature on the Y axis. Let the color represent the cluster. You can change the display-colors of the clusters by clicking on the name of the cluster in the lower part of the window. Set the color of the cluster corresponding to the plant to green. If another cluster was already displayed in green set it to a different color. How well separated are the green points from the other points?

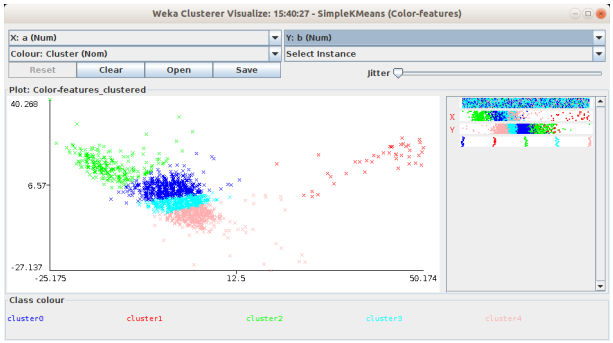

Figure 2.3: Visualization of the clusters.

h) Run the clustering again and use only the a–feature this time. Control the resulting segmentation and check the points in the data–visualization. Is the result better or worse than before?