Detection of BAX Translocation by means of supervised learning: User manual

Luis de la Ossa

April 9, 2014

This set of Matlab scripts implement the method described in "Automatic quantification of the subcellular localization of chimeric GFP protein supported by a two-level Naive Bayes classifier", by Sara Sáez-Atienzar et. al. This software aims at detecting and counting BAX translocation in microscopic images by means of supervised learning.

From the point of view of the user, there are 2 scripts which implement the functionalities described: `trainAndClassify` and `loadModelAndClassify`. Furthermore, there is another script aimed at testing the performance of the proposed technique, namely `testModel`. Next, all of them are explained in detail.

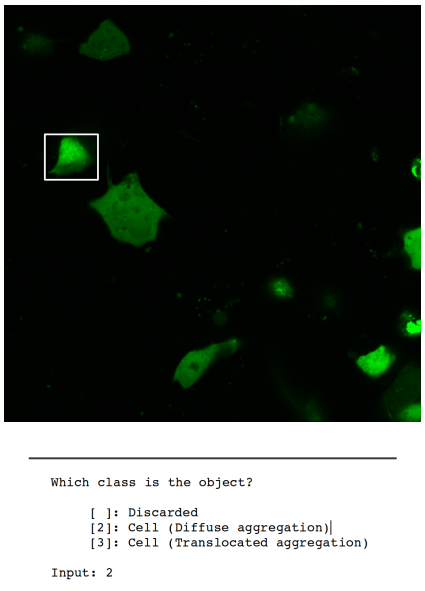
`trainAndClassify.m`

This is the script which implements the main program. Basically, it takes a set of images and uses them to train the model with the intervention of the user. Afterwards, it processes the remaining images, classifying and quantifying the cells.

By default, the images used to train the model must be located in a folder named `training`. The more images contain this folder, the better estimation of the models. When the script is run, it asks for the class of every object detected, as shown in Figure 1.

Once all the training images have been processed, the information is used to estimate the two Naive-Bayes classifiers, which are then stored in a folder named `models`. It is important pointing out that the name of the files must be changed so that they are not replaced in subsequent usages.

After learning the model, the software automatically processes the remaining images. All of them must be stored in the folder named `new`. Besides the statistics, the sofware shows, for each image, the result (Figure 2). When the number of new images is huge, this can be avoided by commenting or deleting the last line of the script.

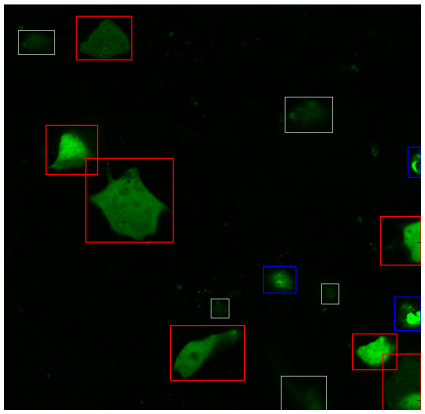


```
Which class is the object?

     [ ]: Discarded
     [2]: Cell (Diffuse aggregation)
     [3]: Cell (Translocated aggregation)

Input: 2
```

Figure 1: Screenshots of the software corresponding to the training phase. The object being processed is surrounded by a white rectangle, and the expert must provide its class.



```
Image1.jpg: (15 objects detected)

Transfected Cells: 10

     Diffuse aggregation: 7
     Translocated aggregation: 3
```

Figure 2: Screenshots of the software corresponding to the results: Objects enclosed by a white rectangle are discarded; Cells surrounded by a red rectangle show a diffuse aggregation of protein; cells surrounded by a blue rectangle show a translocated aggregation of protein.

`loadModelAndClassify.m`

This script also allows processing new images. However, instead of training the classifiers, they are loaded from two files:

- `models/objClassifier.mat`, which contains the Naive Bayes model used to classify the objects between discarded and transfected cells.
- `models/cellClassifier.mat`, which contains the Naive Bayes model used to distinguish the types of transfected cells.

The output of this process is the same shown in Figure 2. Again, by default it shows the resulting images. However, this can be avoided by deleting the last line of the script.

`testModel.m`

This script allows evaluating the models. First of all, it takes all images in folder `training`, and uses them to train the model with the intervention of the user, which must label each object, as described in Figure 1. Once all training images have been processed, models are estimated from the training dataset.

In the second step, the user must label the images contained in folder `test` in the same way.

Last, the model makes the predictions for the images in folder `test`, and compares them with those introduced by the user. The software shows two confusion matrices (Figure 3), one for each model. Moreover, it shows results on accuracy, precision and recall of each model.

```
***   Discarded objects vs Transfected Cells


Discarded   Transfected   <-- Classified as

    9        0               Discarded: 9
    1        21               Transfected: 22

   ------------------------------------------------------------

                          Total: 31
Accuracy: 96.77 %
Precision: 1.00
Recall: 0.95



***   Diffuse vs Translocated Cells


Discarded   Diffuse  Translocated   <-- Classified as

    0        0            0        Discarded: 0
    0        17            0        Diffuse: 17
    0        0            4        Translocated: 4

   ---------------------------------------------------------------------------

                                  Total: 21
Accuracy: 100.00 %
Precision (Translocated): 1.00
Recall (Translocated): 1.00
```

Figure 3: Results displaied by the script `testModel.m`