Centre for Image Analysis
Swedish University of Agricultural Sciences
Uppsala University

Computer Assisted Image Analysis

Exercise 4: Classification

The aim of this exercise is to segment real images into regions using classification. Different methods of classifying images will be tried.

Formality

- You need an account for the computer systems at the Dept. of Information Technology and for Studentportalen. We **strongly** recommend that you work together in groups of two or three people. This way you can get help quicker, and have someone to discuss the lab with.
- The exercise can be corrected orally at the corresponding lab session or the one directly following it, without any written report.
- If you don't finish all the questions or don't attend the lab session, a written report with all the remaining answers has to be handed in as a pdf. The report should be sent by email to one of axel.andersson@it.uu.se, elisabeth.wetzer@it.uu.se, or nadezhda.koriakina@it.uu.se.. Entitle your report **Lab4_LastName1_LastName2**.
- **Deadline:** December 13, 2018.
- Status of your reports will be found in Studentportalen.

1 Getting started

Before you begin, study the text book (GW 12.2.2) and the lecture notes on Maximum Likelihood (ML) classifiers. Do not forget that MATLAB has a Help function. Also, a function or script can be viewed in MATLAB using the command type as

```
type functionname
```

Start MATLAB in a terminal window by writing:

```
matlab &
```

Remember to start MATLAB from a directory where you have write permission.

2 Classification basics

First we examine a synthetic data set. Load the data into MATLAB by writing

```
load cdata
```

The data is of size N×2 where N is the number of samples and we have two data points for each sample, in this case x and y coordinates. You can inspect the data in a 2D scatterplot using the command

```
figure(1);plot(cdata(:,1),cdata(:,2),'.')
```

As you can see it is probable that the data contains two classes.

1. ***Are the two classes possible to separate by using either the x or y coordinate as a single feature? Explain why or why not.***

Second, open the image handBW.pnm and examine the image and the histogram.

```
I = imread('handBW.pnm');   % Read the image
figure(2);imshow(I);        % Show the image
figure(3);imhist(I);        % Show the histogram
```

Your task is to segment the image into three classes hand, object and background as accurately as possible. First try multiple thresholding using the following commands:

```
figure(4);mtresh(I,t1,t2);
```

where t1 and t2 are thresholds that you find suitable from inspection of the histogram.

2. ***Does multiple thresholding give a successful classification of the three classes? Explain why or why not.***

Open the image hand.pnm and examine a scatterplot of the three RGB bands.

```
I2 = imread('hand.pnm');    % Read the image
figure(5);imshow(I2);       % Show the image
R = I2(:,:,1);              % Separate the three layers, RGB
G = I2(:,:,2);
B = I2(:,:,3);
figure(6);plot3(R(:),G(:),B(:),'.') % 3D scatterplot of the RGB data
```

Also examine the three bands R, G and B and their histograms respectively in separate images. To improve the results from multiple thresholding of the grayscale image, single bands, pairs of bands, the grayscale image or all three bands could be used for classification. Training areas for the three classes are available in the image hand_training.png.

```
label_im = imread('hand_training.png');  % Read image with labels
figure(7);imagesc(label_im);             % View the training areas
```

This is an example of classification using two of the bands. Follow the steps below to create training data and do a classification. As the first step we choose the G and B bands as features. Then we split the training areas into training feature vectors and view the vectors in a 2D scatterplot.

```
I3(:,:,1) = G;                                    % Create an image with two bands/features
I3(:,:,2) = B;
[data,class] = create_training_data(I3,label_im); % Arrange the training data into vectors
figure(8);scatterplot2D(data,class);              % View the training feature vectors
```

In the next step we use the feature vectors to train and classify data using the MATLAB function classify and the following steps. First we reshape the test data to vectors. After that we perform classification of the test data using the given training data. The final step is to reshape the classification result to an image and view it.

```
Itest = im2testdata(I3);                                  % Reshape the image before classification
C = classify(double(Itest),double(data),double(class)); % Train classifier and classify the data
ImC = class2im(C,size(I3,1),size(I3,2));                  % Reshape the classification to an image
figure(9);imagesc(ImC);                                   % View the classification result
```

See the help function for the function classify.

3. ***What sort of classification is used? You can use a number of different discriminant functions. The default is 'linear'. What assumptions are made by the classifier in this case?***

Now, test classification on

- The grayscale image
- Single bands from the RGB image

- Pairs of bands from the RGB image
- All three bands in the RGB image

When using all three bands the function scatterplot3D can be used to view the feature vectors in the training data. A histogram can be used to view one dimensional features.

4. ***Have the results improved using classification compared to thresholding? Is the classification more successful in the case with the grayscale image or single bands? Explain. Does it improve the classification to incorporate pairs of bands or the full RGB information? Discuss. Show your results from grayscale classification, one pair of features and full RGB classification.***

3 Classification of multispectral data

Now when you have some tools to use you should try to make a good classification of multispectral image data. A satellite image showing part of Uppsala with seven bands is available. Your task is to segment the image into meaningful classes like forest, water, urban area and agricultural areas. Possibly more classes can be used to obtain a better classification. Use the classification tools described above, like classify, scatterplots of different bands etc. Test the different discriminant functions available in the function classify to get the best possible classification. You can mark training areas using coordinates for rectangles as in the following example.

```
T = zeros(512,512);       % Create an empty image
T(20:30,60:90) = 1;       % Class 1
T(140:150,100:160) = 2;   % Class 2
etc ...
```

It is important that the training areas for a class does not contain data from other classes to obtain a good classification. The image data can be loaded into MATLAB using the command

```
load landsat_data
```

As a help it is also possible to view three bands at the time using color or pseudocolor with the command imshow. Although, the bands 3, 2 and 1 correspond well to the visible spectral bands R, G and B respectively, you can use other combinations of bands in order to visualize differences and improve classification results.

```
figure(10);imshow(landsat_data(:,:,[4,1,3])./255);
```

5. ***Which bands are you using for your classification? What classifier are you using? Which classes are possible to separate in a good way? Illustrate with images and scatterplots. Show images from your classification result and compare with classification using all bands. Discuss your results.***

6. ***Is there any reason not to include all bands if they are not needed to separate the classes?***

4 Classification based on texture

Texture can also be incorporated to improve classification results in grayscale images. Run the code **virus_analysis.m.** This code uses match-based segmentation method (lecture 6) to segment viruses, viruses are ball shape objects in the Figure 1, left. A sample of viruses which is also used as the template is shown in Figure 1, right. The segmentation method finds all objects and save them in the cutouts variable, for this exercise you only need to work with this variable. You can access the contents of cutouts by indexing with curly brace {}, for example you can observe the first object saved in cutouts using the following script:

```
figure,imshow(cutouts{1})      %show the first object stored in cutouts vaiable.
```

The code plot all the objects in cutouts in a single image, (see Figure 2). Please check this figure, you can see that a few of the objects (for example the last one) look different from the template and segmentation algorithm segmented them wrongly. We consider these wrong objects as outliers. Your task is to use co-occurrence matrix descriptors; uniformity and entropy and find the outliers. To calculate the co-occurrence matrix you can use Matlab built in function **graycomatrix**. Please note that you need to choose a proper "offset" value for graycomatrix function so that the entropy is well defined.

7. ***Which "offset" value did you use for the co-occurrence matrix? Explain why. Plot uniformity and entropy for all the objects. You should have a plot similar to Figure 3. Using entropy and uniformity, say which objects are outliers and should be removed from the sequence. Why?***

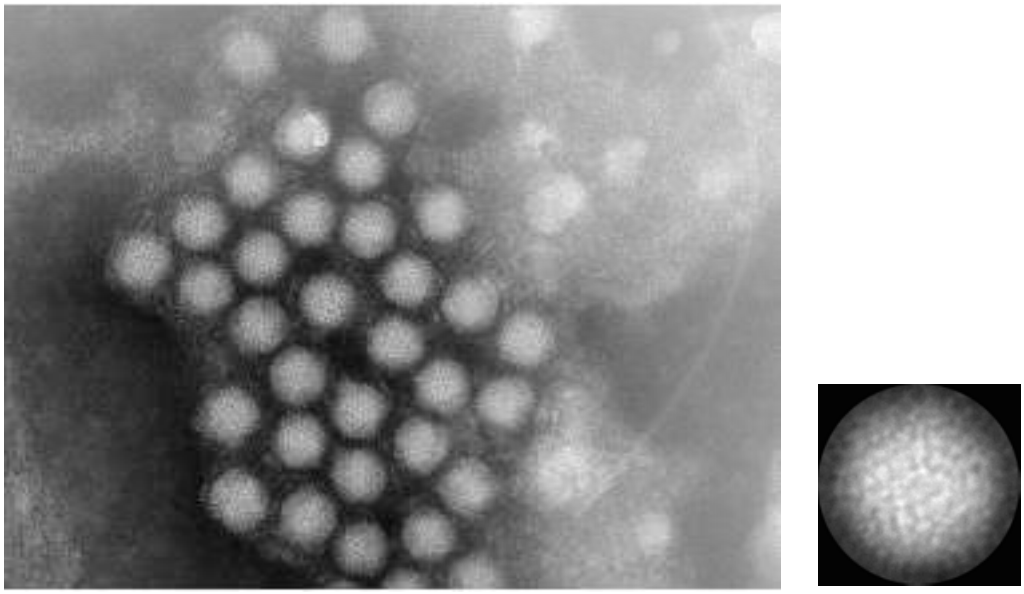

Figure 1: Left: Original image; Right: Desired object.

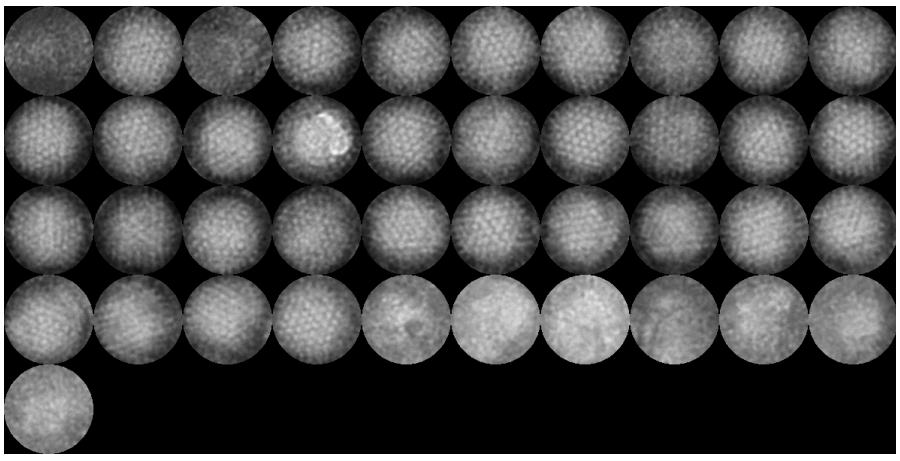

Figure 2: The sequence of the segmented objects (saved in **cutouts**)

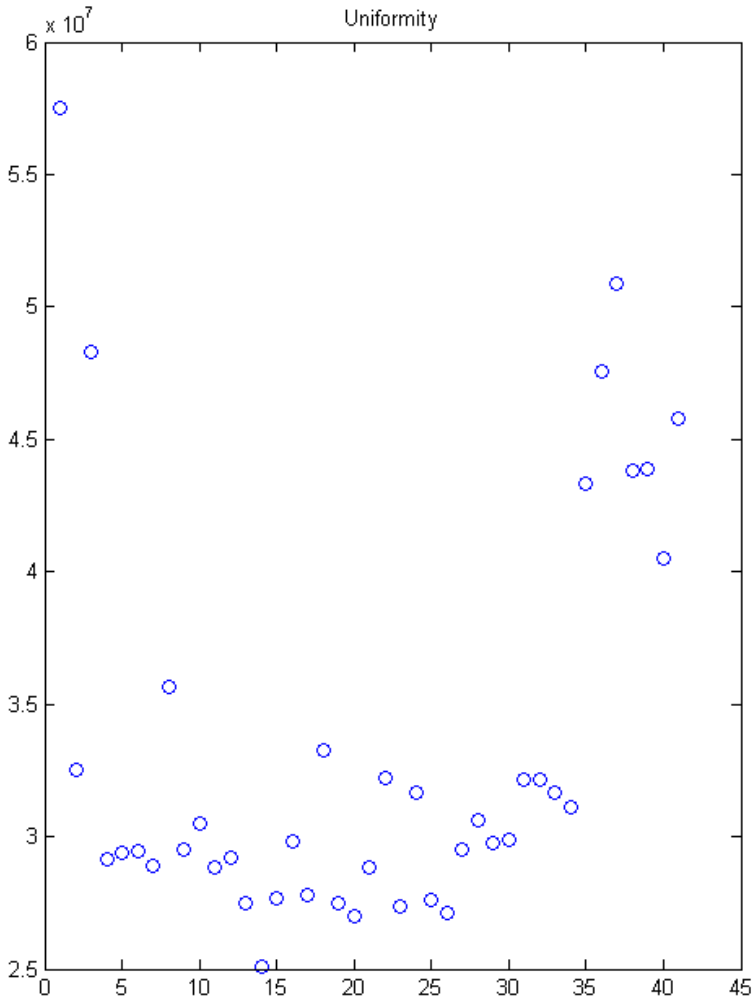

Figure 3: Uniformity plot

5 Your comments

That's it. The exercise is done!

8. ***Any comments on the exercise? Are the instructions good enough? Is the preparation on the lectures enough? Can we improve the exercise in any way? All comments are welcome.***