

2nd Mandatory term project 2022: Feature evaluation and classification

Andreas Kleppe

Adín Ramírez Rivera

Summary

In this mandatory exercise you are going to implement a multivariate Gaussian classifier and use it to classify images with 4 different texture classes. You must implement your own classifier, but you are allowed to use library functions to invert matrices, compute the determinant, and compute the mean vector and the covariance matrix for each class. It is recommended that you use an environment where you have access to a numerically stable library function for matrix inversion (for example Matlab).

In case you use python, a Jupyter notebook version of this exercise is available. The exercise combines feature evaluation, classifier implementation, training a classifier, testing different feature combinations and evaluating classifier performance.

1 Settings

1.1 Languages

We recommend that you use MATLAB or Python.

1.2 Time table

- Exercise available: Thursday 06th of October 2022.
- Deadline: Friday 28th of October 2022 (3 weeks).

1.3 Submission

Your solution must be submitted as a single PDF file containing the problem description, discussion, and the supporting source code. The files should be compressed (.zip or .tar) in a folder named YOURUIOUSERNAME_MANDATORY2.zip/tar, and uploaded through the Devilry system (<http://devilry.ifi.uio.no>) before or on the deadline above.

Questions about submission can be directed to the group teacher Tom-Ruben through discourse (<https://in5520-discourse.uio.no>) and with personal inquiries to his email (t.r.t.kvalvaag@fys.uio.no).

1.4 Evaluation



- There will be two mandatory exercises in IN5520 this year.
- The two mandatory exercises will be evaluated separately.
- Please note that in order to take the exam, both mandatory exercises must be passed.

Since image processing is a field where solutions often are found by experimenting with different methods, we would like to emphasize the following point:

You should analyze the problem and the input images so that you can select suitable methods and parameters. You should not test all available methods/features, even if it is a huge amount of work. Analysis and discussion, of both input and output, are very important.

1.5 How to work

The exercise is an individual work, and each student should deliver a written report. Your report should be genuine, in particular we will check that each report provides its own discussion of all method and parameter choices. Include references if you use external sources.

 The report must contain a description of the problem, theory, chosen methods, results and algorithms used. You have to document all steps in the algorithms, and listings of your own code should be included as appendix. **The code for your classification algorithm should be listed in your report.**

1.6 Data

You have available a separate training and test data sets of three original images and a set of precomputed GLCM matrices computed from the training image.

The images can be found under <https://www.uio.no/studier/emner/matnat/ifi/IN5520/h22/materials/mandatory2/>. There is a zip-file with images in matlab-format (oblig2_matlab.zip), and an alternative zipfile for use in python (oblig2_python.zip). (The reason for this is that the png-format does not handle the scaling of the pre-computed GLCM images very well, thus you should read ascii-files/text-files in python. You can also use a function for reading matlab-files into python.)



Mandatory 2 is not linked directly to Mandatory 1, so you will work on new feature images. These will be input to multivariate classification, not thresholded as in Mandatory 1.

Training data set:

```
mosaic1_train.ext
training_mask.ext

texture1dx0dymin1.ext      # For each texture
texture1dx1dy0.ext         # Angle 90 degrees x, y-1
texture1dx1dymin1.ext      # Angle 0 x+1,y
texture1dxmin1dymin1.ext   # Angle 45 x+1,y-1
texture2dx0dymin1.ext      # Angle 135 x-1,y-1
texture2dx1dy0.ext
texture2dx1dymin1.ext
texture2dxmin1dymin1.ext
texture3dx0dymin1.ext
texture3dx1dy0.ext
texture3dx1dymin1.ext
texture3dxmin1dymin1.ext
texture4dx0dymin1.ext
texture4dx1dy0.ext
texture4dx1dymin1.ext
texture4dxmin1dymin1.ext
```

Test data set:

```
mosaic2_test.ext
mosaic3_test.ext
```

With corresponding mask files: mask_mosaic2_test.ext and mask_mosaic3_test.ext. Note that .ext corresponds to .mat for the Matlab files and .txt for the Python ones.

2 Methodology

Mandatory 2 can be divided into six steps, but must be seen together. The task has the following steps:

1. Choosing GLCM images to work with.
2. Discussing new features by subdividing the GLCM matrices
3. Selecting and implementing the best features from the GLCM matrices
4. Implementing a Gaussian classifier
5. Training the classifier on the chosen features
6. Classifying the test images. Compute the classification accuracy and confusion matrices and discussing the performance of the classification.

 Your report should contain one section for each of these 6 steps.

2.1 Choosing GLCM images to work with

`mosaic1_train.mat` contains 4 different textures (`texture1`, ..., `texture4`). A subimage of each texture was used to precompute GLCM matrices as you did in the first mandatory. GLCM matrices with distances $(\Delta x = 1, \Delta y = 0)$, $(\Delta x = 0, \Delta y = -1)$, $(\Delta x = 1, \Delta y = -1)$ and $(\Delta x = -1, \Delta y = -1)$ for each texture, and $G = 16$. These GLCM images are given above.

 Analyze the GLCM matrices given and select maximum two directions that you expect to be useful for discriminating the textures.

2.2 Discussing new features by subdividing the GLCM matrices

In this exercise you should **not** use any of the GLCM features from the lectures, but implement your own features that are based on only parts of the GLCM matrices, not the entire matrix. Divide the 16×16 GLCM matrix into four quadrants Q_1 , Q_2 , Q_3 and Q_4 of the same size. Create new features by summing the amount of energy/percentage of gray level transitions found in each quadrant, e.g.,

$$F_{Q_1} = \frac{\sum_{i=1}^8 \sum_{j=1}^8 P(i, j)}{\sum_{i=1}^G \sum_{j=1}^G P(i, j)} \quad (1)$$

$$F_{Q_2} = \frac{\sum_{i=1}^8 \sum_{j=9}^G P(i, j)}{\sum_{i=1}^G \sum_{j=1}^G P(i, j)} \quad (2)$$

Features F_{Q_3} and F_{Q_4} should be computed correspondingly.

 Based on just visual inspection of the selected GLCM matrices, discuss if you think all four textures can be separated with these features. If you do not think the textures can be separated using a subdivision into 4 quadrants, you can subdivide ONE of the quadrants into 4 smaller quadrants of equal size. If you choose to do this, discuss which quadrant you should subdivide.

 How many quadrants do you need to discriminate between the four textures?

2.3 Selecting and implementing a subset of these features

Implement the features chosen in step 2 using sliding windows of size 31×31 , $G = 16$, and the direction(s) you chose.  Consider all your features but discuss if you will need all of them in the classification.  Select some of the features and include the corresponding feature images computed from `mosaic1_train.mat` in your report.

2.4 Implement a multivariate Gaussian classifier

The classifier can and should use library functions for matrix inversion and computing the determinant. If you want, you can use library functions for estimating the mean vector and the covariance matrix. However, you must implement the computation of the posterior probability using Bayes rule yourself.  Include the classifier code in this section of your report.

2.5 Training the classifier based on the feature subset from point 3

Compute the overall classification accuracy and the full confusion matrix based on the training data.

 Discuss the performance, what does the confusion matrix show? (Hint: if you run into a singular covariance matrix using a certain combination of features, there might be a reason for that. Try to understand why. If you don't run into singular matrices your feature selection has avoided this little "trap.")

2.6 Evaluation of classification performance on the test data set using the set of features selected in point 3

Now compute the overall classification accuracy and the confusion matrix on the two test images. These images are slightly different from the training image.

Compute the accuracy both on `mosaic2_test.ext` and `mosaic3_test.ext` using the corresponding test masks `mask_mosaic2_test.ext` and `mask_mosaic3_test.ext`. (Note that `.ext` corresponds to `.mat` for the Matlab files and `.txt` for the Python ones.)

 Compare the test performance, and discuss the differences in accuracy you see.

2.7 Voluntary extension

Train using a library function for another classifier (e.g., k NN or SVM). Train on `mosaic1_train`. Use `mosaic2_test` as validation dataset (e.g., for hyperparameter selection), and present your results on `mosaic3_test`.

3 General remarks

-  You may use libraries and pre-programmed functions, as long as you cite your source(s) and do not contradict the restrictions on this assignment.
-  You should not test a massive list of GLCM features/methods combinations.
-  You should give a good discussion based on your selection of features and your results.