Models of Higher Brain Functions Programming Tutorial

Exercise Sheet Nr. 1

Ingo Fründ & Tiziano Zito

May 3rd, May 10th, and May 17th 2010

1.1

Write a function that allows to import pictures from the collection located at:
`/home/mohbf/images`
The output of the function should be a numpy array containing the gray scale values of the imported picture.
Pictures are stored in TIFF format, where every pixel is represented by a little-endian 16bit signed integer gray scale value. You can use the module `Image` from the Python Imaging Library to import the TIFFs.
Visualize the resulting array using matplotlib. Remember to disable automatic interpolation with something like:
`matplotlib.rcParams['image.interpolation'] = 'nearest'`

1.2

Import an image and add noise to the image. How does the image change? Try the following types of noise:

- White noise (each pixel is independent from the others) with a gaussian distribution with standard deviations $100, 1000, 5000, 10000, 30000$.
- Filtered gaussian noise. Generate white noise and then replace each pixel with the average noise value in a square centered on that pixel. Take squares of 3×3, 5×5, 9×9, 15×15 and 29×29 pixels.

1.3

Write a function that extracts a random patch from a random image in the database. Be sure that it can cope properly with image boundaries. Write a function to normalize a patch such that it has zero mean and unit variance. Write another function to normalize a set of patches, such that the average patch is zero and the variance thereof is unitary.

1.4

- Generate 500 16×16 pixel image patches from the image data base and normalize the set.
- Generate 500 16×16 pixel white noise patches and normalize the set.
- Generate 500 16×16 pixel patches of filtered noise. Average over a 9×9 pixel square and normalize the set.

1.5

Write a function to calculate and to plot pairwise pixel correlations over pixel distance. Note that you can either:

- Calculate the correlation over all pixel pairs on a single image and plot it as a function of the euclidean distance: $C(d) = C(||\mathbf{x} - \mathbf{y}||)$, **or**
- Extract random patches from random images and calculate the correlations with respect to a reference pixel position. In this case you can plot the correlation as a matrix, i.e. the correlation as a function of the pixel position: $C(\mathbf{x}, \mathbf{y})$. Use at least 5000 41×41 (note the odd number) patches.

Visualize a 1-D projection of the correlation by averaging over north/south/east/west directions. Plot it on a linear and a log scale, and fit a straight line in the log scale plot. What does the straight line fit tell us?
Do the same for images from the database, for white noise and for filtered white noise. What are the differences?

1.6

Train principal component analysis (PCA) on natural image, white noise, and filtered white noise patches in vector representation. Use at least 5000 40×40 patches and analyze the first 25 basis vectors. How do the basis vectors look like?
To perform PCA you can use the package `mdp`.

1.7

Train independent component analysis (ICA) on natural image patches in vector representation. Use at least 5000 32×32 patches and reduce the dimensionality with PCA to 256 dimensions. How do the resulting basis vectors look like?
To perform ICA you can use the package `mdp` (use the `FastICA` implementation, with a **symmetric** optimization approach and a **tanh** non-linearity).