02458 Cognitive Modeling

PCA/ICA – Homework 3.1

Load an image (in Matlab using the imread.m function). It can be the image of Mona Lisa that you find in the exercise folder or it can be any image that you choose. It is important that the image has thousands of pixels and that it is grayscale. (You can transform it to gray scale either by using the rgb2gray.m function in Matlab's image processing toolbox or by simply extracting one of the three color channels in the image). We'll call this image matrix ***I***.

Now transform the image to a set of patches each 10 by 10 pixels. Transform the patches to 100-dimensional vectors (e.g. using reshape.m) and catenate them to form a matrix, ***S***, where each row is a patch and each column is a pixel.

Perform PCA on the transformed matrix using.

It can return the principal components (also called the *loadings* or *coefficients*) in a 100-by-100 matrix, ***X***. The principal components are the columns of ***X***. It can also return the transformed data, ***W*** (also known as *scores*), which has the same size as ***S***.

Now, $\boldsymbol{XW^T}$, is a reconstruction of ***S***, except for the scaling. Test this by transforming $\boldsymbol{XW^T}$ back into image-space and use imagesc.m to display the image. For comparison display the original image matrix, ***I***, in another figure. The two images should look identical. Also check that the difference between the original image and the reconstruction is negligible.

The princomp.m function can also return the amount of variance each principal component contains (also known as the *eigenvalues*). Have a look at the eigenvalues. How many eigenvalues does it take to explain 95% of the total variance?

The columns of ***X*** are the principal components ordered according to how much variance they contain. Take some (like 6) of the first ones and transform them back to image space. Scale the resulting images on the *same scale* and display them (in Matlab using the image.m, *not* imagesc.m function). What do they look like?

Now reconstruct ***S*** from $\boldsymbol{XW^T}$ using only the first six principal components. Transform ***S*** back to image-space and display the image using imagesc.m. What does it look like? How does it compare to the original image?