# Part II: Feature Extraction

## Motivation

A significant use case for MRI is the identification of neurodegenerative diseases and/or tumours within the brain. MRI scans produce enormous amounts of data which can be extremely difficult to analyse manually. As such, a combination of mathematical and machine learning techniques are often employed in conjunction to aid such analysis. We will demonstrate one of the common mathematical techniques used within this field. For practicality’s sake, black and white images of human faces will be used as a demonstrative proxy for MRI brain scans. Where while in the latter context, we would be interested in identifying tumours and/or biomarkers of disease, in the context of our proxy we will simply be attempting to detect moustaches. In a real implementation, machine learning would most likely be used in conjunction with the mathematical technique explained below, however since we will be mainly focusing on the mathematics, our detector will be highly rudimentary at best and will not rely on any machine learning techniques. It should be noted that although our chosen proxy is two dimensional while MRI data is three dimensional, extending the maths to three dimensions is extremely simple.

## Mathematical foundation

Say we have a set of images of human faces denoted as matrices . Our goal is to use this data to derive some method for identifying moustaches within the images. The matrices are first vectorized, i.e., their columns are stacked on top of one another to form vectors:

where . The resultant vectors are then stacked column-wise to form a matrix containing the data from all images:

We will identify the “average face” by taking the column-wise mean of :

This can be used to mean centre :

where is a column vector of ones. We will now consider the reduced singular value decomposition of :

where , and . This can be visualised as follows:

where , and and denote the columns of and respectively. The RHS can be rearranged to form a sum of rank one matrices:

Since by their construction we have that , we also have that . As such, we can see that the relative contribution of each matrix towards the reconstruction of is determined solely by the value of . Since , it can be deduced that said contribution monotonically decreases as increases. It turns out that in most instances this happens very quickly. As such, can often be very well approximated by

where or even . Returning to matrix format this can be expressed as

where , and . From this we can write an equation for the column of :

where is the row of or more intuitively the column of .

## Eigenfaces

The columns of are called eigenfaces. They represent the principal components of the data. More specifically they are the vectors that minimise total squared reconstruction error,

or equivalently,

for . To be very clear, when we use a rank approximation, is just the first columns of . If the system is underdetermined, that is, , then is simply the minimum-norm solution among all valid orthonormal bases. Eigenfaces can be visualised by de-vectorizing the columns of and viewing the resulting matrices as images. The eigenface of is given by:

When visualised, eigenfaces often take the form of ghostly human faces, each representing different ways in which the faces in the dataset deviate from the mean face. The span of a set of eigenfaces is called a -dimentional eigenface space.

## Projection onto subspaces

Recall that:

Let . This enables the expression above to be rewritten as:

Having done this one can clearly see that is simply the coordinate vector of with respect to the basis . Note that is simply a projection of onto the lower rank bearing basis , i.e.:

Now let’s say we have some vectorized image and we want to find its closest representation with respect to the basis . To do this we would first have to mean centre the image:

We would then find the projection of onto with respect to . Since is orthonormal, said projection is given by:

This can then be expressed in terms of the standard basis as follows:

Adding the mean returns a vectorized approximation of the original image:

It is fairly easy to see that is essentially just a vector of the amounts of each of the eigenfaces present in the mean centred image. As we will see, this particular result is what makes this method so useful.

## Moustache detector

With the mathematical background out of the way, we will now explain how this technique was used this to create a rudimentary moustache detector. For our detector we set . This decision was made based on the following graph:

A graph of a number of values

AI-generated content may be incorrect.

As can be seen, singular values fall away very rapidly. Singular value 47 was the first to drop below five figures. Singular value 47 was also visually the approximate point at which we deemed the computational cost to outweigh the increase in detector accuracy. It is important to note that this choice is entirely subjective. Upon visual inspection of the eigenfaces we were able to see that the 13th eigenface, , clearly corresponded to the feature of interest, a moustache. Recall that for some mean centred vectorized image we have that it’s projection onto in terms of given by is essentially just a vector of the amounts of each of the eigenfaces present in .As such, since corresponds to moustaches, we can simply observe the magnitude of the 13th element of , i.e., , to gauge the amount of variation from the mean face explainable by a moustache. With this logic in mind, we found and subsequently for . We then set about finding a suitable moustache detection threshold that best made true the statement

for .

This was achieved by numerically solving the following maximisation problem:

where the square brackets are the Iverson brackets. The results can be seen in the following graph:

A graph showing the difference between a moustache level and a level

AI-generated content may be incorrect.

As such we set . While the graph would seem to suggest that for the given choice of , the detector boasts an exceptional degree of accuracy, namely 100%, it is important to recognise that this accuracy reading is going to be high by construction as we are testing against the data on which the detector was trained and optimised for. As such, the detector is likely to exhibit a lower degree of accuracy when tested against unseen data. Unfortunately, we do not have a validation set upon which to perform such a test however the exceptional accuracy of the detector given the training data probably does suggests at least some degree of model utility.