# Practice 2 – Dimensionality reduction

## Pattern Recognition

## 1. Introduction

In this practice we will work with two dimensionality reduction techniques: PCA and LDA. Sometimes the descriptors of our data (feature vectors) are very large, for example, the images that we used in the previous practice had a size of 128x128 pixels, therefore, if we take as the descriptor the intensities of all the pixels, our descriptors will be of size $1 \times 2^{14}$. Working with descriptors of such a large dimension has many disadvantages and dimensionality reduction techniques allow us to reduce the size of these vectors using different criteria. Some advantages of reducing the dimensionality of our descriptors are the following:

- Being able to visualize the data.

- Reduction of computation time and memory when working with the data.

- Best results in classification or clustering algorithms (Reduction of noise).

- In general, we can summarize these advantages as "Reducing the negative effects of dimensionality".

In this practice, we will use the same databases as in the previous one to prove the effect of two techniques of dimensionality reduction: PCA and LDA. At the same time, we will see if using these two techniques we can improve the results in the classification task using template matching.

## 2. PCA and LDA (Fisher discriminant analysis)

Both PCA and LDA are very used dimensionality reduction techniques in the area of Pattern Recognition. Let's imagine we have a matrix X of size N × D containing N samples of D dimensions (in our case N is the number of faces in the database and D is the number of pixels in the image, $2^{14}$). The main idea of these two techniques is to find a projection matrix U with size D × M such that:

$$X' = XU \tag{1}$$

Where X’ is an array of size N × M where there are the N samples projected in a space of dimension M smaller than D. Therefore, the columns of U contain the foundations of this new space of reduced dimensions in which the data is projected.

PCA and LDA are methods that look for an U matrix following different criteria. PCA does not take into account the labels (in our case the type of expression) of the data, and simply look for an U matrix such that the projected data X' have maximum variance, i.e. when we project the data in the new space of reduced dimension, we lose as little information as possible.

On the other hand, LDA looks for an U matrix taking into account the labels corresponding to each sample, and looks for the projected data in the reduced dimension space to maximize the Fisher's criteria. This criteria attempts to minimize the distance of the samples of a class with respect to the centroid of the class and, at the same time, to maximize the distance between the centroids of the different classes.

In Figure 2 we can observe a bunch of points corresponding to two different classes in a 2-dimensional space. Imagine that for some reason we needed to reduce the dimensionality of our samples to 1. If we wanted to lose as little information as possible about the data, we would use PCA. On the other hand, if we wanted to find a basis in which the data satisfy the Fisher criteria as well as possible, we would use LDA. LDA is usually the used option when we want to do classification.

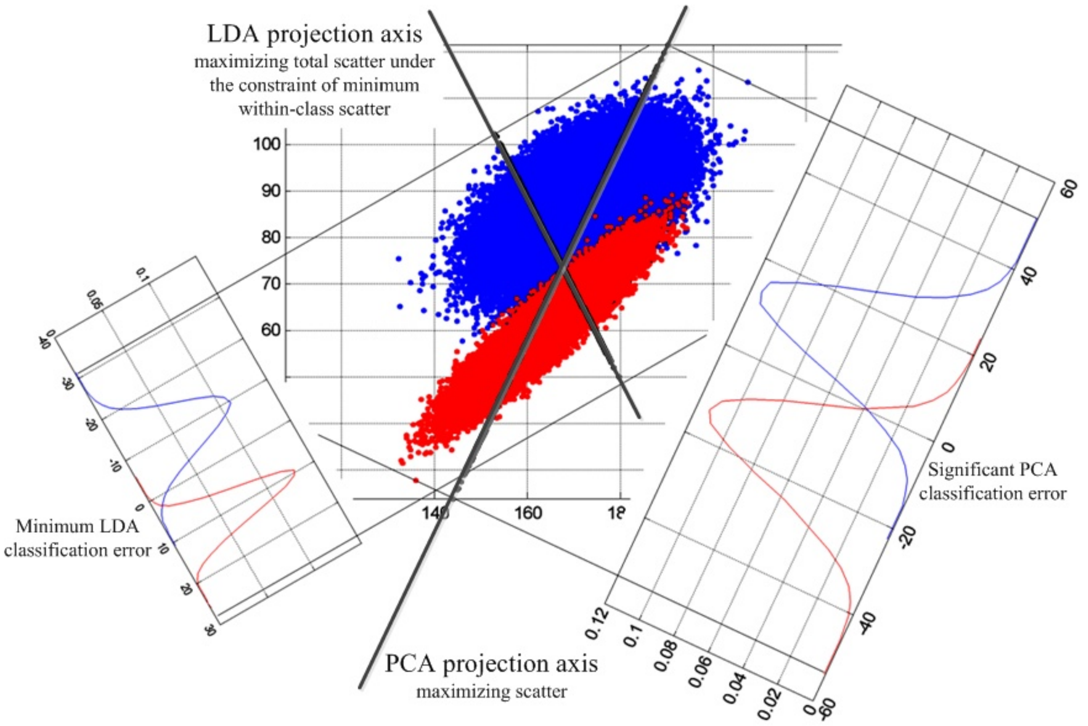

*Figure 1: Projeccions obtined with PCA and LDA.*

## 3. Database and code provided

In this practice we will work with the same database as in the previous one. The main script that you have to run and modify is the p2DR.m. Furthermore, we provide the following support functions:

- extractData.m: This function is equal to the one from the previous practice but with an extra returned parameter called stringLabels. This parameter is a vector of labels equal to the vector *labels* but with strings (the names of expressions) instead of numbers.

- extractFeaturesFromData.m: This function receives as parameters the image data that we obtain with extractData and creates an array of N × D where N is the number of images and D the dimension of our descriptor. The kind of descriptor will be given for the second parameter of the function.

- reduceDimensionality.m. This function will allow us to reduce the dimensionality of our data using PCA and LDA, and at the same time obtain the mean of our data and the projection matrix obtained with these two methods. The function receives 4 input parameters:

  - data: N × D matrix with the descriptors of dimension D of the N samples.

  - drMethod: string with the desired dimensionality reduction technique ('PCA' or 'LDA').

  - dimensions: total number of dimensions that we want for our projected samples.

  - labels: in the case of using 'LDA', vector of labels with the class of each sample of data from the data array.

- projectData.m: This function projects the data from the data array to the new space of reduced dimension by substracting the mean *meanProjaction* to all of the samples and using the projection matrix *projectionBasis*.

- reprojectData.m: This function projects the data already projected in the space of reduced dimension to the original space by using the inverse of the projection matrix *projectionBasis* and adding the mean defined in *meanProjection*.

- gscatter3.m: Function that allows to make a plot of data in 3 dimensions and paint the data belonging to different classes with different colors. The 3 first parameters of the function are the coordinates x, y and z of the data and the 4th is the vector of data labels. You have an example of how to use this function in the p2dr.m script.

With the code you will also see a folder called drtoolbox. This folder contains a toolbox for Matlab where different techniques of dimensionality reduction are implemented and that uses the function reduceDimensionality.m. You do not have to modify anything in the code of this folder but if someone wants to investigate about this toolbox there is a README and you can find more information at http://homepage.tudelft.nl/19j49/Matlab_Toolbox_for_Dimensionality_Reduction.html.

## 4. Work to do

The objective of this practice is to familiarize you with the concepts of dimensionality reduction using PCA and LDA. For this reason, we propose a series of tasks that you can implement in the p2DR.m script using the functions of the provided code. The results of these tasks, the images, and the analysis that you can do of the results have to be included in a report that you will have to deliver.

1. Extract the descriptor based on the intensities of the pixels of all samples of the database and project them in a space of 3 dimensions using PCA. Plot the data in the new 3D space.

2. Plot the mean obtained with PCA and the 3 bases of the projection matrix that you have obtained. The mean and each vector of the base will have dimension $1 \times 2^{14}$ but you can use the reshape function to make it have dimension 128 × 128 and plot it as an image using the imagesc or surfing. Interpret results.

3. Reduce the dimensionality of the data of the entire database with PCA. After that, reproject them again to the original space. Take an image of the database and plot it after reducing its dimensionality and reprojecting it in the original space. Observe how does this image have changed in respect to the original one. Repeat the experiment reducing the dimensionality to 2, 5, 10, 50, 100, 300 and 500 dimensions. Comment the results.

4. Do the same as in the previous exercise but now reproduce the whole database. After that, calculate the quadratic error between the reprojected data and the original one for all of the images. Add the error of all images and see how it evolves when we use PCA with 2, 5, 10, 50, 100, 300 and 500 dimensions. Make a graph of this evolution and comment on the results. To how many dimensions do you think that we could reduce the samples from this database without losing a lot of information? Why?

5. Reduce the size of your data using PCA to the number of dimensions that you have answered in the previous section. Now, use LDA to reduce the dimensions to 3. Plot the data in a 3D space and compare it with the visualization obtained in the first exercise. Comment the results.

6. Apply any of the methods you used in the previous practice for classifying expressions using template matching, but first reduce the dimensionality of your data with PCA and LDA. To do this you can modify the code of the function testTemplateMatchingWithDR.m writing the code to reduce the dimensionality of the data. You can add error measures as in the previous practice by modifying the code classifyWithTemplateMatching.m

BONUS: Make the classification of expressions using Support Vector Machines and compare the accuracy obtained with the one from the previous exercise.

## 5. Delivery and evaluation.

The evaluation of this practice will be based on the report that you have to deliver answering the questions from the previous section. You should include the relevant figures to make the explanations more clear and justify the answers properly.