# Práctica 4:
# Análisis de Componentes Principales.
# Reconocimiento facial

En esta práctica se va a utilizar el análisis de componentes principales (PCA) en un sistema de reconocimiento facial. La idea del PCA es la siguiente:

PCA es una metodología muy utilizada para identificar patrones en datos que generalmente están expresados en un número de dimensiones muy alto. PCA convierte los datos a una representación mucho más sencilla (menos dimensiones) que expresan las similitudes o diferencias entre los datos que queremos analizar. Esta reducción en la dimensión nos permite reducir los datos almacenados sin demasiada perdida de información. La reducción de la dimensión se realiza calculando una nueva base formada por los vectores propios de la matriz de correlación de los datos analizados.

Lo que hacemos en esta práctica es:

1.- Adquirimos los datos de las imágenes y los transformamos en vectores
2.- Calculamos y restamos la media de estos vectores
3.- Calculamos la matriz de covarianza, calculamos sus vectores propios, los ordenamos en función del valor propio y tomamos los mayores vectores propios asociados a los mayores valores propios.
4.- Calculamos las coordenadas a partir de la nueva base. Las nuevas coordenadas tienen que tener tantos elementos como vectores propios hayamos elegido.
5.- Calcular la coordenada media de cada persona.
6.- Clasificación de una nueva imagen con la que compararemos.

La clase a la que pertenece la nueva imagen viene dada por:

$$\arg \min_{Pi} \sum ||\omega 21 - \omega Pi||$$

El algoritmo esta adjuntado con la práctica, tiene muchos comentarios para facilitar su lectura. Además, tiene alguna parte de código comentada, porque ha sido usada para pruebas, o se usaba en un principio pero luego fue modificada para hacer pruebas...

Adjunto también diferentes carpetas con diferente número de caras, porque he necesitado hacerlo de esta manera. También adjunto una carpeta con imágenes reconstruidas, donde el nombre de la imagen es del estilo: prob2$4$200 , donde el primer dígito significa X vectores propios, el segundo X número de imágenes (X/4 por cara) y el último dígito corresponde a la imagen que se ha insertado.

# COMENTARIOS

• Porcentaje de personas bien clasificadas

123/240 = 51.25%  DE TODAS!

Este es un dato que carece de sentido, por lo que a continuación se reflejan unos datos más trabajados:

aciertos en %

NUMimagenesxcara 5, NUMVECTS  1 : aciertos-> 0
NUMimagenesxcara 4, NUMVECTS  1 : aciertos-> 50
NUMimagenesxcara 3, NUMVECTS  1 : aciertos-> 0
NUMimagenesxcara 2, NUMVECTS  1 : aciertos-> 0
NUMimagenesxcara 1, NUMVECTS  1 : aciertos-> 25
NUMimagenesxcara 5, NUMVECTS  2 : aciertos-> 0
NUMimagenesxcara 4, NUMVECTS  2 : aciertos-> 0
NUMimagenesxcara 3, NUMVECTS  2 : aciertos-> 0
NUMimagenesxcara 2, NUMVECTS  2 : aciertos-> 50
NUMimagenesxcara 1, NUMVECTS  2 : aciertos-> 50
NUMimagenesxcara 5, NUMVECTS  3 : aciertos-> 25
NUMimagenesxcara 4, NUMVECTS  3 : aciertos-> 0
NUMimagenesxcara 3, NUMVECTS  3 : aciertos-> 25
NUMimagenesxcara 2, NUMVECTS  3 : aciertos-> 100
NUMimagenesxcara 1, NUMVECTS  3 : aciertos-> 100
NUMimagenesxcara 5, NUMVECTS  4 : aciertos-> 25
NUMimagenesxcara 4, NUMVECTS  4 : aciertos-> 25
NUMimagenesxcara 3, NUMVECTS  4 : aciertos-> 25
NUMimagenesxcara 2, NUMVECTS  4 : aciertos-> 75
NUMimagenesxcara 1, NUMVECTS  4 : aciertos-> 75
NUMimagenesxcara 5, NUMVECTS  5 : aciertos-> 0
NUMimagenesxcara 4, NUMVECTS  5 : aciertos-> 25
NUMimagenesxcara 3, NUMVECTS  5 : aciertos-> 75
NUMimagenesxcara 2, NUMVECTS  5 : aciertos-> 75
NUMimagenesxcara 5, NUMVECTS  6 : aciertos-> 25
NUMimagenesxcara 4, NUMVECTS  6 : aciertos-> 50
NUMimagenesxcara 3, NUMVECTS  6 : aciertos-> 75
NUMimagenesxcara 2, NUMVECTS  6 : aciertos-> 50
NUMimagenesxcara 5, NUMVECTS  7 : aciertos-> 50
NUMimagenesxcara 4, NUMVECTS  7 : aciertos-> 25
NUMimagenesxcara 3, NUMVECTS  7 : aciertos-> 75
NUMimagenesxcara 2, NUMVECTS  7 : aciertos-> 50
NUMimagenesxcara 5, NUMVECTS  8 : aciertos-> 50
NUMimagenesxcara 4, NUMVECTS  8 : aciertos-> 0
NUMimagenesxcara 3, NUMVECTS  8 : aciertos-> 75
NUMimagenesxcara 2, NUMVECTS  8 : aciertos-> 100
NUMimagenesxcara 5, NUMVECTS  9 : aciertos-> 50
NUMimagenesxcara 4, NUMVECTS  9 : aciertos-> 25
NUMimagenesxcara 3, NUMVECTS  9 : aciertos-> 75
NUMimagenesxcara 5, NUMVECTS  10 : aciertos-> 50
NUMimagenesxcara 4, NUMVECTS  10 : aciertos-> 50

NUMimagenesxcara 3, NUMVECTS  10 : aciertos-> 75
NUMimagenesxcara 5, NUMVECTS  11 : aciertos-> 50
NUMimagenesxcara 4, NUMVECTS  11 : aciertos-> 50
NUMimagenesxcara 3, NUMVECTS  11 : aciertos-> 100
NUMimagenesxcara 5, NUMVECTS  12 : aciertos-> 25
NUMimagenesxcara 4, NUMVECTS  12 : aciertos-> 25
NUMimagenesxcara 3, NUMVECTS  12 : aciertos-> 100
NUMimagenesxcara 5, NUMVECTS  13 : aciertos-> 50
NUMimagenesxcara 4, NUMVECTS  13 : aciertos-> 75
NUMimagenesxcara 5, NUMVECTS  14 : aciertos-> 25
NUMimagenesxcara 4, NUMVECTS  14 : aciertos-> 75
NUMimagenesxcara 5, NUMVECTS  15 : aciertos-> 25
NUMimagenesxcara 4, NUMVECTS  15 : aciertos-> 100
NUMimagenesxcara 5, NUMVECTS  16 : aciertos-> 75
NUMimagenesxcara 4, NUMVECTS  16 : aciertos-> 100
NUMimagenesxcara 5, NUMVECTS  17 : aciertos-> 50
NUMimagenesxcara 5, NUMVECTS  18 : aciertos-> 50
NUMimagenesxcara 5, NUMVECTS  19 : aciertos-> 75
NUMimagenesxcara 5, NUMVECTS  20 : aciertos-> 100

Es realmente difícil sacar una conclusión de estos datos, ya que no existe una relación muy clara entre mayor número de aciertos por número de vectores propios o número de imágenes por cara elegidos. Se aprecia, es verdad, que el aumento en el número de vectores propios utilizado, proporciona una mayor tasa de acierto, pero es bastante desconcertante como influye más bien poco , en principio el número de imágenes por persona que elegimos. Incluso, en ocasiones, los datos denotan que es mejor utilizar entre 2-3 imágenes que un número mayor de ellas.

• Cómo afecta el número de vectores propios que utilices para calcular la nueva base en la clasificación?

Es más clara la importancia que tiene el uso de mayor número de vectores propios que el uso de un mayor número de imágenes por persona. Cuantos más vectores propios utilicemos, más tasa de acierto tenemos. Además, no influye mucho en el rendimiento del algoritmo, los tiempos de ejecución son muy aproximados (para cantidades altas de información, obviamente, si que tendrán importancia), pero a nuestro nivel, nos viene mucho mejor un nº alto de vectores propios.

• Mostrar gráficamente (en forma de imagen) los vectores propios que hayas  utilizado para calcular la nueva base (a estas imágenes se les llama también  eigenfaces o caras propias). ¿Qué características tienen?

He capturado todas las reconstrucciones (con un script) en la carpeta “reconstruidas”. Podemos ver como las gafas son una de las características que más fuertemente se marcan. Por otro lado, con apenas 1 imagen por persona y 3 vectores, se pueden apreciar bastante bien las caras reconstruidas (tasa acierto 100%) o incluso claramente con 1 imagen por persona y 4 vectores (tasa acierto, por el contrario 75%, aunque se distingan mejor).

• Qué ocurre si en lugar de tomar 5 imágenes para cada persona tomamos un número menor?

Con las imágenes delante, se aprecia claramente cómo un mayor número de ellas, hace la imagen más distorsionada y menos clara. Esta puede ser una de las razones por la que no encontraba una relación clara entre un mayor nº de imágenes y la tasa de acierto.

• Cómo afecta el número de vectores propios en la reconstrucción de las imágenes?

Guardando relación con la pregunta anterior, se puede llegar a la conclusión de que lo mejor para tener una tasa de aciertos alta es que se asemejen el nº de imágenes y el nº de vectores propios utilizados, ya que con muchas imágenes es difícil obtener una reconstrucción nítida y con pocos vectores propios, la información que podemos almacenar se resiente.

• Es PCA un buen método de reducción/reconstrucción?

En mi opinión si que nos proporciona un gran ahorro de datos respecto al tratamiento normal de imágenes y los resultados son bastante aceptables, siempre que se usen un nº razonable de imágenes y vectores propios. Cabe resaltar que tampoco se percibe una mayor carga de trabajo para el procesador aumentando el nº de vectores propios al máximo posible (como pasaba en el algoritmo del gradiente con 'alpha', por ejemplo) , por lo que no veo razón aparente para usar menos vectores propios que el nº de imágenes utilizadas, más sabiendo que se obtienen peores resultados.

FIN PRÁCTICA 4