Trabajo Final de Reconocimiento de Patrones: Identifiación utilizando PCA, ICA y LDA.

Mauricio Delbracio, Matías Mateu

8 de marzo de 2006

Resumen

En este documento, se presenta los resultados del trabajo con tres algoritmos clásicos de extracción y selección de características utilizados para problemas de reconocimiento de caras: Principal Component Analysis (`PCA`), Independent Component Analysis (`ICA`) y Linear Discriminant Analysis (`LDA`). Se evalúan estos algoritmos aplicados a la base de caras creada por nuestro grupo de Proyecto de Fin de Carrera, la `IIE Database`. Se compara la performance entre estos algoritmos en la identificación de personas enroladas en la base.

1. Introducción

Se aborda en este trabajo final de Reconocimiento de Patrones el problema de la identificación de caras. En los sistemas de identificación, la identidad es desconocida y por tanto el algorimto deberá descifrar la identidad correspondiente siempre que la imagen contenga una cara (problema de la localización) y que la eventual persona haya sido previamente enrolada a la base (problema de entrenamiento).

En identificación de caras, como en la mayoría de los problemas de procesamiento de imágenes, son extraídas ciertas características de cada imagen antes de ser clasificadas. Se busca seleccionar aquellas características relevantes para poder discriminar de modo de no trabajar con la imagen original. El trabajo con la imagen cruda presenta principalmente dos problemas: alta variabilidad inclusive para una misma persona y muy alta dimensionalidad. Sin embargo, uno no conoce a priori cuáles son esas características relevantes y discriminantes. De este problema surgen diversas soluciones posibles, cada una inspirada en ideas más o menos intuitivas acerca de cómo obtener estas características.

En este trabajo nos enfocamos en tres algoritmos bien conocidos en el mundo de reconocimiento e identificación de caras: `PCA`, `ICA` y `LDA`, todos ellos algoritmos de proyección en subespacios de menor dimensión que a través de argumentos estadísticos obtienen representaciones

de baja dimensionalidad que permiten pasar a la siguiente etapa en el procesamiento, la clasificación de las imágenes.

Se estudia entonces la performance de los algoritmos de clasificación utilizando el algoritmo de k-NN, con la norma euclideana. Se observa en la bibliografía estudios similares para otras normas como la norma de Mahalanobis o la norma del Coseno.

2. Background de los Algoritmos

2.1. Preprocesamiento y Normalización de la `IIE Database`

Se utilizó dos herramientas que permitieron normalizar la galería de imágenes. Por un lado aplicamos la herramienta de preprocesamiento y normalización del paquete `CSU` [5]. Este paquete es un sistema de evaluación de algoritmos de identificación de caras que trabaja sobre la `FERET` Database [6]. Por otro lado, para poder ingresar las imágenes al sistema de preprocesamiento se realizó extracción semiautomática de las coordenadas de los ojos, dado que esto es un requerimiento de la herramienta `CSU`. Esta extracción de coordenadas se realizó primero aplicando el algoritmo de detección de caras `esCARAbajo` [7] y luego corrigiendo manualmente las coordenadas de los ojos.

Sintéticamente los pasos fueron los siguientes:

- Detección semiautomática de coordenadas de los ojos.
- Conversión de entero a flotante: lleva los 256 niveles en sus equivalentes de punto flotante.
- Normalización geométrica: alinea las coordenadas de los ojos especificadas.
- Enmascaramiento: recorta la imagen usando una máscara elíptica y los bordes de la imagen de modo tal que sea visible la cara desde la frente al mentón.
- Ecualización de histograma: aplicado a la parte que no fue enmasacarada.
- Normalización de píxeles: escala los valores de los píxeles para obtener una media de cero y una varianza unitaria de estos valores.

Los últimos cinco pasos corresponden a la ejecución de *csuPreprocessNormalize* del sistema `CSU`. Luego de este preprocesamiento, obtuvimos la base normalizada pronta para utilizar como galería de imágenes de nuestro trabajo.

2.2. Principal Components Analysis: `PCA`

`PCA` es una técnica tradicional de proyección sobre un subespacio para reconocimiento de caras, es probablemente la más utilizada también. Se tiene un set de imágenes de entrenamiento I. Como primer

paso se computa la imagen promedio de I y se le resta a cada una de las imágenes de entrenamiento, obteniendo el set de datos:

$$i_1, i_2, ..., i_n \in I - \bar{I} \tag{1}$$

Luego se compone una matriz X tal que cada columna es una imagen de muestra. XX^T es la matriz de covarianza de las muestras de entrenamiento y las componentes principales de la matriz de covarianza se computan resolviendo

$$R^T(XX^T)R = \Lambda \tag{2}$$

donde Λ es la matriz diagonal de valores propios y R es la matriz de vectores propios ortonormales.

Se puede ver geométricamente que R es una matriz de cambio de base que rota los antiguos ejes a los ejes propios, donde el vector propio con valor propio más grande se corresponde con el eje de máxima varianza, el asociado con el segundo más grande se corresponde con la segunda mayor varianza y es ortogonal con el anterior y así sucesivamente.

Finalmente, nos quedamos con los n vectores propios de mayor valor propio asociado. El parámetro de compresión en este caso es precisamente n dado que indica la dimensión del vector de características que va a representar a la imagen original. Cada coeficiente de la representación de la imagen se obtiene proyectándola sobre cada uno de los vectores de la base `PCA`.

2.3. Independent Components Analysis: `ICA`

`ICA` es una herramienta de análisis cuyo objetivo es descomponer una señal observada (imagen de una cara) en una combinación lineal de fuentes independientes. Surge de la técnica conocida por su sigla `BSS`, o Blind Sepparation Source, que intenta obtener las fuentes independientes a partir de combinaciones de las mismas.

Mientras que `PCA` decorrelaciona las señales de entrada utilizando estadísticos de segundo orden (minimizando el error cuadrático medio de proyección, i.e.: `KLT`), `ICA` minimiza mayores órdenes de dependencia.

El número de observaciones $N(1 \leq i \leq N)$ debe ser mayor o igual al número de fuentes originales $M(1 \leq j \leq M)$. En general se utiliza $N = M$. Asumiendo que cada X_j es una combinación desconocida y diferente de los "vectores fuentes" originales, `ICA` expande cada señal X_j en una suma ponderada de vectores fuente. Encontramos aquí una fuerte similitud con `PCA`.

Sea $\mathbf{S}$ la matriz de señales independientes y $\mathbf{X}$ la matriz de observación. Si A es la matriz de combinación desconocida, el modelo de combinación se puede escribir como:

$$\mathbf{X} = A.\mathbf{S} \tag{3}$$

Asumiendo que las señales fuente son independientes unas de las otras y que la matriz A es invertible, el algoritmo `ICA` tratará de encontrar la matriz de separación W, tal que:

$$\mathbf{U} = W.\mathbf{X} = W.A.\mathbf{S} \tag{4}$$

donde $\mathbf{U}$: estimación de las componentes independientes.

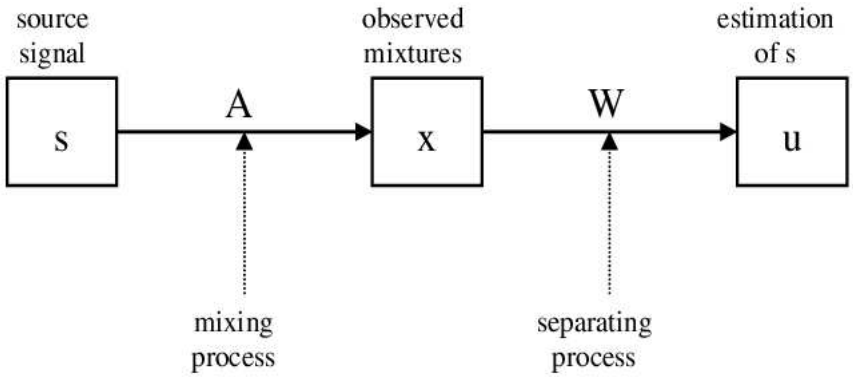

Figura 1: Esquema Blind Source Separation

La figura 1 muestra un esquema de lo anterior.

La esencia de los algoritmos que implementan `ICA` es la búsqueda de la matriz W según cierto método iterativo de optimización. Para una matriz U vista como arreglo de vectores, los vectores son estadísticamente independientes cuando

$$f_U(U) = \prod_i f_{U\,i}(U_i) \tag{5}$$

En esta implementación de `ICA` utilizamos el algoritmo `FastICA`[3], probablemente uno de los algoritmos más generales, el cual maximiza:

$$J(y) \simeq C[E\{G(y)\} - E\{G(v)\}]^2 \tag{6}$$

En donde G : función no cuadrática, v : densidad de probabilidad gaussiana y C es una constante mayor a cero. Se puede demostrar que maximizando una función de estas características se obtiene un óptimo

en el sentido de independencia buscado.

FastICA es un algoritmo ampliamente explorado en esta área. Desde el punto de vista de la performance de los algoritmos que implementan ICA se ha demostrado empíricamente que existen diferencias muy pequeñas y que todos obtienen un óptimo muy similar de componentes independientes.

En la literatura de esta temática se argumenta que ICA no tiene ventajas como algoritmo de clasificación perse [1], [2]. En la práctica, y en nuestro caso, se suele utilizar PCA como preprocesador: se aplica PCA para proyectar las observaciones en un subespacio de dimensión m (utilizamos $m = 100$). De esta forma se controla el número de componentes o dimensión de la base producida por ICA.

En este trabajo utilizamos una implementación de ICA clásica y una alternativa:

- En la implementación clásica alimentamos el algoritmo FastICA con las primeras m componentes de las imágenes de entrenamiento proyectadas en la base PCA. A la salida, el algoritmo nos devuelve la matriz de separación W. Para obtener el representante de una imagen de prueba entonces primero proyectamos sobre la base PCA y obtenemos los primeros m coeficientes y luego proyectamos sobre la matriz entrenada W. Con este representante de la imagen original es que se alimenta el algoritmo de clasificación.
- En la implementación alternativa se alimenta el algoritmo FastICA directamente con los primeros m vectores de la base PCA. A la salida se recoge la matriz U que está compuesta en sus columnas por las coordenadas independientes de la base PCA. Luego las imágenes de prueba se proyectan sobre la matriz U. El vector de dimensión m resultante es lo que se toma como el representante de las imágenes originales.

2.4. Linear Discriminant Analysis: LDA

LDA o *Linear Discriminant Analysis* es una técnica de aprendizaje supervisado para clasificar datos. La idea central de LDA es obtener una proyección de los datos en un espacio de menor (o incluso igual) dimensión que los datos entrantes, con el fin de que la separabilidad de las clases sea la mayor posible. Es una técnica supervisada ya que para poder buscar esa proyección se debe entrenar el sistema con patrones etiquetados. Es importante aclarar que LDA no busca en ningún momento minimizar el error de representación cometido, como sí lo hacía PCA.

Existen varias implementaciones de LDA, entre ellas se encuentra Fisher-LDA [8]. Para explicarlo vamos a considerar la versión más simple del problema:

Encontrar el vector w de proyección, que proyecte los datos a un espacio uni-dimensional de manera de obtener la mayor separabilidad entre sus clases.

Formalizando, tenemos $x_1..x_n$ patrones d-dimensionales etiquetados en c clases. Cada clase cuenta con N_c patrones. Se busca w, para obtener $y_i = w^T x_i$ proyecciones uni-dimensionales de los patrones.

Lo que busca `Fisher-LDA` es maximizar la siguiente función objetivo:

$$J(w) = \frac{w^T S_B w}{w^T S_W w} \tag{7}$$

donde S_B es la matriz de dispersión inter-clase y S_W es la matriz de dispersión intra-clase. Siendo más precisos:

$$S_B = \sum_c N_c(\mu_c - \mu)(\mu_c - \mu)^T \tag{8}$$

$$S_W = \sum_c \sum_{i \in c} (x_i - \mu_c)(x_i - \mu_c)^T \tag{9}$$

Siendo μ_c la media de cada clase, μ la media de todos los datos, N_c la cantidad de patrones de la clase c.

`Fisher-LDA` busca encontrar el vector w de proyección que maximice el "cociente" entre la matriz de dispersión inter-clase y la matriz de dispersión intra-clase.

Operando se puede ver que el w que maximiza la función objetivo debe cumplir:

$$S_B w = \lambda S_W w \tag{10}$$

Si S_W es no singular podemos resolver el clásico problema de valores propios para la matriz $S_W^{-1} S_B$:

$$S_W^{-1} S_B w = \lambda w \tag{11}$$

Si ahora sustituimos la solución en (7) obtenemos lo siguiente:

$$J(w) = \frac{w^T S_B w}{w^T S_W w} = \lambda_k \frac{w_k^T S_B w_k}{w_k^T S_W w_k} = \lambda_k \text{ con } k = 1..d \tag{12}$$

siendo w_k vector propio k de valor propio λ_k.

En consecuencia, para maximizar la solución debemos considerar el vector propio con mayor valor propio asociado.

Claro está que este desarrollo valió para el caso en que queremos proyectar los datos sobre un espacio uni-dimensional. Se puede ver sin mayor esfuerzo [9] que para el caso de querer proyectar sobre un espacio m-dimensional, se debe resolver el mismo problema y elegir los m

vectores propios con valores propios asociados más grandes.

En nuestro caso particular en donde se trabajó con imágenes (datos de alta dimensión) se aplicó, como en el caso de `ICA`, una primera etapa de `PCA` para reducir la dimensionalidad de los datos. Los datos fueron reducidos a dimensión 100. Cabe acotar que existen formas directas de aplicar `LDA` (`D-LDA`) que no fueron objeto de estudio en este proyecto.

Al igual que en los casos anteriores, para la clasificación se utilizó el algoritmo `k-nn`.

3. Experimentos - Resultados

Para evaluar los algoritmos propuestos, se utilizó la `IIE Database`. La base consta de 24 imágenes por persona y está formada por un total de 49 individuos. Entre las diferentes imágenes, se encuentran dos sets obtenidos con una semana de diferencia, imágenes con diferentes expresiones y en diferentes condiciones de iluminación.

Realizamos tres tipos de experimentos:

- **Test I**: Entrenamiento: 4 imágenes de frente. Evaluación: 2 imágenes de frente (tomadas una semana después).
- **Test II**: Entrenamiento: 2 imágenes de frente. Evaluación: 2 imágenes de frente (tomadas en el mismo momento).
- **Test III**: Entrenamiento: 4 imágenes de frente. Evaluación: 8 imágenes de varias expresiones, partes ocluídas o problemas de iluminación.

Luego de proyectar los datos en los diferentes subespacios, se aplicó `k-nn` con métrica euclideana. Se estudió la performance del mismo al aplicarlo con parámetro $k = 1, 3, 5$.

A continuacion mostramos algunos de los resultados obtenidos.

Test I	**Dimensión**				**max**	
$k = 1$	10	30	50	70	RR_{max}	dim
`PCA`	71.43 %	86.73 %	88.78 %	88.78 %	88.78 %	40
`ICA`	69.39 %	86.73 %	91.84 %	91.84 %	91.84 %	50
`LDA`	86.73 %	95.92 %	95.92 %	92.86 %	96.94 %	39

Test I	**Dimensión**				**max**	
$k = 3$	10	30	50	70	RR_{max}	dim
`PCA`	70.40 %	84.69 %	86.73 %	86.73 %	87.75 %	42
`ICA`	68.36 %	85.71 %	90.81 %	91.83 %	91.83 %	70
`LDA`	85.71 %	95.91 %	95.91 %	92.85 %	97.95 %	56

La tabla anterior muestra la performance (RR[1]) de los algoritmos

[1] RR: Recognition Rate - Porcentaje de Reconocimiento

frente al Test I al variar la dimensión del subespacio de proyección. La primer tabla clasifica utilizando el algoritmo k-NN con $k = 1$ (vecino más cercano). La segunda utiliza el mismo algoritmo pero con $k = 3$. Ambos casos dan resultados muy similares. Observando los resultados vemos que a partir de 40 coeficientes éstos prácticamente no varían. En ambos casos el algoritmo que mejor se desempeña es `LDA` logrando un altísimo porcentaje de reconocimiento. En la figura 2 se muestra la evolución de los algoritmos al considerar mayor número de componentes.

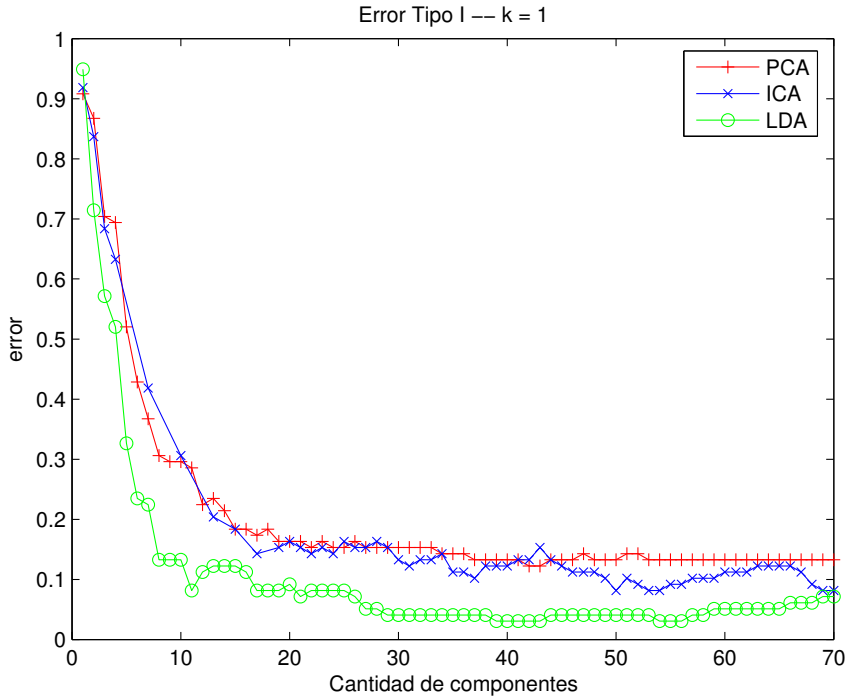

Figura 2: Evolución del error cometido al clasificar el Test I en función de la cantidad de componentes tomados. Clasificación mediante 1-NN. Test I.

A continuación mostramos los resultados obtenidos al realizar el Test II. Vemos que en este caso, los resultados son mejores que en el caso anterior. En este Test, todos los algoritmos tienen un muy buen desempeño. Cabe acotar que las imágenes utilizadas para el entrenamiento y para la validación fueron adquiridas en la misma sesión fotográfica.

Test II	**Dimensión**				**max**	
$k = 1$	10	30	50	70	RR_{max}	dim
PCA	86.73 %	94.89 %	95.91 %	95.91 %	96.93 %	33
ICA	88.77 %	97.95 %	97.95 %	94.89 %	97.95 %	29
LDA	78.57 %	94.89 %	97.95 %	93.87 %	98.97 %	41

Según la literatura existen variantes de los métodos explicados anteriormente. Por ejemplo, para el método PCA, hay autores [5] que sostienen que las primeras componentes no sirven para diferenciar las imágenes de la base. En nuestro caso implementamos la clasificación sin tomar en cuenta los primeros dos componentes (PCA2). Respecto a ICA, se obtuvieron los resultados mediante el método original y su variante (ICA2). En todos estos casos el algoritmo utilizado fue 1-NN. Para el algoritmo LDA mostramos los resultados de aplicar la clasificación k-NN, con $k = 1, 3, 5$. En la figura 3 se muestran algunos resultados.

Test I	**Dimensión**		**Max**	
Método	30	50	RR_{max}	dim
PCA	86.73 %	88.78 %	88.78 %	40
PCA2	84.69 %	90.82 %	90.82 %	44
ICA	86.73 %	91.84 %	91.84 %	50
ICA2	85.71 %	87.75 %	87.75 %	24
LDA k1	95.92 %	95.92 %	96.94 %	39
LDA k2	95.91 %	95.91 %	97.95 %	56
LDA k3	94.89 %	95.91 %	97.95 %	49

Aplicando los algoritmos al Test III - el más difícil - los resultados no fueron alentadores como en el resto. La siguiente tabla muestra la codificación del tipo de foto en una letra.

m : sonriente
n : sorprendido
o : enojado
p : guiñada
q : bufanda
r : lentes normales
s : lentes negros
t : iluminación no uniforme

A continuación se presenta un resumen de lo obtenido frente al Test III. Los resultados mostrados corresponden a representaciones de dimensión 50.

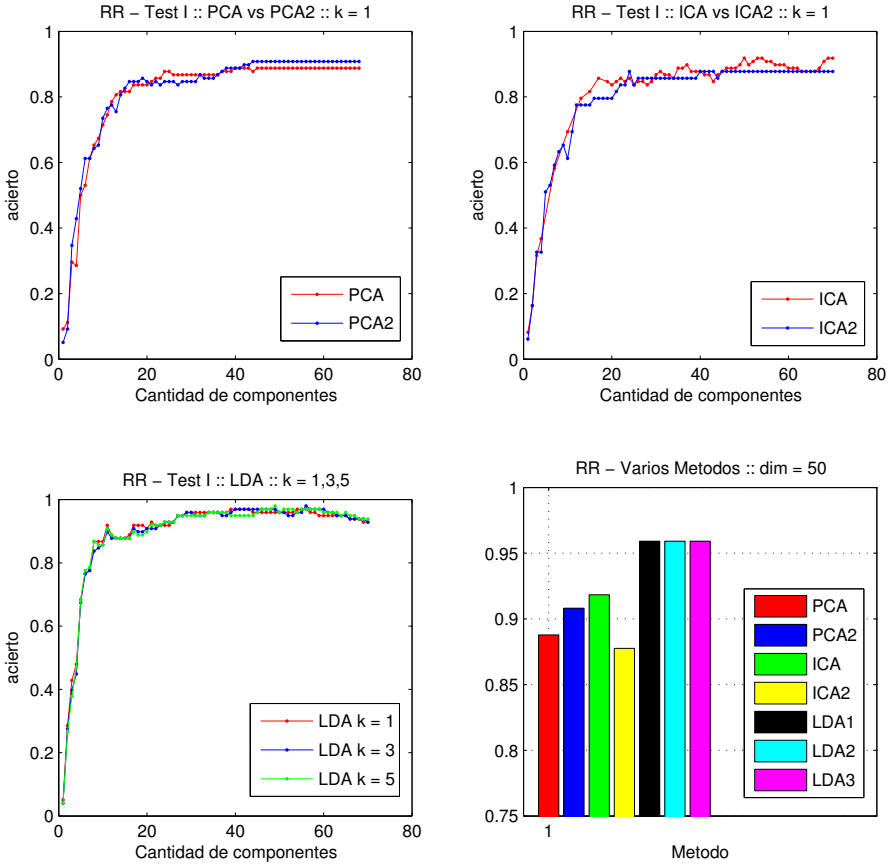

Figura 3: Resultados varios para el Test I. `PCA2`: `PCA` quitando los dos primeros componentes para la clasifiación. `ICA2`: Arquitectura `ICA` entrenada directamente mediante la base `PCA`. `LDA` para $k = 1, 3, 5$. En los tests `ICA` y `PCA` se utilizó 1-NN para clasificar.

Test III	**Tipo de Foto**							
Método	m	n	o	p	q	r	s	t
`PCA`	81.6 %	67.3 %	71.4 %	79.5 %	28.5 %	59.1 %	18.3 %	20.4 %
`PCA2`	77.5 %	61.2 %	65.3 %	69.3 %	28.5 %	61.2 %	12.2 %	20.4 %
`ICA`	73.4 %	73.4 %	73.4 %	73.4 %	20.4 %	57.1 %	24.4 %	2.0 %
`ICA2`	81.6 %	67.3 %	71.4 %	79.5 %	28.5 %	61.2 %	18.3 %	20.4 %
`LDA`	87.7 %	81.6 %	83.6 %	83.6 %	65.3 %	81.6 %	32.6 %	26.5 %

Finalmente, en la figura 4 mostramos un ejemplo de bases de caras utilizadas para proyectar las imágenes originales en espacios de menor dimensión.

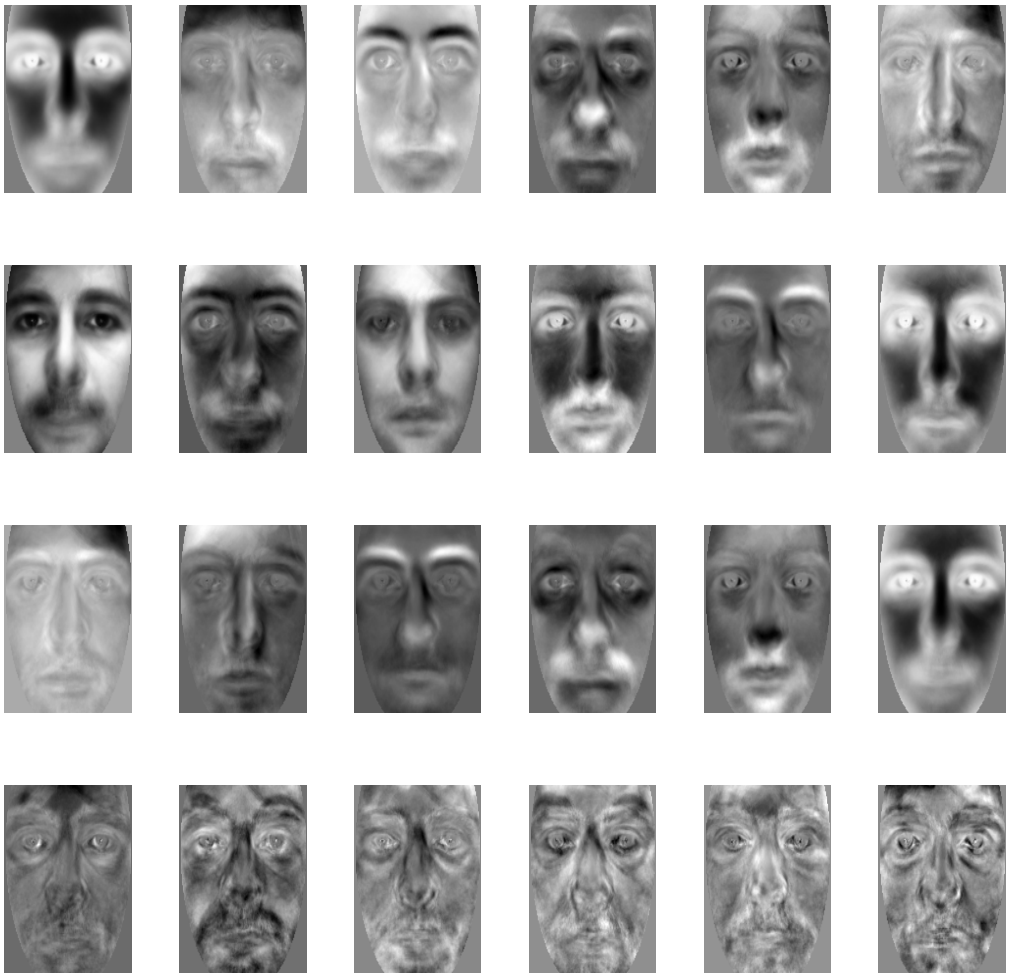

Figura 4: Imágenes generadoras de los diferentes subespacios de dimensión 6. En orden descendente: base `PCA`, `ICA`, `ICA2`, `LDA`.

4. Conclusiones

Se estudió la performance de tres algoritmos bien conocidos en el mundo del Reconocimiento de Caras. Se encontró que todos ellos son fuertemente sensibles a las imágenes con las que fueron entrenados. Su performance baja significativamente frente a oclusiones y a problemas de iluminación.

De manera global podemos concluir que el algoritmo que obtiene mejores resultados al clasificar es `LDA`, el único de los algoritmos que se entrena supervisadamente. En el caso de `ICA` concluimos que mejora

visiblemente los resultados que `PCA`, aún estando lejos de la performance de `LDA`. Vimos que `PCA` es un buen reductor de dimensión (útil como compresor), que por sí solo no logra obtener buenos resultados de clasificación. `LDA` logra obtener un buen RR frente a variaciones de expresiones, como pueden ser una sonrisa o un enojo.Frente a oclusiones se vio que ninguno de los algoritmos logra una performance aceptable. Esto es debido al caracter holístico de los algoritmos implementados.

Respecto a las variantes de los algoritmos utilizados, se concluye que `ICA2` no logra obtener una performance tan buena como la de `ICA` - clásico - obteniendo valores similares a los de `PCA`. En cuanto a `PCA2` se obtiene una mejora sensible frente a `PCA` en la mayoría de los casos.

Se observó que la performance de los algoritmos, no tuvo un cambió significativo al variar el parámetro k del clasificador por vecino más cercano.

Referencias

[1] C. Havran, L. Hupet, J. Lee, L. Vandendorpe, M. Verleysen,"Independent Component Analysis for Face Authentication," *Proceedings-Knowledge-based Intelligent Information and Engineering Systems,*(Crema, Italy), (2002).

[2] A. Draper, K. Baek, M. S. Barlett, J. Ross,"Recognizing Faces with PCA and ICA ," *for Special Issue on Face Recognition,*(Colorado State, USA), (2002).

[3] `FastICA` (GPL) package for Matlab, *http://www.cis.hut.fi/projects/ica/fastica/,* (Laboratory of Computer and Information Science, Neural Network Research Group, University of Helsinki, Finland.) (ult. visita: febrero de 2006).

[4] M.S. Barlett, J.R. Movellan and T.J. Sejnowski, "Face Recognition by Independent Component Analysis", *IEEE Transaction on Neural Networks* vol. 13, pp. 1450-1462, 2002., volume 13,(2002): 1450-1462.

[5] "The `CSU` Face Identification Evaluation System" *http://www.cs.colostate.edu/evalfacerec/* Computer Science Departament, Colorado State University. (ult. visita: Febrero de 2006).

[6] "The Feret Database", *http://www.nist.gov/humanid/colorferet-/home.html,* (ult. visita: Febrero de 2006)(.NIST,USA,2002.)

[7] C. Aguerrebere, G Capdehourat, M.Delbracio, M.Mateu, "Página web del proyecto final del curso de Tratamiento de Imágenes", *http://iie.fing.edu.uy/investigacion/grupos/gti-/timag/trabajos/2005/caras/index.htm,* (ult. visita: Febrero de 2006).

[8] M.Welling, "Fisher Linear Discriminant Analysis" *Department of Computer Science, University of Toronto*

[9] Duda, Hart, "Pattern Classification (2nd Edition)" *Wiley* 0471056693 (2000)