Trabajo Práctico 2

Jueves 15 de Mayo de 2014 Métodos Numéricos

Reconocimiento de rostros mediante técnica PCA.

Integrante	LU	Correo electrónico
Pedro Rodriguez	197/12	pedrorodriguezsjs@hotmail.com
Martín Caravario	470/12	martin.caravario@gmail.com
Esteban Fernandez	691/12	esteban.pmf@gmail.com

Facultad de Ciencias Exactas y Naturales
Universidad de Buenos Aires
Ciudad Universitaria - (Pabellon I/Planta Baja)
Intendente Güiraldes 2160 - C1428EGA
Ciudad Autonoma de Buenos Aires - Rep. Argentina
Tel/Fax: (54 11) 4576-3359
http://www.fcen.uba.ar

Índice

Resumen

Actualmente es posible crear nuevas medidas de seguridad, en particular, para establecimientos donde es posible que concurran personas clasificadas como peligrosas. Dicha posibilidad está basada en el poder de cálculo de los computadores actuales que nos permiten, mediante aplicaciones del aĺgebra lineal, acceder a estas medidas en cuestión. Una de ellas, en este caso, es la del reconocimiento de rostros de las personas, método que desarrollaremos mediante el uso de una técnica conocida como PCA (análisis de componentes principales). Con dicho desarrollo podríamos determinar, comparando con nuestra base de datos, dada una fotografia, quién es la persona que ingresó, por ejemplo, al establecimiento y determinar su derecho de admisión dependiendo de sus antecedentes.

Palabras clave : PCA - Reconocimiento de rostros - Analisis de componentes principales.

1. Introducción

El siguiente informe tiene como finalidad poner al tanto al lector sobre una aplicación de álgebra lineal que será aprovechada para implementar un sistema computacional, como se dijo en el resumen, de reconocimiento de rostros. A continuación se presentarán los fundamentos teóricos que crean las bases de nuestra implementación.

- Se utilizará una técnica llamada Análisis de Componentes Principales. La misma se encarga de reducir, arbitrariamente, la dimensionalidad de un conjunto de datos con el fin de analizar las causas, de aquí el nombre, principales que influyen en la variación de los mismos en conjunto y ordenarlas por importancia.[1]
- Se busca crear un método predictivo mediante la aplicación de esta técnica, se reprensetarán los datos de una forma mejor (mediante un cambio de base) en términos de mínimos cuadrados (es decir, dada una cantidad de datos, compuestos por variables dependientes e independientes entre sí, y un conjunto de funciones continuas, encontrar una de estas ultimas que se ajuste mejor (se aproxime a los datos en cuestión) según el criterio de mínimo error cuadrático.
- Esta nueva representación de los datos obtenida mediante el cambio de base guardará la información más relevante en términos de la covarianza entre los datos muestrales (es decir, la base de datos): en consecuencia, el primer eje en el nuevo sistema de coordenadas será la que contenga la captura de la mayor covarianza entre los datos, a dicho eje se lo conocerá como Primer Componente Principal. Se analizará entonces la matriz de covarianza a través del cálculo de su descomposición en valores singulares. Se analizará la precisión de este método implementado verificando la tasa de aciertos a la hora de reconocer personas mediante diversas instancias que serán el punto de partida para ejecutar nuestros algoritmos.

2. Desarrollo

En el presente TP, realizamos tres experimentos, cada uno bastante abarcativo, a partir de los cuales estudiamos el sistema computacional que implementamos, como esta explicado en la introducción.
Antes de comprobar la precisión de nuestra implementación a través de los test de prueba, es necesario aclarar el procedimiento seguido para obtener el sistema de reconocimiento finalizado (realizado a partir del aprovechamiento de la técnica PCA que funda la base del funcionamiento del mismo):

Teorema 1: Una matriz $A \in \mathbb{R}^{nxn}$ es diagonalizable si y solo si A tiene n autovectores linealmente independientes.

Teorema 2: Si $A \in \mathbb{R}^{nxn}$ es simétrica, entonces existe una base ortonormal de autovectores $v_1, \ldots, v_n$ asociados a $\lambda_1 \ldots \lambda_n$. Por ende, existe P, y $P^{-1} = P^t$, luego $A = PDP^t$.

Implementación del Método de la Potencia : Se utilizará un algoritmo, que implementará el método de la potencia que será el encargado de, a partir de un vector cualquiera, transformarlo y normalizarlo, de manera tal que, luego de un número considerable de iteraciones de este proceso el mismo termine convergiendo al primer autovector principal, el cual tendrá asociado el autovalor más grande en módulo en comparación al resto que aún faltarían calcular para completar la base de autovectores (es una base ya que se utilizarán matrices simétricas, por lo que este método funciona sin problemas).
El pseudocódigo de este método es el siguiente:

$MetodoPotencia(B, x_0, niter)$
$v \leftarrow x_0$
  Para $i = 1 \ldots niter$
  P$v \leftarrow \frac{B_v}{||B_v||}$
  $\lambda \leftarrow \frac{v^t B v}{v^t v}$
  Fin Para
Devolver λ, v

Con esto habremos encontrado un único autovector y autovalor(el mayor en módulo de la matriz), para continuar utilizando éste método y encontrar los siguientes autovalores y autovectores utilizaremos la deflación en un algoritmo:

Deflación: Sea B una matriz con autovalores distintos $|\lambda_1| > |\lambda_2| > \ldots > |\lambda_n|$ y una base ortonormal de autovectores. Entonces la matriz $B - \lambda_1 v_1 v_1^t$tiene autovalores $0, \lambda_2, \ldots, \lambda_n$ con autovectores asociados $v_1, \ldots, v_n$.
Observación: en nuestro caso alcanza que con que los primeros k tengan magnitudes distintas.

[1] `http://www.cs.princeton.edu/picasso/mats/PCA-Tutorial-Intuition_jp.pdf`

Finalmente la deflación consistirá en iterar el método de la potencia k veces sobre la matriz B de la cual queremos calcular sus componentes principales(cantidad de componentes principales a considerar, parámetro del algoritmo principal del programa) sobre B e ir guardando en cada iteración los autovectores $v_1, \ldots, v_k$ y $\lambda_1, \ldots \lambda_k$ de manera apropiada para luego poder utilizarlos. La única sutileza, es que en cada iteración es necesario reasignar B de la siguiente manera: $B \leftarrow B - \lambda_1 v_1 v_1^t$, para no calcular siempre el mismo autovector sino ir calculando el autovalor deseado.

Procedimiento: Dada una base de datos de fotografías de personas, consideramos los píxeles como variables distintas. Sea n la cantidad de imágenes totales de la base, y m la cantidad de píxeles de cada una de ellas (cabe destacar que es precondición, para nuestra implementación, que ésta última cantidad coincida en todas las fotos de la base a tomar, en particular, las imágenes deben tener la misma cantidad de píxeles de alto entre sí y, también la correspondiente de ancho, entre sí).

Vamos a vectorizar cada imágen para, así, confeccionar la matriz X que tendrá en cada una de sus filas una fotografía distinta, de las que fueron obtenidas de la base de datos, de m pixeles (es decir, la matriz X finalmente, será de n filas por m columnas).

Como no se pierde generalidad en el procedimiento, se muestra a continuación como quedaría X usando muestras de n datos (en el caso a analizar luego en este informe, los mismos serán las imágenes) de $\mathbb{R}^2$ (es decir $m = 2$):

Sean $x^{(1)} ... x^{(n)}$ una muestra de n datos con $x^{(i)}$ perteneciente a $\mathbb{R}^2$. Tomaremos, como se nombró previamente, los mismos de forma vectorial, para luego trasponerlos, de manera tal que a la hora de generar X, cada dato sea una fila de la misma y en cada columna se vean reflejadas las diferentes componentes (variables) de los datos obtenidos de la muestra.
Una vez confeccionada X, nos quedaría:

$$X = \begin{pmatrix} {x^{(1)}}^t \\ {x^{(2)}}^t \\ \vdots \\ {x^{(n)}}^t \end{pmatrix} = \begin{pmatrix} 26{,}4320 & 27{,}7740 \\ 26{,}8846 & 26{,}5631 \\ \vdots & \\ 16{,}0650 & 24{,}210 \end{pmatrix}$$

Nuestro objetivo será obtener la matriz de covarianzas de X, a la cual llamaremos Mx, la misma nos mostrará como están relacionadas las variables las unas con las otras, indicándonos así, justamente, que tanto varían entre ellas y en conjunto.
Será necesario centrar los datos respecto a los ejes de coordenadas. Tomamos entonces la media de cada una de las variables que conforman nuestros datos a analizar:
Sea $\mu = 1/n(x^{(1)} + x^{(2)} + \cdots + x^{(n)})$
Para este ejemplo en $\mathbb{R}^2$ resultaría: $\mu = (29{,}3623, 29{,}7148)$
Dicha necesidad de centrar los datos a partir del μ calculado se verá reflejada en la cuenta que se hará a continuación para obtener la matriz buscada: Mx.

Definición:

Covarianza: como se dijo previamente, es una medida que nos da una idea de cuánto variación hay entre un par de variables. Mientras mayor sea ésta, mayor será la dependencia/relación entre las mismas.

Sea v =(1, ..., 1). Se define la covarianza entre dos variables x_k y x_j como:

$$\sigma_{x_k x_j} = \sum_{i=1}^{n} (1/n-1)(x_j^{(i)} - \mu_j)(x_k^{(i)} - \mu_k) = (1/n-1)(x_k - \mu_k v)^t (x_j - \mu_j v)$$

Nota: si $k = j$ entonces estariamos obteniendo la Varianza de una variable particular.

Dicho esto, procedemos a mostrar como resultaria la matriz Mx tras realizar el cálculo de la covarianza entre sus variables (siguiente con nuestro ejemplo en $\mathbb{R}^2$):

$$X = \begin{pmatrix} 26{,}4320 - \mu_1 & 27{,}7740 - \mu_2 \\ 26{,}8846 - \mu_1 & 26{,}5631 - \mu_2 \\ \vdots & \\ 16{,}0650 - \mu_1 & 24{,}210 - \mu_2 \end{pmatrix}$$

$$Mx = (1/(n-1))X^t X = \begin{pmatrix} \sigma_{x_1 x_1} & \sigma_{x_1 x_2} \\ \sigma_{x_1 x_2} & \sigma_{x_2 x_2} \end{pmatrix} = \begin{pmatrix} 66{,}2134 & 27{,}1263 \\ 27{,}1263 & 12{,}5491 \end{pmatrix}$$

Ahora, nuestro objetivo es buscar una transformación de los datos de manera tal que disminuya la redundancia entre los mismos (es decir, disminuir la covarianza): Realizaremos un cambio de base: $X^t = PX^t$ mediante la diagonalización de la matriz de covarianza. Al hacerlo buscamos variables que tengan covarianza 0 entre sí y la mayor varianza posible, entonces podremos ver la matriz Mx como una diagonal, concentrando en dicha diagonal (valga la redundancia) las varianzas de las variables, para luego buscar las que presenten mayor medida.

La idea es escribir $Mx = PDP^{-1}$
El cambio de base, como se dijo, será $X^t = PX^t$:
Sea P ortogonal y Mx la matriz de covarianza de X.

$$Mx = \frac{1}{n-1}X^tX = \frac{1}{n-1}(PX^t)(XP^t) = P\frac{X^tX}{n-1}P^t = PMxP^t$$

Como Mx es simétrica, entonces existe V ortogonal tal que $Mx = VDV^t$.

$$Mx = PMxP^t = P(VDV^t)P^t \text{ tomamos } P = V^t \text{ Entonces } (V^tV)D(VV^t) = D$$

Aclaración: se presentarán y desarrollarán dos métodos, uno para imágenes chicas (Método 0, imágenes de 28x23 píxeles) el cual directamente, dada una matriz de 644x644 calcula los autovectores de la matriz de covarianzas Mx mediante el método de la potencia por deflación y otro para imágenes grandes(Método 1, imágenes de 112x92 píxeles) que aplica el método de la potencia por deflación sobre una matriz diferente, más pequeña que la utilizada para el método 0, y a partir de estos resultados, mediante manipulaciones algebraicas y matemáticas obtenemos los mismos autovectores que los obtenidos en el método 0.
Esta posibilidad se ve reflejada directamente en el tamaño de las imágenes, ya que, como se vio anteriormente, es necesario realizar $\frac{X^tX}{n-1}$ que es, justamente, Mx; pero X $\in \mathbb{R}^{nxm}$ donde, m en este caso es $23 * 28 = 644$.
Entonces, al hacer X^tX, el resultado nos dará una matriz de 644 filas y 644 columnas; si bien es grande la matriz, es posible continuar calculando sin problemas, es decir terminando en un tiempo razonable el cómputo, los autovectores y autovalores que es lo que se consigue aplicando iterativamente dicho método nombrado.
Luego de desarrollar el método 0, se mostrará una alternativa que nos permitirá resolver el problema en cuestión cuando se utiliza una base de datos con imágenes de mayor tamaño (en este caso, se utilizarán imágenes de 112 por 92 píxeles). Se puede ver entonces que el método 0 queda descartado para obtener la matriz de covarianzas y calcular directamente mediante dicha matriz los autovectores y autovalores; esto se debe a que si hacemos X^tX, en este caso m es $112 * 92 = 10304$, por lo que la matriz resultante de esa multiplicación será de 10304 filas y 10304 columnas, haciendo que se demore demasiado tiempo en la multiplicación nombrada y los cálculos que vienen a continuación tendrán un tiempo de cómputo demasiado elevado, lo cual no se justifica. Dicha alternativa es el Método 1 y se presentará a continuación del 0.

Método 0: Las columnas de V (autovectores de Mx) son las componentes principales de los datos. En caso de que m sea grande, es posible tomar sólo un subconjunto de las componentes principales para estudiar, en particular, las que nos brinden la mayor proporción de la varianza de los datos, esto se verá reflejado en el valor singular que nos brindará cada autovalor asociado a dichas componentes principales (es decir, los autovectores de Mx). Dadas las dimensiones de las imágenes, en este caso las pequeñas, para hallar los autovalores y autovectors que necesitaremos más adelante para el reconocimiento de rostros es posible utilizar directamente la Deflación sobre la matriz de covarianzas Mx.

Método 1: Este método, como se nombró en la aclaración, tiene como finalidad aplicar la técnica PCA a las bases de datos de imágenes de mayor tamaño. Esta consiste en hallar los autovectores de Mx a través de la descomposición en valores singulares (es decir, la descomposición SVD) de las matrices que se presentarán a continuación. La idea es definir $A = \frac{X}{\sqrt{n-1}}$, donde r es el rango de A, de manera tal que $Mx = A^tA$, y si, ahora, consideramos la descomposicion en valores singulares de A, la cual sería $A = U\Sigma V^t$, entonces $Mx = A^tA = V\Sigma^tU^tU\Sigma V^t = V\Sigma^t\Sigma V^t$.
Las columnas de V contienen los autovectores de Mx (por Teorema 2), donde $\Sigma = \begin{pmatrix} D & 0 \\ 0 & 0 \end{pmatrix}$
Con $D \in \mathbb{R}^{rxr}$ con $d_{ii} = \sigma_i > 0$ los valores singulares, $\sigma_i \geq \sigma_{i+1}$
Aclaramos: $\sigma_i = \sqrt{\lambda_i}$, λ_i autovalor de A^tA.
$V = [v_1, v_2 \dots v_m]$, $A^tAv_i = \lambda_i v_i$
$u_i = \frac{1}{\sigma_i}Av_i, i = 1 \dots r$ (el resto se extiende a base ortonormal).

El problema se reduce a encontrar los autovectores de Mx, es decir V, pero no podremos utilizar A^tA ya que es una matriz muy grande (para el caso en consideración, las imágenes más como se advirtió en la aclaración previa, pero cabe destacar que este procedimiento vale también para la base de datos de imágenes chicas), por lo que analizaremos como podemos utilizar AA^t que ya no seria de 10304 filas por 10304 columnas sino que será de n filas por n columnas ($n =$ cantidad de imágenes).
Utilicemos entonces AA^t:

$$A = U\Sigma V^t \implies A^t = V\Sigma U^t \implies AA^t = U\Sigma V^tV\Sigma U^t = U\Sigma\Sigma U^t$$

Como AA^t es simétrica entonces, podemos diagonalizarla con una base de autovectores ortonormal. Es decir $P = U, P^{-1} = U^t$y $D = \Sigma^2$, donde U es la matriz que contendrá los autovectores asociados a AA^t. Diremos que $v_i = A^tu_i$ (i = 1, ... k) son los autovectores asociados a la matriz de covarianzas $Mx = A^tA$, justamente lo que buscábamos. Antes que nada es necesario decir que para obtener cada u_i hay que aplicar el método de la potencia a la matriz AA^t, la cual será de n filas por n columnas (donde n era la cantidad de imágenes de la base de datos considerada). Esta matriz, es mucho mas chica que la cantidad total de píxeles de las imágenes, por lo que claramente

puede usarse dicho método en un tiempo aceptable, obviamente por ser de menor dimensionalidad, menor que el utilizado para calcular de forma directa los autovectores de A^tA.
Daremos ahora la prueba de la proposición que afirma que $v_i = A^tu_i$ (i = 1, ... k). es autovector de la matriz de covarianzas $Mx = A^tA$.

Sea u_i autovector de $AA^t = U\Sigma\Sigma U^t$, y λ_i su autovalor asociado.
Entonces vale que:

$AA^tu_i = \lambda_i u_i$, multiplicamos a ambos lados por A^t, nos queda $A^tAA^tui = A^t\lambda_i u_i$, si agrupamos de la siguiente manera: $A^tA(A^tu_i) = \lambda_i(A^tu_i)$ se puede apreciar que si renombramos $v_i = A^tu_i$ la cuenta nos quedaria $A^tAv_i = \lambda_i v_i$.

De esto deducimos entonces que, por definición de autovectores, $v_i = A^tu_i(i = 1, \dots n)$ es autovector de $A^tA = Mx$.

Finalmente, una vez encontrados los autovectores de Mx (mediante, en este caso, con el método 1, o bien, con el método 0) es decir V, resta utilizarlos de manera conveniente como mostraremos a continuación.

Reconocimiento de caras: La idea es utilizar el cambio de base planteado previamente, transformando cada imagen convenientemente. Además reduciremos la dimensión de los datos utilizando sólo algunas de las variables (eligiendo aquellas que capturan una fracción mayor de la varianza).

Procedimiento: Para reducir la dimensión, como dijimos, tomaremos k (que puede ser un parámetro arbitrario) componentes principales a considerar. Es decir, tomaremos $V = [v_1, v_2 \dots v_k]$
Luego, aplicaremos el cambio de base a cada muestra $x^{(i)}$ de la base de datos, definimos asi
$tc(x^i) = V^tx^i = (v_1^t, \dots, v_k^tx^i)$
Después, cuando se nos pasa una imágen x, a reconocer, la centramos con la media de las variables calculada previamente, y dividimos por la raiz de la cantidad de imagenes menos 1, es decir, $x^* = \frac{x-\mu}{\sqrt{n-1}}$.
Una vez calculado $tc(x*)$ (vector) consideramos cada $tc(x^{(i)})(i = 1 \dots n)$ con el fin de realizar la resta entre el primero y cada una de las transformaciones caracterisitcas de las imágenes de la base de datos por separado, a cada resultado le tomaremos la norma 2, para, así, determinar cual es la que presenta menor distancia (por definición de norma) entre sí. (es decir busco $min_{i,i=1\dots n}(||x^* - x^{(i)}||)$. Así, sea el mínimo i que verifique la menor distancia calculada por la norma, diremos que x_i es la imagen más cercana (parecida). Este x_i vamos a llamarlo el "vecino más cercano" a x^*. Esto lo decidimos así ya que obtendriamos la correspondencia de la imágen a reconocer con la más cercana de la base de datos en términos de distancias vectoriales; siempre teniendo en consideración que estamos tomando los rasgos, o sea las variables, más relevantes, es decir, los que presentan mayor varianza.
Antes de dar por terminada la sección de desarrollo, vamos a aclarar que en la etapa de desarrollo del TP, tuvimos varios traspiés. Uno de estos fue que el código con la implementación de todo el procedimiento que describimos más arriba tenía dos bugs: el primero, en algunas funciones trabajamos mal con las matrices en la función tc, utilizando filas de las mismas como si fueran columnas y viceversa. También, en la función producto, que calcula el producto entre una matriz y un vector, al multiplicar B*v, sólo multiplicaba las primeras v.size() filas de B por el vector v. Entonces, para el método 1, lo que nos sucedió fue que al multiplicar res = At*U, donde At es de 644x205 y U de 205*15, obteniamos en res una matriz de 205*15, en lugar de 644*15. Con lo cual, perdíamos datos y en consecuencia, la efectividad en el reconocimiento de caras era baja pero no tanto.
Como al principio no encontramos este bug, realizamos varios experimentos con ese código. Creemos que no es importante incluir estos experimentos pues los resultados no son ptimos, pero sí creemos que vale la pena mencionar lo sucedido puesto que haberlos hecho y ver que la tasa de aciertos podía ser de tan sólo del 50 por ciento nos hizo pensar en principio que este método no era tan bueno, lo cual nos llevó a investigar más sobre el mismo y convencernos que en realidad sí se podía alcanzar una mayor efectividad en el reconocimiento de caras. Y gracias a esto pudimos finalmente encontrar el bug y mejorar el código con la implementación del método descripto.
Damos por terminada la sección de desarrollo, pasaremos a dar indicaciones sobre como compilar y ejecutar nuestra implementación de la técnica en cuestión, y finalmente plantearemos hipótesis para los tests seguidas de sus respectivas discusiones y conclusiones.

Como compilar los archivos fuente y ejecutar tests: Para compilar los archivos fuente del TP, en Linux (Ubuntu), es necesario abrir una terminal (ctrl + alt + t), acceder a la carpeta donde se encuentran los archivos fuente input.cpp (mediante el comando 'cd 'ruta de la carpeta").
Luego, teclear el siguiente comando en la consola:

g++ input.cpp -o input

Una vez hecho esto, nos aparecerá en la misma carpeta donde se encuentra input.cpp, un archivo con nombre 'input', el cual ejecutaremos desde la línea de comandos para realizar los tests de la siguiente manera (este programa toma 3 parametros, los cuales se separan por un espacio cuando se los escribe en la linea de comandos, el primero es un archivo con informarción sobre los parámetros de entrada, el segundo es donde se guardarán los resultados, y el tercero, Option, es el número que selecciona el método para resolver el problema de encontrar los autovectores de la matriz de covarianzas, 0 para utilizar el Método 0 , 1 para usar el Método 1):

./input 'ruta de acceso del archivo .in,' 'ruta de acceso del archivo .out ' Option

Dicho archivo .in que se pasa como parámetro en el comando que aparece arriba tendrá información sobre k (cantidad de componentes principales a considerar), la cantidad de personas, de imágenes por personas en la base de datos, la cantidad de píxeles de las imágenes, las imágenes a utilizar de cada persona y, por último, las imágenes a reconocer mediante nuestra implementación.
Para poder levantar las imágenes y utilizarlas en nuestro código de una manera mas cómoda es necesario correr los siguientes scripts en Matlab que transformarán las imágenes en cuestión a archivos de texto con la forma, en su contenido, de matrices de números, donde, cada uno de ellos representan el valor de un píxel de la imágen a transformar. Los mismos pueden encontrarse en las carpetas ./data/ImagenesCaras y la carpeta ./data/ImagenesCarasRed; sus nombres respectivos son scriptB.m y scriptS.m. Una vez encontrados, es necesario abrir Matlab y posicionarse desde dicho programa en la carpeta donde se encuentran estos scripts, luego correrlos para realizar la transformación deseada. Una vez hecho esto, en cada carpeta de imágenes de los sujetos se podrá encontrar un archivo .in asociado para cada imágen transformada, el cual será interpretado de manera correcta por nuestro programa como la imágen a levantar de la base de datos.
Los resultados del reconocimiento de rostros serán devueltos por pantalla, es decir, por la consola.También se imprime por pantalla (y en el archivo .out, segundo parámetro que se pone en el comando de ejecución del programa) los k valores singulares calculados.

- Hipótesis:

.

Hipótesis del experimento 1 :
Para este experimento consideraremos 2 bases de datos (imágenesCaras e imágenes carasRed). En imágenesCaras tendremos imágenes de 41 sujetos distintos, con una resolución de imágen de 112x92 píxeles, mientras que la base imágenesCarasRed, tendrá las mismas imágenes que la anterior, pero con una resolución de 28x23 píxeles. También iremos variando los métodos utilizados, ya sea el método normal, o el alternativo. La idea de este experimento consiste en variar la cantidad de componentes principales a ser consideradas, y a la vez variar la cantidad de imágenes utilizada en la base de datos. Creemos que a mayor cantidad de componentes principales consideradas, la transformación característica tendrá mayor precisión y la tasa de efectividad será mayor, ya que el vector construido obtendrá mas rasgos de cada cara (pensamos que cada componente principal de la base de datos tiene información sobre los puntos de la matriz en los que hay mayor variación entre imágenes) y la diferencia entre el sujeto a identificar y el sujeto que le corresponda será menor. Con respecto a la cantidad de imágenes por sujeto en nuestra base, consideramos que tendremos mayor tasa de aciertos si tenemos más imágenes por persona, ya que al tener mas imágenes, la transformación característica de cada sujeto captará más rasgos de cada uno de ellos, o estaremos considerando los mismos rasgos de la persona, pero vistos de diferentes ángulos, lo cual es muy útil pues la imágen del sujeto a identificar puede estar tomada desde un ángulo disinto al que fueron tomadas las imágenes que tenemos en nuestra base de datos.
Finalmente creemos que el tiempo que tomarán los métodos es proporcional a la cantidad de imágenes a considerar.

Hipótesis del experimento 2 :
Para este experimento consideraremos una única base de datos (imágenesCarasRed),en la cual cada imágen tiene una resolución de 28x23 pixeles. En el experimento iremos variando las componentes tomadas, es decir el k, y trataremos de predecir como influye el aumento o disminución de este con respecto a la tasa de aciertos. También nos preguntaremos si existe un k óptimo para el cual la tasa de aciertos es máxima. Finalmente variaremos la cantidad de sujetos en nuestra base de datos , y observaremos como afectan estos cambios en los resultados obtenidos Para nosotros, cuanto más grande sea el k, la tasa de aciertos será mayor Es decir que el crecimiento o decrecimiento de la tasa de aciertos es proporcional a la cantidad de componentes principales utilizados para calcular la transformación característica. Creemos que el k óptimo, en principio, será el más grande que utilizemos en la experimentación, y que la tasa de aciertos variará con respecto a la cantidad de sujetos que tengamos en la base de datos, es decir que a menor cantidad de sujetos en nuestra base y mayor k, la tasa de aciertos será muy grande, mientras que con más sujetos y utilizando el mismo k,

la tasa será menor.

Hipótesis del experimento 3 :
Para este experimento consideraremos una base de datos: imagenesCaras. En imágenesCaras tendremos imágenes de 41 sujetos distintos, por otro lado, también su utilizarán 21 sujetos distintos, con una resolución de imágen de 112x92 píxeles. Aplicaremos unicamente el método 1 para el reconocimiento de las personas, ya que no será posible utilizar el 0 debido a lo mencionado en la aclaración hecha en el desarrollo.
El objetivo de este experimento es ver si existe una relación entre los parámetros que toma nuestra implementación. En principio, esperamos que las tasas de acierto sean similares a los de las imágenes de menor resolución pero no con el mismo k, ya que, al ir aumentando la cantidad de imágenes totales y además, considerando que las imágenes debido a sus dimensiones ahora poseerán mayor información a considerar para poder diferencirlas, dicho valor deberá ser mayor. Por lo que se espera ver una relación entre la cantidad de imágenes totales a considerar de la base por persona, la cantidad de personas de la misma y el valor de k utilizado para obtener la mejor tasa de aciertos. Aquí es donde analizaremos dicha relación entre parámetros.

3. Resultados

Resultados del experimento 1 :
Métodos: 0 y 1
K: 15
Base utilizada: Imágenes carasRed
Sujetos: 41

Cantidad Imágenes	Tasa de aciertos
10	100 %
8	95 %
5	83 %
1	61 %

Métodos: 0 y 1
K: 10
Base utilizada: Imágenes carasRed
Sujetos:41

Cantidad Imágenes	Tasa de aciertos
10	100 %
8	92 %
5	83 %
1	61 %

Métodos: 0 y 1
K: 5
Base utilizada: Imágenes carasRed
Sujetos:41

Cantidad Imágenes	Tasa de aciertos
10	100 %
8	75 %
5	70 %
1	49 %

Métodos: 0 y 1
K: 15
Base utilizada: Imágenes carasRed
Sujetos:20

Cantidad Imágenes	Tasa de aciertos
10	100 %
8	85 %
5	90 %
1	65 %

Métodos: 0 y 1
K: 10
Base utilizada: Imágenes carasRed
Sujetos:20

Cantidad Imágenes	Tasa de aciertos
10	100 %
8	90 %
5	90 %
1	65 %

Métodos: 0 y 1
K: 5
Base utilizada: Imágenes carasRed
Sujetos:20

Cantidad Imágenes	Tasa de aciertos
10	100 %
8	70 %
5	70 %
1	45 %

Comparación de tiempos:
Base utilizada: Imágenes carasRed
Sujetos:20
Imágenes por persona: 8

K	Método 0	Método 1
5	10s	0.7s
10	19s	1.3s
15	29s	2s

Base utilizada: Imágenes carasRed
Sujetos:20
Imagenes por persona: 5

K	Método 0	Método 1
5	9s	0.3s
10	19s	0.5s
15	28s	0.7s

Base utilizada: Imágenes carasRed
Sujetos:20
Imágenes por persona: 1

K	Método 0	Método 1
5	9s	0.01s
10	19s	0.03s
15	28s	0.04s

Base utilizada: Imágenes carasRed
Sujetos:41
Imágenes por persona: 8

K	Método 0	Método 1
5	41s	11s
10	82s	22s
15	123s	32s

Base utilizada: Imágenes carasRed
Sujetos:41
Imágenes por persona: 5

K	Método 0	Método 1
5	41s	4s
10	81s	8s
15	125s	12s

Base utilizada: Imágenes carasRed
Sujetos:41
Imágenes por persona: 1

K	Método 0	Método 1
5	40s	0.2s
10	80s	0.3s
15	121s	0.5s

Base utilizada: Imágenes Caras (únicamente utilizado el Método 1)
Sujetos: 41
Imágenes por persona: 2

K	Tiempo	Tasa de acierto
5	5,8s	51,2 %
10	6,3s	75 %
15	6,9s	80,4 %
15	7,1s	80,4 %

Sujetos: 41
Imágenes por persona: 5

K	Tiempo	Tasa de aciertos
5	41,1s	51,1 %
10	41,8s	75,6 %
15	42,8s	75,6 %
19	44s	80,4 %

Sujetos: 41
Imágenes por persona: 9

K	Tiempo	Tasa de aciertos
5	126,1s	82,9 %
10	139,6s	92,6 %
15	151s	90,2 %
19	157,1s	90,2 %

Sujetos: 20
Imágenes por persona: 2

K	Tiempo	Tasa de aciertos
5	1,48s	70 %
10	1,61s	85 %
15	1,82s	85 %
19	2,04s	85 %

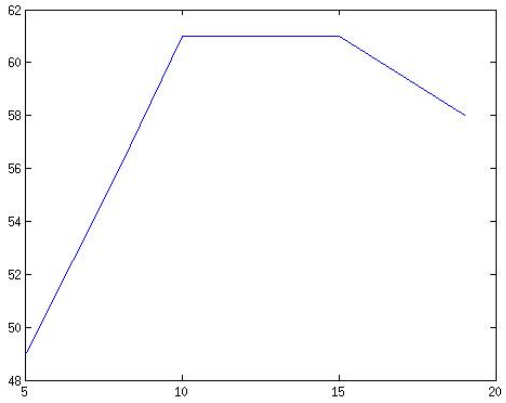
Sujetos: 20
Imágenes por persona: 5

K	Tiempo	Tasa de aciertos
5	8,6s	75 %
10	9,4s	95 %
15	10,1s	95 %
19	10,8s	95 %

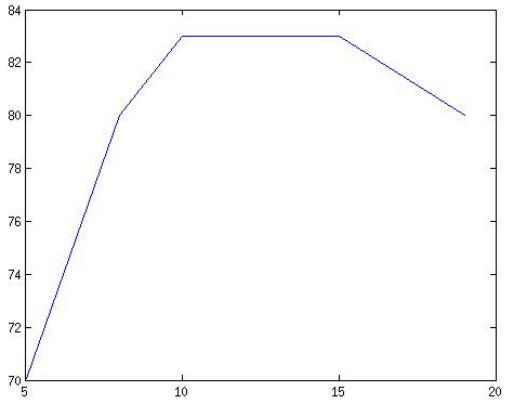
Sujetos: 20
Imágenes por persona: 9

K	Tiempo	Tasa de aciertos
5	30,2s	75 %
10	32,4s	85 %
15	34,2s	85 %
19	121s	85 %

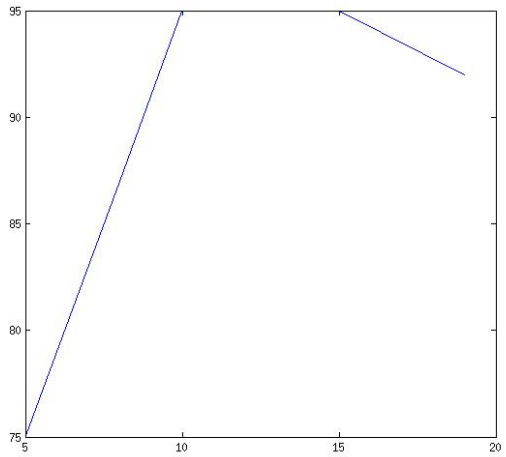
Resultados del experimento 2 :
41 Sujetos y k = 5,8,10,15,19. En el eje x estan representados los k y en el eje y la tasa de aciertos(en porcentaje)



Unica imagen considerada por persona

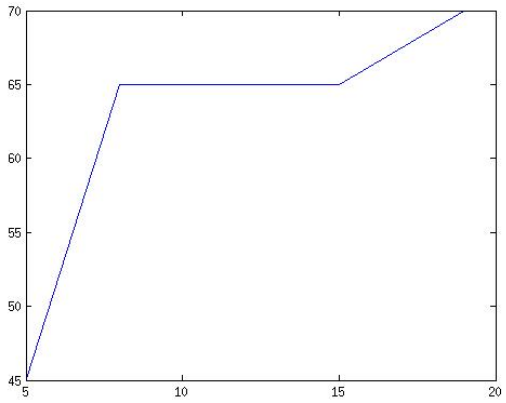


5 imagenes consideradas por persona

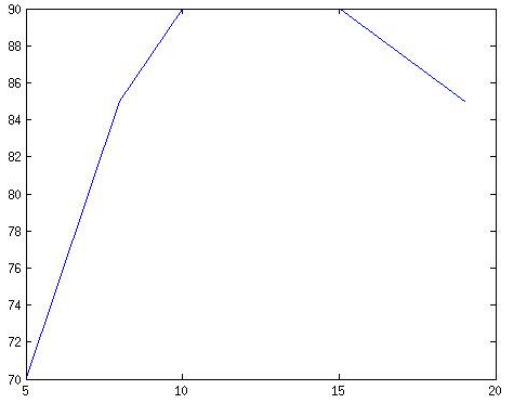


8 imagenes consideradas por persona

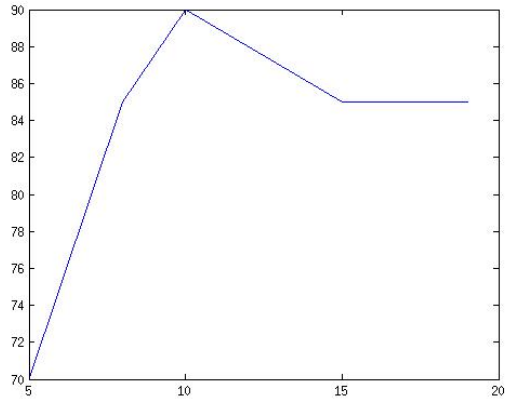
20 sujetos y k = 5,8,10,15,19. En el eje x estan representados los k y en el eje y la tasa de aciertos(en porcentaje)



Unica imagen considerada por persona



5 imagenes consideradas por persona

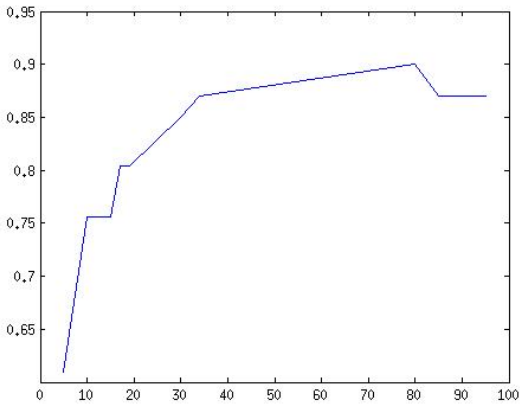


8 imagenes consideradas por persona

Resultados del experimento 3 :

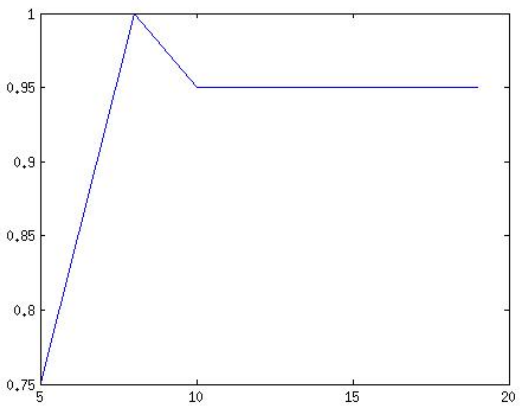
El eje horizontal de los gráficos representa el valor de k, el vertical indica la tasa de acierto del experimento. (Es decir, cuantas caras se lograron reconocer de la base de datos efectivamente).

41 sujetos, $k = 5, 10, 15, 17, 19, 30, 34, 80, 85, 90, 95$.



5 imagenes consideradas por persona

21 sujetos, $k = 5, 8, 10, 15, 19$.



5 imagenes consideradas por persona

4. Discusión

Sobre el experimento 1 Como esperabamos, el tiempo de ejecución del método 1 fue mucho menor que el del método 0. Sin embargo, la tasa de efectividad fue exactamente la misma en ambos casos. Es decir, el método 1 es mucho más efectivo que el método 0. Además, como era de esperar, a mayor cantidad de componentes principales consideradas, mayor era la tasa de efectividad. Con lo cual, esto nos da la pauta que las componentes principales de una imagen realmente capturan datos relevantes sobre los aspectos más distintivos de una imagen. Al menos en comparación con la habilidad de diferenciar una imagen de otra para el ojo humano. También se observa que hay una relación proporcional entre el k y el tiempo de cómputo, es decir a mayor k, mayor tiempo tarda el programa, ya sea tanto para el método 0 como para el método 1. Otro detalle observado en los resultados fue que cuantos menos sujetos tengamos en nuestra base, los tiempos serán menores, con ambos métodos. Particularmente notamos que los tiempos de cómputo del método 0 no variaban demasiado, dependiendo de cuantas imagenes por persona tengamos en la base, mientras que con el método 1 se observo una diferencia de tiempos notable. Esto nos llamó considerablemente la atención, puesto que al usar una menor cantidad de imágenes por persona, en principio, uno pensaría que se tarda menos tiempos en trabajar con las mismas. Sin embargo, después de pensarlo un poco, nos dimos cuenta que en el método 0 se trabaja siempre con matrices de tamaño 644x644 (donde 644 es la cantidad de variables de cada imagen) y por eso, al contrario de lo que sucede en el método 1, en el método 0 el tamaño de la matriz con la cual trabajamos es siempre igual, independientemente de la cantidad de imágenes que coloquemos en nuestra base de datos. Pensamos que es esta una de las razonas por las cuales el método 1 va a ser mucho más óptimo y efectivo que el 0.

Sobre el experimento 2 : En el primer gráfico se ve claro que hay un valor de k en el cual la efectividad en el reconocimiento de caras se hace máxima, y a partir de ese valor, por más que uno incremente el valor de k, la efectividad es siempre la misma hasta cierto valor de k, a partir del cual la efectividad comienza a bajar. Esto nos hace pensar: ¿Y si aumentamos la cantidad de imágenes consideradas por persona, incrementará la efectividad? En efeto, en los gráficos subsiguientes se ve que si aumentamos la cantidad de imágenes de la base de datos para una misma cantidad de componentes principales consideradas, la efectividad en el reconocimiento aumenta considerablemente. ¿Qué nos dice el hecho de que para un cierto valor de k la efectividad se estanque cuando hay pocas imágenes y todavía no se haya estancado si hay muchas? Esto se ve que sucede al comparar las primeras 3 imágenes que aparecen en la sección Resultados del experimento 2. Creemos que puede deberse a que para el conjunto de imágenes que tenemos en la base de datos, hay una cierta cantidad k' de componentes principales que nos las permite diferenciar a todas. Luego, si utilizamos más componentes principales que ese valor óptimo de k, los resultados van a seguir siendo los mismos pues ya las primeras k' componentes principales van a haber diferenciado todas las imágenes, y las subsiguientes no aportarán ningún dato relevante de la imagen. En los gráficos de 41 sujetos 1 y 2, la efectividad se estanca porque por más, que logramos diferenciar lo máximo posible las 5 imágenes que tenemos de cada persona, esto no alcanza para matchear alguna de estas imágenes a la imagen que nos pasan para ser testeada, pues esta no se parece lo suficientemente a ninguna de estas cinco imágenes. Además, para valores de k mayores que 15, la efectividad disminuye. Luego, si incrementamos la cantidad de imágenes de la base de datos (gráfico 3 con 41 personas) sí obtenemos mejores resultados y la efectividad no se estanca para valores tan pequeños de k como sucedía en los casos 1 y 2. Esto probablemente sea porque al tener más imágenes, la cantidad de rasgos distintivos a diferenciar es mayor (porque hay más imágenes), con lo cual para capturarlos a todos se necesita una cantidad mayor de componentes principales.
Algo muy destacable, que sucede en todos los ejemplos es que siempre hay un intervalo de valores para k en el cual la efectividad del método se hace máxima y luego baja considerablemente. Creemos que esto podría deberse seguramente a que siempre alguna de las imágenes de la base de datos tiene algun rasgo muy disinto que las otras (por ejemplo, podría suceder que en alguna imagen de la base de datos el sujeto estuviera con anteojos puestoso con gorra) y esto hace que cuando tomamos un k lo suficientemente grande como para tener en consideración este rasgo, al estar considerando pocas imágenes y no tener una mayor cantidad de rasgos en los cuales la imagen a testear y las de la base de datos pudieran coincidir, la identificación fallara, y se matcheara al sujeto con el equivocado.

Pensamos que probablemente, este efecto debería disminuir al aumentar la cantidad de imágenes de la base de datos.Sin embargo, cuando consideramos bases de datos con mayor cantidad de imágenes, esto siguió sucediendo sólo que en un rango de efectividad mayor. Es decir que a partir de un valor de k, la efectividad comenzó a bajar, con la misma pendiente que en todos los casos, pero la efectividad en esos casos nos daba mayor que antes. Esto pordría querer decir que al considerar demasiadas componentes principales, empezamos a tomar en cuenta más rasgos de los rostros de los que son deseables. Por otro lado, si para un intervalo la efectividad se mantiene constante y luego comienza a subir(como en el primer gráfico con 20 sujetos), entonces quiere decir que considerar más componentes principales no iba a ser nocivo sino que iba a mejorar la efetividad y probablemente habría que utilizar un k mucho mayor para lograr identificar ese k=k' máximo, en el cual la efectividad se estancara. Con lo cual, hay casos en los cuales considerar más componentes principales después del estancamiento es recomendable pues puede suceder que el tras esa fase de estancamiento, la efectividad siga subiendo antes de llegar a otra meseta y finalmente (como sucede en casi todos los tests que hicimos), comenzar a decrecer.

Sobre el Experimento 3 :
Como se puede observar en el primer gráfico, el valor de k es elevado en comparación a los utilizados para obtener, en los demás experimentos, la tasa de acierto más óptima. Creemos que mientras más personas haya en la base de datos, y mientras menos imágenes por persona haya en dicha base, será necesario tomar una cantidad de componentes principales mayor, ya que, por tener pocas imágenes por persona, será requerido distinguir una mayor cantidad de rasgos entre cada imágen, es decir, componentes principales; esta es la razón por la cual creemos que con $k = 80$ se obtuvo la mejor tasa de aciertos. Si hubieran muchas más imágenes que nos brinden más información relevante sobre una persona en particular, creemos que no será necesario un k tan grande, ya que con pocas componentes principales seriamos capaces de distinguir en ese caso a las personas de la base de datos. Se puede ver, en el segundo gráfico de este experimento en cuestión, que no hizo falta tomar un k tan elevado para obtener la máxima tasa de aciertos, es decir, con $k = 8$ se consiguió reconocer efectivamente todos los rostros. Tampoco hay muchas imágenes por persona por lo que podemos concluir lo siguiente: esto nos indica que, mientras menos personas haya en la base de datos, menos imágenes van a ser necesarias para obtener una tasa de acierto elevada considerando un valor de k pequeño.
Una aclaración final sobre esta discusión que hay que hacer, es que el comportamiento de la tasa de acierto en relación con la variación de k se ve afectada, de forma proporcional, de igual manera que como se vió en experimentos anteriores. Es decir, siempre existe un k ligado a la cantidad de imágenes y cantidad de personas, con el cual se obtiene la máxima tasa de acierto. Para todos valor menor que dicho k óptimo, la tasa de aciertos será menor y, para k mayores (es decir, habiendo tomado más rasgos de los necesarios, obteniendo así componentes principales de menor peso que desbalancearán el peso que ejercían las primeras componentes que si capturaban la mayor covarianza entre variables) ocurrirá que, de a poco, se reduzca la tasa de acierto hasta llegar a un punto donde las componentes prinicpales que se tomen luego no ejerzan ningún tipo de peso, en consecuencia, ningún tipo de variación en la tasa de acierto manteniendo la tasa de aciertos constantes, como ocurre en los experimentos del informe.

5. Conclusiones

Conclusión del experimento 1 : Podemos concluir que, para las imágenes de tamaño reducido, los métodos 0 y 1 tienen exactamente la misma tasa de aciertos para cualquier base y para cualquier imagen a testear. Además, el método 0 tarda un tiempo mucho mayor en ejecutarse. Además, pudimos concluir que si dejamos fija la cantidad de imágenes por persona y aumentamos la cantidad de sujetos en nuestra base, el tiempo de cómputo para cualquiera de los dos métodos será mayor y la tasa de aciertos menor. También, observamos que en el método 0, cuando se va aumentando el valor de k, el tiempo de cómputo va aumentando enormemente mientras que en el método 1, por más que aumentemos mucho el valor de k, el tiempo de cómputo se mantiene masomenos constante.(menos de 1 segundo, para las imágenes de dimensión pequeña).
Por último, en lo concerniente a tiempos de cómputo, se puede concluir que en el método 1, cuanto menor es la cantidad de imágenes utilizadas en la base de datos, menor es el tiempo de ejecución del programa. Sin embargo, en el método 0, no importa la cantidad de imágenes por persona en la base de datos, el tiempo de ejecución siempre es mayor que el del método 1, con lo cual en principio podemos pensar que siempre que se quiere ejecutar este programa, podemos decir que lo más recomendable es usar el método 1, que tiene la misma efectividad que el método 0 pero tiene menor tiempo de cómputo (pues trabaja con una matriz más pequeña).
Para el caso de las imágenes de mayor tamaño, el método varia sus tiempos de ejecución mientras más imágenes de la base de datos sean considerados para el experimento. También aumenta el tiempo de cómputo mientras más valores singulares sean cálculados. Este método es el unico posible a aplicar en las imágenes de gran dimensión, no se muestran tiempos de estos experimentos con el Método 0 ya que no finalizan en tiempo razonbale, por realizar cuentas sobre matrices muy grandes.

Conclusión del experimento 2 : Una conclusión básica y muy importante es que siempre hay un conjunto de valores para k que es óptimo para el ojo humano (que hace que los resultados obtenidos se correspondan a lo que nosotros esperaríamos obtener al analizar a ojo las imágenes). Es decir: quizás si estuviéramos utilizando este método para diferenciar alguna característica de algún órgano interno de una persona, o una huella digital, querríamos considerar un k mucho mayor, para ser capaces de diferenciar con mayor precisión los rasgos del mismo.
Otra conclusión importante, es que cuantas más imágenes tengamos en nuestra base de datos, mayor será la efectividad en la identificación de rostros, pues tendremos más caracterśticas de cada rostro, lo que hará que todo par de rostros sea más diferenciable a la hora de aplicar el método utilizado en el TP.

Conclusión del experimento 3 Una conclusión, nombrada en la discusión, es que hay una dependencia entre el k, la cantidad de personas distintas de la base de datos, y la cantidad de imágenes por persona a tomar en consideración de dicha base. Dicha dependencia obliga, a la hora de implementar este sistema de reconocimiento de caras, pensar si está dispuesto a tomar un mayor tiempo de cómputo a la hora de calcular más componentes principales para resolver el problema. O bien siempre mantener una cantidad proporcional de imágenes por persona dependiendo de la cantidad total de estas últimas. Finalmente concluimos que siempre hay un k óptimo para el cual se obtiene, dada una cantidad determinada de personas y de fotografías asociadas a estas, la mayor tasa de acierto.

Conclusión Final del Informe Como conclusión, podemos sacar que el método 1 es tan efectivo como el 0 a la hora de la identificación de rostros, pero que el método 1 es mucho más veloz ya que trabaja con matrices más pequeñas, y a partir de estas obtiene exactamente las mismas componentes principales. Esto quiere decir que para realizar el análisis de una imagen, no siempre es necesario trabajar con todos los puntos de la imagen sino que muchas veces es necesario (o al menos conveniente) encontrar una manera alternativa de obtener los datos de una imagen, a partir de un subconjunto de los datos que tenemos en la base de datos y no de todos, ya que muchas veces el tiempo computacional de trabajar con una cantidad tan grande de datos es inaceptable.
También podemos concluir que las matemáticas (en este caso el álgebra lineal) pueden contribuir y de hecho contribuyen increíblemente en la computación y en los cálculos computacionales. Es decir, matemática y computación van de la mano. Puesto que si no hubieramos tenido los conocimientos

suficientes de álgebra lineal para darnos cuenta de que para obtener las componentes principales de la matriz A era suficiente con obtener los autovectores de AAt y no los de AtA, nunca hubiéramos sido capaces de mejorar el tiempo de ejecución de nuestro algoritmo identificador de rostros, tornando el método de análisis de imágenes por medio de las componentes principales en inviable.
Algo interesante para realizar a futuro sería implementar este mismo modelo para otro tipo de imágenes, como por ejemplo de paisajes o huellas digitales y ver si los resultados son tan buenos como para el análisis de caras que realizamos en este TP. También sería interesante probar con imágenes de otros tamaños y ver si la efectividad así como los resultados son similares a los aquí obtenidos.

6. Apéndices

6.1. Enunciado

Laboratorio de Métodos Numéricos - Primer Cuatrimestre 2014
Trabajo Práctico Número 2: Tu cara me suena

Introducción

A menos de cincuenta días del inicio de la gran fiesta del fútbol mundial, el Equipo de Desarrollos de Métodos Numéricos ha recibido un pedido reciente por parte del Comite de Seguridad de la Organización del Mundial Brasil 2014. Informaciones de último momento aseguran que una gran cantidad de *barras* se están organizando para *copar* las tribunas de los (aún no terminados) estadios del país organizador. Debido al retraso en la implementación del sistema FIFA Plus por cuestiones técnicas, el evento será cancelado a menos que nuestro equipo logre desarrollar un software de reconocimiento facial efectivo en los próximos 20 días.

El Comité Organizador ha puesto a nuestra disposición todos sus recursos a fin de evitar lo que sería la mayor vergüenza en la historia de la entidad. Entre otras cosas, se dispone de una base de datos de fotografías digitalizada de todas aquellas personas que alguna vez hayan ingresado a algún estadio de fútbol en algún país del planeta. El objetivo del trabajo es desarrollar un software que, tomando esta información como entrenamiento, pueda detectar en tiempo real si los asistentes a alguno de los partidos corresponde a un *barra*.

Como instancias de entrenamiento, se tiene un conjunto de N personas, cada una de ellas con M imágenes (eventualmente) distintas sus rostros en escala de grises del mismo tamaño y resolución. Cada una de estas imágenes sabemos a qué persona corresponde.

Para $i = 1, \ldots, n$, sea $x_i \in \mathbb{R}^m$ la i-ésima imagen de nuestra base de datos almacenada por filas en un vector, y sea $\mu = (x_1 + \ldots + x_n)/n$ el promedio de las imágenes. Definimos $X \in \mathbb{R}^{n \times m}$ como la matriz que contiene en la i-ésima fila al vector $(x_i - \mu)^t/\sqrt{n-1}$, y

$$A = U \Sigma V^t$$

a su descomposición en valores singulares, con $U \in \mathbb{R}^{n \times n}$ y $V \in \mathbb{R}^{m \times m}$ matrices ortogonales, y $\Sigma \in \mathbb{R}^{n \times m}$ la matriz diagonal conteniendo en la posición (i, i) al i-ésimo valor singular σ_i. Siendo v_i la columna i de V, definimos para $i = 1, \ldots, n$ la *transformación característica* del dígito x_i como el vector $\mathbf{tc}(x_i) = (v_1^t\, x_i, v_2^t\, x_i, \ldots, v_k^t\, x_i) \in \mathbb{R}^k$, donde $k \in \{1, \ldots, m\}$ es un parámetro de la implementación. Este proceso corresponde a extraer las k primeras *componentes principales* de cada imagen. La intención es que $\mathbf{tc}(x_i)$ resuma la información más relevante de la imagen, descartando los detalles o las zonas que no aportan rasgos distintivos.

Dada una nueva imagen x de una cara, que no se encuentra en el conjunto inicial de imágenes de entrenamiento, el problema de reconocimiento consiste en determinar a qué persona de la base de datos corresponde. Para esto, se calcula $\mathbf{tc}(x)$ y se compara con $\mathbf{tc}(x_i)$, para $i = 1, \ldots, n$, utilizando un criterio de selección adecuado.

Enunciado

Se pide implementar un programa en `C` o `C++` que lea desde archivos las imágenes de entrenamiento correspondientes a distintas personas y que, utilizando la descomposición en valores singulares y el número de componentes principales k mencionado anteriormente, calcule la transformación característica de acuerdo con la descripción anterior. Se debe proponer e implementar al menos un método que, dada una nueva imagen de una cara, determine a que persona de la base de datos corresponde utilizando la transformación característica.

Con el objetivo de obtener la descomposición en valores singulares, se deberá implementar el método de la potencia con deflación para la estimación de autovalores/autovectores. En este contexto, la

factibilidad de aplicar este método es particularmente sensible al tamaño de las imágenes de la base de datos. Por ejemplo, considerar imágenes en escala de grises de 100×100 píxeles implicaría trabajar con matrices de tamaño 10000×10000. Una alternativa es reducir el tamaño de las imágenes, por ejemplo, mediante un submuestreo. Sin embargo, es posible superar esta dificultad en los casos donde el número de muestras es menor que el número de variables. Se pide desarrollar las siguientes sugerencias y fundamentar como utilizarlas en el contexto del trabajo.

- Dada una matriz y su descomposición en valores singulares $A = U\Sigma V^t$, encontrar la descomposición en valores singulares de A^t. Cómo se relacionan los valores singulares de A y A^t?
- Dada la descomposición en valores singulares de A, expresar en función de U, Σ y V las matrices A^t, A^tA y AA^t. Analizar el tamaño de cada una de ellas y deducir como relacionar las respectivas componentes principales. Combinar con el item anterior para el cómputo de los componentes principales.

En base a este análisis, se pide desarrollar una herramienta alternativa que permita trabajar bajo ciertas condiciones con imágenes de tamaño mediano/grande.

Junto con este enunciado se provee una base de datos de imágenes correspondiente a 41 personas, con 10 imágenes por cada una de ellas. Esta base de datos se encuentra disponible en dos resoluciones distintas: 92×112 y 23×28 píxeles por cada imagen. La segunda corresponde a un submuestreo de la base original. En relación a la experimentación, se pide como mínimo realizar los siguientes experimentos:

- Analizar para cada una de las variantes qué versión de la base de datos es posible utilizar, en base a requerimientos de memoria y tiempo de cómputo.
- Para cada una de las variantes propuestas, analizar el impacto en la tasa de efectividad del algoritmo de reconocimiento al variar la cantidad de componentes principales considerados. Estudiar también como impacta la cantidad de imágenes consideradas para cada persona en la etapa de entrenamiento.
- En caso de considerar más de una posibilidad para determinar a que persona corresponde una nueva cara, considerar para cada una la mejor configuración de parámetros y compararlas entre ellas.

El objetivo final de la experimentación es proponer una configuración de parármetros/métodos que obtenga resultados un buen balance entre la tasa de efectividad de reconocimiento de caras, la factibilidad de la propuesta y el tiempo de cómputo requerido.

Programa y formato de archivos

Se deberán entregar los archivos fuentes que contengan la resolución del trabajo práctico. El ejecutable tomará tres parámetros por línea de comando, que serán el archivo de entrada, el archivo de salida, y el método a ejectutar (0 método estándar, 1 método alternativo).

Asumimos que la base de datos de imágenes se encuentra organizada de la siguiente forma: partiendo de un directorio raíz, contiene un subdirectorio por cada una de las personas donde se encuentran las imágenes de entrenamiento. El nombre de las imágenes será un número y su extensión una archivo `.pgm` (ejemplo: `1.pgm`, `2.pgm`, etc.). De esta forma se puede saber a que persona corresponde cada imagen simulando un etiquetado de las mismas.

Para facilitar la experimentación, el archivo de entrada con la descripción del experimento sigue la siguiente estructura:

- La primera línea contendrá el *path* al directorio que contiene la base de datos, seguido de 5 números enteros que representan la cantidad de filas y columnas de las imágenes de la base,

cuantas personas (p) y cuantas imágenes por cada una de ellas ($nimgp$), y cuántas componentes principales se utilizarán en el experimento (k). A continuación de muestra un ejemplo de una base de datos de imágenes de 112×92, con 41 sujetos, 5 imágenes por sujeto y tomando 15 componentes principales.

```
../data/ImagenesCaras/ 112 92 41 5 15
```

- A continuación, el archivo contendrá p líneas donde en cada una de ellas se especificará la carpeta correspondiente a la p-ésima persona, seguido de $nimgp$ numeros enteros indicando las imágenes a considerar para el entrenamiento. La siguiente línea muestra como ejemplo 5 imágenes (2, 4, 7, 8 y 10) a considerar para la persona `s10`.

  ```
  s10/ 2 4 7 8 10
  ```

- Finalmente, se especificará un número $ntest$ de imágenes (preferentemente no contenidas en la base de imágenes) para las cuales se desea identificar a quien pertenecen. Cada una de ellas se especificará en una nueva línea, indicando el *path* al archivo seguido del número de individuo al que pertenece (relativo a la numeración establecida en el punto anterior). La siguiente línea muestra un ejempo de una imagen a testear para el sujeto número 1:

  ```
  ../data/ImagenesCaras/s1/9.pgm 1
  ```

El archivo de salida obligatorio tendrá el vector solución con los k valores singulares de mayor magnitud, con una componente del mismo por línea. Adicionalmente, en caso de considerarlo conveniente, se podrán agregar archivos adicionales opcionales con más información que la descripta en la presente sección.

Junto con el presente enunciado, se adjunta una serie de scripts hechos en `python` y un conjunto instancias de test que deberán ser utilizados para la compilación y un testeo básico de la implementación. Se recomienda leer el archivo `README.txt` con el detalle sobre su utilización.

Fecha de entrega

- *Formato electrónico:* Jueves 15 de Mayo de 2014, hasta las 23:59 hs., enviando el trabajo (informe+código) a `metnum.lab@gmail.com`. El subject del email debe comenzar con el texto `[TP2]` seguido de la lista de apellidos de los integrantes del grupo.
- *Formato físico:* Viernes 16 de Mayo de 2014, de 17:30 a 18:00 hs.

7. Referencias

-R. Burden y J.D.Faires, Análisis numérico, International Thomson Editors, 1998
-D. Watkins, Fundamentals of matrix computations, John Wiley & Sons, 1991
-http://people.maths.ox.ac.uk/richardsonm/SignalProcPCA.pdf