Métodos Numéricos Avanzados
Trabajo Práctico Nº1

"Facial Recognition"

3 de Octubre de 2019

Comisión: S

Grupo 5

Ferrer Tomás 57207
Lo Coco Juan Pablo 57313
Lund Marcos 57159
Princ Guido 57334
Zuberbuhler Ximena 57287

Índice

Resumen

El objetivo de este trabajo práctico es desarrollar un sistema informático capaz de reconocer e identificar caras dada una imagen mediante la implementación de dos técnicas distintas (Análisis de componentes principales y análisis de componentes principales con kernels).

Palabras Claves

PCA: Análisis de Componentes Principales (por sus siglas en inglés)
KPCA: Análisis de Componentes Principales con Kernels (por sus siglas en inglés)
Eigenface: autocara
SVM: Máquina de Soporte Vectorial (por sus siglas en inglés)

Introducción

Durante el siglo XX se han desarrollado distintas técnicas con la finalidad de automatizar el reconocimiento facial, comenzando por las mediciones de Bledsoe en la década de los 60. Las primeras metodologías implicaban complejas mediciones manuales, y, debido a las limitaciones técnicas de la época, resultaban inaccesibles para la gran mayoría de las personas. A fines de los años 80, sin embargo, Sirovich y Kirby dieron origen al enfoque Eigenface mediante la aplicación de álgebra lineal, consiguiendo describir y codificar un rostro de manera mucho más simple y abreviada, basándose sólo en las características más generales del mismo.

Este informe está basado en el enfoque de las autocaras (eigenfaces) y pretende dar como resultado un sistema capaz no sólo de reconocer rostros, según los datos ingresados, sino también de identificarlos.

A continuación se describirá detalladamente la metodología utilizada para el desarrollo del sistema, así como las pruebas realizadas para determinar el funcionamiento del mismo. Luego se analizarán los resultados de dichas pruebas, y finalmente se desarrollarán conclusiones basadas en los resultados obtenidos.

Metodología

Desarrollo

Para el desarrollo del sistema se partió de los scripts implementados en Python provistos por la cátedra. Se utilizó la distribución "Anaconda" de Python (en su versión 3.7) debido a su portabilidad y a la fácil integración con las principales librerías matemáticas.

En primer lugar, se desarrolló una implementación del algoritmo de descomposición QR (mediante el uso de Gram-Schmidt) con el fin de obtener autovalores y autovectores, utilizados tanto en el script de PCA como en el de KPCA. Una vez implementado el algoritmo se procedió a la integración con los códigos, y posterior testeo de ambos códigos.

En segundo lugar se usó la herramienta OpenCV para utilizar la cámara de la computadora personal y reconocer automáticamente rostros que pudieran aparecer en pantalla, según lo implementado en el archivo main.py. Se itera continuamente aplicando dibujos de rectángulos sobre los rostros reconocidos en los distintos instantes, pudiendo el usuario ordenar al sistema que ejecute el procedimiento de reconocimiento de los mismos pulsando la tecla 'R'.

A continuación se envía al algoritmo PCA o KPCA, según correspondiera, cada uno de los rostros como archivos .pgm (del mismo tamaño de las imágenes del conjunto de datos público utilizado en las pruebas, 92x112 píxeles). Al ya tener disponibles las autocaras obtenidas del entrenamiento, se proyecta la imagen sobre el subespacio generado por las mismas y, mediante el uso de la librería sklearn y utilizando un SVM que realiza un aprendizaje supervisado, se clasifica la entrada con la etiqueta correspondiente a la persona a la que fue relacionada. Los nombres correspondientes a cada persona se detallan en el archivo name_db.txt dentro del directorio att_faces.

Cabe destacar que se transformó cada imagen a un vector de tamaño (92x112) para poder representar las mismas en columnas y filas de matrices y luego poder trabajar sobre ellas, siendo el valor de cada píxel el analizado en el desarrollo del algoritmo de componentes principales para cada muestra.

Por último, se imprime en pantalla dicha etiqueta, congelando la imagen por un breve instante para poder apreciar los resultados. Para salir del programa se debe presionar la tecla 'E'.

Implementación y Modelo Matemático

Para la implementación del método de descomposición QR, se aplicó el siguiente algoritmo de Gram-Schmidt modificado:

$\mathbf{R}_1 = \mathbf{0}$
for $k = 1 \rightarrow n$ **do**
  $\mathbf{R}_1(k,k) = \|\mathbf{A}(1:m,k)\|_2$
  $\mathbf{Q}_1(1:m,k) = \frac{\mathbf{A}(1:m,k)}{\mathbf{R}_1(k,k)}$
  for $j = k+1 \rightarrow n$ **do**
    $\mathbf{R}_1(k,j) = \mathbf{Q}_1(1:m,k)^T \mathbf{A}(1:m,j)$
    $\mathbf{A}(1:m,j) = \mathbf{A}(1:m,j) - \mathbf{Q}_1(1:m,k)\mathbf{R}_1(k,j)$
  end for
end for

Figura 1: Pseudocódigo de Gram-Schmidt modificado

Este algoritmo es una aproximación en código del método visto en clase, que garantiza que no ocurran errores de aproximación que luego son arrastrados y amplificados en los pasos siguientes, como sí ocurre en el método original. Dado un epsilon, se itera tantas veces como sea necesario hasta que las diferencias de los valores en la diagonal de R entre un paso y el siguiente cumplan la cota. Dichos elementos componen los autovalores de la matriz A, por lo que se asegura de esta manera obtener autovectores precisos. Asimismo, para obtener los autovectores se calculó el producto de Q_i para todos los pasos.

Los autovectores obtenidos se corresponden a las distintas autocaras, las componentes principales del conjunto de entrenamiento utilizado. En PCA, para no trabajar con dimensiones del orden de $1x10^5$ se calcularon los autovectores de una matriz más pequeña, para luego calcular los autovectores de la matriz de imágenes, al estar relacionados, según lo indicado en [1]. Para el caso de KPCA, se utilizó un kernel polinomial de grado 2 para correlacionar píxeles en órdenes elevados en el cálculo de las componentes principales, según lo hecho en [2] y [3].

Pruebas

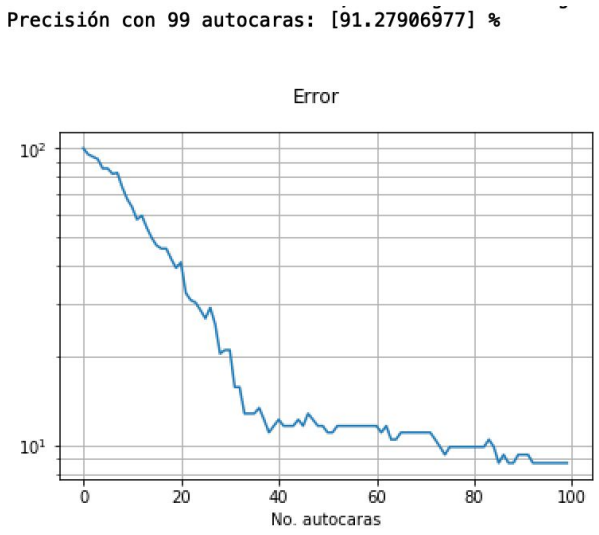
Para probar ambos scripts se utilizó un conjunto de datos público [6] que consiste en diez fotos de los rostros de cuarenta personas distintas en formato PGM (en escala de grises), con una resolución de 92x112. Cada foto se distingue por variaciones gestuales, e incluso por variaciones en los ángulos de captura. Seis de las fotos de cada persona se usaron para entrenamiento en PCA, y nueve para entrenamiento en KPCA.

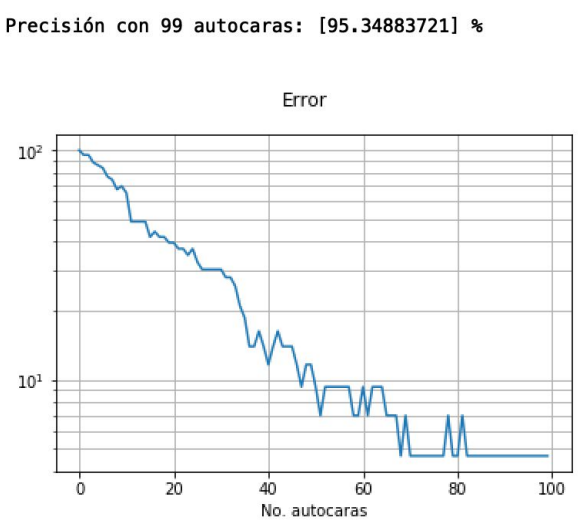
En primer lugar, se varió la cantidad de autocaras utilizadas para la evaluación posterior al entrenamiento, tomando valores entre uno y cien, y observando la precisión alcanzada obtenida con el conjunto de testeo.

La segunda prueba realizada consistió en variar la condición de corte de la ejecución de la descomposición QR (nótese la ejecución cíclica de este código). En todos los casos se comparó cada valor de la diagonal de la matriz R obtenida de la descomposición QR con cada valor de la diagonal de la matriz R obtenida en la iteración anterior; se varió el valor de la tolerancia para la diferencia entre los correspondientes pares de valores de las diagonales.

Para la prueba final se extendió el conjunto de datos de entrenamiento, añadiéndose fotos de los autores de este documento en el mismo formato y con la misma resolución para mantener la consistencia.

Resultados





Figuras 2 y 3. Gráficos al probar detectar 100 autocaras. (PCA a la izquierda. KPCA a la derecha)

En cuanto al número de autocaras utilizadas para la fase de evaluación, se fijó el número deseable en cien autocaras, al observarse una caída constante y cierta estabilidad hacia el final (en las Figuras 2 y 3), cercanos a ese valor. En base a las pruebas mencionadas anteriormente, se comprobó que KPCA provee mejores resultados que PCA, aunque el tiempo de entrenamiento fue mayor. Se fijó el valor de la tolerancia para el cálculo de autovectores (error) en $1x10^{-2}$ ya que la diferencia entre este valor y menores valores no resultó apreciable.

Figura 4. Ejemplo de imagen con reconocimiento

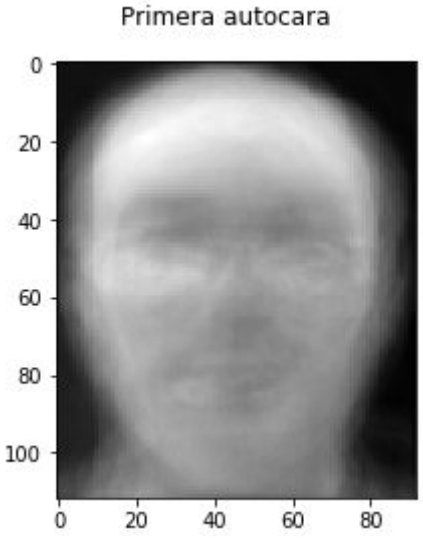


Figura 5: Primera autocara obtenida al ejecutar PCA

Conclusiones

- Al mantenerse constante la precisión del testing en un punto cercano a las 100 autocaras utilizadas, y con valores aceptables superiores al 90%, se tomó dicha configuración como la establecida.
- No se observaron cambios significativos al variar el epsilon utilizado en el método de Gram-Schmidt. Por lo tanto, se eligió un valor de orden de magnitud $1x10^{-2}$ para evitar un excesivo uso de memoria y tiempo, en la obtención de los autovalores y autovectores.
- El uso de SVM resultó ser muy eficiente con el uso de espacios de gran dimensionalidad, como el que fue generado por las autocaras, incluso trabajando en el entrenamiento con conjuntos de imágenes muy similares y de baja resolución.
- Para ciertos casos, la detección resultaba ser más errónea por los espacios vacíos que habían al tomar la imagen con la cámara (por ejemplo cuando uno se coloca de perfil, el rectángulo que captura la imagen toma un gran espacio vacío, o con fondos no blancos). Esto se podría mejorar a futuro con algoritmos de detección de rostros más avanzados, como los expuestos en la bibliografía.
- Se podría agregar funcionalidad a futuro para permitir agregar más personas a la base de datos y realizar un nuevo training, ya que esos parámetros permanecen fijos con la configuración actual.
- La detección falla cuando parte de la imagen dentro del rectángulo que se usa de muestra no se encuentra al alcance de la cámara. Esto sucede porque parte del rostro de la persona se encuentra visible. Se podría solucionar al mejorar la lógica de la obtención de la imagen en los límites.

Bibliografía

[1] Matthew Turk and Alex Pentland. Eigenfaces for recognition. Journal of Cognitive Neuroscience, 3(1):71–86, 1991. PMID: 23964806.
[2] Kwang In Kim, Keechul Jung, and Hang Joon Kim. Face recognition using kernel principal component analysis. IEEE Signal Processing Letters, 9(2):40–42, Feb 2002.
[3] Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. Nonlinear component analysis as a kernel eigenvalue problem. Technical report, MaxPlanck-Institut für biologische Kybernetik, 1996.
[4] Apunte de Descomposición QR provisto por la cátedra.
[5] medium.com/@spot_blog/una-breve-historia-del-reconocimiento-facial-vision-blog
[6] cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html