

TP1: Reconocimiento Facial

Grupo VII

Eugenio Damm

Agustín Calatayud

Qué es el Reconocimiento Facial?

Es una aplicación dirigida por una computadora que identifica automáticamente a una persona en una imagen digital.

Esto se lleva a cabo mediante el análisis de las características faciales del sujeto y comparandolas con una base de datos.

Abarca diversas disciplinas como procesamiento de imágenes, reconocimiento de patrones y redes neuronales entre otras.

Se utiliza principalmente en sistemas de seguridad para el reconocimiento de usuarios.

En qué se basa nuestro programa?

Análisis de las componentes principales para conseguir las direcciones de mayor varianza.

Búsqueda de autovalores y autovectores.

Obtención de Right Singular Vectors para manejar un número menor de datos.

Proyección sobre las autocaras teniendo así un volumen menor de datos en la base de datos, para la posterior comparación.

Descomposición QR

Para la descomposición QR de nuestras imágenes utilizamos dos métodos distintos:

- Gram-Schmidt
- Householder

Obtención de Autovalores y Autovectores

Para obtener los autovalores y autovectores utilizamos el método iterativo QR.

Extraemos los autovalores de la diagonal de la matriz “T” triangular superior.

Los mismos están ordenados de manera descendente.

Extraemos los autovectores de las columnas del producto de las matrices “Q”.

Obtención de Right Singular Vectors

Para manejar un tamaño menor de datos se obtienen los right singular vectors.

Esto se hace por medio de la obtención de los autovalores y autovectores de $A * A'$

Con ambos se pueden obtener los right singular vectors. Que usaremos como autocaras en PCA.

Métodos PCA y KPCA

Para el método de PCA generamos la imagen media, y la restamos, para luego con los datos centralizados obtener los Right Singular Values.

Luego se generaron las autocaras con las cuales se puede analizar el error producido a medida que la cantidad de las mismas aumentaba.

Para el método de KPCA se utilizamos un Kernel de grado 2 para generar la separación no lineal de los datos.

En vez de restar una imagen media utilizamos una transformación que involucra al Kernel, a una matriz “U” de 1’s y nuestra cantidad de imágenes.

Demostración

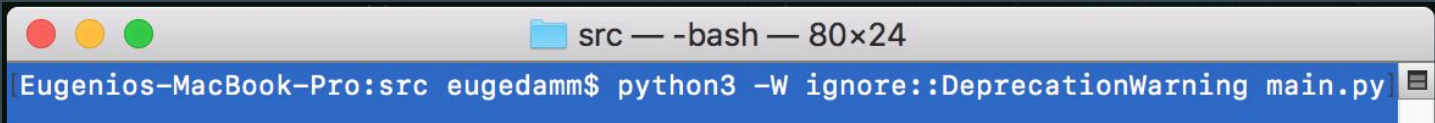
```
[DEFAULT]
V_SIZE = 112
H_SIZE = 92

IMAGES_PER_PERSON = 10
NUMBER_OF_PEOPLE = 42
IMAGE_DIR = ../att_faces/

TEST_NUMBER = 4
TRAINING_NUMBER = 6

#KPCA or PCA
METHOD = KPCA
#TEST or path to face to predict eg: ../att_faces/s1/1.pgm
QUERY = /Users/eugedamm/Desktop/euge_test.pgm
NUMBER_OF_EIGENFACES = 60
```

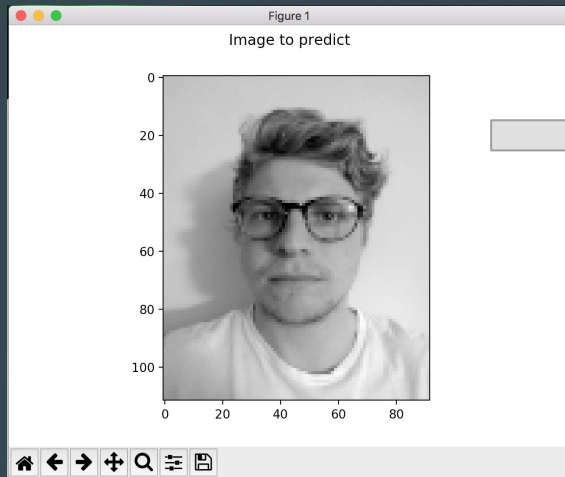
Agregamos el path a la imagen que queremos intentar reconocer



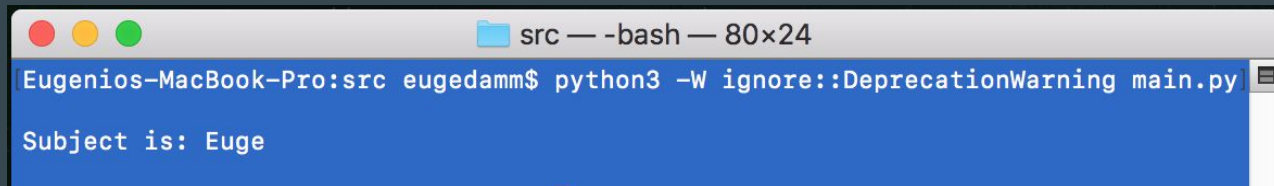
A terminal window titled 'src — -bash — 80x24' with standard macOS window controls (red, yellow, green buttons). The prompt is 'Eugenios-MacBook-Pro:src eugedamm\$'. The command entered is 'python3 -W ignore::DeprecationWarning main.py'. The command is highlighted in blue.

```
src — -bash — 80x24
Eugenios-MacBook-Pro:src eugedamm$ python3 -W ignore::DeprecationWarning main.py
```

Ejecutamos nuestro programa



Se genera una vista previa de la imagen que queremos predecir



```
src — -bash — 80x24
Eugenios-MacBook-Pro:src eugedamm$ python3 -W ignore::DeprecationWarning main.py
Subject is: Euge
```

La terminal nos indicará con qué cara de la base de datos coincide

Resultados

Al ejecutar la prueba tanto con PCA como con KPCA obtuvimos los siguientes resultados:

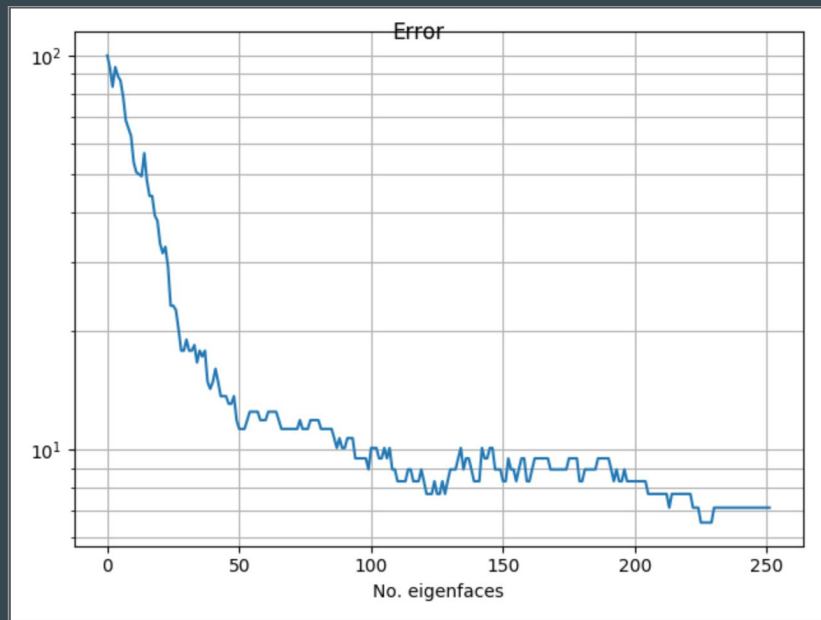


Figura 1: Error obtenido con KPCA

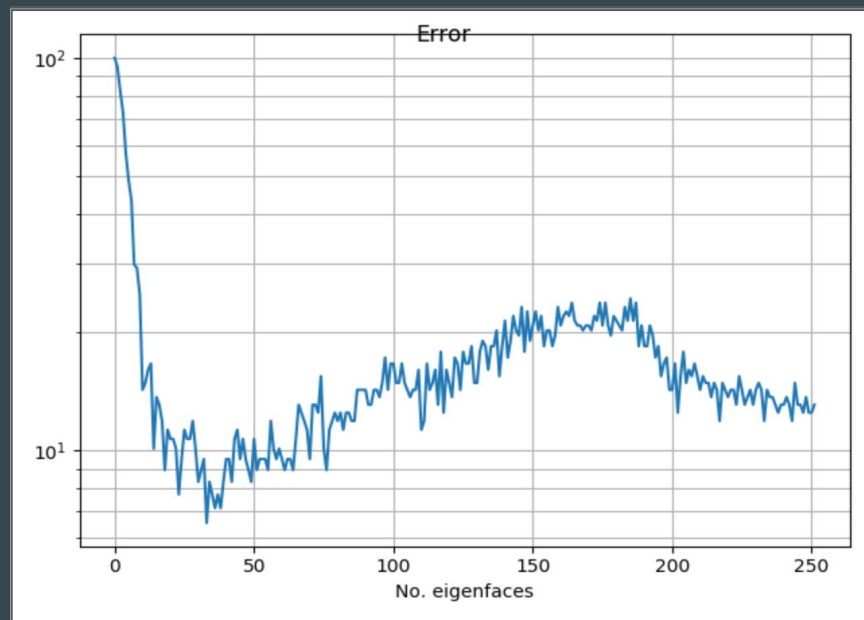


Figura 2: Error obtenido con PCA