COMPRESIÓN DE IMAGEN USANDO SVD

RETO FIN DE SEMANA MÓDULO DE ÁLGEBRA
KEEPCODING

Pablo Indias Salguero

I.- Descomposción en Valores Singulares (SVD)

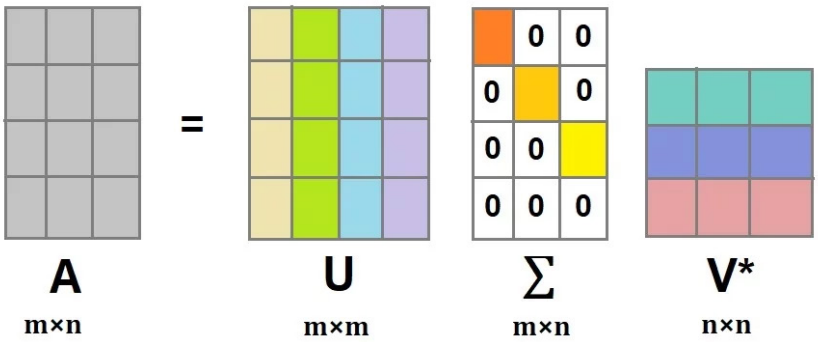
La Descomposición en Valores Singulares es un método matemático por el cual se consigue descomponer una matriz en tres compononentes:

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

- U : Una matriz ortogonal de dimensión m, cuyas columnas son los vectores singulares de izquierdos de A

- Σ: Una matriz Sigma compuesta por la diagonal de los valores singulares. Estos valores extrictamente > 0 y reducen su valor a lo largo de n

- V^T: Es la transpuesta de una matriz ortogonal (V) de dimensión n. Contiene los vectores singulares de A



Los valores singulares en la matriz Σ proporcionan una medida de la importancia o la energía de cada componente de la matriz original. Esta descomposición permite simplificar muchos problemas de análisis y procesamiento de datos, tales como la reducción de dimensionalidad, la compresión de imágenes, y la resolución de sistemas de ecuaciones lineales.

I.I .- SVD para comprimir imágenes

- Compresión de Imágenes:

SVD permite reducir la cantidad de datos necesarios para representar una imagen sin perder significativamente la calidad visual. Al mantener solo los valores singulares más grandes y sus vectores singulares asociados, se puede aproximar la imagen original con menos información, lo que resulta en una imagen comprimida.

Ejemplo: Una imagen de alta resolución puede ser representada con una versión comprimida utilizando solo una fracción de sus componentes singulares, reduciendo así el almacenamiento necesario.

- Eliminación de Ruido:

Las imágenes pueden contener ruido o detalles no deseados. Utilizando SVD, es posible filtrar los valores singulares pequeños que corresponden a este ruido, conservando los componentes principales que representan la información útil de la imagen.

En imágenes médicas, SVD puede ayudar a eliminar artefactos y realzar las características importantes para un mejor diagnóstico.

- Reconstrucción de Imágenes:

SVD es útil para reconstruir imágenes dañadas o incompletas. Al utilizar los valores singulares y vectores asociados, se puede recuperar la información perdida y reconstruir la imagen con alta fidelidad.

Ejemplo: En restauración de fotografías antiguas, SVD puede ayudar a rellenar partes dañadas o borradas de la imagen.

- Reducción de Dimensionalidad:

SVD reduce la dimensionalidad de los datos de la imagen, facilitando el procesamiento y análisis. Esto es especialmente útil en tareas de reconocimiento de patrones y clasificación de imágenes.

Ejemplo: En el reconocimiento facial, SVD puede reducir la cantidad de datos necesarios para identificar características faciales, mejorando la eficiencia de los algoritmos de reconocimiento.

II.- SVD para comprimir una imagen

Como ya hemos comentado, el método SVD nos puedde servir para comprimir una imagen. Vamos a hacer este ejercicio en Python.

Para ello usaremos:

- La implementación del algoritmo **`scipy.linalg`**
- La librería **`matplotlib`** para las gráficas
- El error SSE (Sum of Squared estimate of Errors). Definición:

$$SSE = \sum_{i=1}^{n} (x_i - \widehat{x}_i)^2$$

Donde:

· x_i son los valores de la matriz original X
· $\hat{x}_i$ son los valores de la matriz reconstruida $\widehat{X}$

Por último, la imagen que utilizaremos para este ejercicio será la siguiente:

Fig 1. Face `scipy.datasets`

Primero guardamos la imagen en una variable para poder trabajar en forma de matriz.

```
import matplotlib.pyplot as plt
from scipy import misc, datasets
%matplotlib inline

# Load image
A = datasets.face(gray=True)

plt.imshow(A, cmap=plt.cm.gray)
```

Implementamos la función para calcular el error usando numpy

```
import numpy as np

def sse_score(X, X_hat):
    return np.sum((X - X_hat)**2)

X = np.array([[1, 2], [3, 4]])
X_hat = np.array([[1.01, 1.75], [2.81, 3.99]])
sse = sse_score(X, X_hat)
print(sse)
```

Usamos la función de DVD de numpy

```
import numpy as np
from scipy import linalg

def svm(X):
    # Definimos la función donde usamos linalg.svd de scipy
    U, S, Vt = linalg.svd(X, full_matrices = False)
    return U, S, Vt # S es una matriz diagonal
```

Implementamos la función para reconstruir la imagen

```
U,S,Vt = svm(A)
def reconstruction(U, S, Vt, n_val):
    #Obtenemos la diagonal de S
    S = np.diag(S)
   #Calculamos la matriz S reducida dejando filas hasta n_valores y columna
hasta n_valores
    S_red =S[:n_val , :n_val]
    print('Dimemsión de S_red: ', S_red.shape)
     #Calculamos U reducida dejando todas las filas y tomando las columnas
hasta n_valores
    U_red=U[:,:n_val]
    print('Dimemsión de U_red: ', U_red.shape)
   #Calculamos Vt reducida tomando las filas hasta n_valores y dejando todas
las columnas
    Vt_red=Vt[:n_val,:]
    print('Dimemsión de Vt_red: ', Vt_red.shape)
    # Calculamos el resultado haciendo la multiplicación entre U_red, S_red
y Vt_red
    result = np.dot(U_red,np.dot(S_red, Vt_red))
    return result


A_hat = reconstruction(U, S, Vt,2)
A_hat
```

Implementamos la compresión

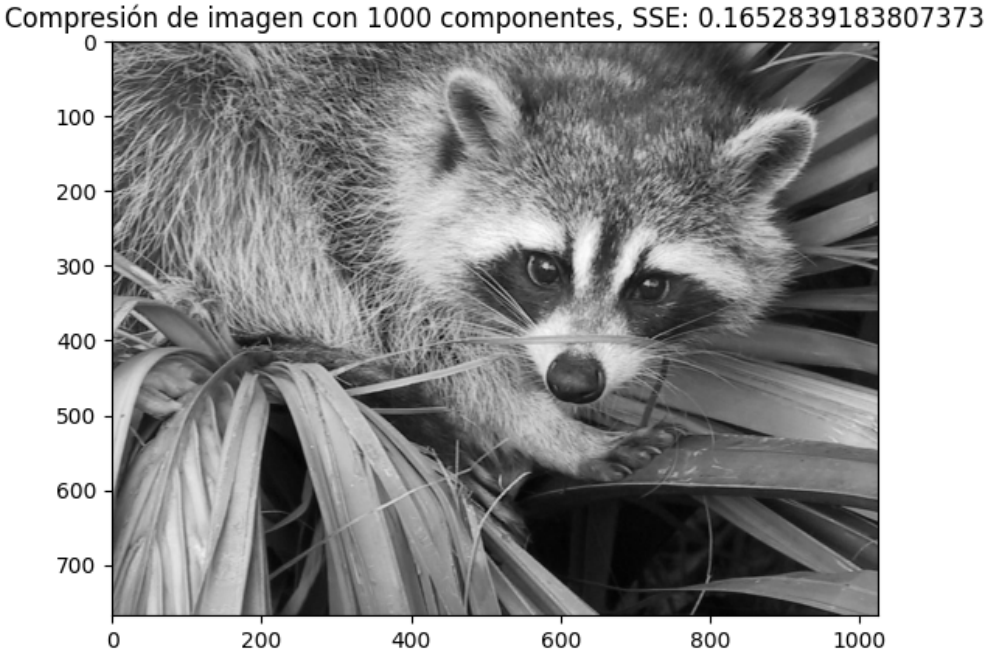
```
### TODO: Función que recibe una imagen A y devuelve la imagen comprimida
### Tiene como entrada A y el número de componentes para realizar la reducción de
dimensionalidad
### Devuelve la imagen comprimidad y el error de reconstrucción

def image_compression(A, n_comp):
    # TODO 1: Aplicar SVD (usando la función que hemos creado)

    U,S,Vt = svm(A)

     # TODO 2: Reconstruir usando solo el número de componentes n_comp (usando la
función que hemos creado)

    A_hat = reconstruction(U, S, Vt,n_comp)

    # TODO 3: Calcular el error

    sse = sse_score(A, A_hat)

   return A_hat, sse # A_hat es la matriz comprimida y sse es su error respecto de A

racoon = misc.face(gray=True)
racoon_hat, sse = image_compression(racoon, 100)
```

Por último, graficamos el resultado

```
import matplotlib.pyplot as plt
# TODO Dibuja las graficas

# 1 Crear gráfica con plt.figure()

plt.figure()

# 2 Elegir un n_comp y aplicar la función image_compression()

n_components = 20
racoon = misc.face(gray=True)
racoon_hat, sse = image_compression(racoon, n_components)

# 3 Usar plt.imshow(A_hat, cmap=plt.cm.gray), donde A_hat va a ser la matriz com-
primida resultante del paso anterior

plt.imshow(racoon_hat, cmap=plt.cm.gray)

# 4 Añadir un título a la gráfica

plt.title(f"Compresión de imagen con {n_components} componentes, SSE: {sse}")

plt.show()
# Repetir para distintas compresiones (distinto n_comp)
```

Obteniendo como resultado la siguiente imagen:

Si cambiamos la variable del número de valores singulares utilizados a 1000 obtenemos la siguiente imagen:



Podemos comprobar que la imagen es mucho más nítida porque la compresión es menor y que la pérdida de datos es menor.

III.- Conclusión

La Descomposición en Valores Singulares (SVD) ha demostrado ser una herramienta muy versátil en el procesamiento y análisis de imágenes. A lo largo de este trabajo, hemos explorado cómo la SVD descompone una matriz en sus componentes fundamentales, permitiendo aplicaciones prácticas que van desde la compresión de imágenes hasta la eliminación de ruido y la reconstrucción de datos perdidos.

En particular, hemos visto cómo la SVD puede reducir significativamente la cantidad de información necesaria para representar una imagen sin comprometer su calidad visual, lo que es crucial para el almacenamiento y transmisión eficiente de datos.

Además, la SVD facilita la reducción de dimensionalidad, haciendo que el análisis de grandes conjuntos de datos de imágenes sea más manejable y eficiente, lo cual es beneficioso en aplicaciones como el reconocimiento de patrones y la clasificación de imágenes.

En resumen, la SVD no solo mejora la eficiencia y efectividad en el manejo de imágenes, sino que también abre nuevas posibilidades para el análisis y procesamiento de datos en diversas disciplinas. La comprensión y aplicación de la SVD son, por lo tanto, fundamentales para avanzar en tecnologías de procesamiento de imágenes y en el desarrollo de algoritmos más robustos y eficientes. Este trabajo pone de manifiesto la importancia y las múltiples aplicaciones de la SVD, subrayando su relevancia en el ámbito académico y profesional.