Eigenfaces Using SVD & PCA

By: Emiliana Geronimo & Brianna Cirillo

TABLE OF CONTENTS

01

WHAT ARE EIGENFACES?

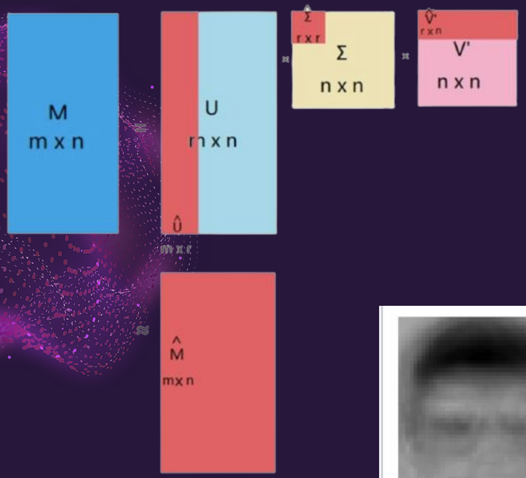
DEF: a set of eigenvectors used in a computer vision problem of human face recognition

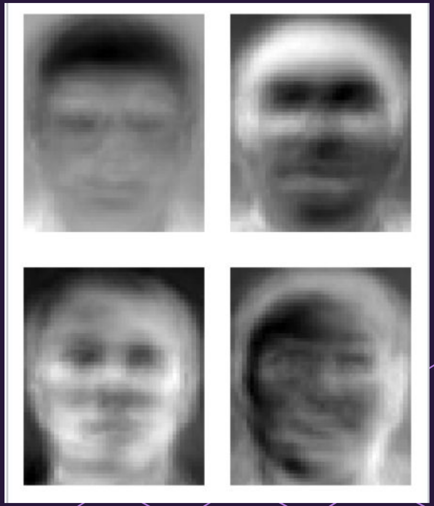
The term *Eigenface* was first introduced by Sirovich and Kirby in 1987.

Eigenfaces describe a set of bases (of features) that are derived by using PCA and SVD to project a higher-dimensional face-image space to a lower dimension.

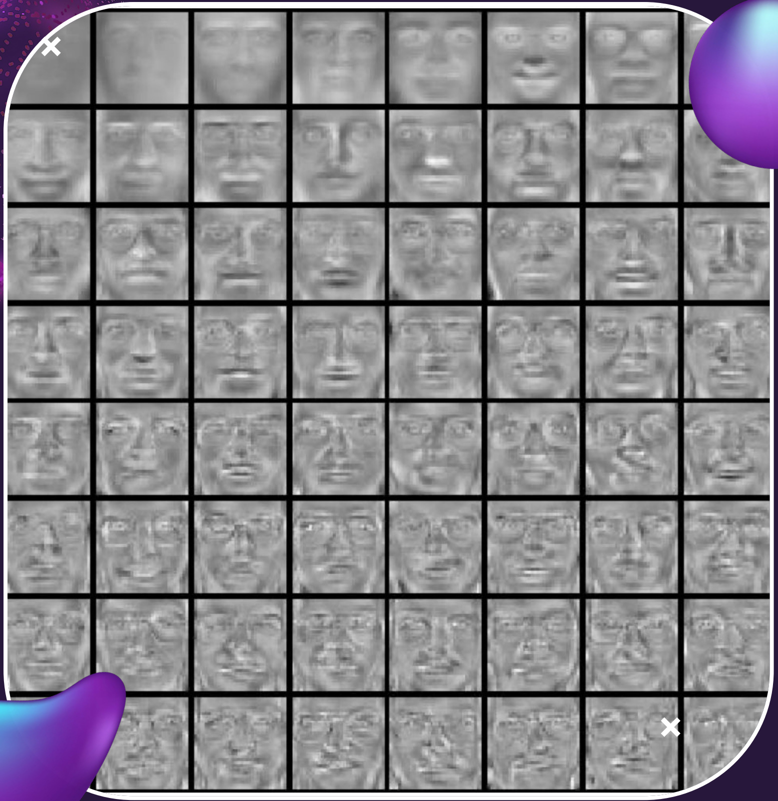
02 SVD

- Facial images can have millions of pixels, leaving us with a multi-million dimension vector space
- SVD can be used to compress images by **truncating** SVD matrices to lower dimensions
- Components will be in order of importance based on singular values





03 PCA

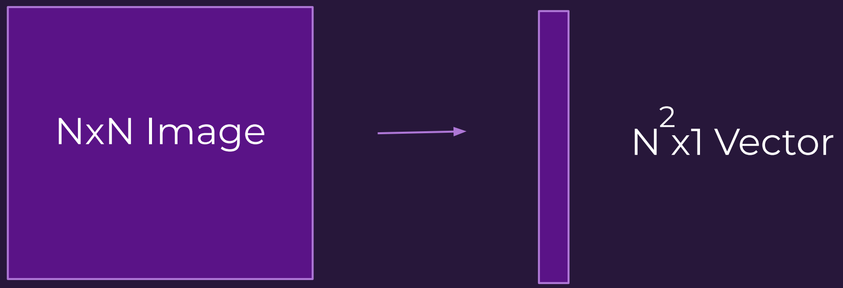


- Purpose of deriving principal components: **REDUCE** the number of variables while still retaining pertinent information
- Principal components are represented as a linear combination of the principal components. Eigenfaces are **LINEAR COMBINATIONS** of the faces
- The top eigenvectors (largest) are used to create eigenfaces that allow for comparison and recognition of facial features

SVD vs PCA

- SVD:
- **PROS**: more numerically stable, handle rank-deficient matrices
- **CONS:** computationally expensive, time consuming
- PCA:
- **PROS:** computationally inexpensive, more suitable for large data sets
- **CONS:** not as numerically stable, cannot handle rank-deficient matrices as well
- Which is better? **IT DEPENDS**

04 Eigenfaces



Main Idea

Let Γ be a vector, corresponding to a NxN face image.
We want to represent Γ into a low-dimension space such that

$$\Phi = \Gamma - \text{mean face}$$

$$\hat{\Phi} - \text{mean} = w_1u_1 + w_2u_2 + \dots + w_Ku_K$$

Computation of Eigenfaces

STEP 1

Obtain face images $I_1, I_2, \ldots, I_M$

Note: Images must be the same size and centered

STEP 2

Create a vector Γ_i for every image I_i

STEP 3

Compute the average face vector:

$$\Psi = \frac{1}{M} \sum_{i=1}^{M} \Gamma_i$$

STEP 4

Subtract the mean face

$\Phi_i = \Gamma_i - \Psi_i$

STEP 5

Compute the covariance matrix

$$C = \frac{1}{M}\sum_{n=1}^{M}\Phi_n \Phi_n^T = AA^T$$

Where

$$A = [\Phi_1 \Phi_2 \ldots \Phi_M]$$

STEP 6

Compute the normalized eigenvectors u_i of AA^T

STEP 7

Keep only K eigenvectors, corresponding to the K largest eigenvalues

0 5

In-Depth Example

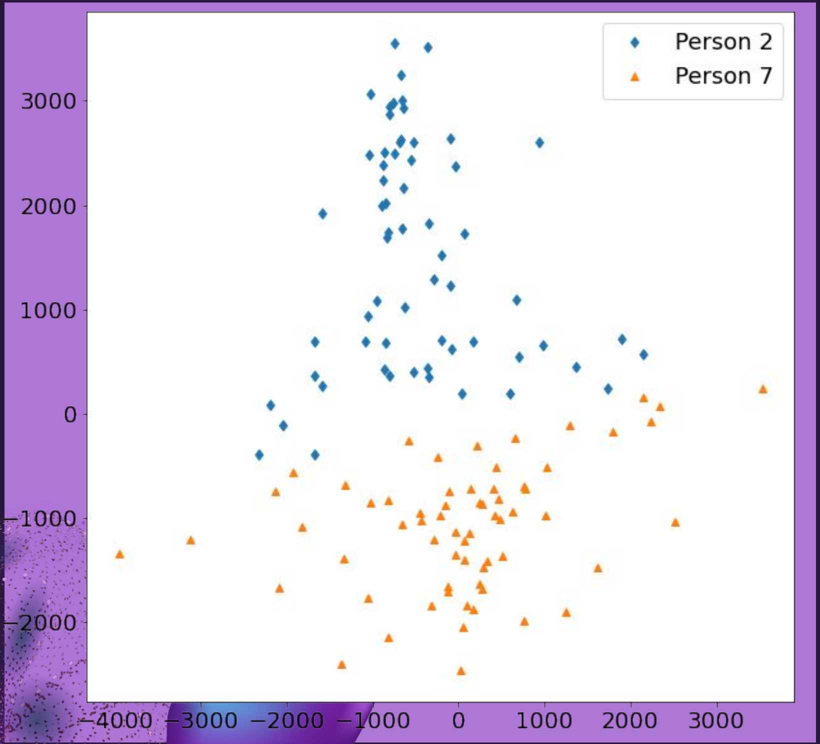
- Images used are from the Yale Faces B data set, which had 38 participants
- Participants sat in a geodesic dome where lights would flash at different angles to capture each participants face in different lighting conditions
- This resulted in 64 pictures of each participants face
- Each image has a resolution of approximately 32,000 pixels

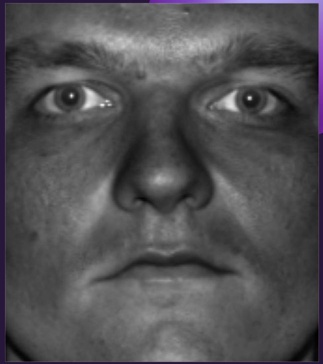

Facial Reconstruction

- Compute the mean face or the average column vector and subtract it from all of the faces in the matrix
- Then compute the SVD of the data
- Calculate the eigenfaces for each column, which captures different features
- Can then use a linear combination of these eigenface to recreate the image of a person


Original Image
r = 25
r = 50
r = 100
r = 200
r = 400
r = 800
r = 1600

Image Classification





- Randomly chose person 2 (left) and person 7 (right)
- Projected their faces into the 5th and 6th principle component and graph them together

06 Takeaways

- Facial recognition technique by breaking down all the important features that make up a face and using those features to create a special set of images called Eigenfaces.
- SVD: factorization technique
- PCA: linear combination technique
- Eigenfaces method: higher success rate in comparison to other methods; ease and speed of recognition with respect to other methods

WORK CITED

EigenFaces and A Simple Face Detector with PCA/SVD in Python | sandipanweb

SVD and Eigenfaces - Nextjournal

SVD: Eigenfaces 1 [Python]

SVD: Eigenfaces 2 [Python]

SVD: Eigenfaces 3 [Python]