LINEAR ALGEBRA

MINI PROJECT REPORT

ON

FACE RECOGNITION USING SINGULAR VECTOR DECOMPOSITION

Contents

Introduction

Facial recognition is a biometric technology that uses distinguishable facial features to identify a person. Today, it's used in a variety of tasks such as unlocking our phone, going through security at the airport and purchasing products at stores. While on one hand, Facebook automatically tags people in the image, and we consider it a convenience, on the other hand, the very same facial recognition could be used to track us without our permission while we are leisurely walking down a street. As such, facial recognition brings several pros and cons along with it.

Face recognition has innumerable applications, the most significant of which are:

i) Criminal Identification: Photographs of individuals can be matched against large databases of criminals, thus prompting necessary action.

ii) Phone unlocking: Almost all recent phones now have the feature to use face recognition for unlocking phones.

iii) Tracing missing people: Face recognizers can be installed at public places to find missing children or victims of human trafficking, by using them against a database of all missing faces registered.

iv) Facebook: Identifying people on social media and tagging them, the most well-known application.

v) Airline Travel: Airlines have already started using face recognition to help people check-in faster.

vi) Forensic investigations: Facial recognition aids forensic investigations by automatically recognizing individuals in security footage or other videos.

Thus, all applications of face-identification are varied, ranging over several domains.

Automatic recognition of faces, being considered a widely used application in the current scenario, we aim to build a facial recognition model that, given a set of faces, trains the model on it using SVD.

Literature Review

Use of Singular Vector Decomposition to recognise faces

Earlier facial recognition technology was considered as an idea of science fiction. But in the past decade, facial recognition technology has not only become real — but it's widespread. Today, people can easily read articles and news stories about facial recognition everywhere.

Facial recognition technology along with Artificial Intelligence and Deep Learning technology are benefiting several industries. These industries include law enforcement agencies, airports, mobile phones, manufacturing companies, home appliance manufacturing companies, etc.

Woodrow Wilson Bledsoe was the father of facial recognition. In the 1960s, Bledsoe created a system that could organize faces' photos by hand using the RAND tablet. The tablet is a device people could use to enter vertical and horizontal coordinates on a grid with the help of a stylus that released electromagnetic pulses. People used that system to manually record the coordinate areas of facial features like eyes, nose, mouth, and hairline.

The manually recorded metrics could be later saved in a database. And when the new photograph of an individual was entered into the system, it was able to get the most closely resembled image via database. During this period, face recognition was untouched by technology and computer processing power. Still, it was the first and foremost step taken by Bledsoe to prove that face recognition was a practical biometric.

It was in the 1970s when Harmon, Goldstein, and Lesk made the manual facial recognition system more accurate. The three used 21 facial markers including lip thickness and hair color, to detect faces automatically. However, Bledsoe's system was still computed with actual biometrics, done manually.

Eigenfaces

Sirovich and Kirby started using linear algebra to the issue of facial recognition in 1988. The approach they used was called the Eigenface approach. The rendering began as a search for

low-dimensional facial images representation. The team was able to prove that feature analysis on collected pictures in the database could form a set of basic features.

They were also able to explain how less than a hundred values could be used to code a face image precisely.

In 1991, Pentland and Turk worked further on the Eigenfaces approach by finding ways to detect faces within images. Pentland and Turk's work was the first automatic face recognition attempt. They used technological and environmental factors for their approach.

Then in the 1993-2000s period, DARPA and NIST released the FERET program to encourage the commercial facial recognition market. In 2002, law enforcement officials applied facial recognition in critical technology testing.

The approach used in a study [1] involved using a mathematical model called Eigen face that measures variability within a set of images, and using them to classify new faces in terms of the ones already seen. A covariance matrix that measured the relation between all dimensions pair-wise so as to eliminate those dimensions which were closely related and thus eliminating redundant information. The low-dimensional image was then constructed using eigen vectors found using SVD of the covariance matrix. The set of eigen vectors were named "Eigen faces" since they formed the basis of the "face" subspace. The Eigen faces were later used to find the projection of each test image along these dimensions, thus expressing each face in terms of these Eigen vectors. Once this was down, each test image was projected along these dimensions (Eigen vectors) too, to find their projections and thus the face in the test image it had a value closest too.

In this approach [1], the number of Eigen vectors were equal to the number of different faces fed. We however, tried to improve on the efficiency using PCA on top of SVD. There were several dimensions that had very low Eigen values, or in other words less information. Considering them in our computations hardly made any difference. Thus, we dropped those dimensions by plotting a scree plot. A scree plot displays the eigenvalues of factors or principal components in an analysis as a line plot. It was used to determine the number of principal components to keep in the principal component analysis. The components that had a

high singular value were retained in the “Eigen face” thus eliminating the redundant Eigen vectors. The results were however extremely satisfactory.

Thus, adding Principal Component Analysis to the recognition procedure, helped improve the efficiency of face recognition computation-wise, manifold times.

Terms and their application to our use-case

Covariance

Covariance measures the relationship between how two variables change. A high positive covariance means that a positive change in one variable correlates to a positive change in the other. A highly negative covariance implies a positive change in one variable correlates to a negative change in the other. If the two variables are independent, then their covariance is zero.

$$\mathrm{Var}(X) = \Sigma (X_i - X)^2 / N = \Sigma x_i^2 / N.$$

$$\mathrm{Cov}(X, Y) = \Sigma (X_i - X) (Y_i - Y) / N = \Sigma x_i y_i / N.$$

where X and Y are means of the corresponding dimensions (which is computed prior to the calculation).

Covariance matrices

Covariance matrix consists of the covariances between each pair of variables in the other matrix positions and the variances of the variables along the main diagonal.

$$\Sigma = \begin{pmatrix} E[(x_1 - \mu_1)(x_1 - \mu_1)] & E[(x_1 - \mu_1)(x_2 - \mu_2)] & \cdots & E[(x_1 - \mu_1)(x_p - \mu_p)] \\ E[(x_2 - \mu_2)(x_1 - \mu_1)] & E[(x_2 - \mu_2)(x_2 - \mu_2)] & \cdots & E[(x_2 - \mu_2)(x_p - \mu_p)] \\ \vdots & \vdots & \ddots & \vdots \\ E[(x_p - \mu_p)(x_1 - \mu_1)] & E[(x_p - \mu_p)(x_2 - \mu_2)] & \cdots & E[(x_p - \mu_p)(x_p - \mu_p)] \end{pmatrix}$$

SVD

According to SVD, any matrix A can be factorized as:

$$A = U S V^T$$

where U and V are orthogonal matrices consisting of orthonormal eigenvectors of AA^T and A^TA respectively. S is a diagonal matrix with r elements equal to the root of the positive eigenvalues of AA^T or A^TA (both matrices have the same positive eigenvalues anyway). The diagonal elements of S are composed of singular values.

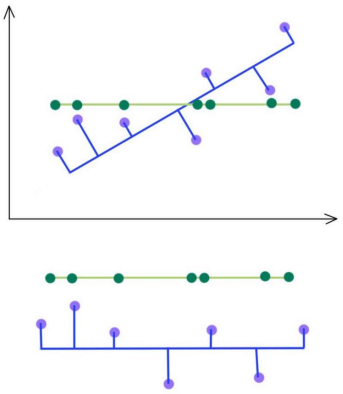
To standardize the solution, we order the eigenvectors such that vectors with higher eigenvalues come before those with smaller values.

Principal Component Analysis

A superset of SVD, PCA uses SVD in its computation. Similar to SVD, it is used to reduce the dimensionality of the image. Reducing the number of variables, and thus the dimensions comes at the expense of accuracy, but we trade a little accuracy for the significant reduction in computation. Thus, we reduce the dimensions, retaining as much information as possible.

Geometrically speaking, principal components represent the directions of the data that explain a maximal amount of variance, that is to say, the vectors that capture most information of the data. The relationship between variance and information here, is that, the larger the variance carried by a line, the larger the dispersion of the data points along it, and the larger the dispersion along a line, the more the information it has.

SVD selects a projection that maximizes the variance of their output. Hence, PCA will pick the blue line over the green line if it has a higher variance.



We keep the eigenvectors that have the top *k*th highest singular value. Hence, we can reduce the number of dimensions in our image, by considering only the principal components that contain the most information.

Implementation

This project involves 3 phases:

- Getting a dataset of sample images of people we want to recognize.
- Calculate the eigenfaces which represents shared variability among the faces of the sample faces.
- From the sample images, classify new images of the faces.

The subspace spanned by the eigen faces(eigen vectors of the covariance matrix) contains faces of all sample people. Hence any face can be represented as a linear combination of these eigenfaces.

The following are the steps used in this process:

1. Mathematical Representation of Image

In short, a vector.

We refrain to grayscale images so as to reduce the amount of unnecessary information. Say each image is an n x m matrix of pixel values R, integers ranging from 0 to 255. Pixel values of each image are collected and stored as columns in a matrix.

A = (a1,1 , … , a1,m, a2,1 , …. , a2,m, … an,1, … an,m)

We first find the “mean face” which corresponds to the center of gravity of our sample of faces, the average pixel intensity values for each pixel.

Mean image obtained for dataset Face95

Next, calculate the deviation of all the training images from the “mean face”. For each vector subtract mean from the pixel vector representation. Let’s call all these vectors “difference faces”.

Four difference faces obtained from Face94 dataset.

Note that, if we visualise this vector as a matrix, we’ll see that some faces are dark (difference value close to 0; pixel value of black = 0), while some are white (pixel and thus difference close to 255[white]). These dark faces thus indicate the more agreeable position of their faces as compared to the relatively white images that are more unique.

2. Eigen Subspace

In order to account for all faces, we require a “face space”, similar to eigen spaces. And just as an eigen space is made up of eigen vectors, similarly our face space is made of eigen faces.

We know that :

first r columns of U : column space of A

last m-r columns of U : left nullspace of A

first r columns of V : row space of A

last n-r columns of V : nullspace of A

Hence, we can use SVD to find the column space of the given faces.

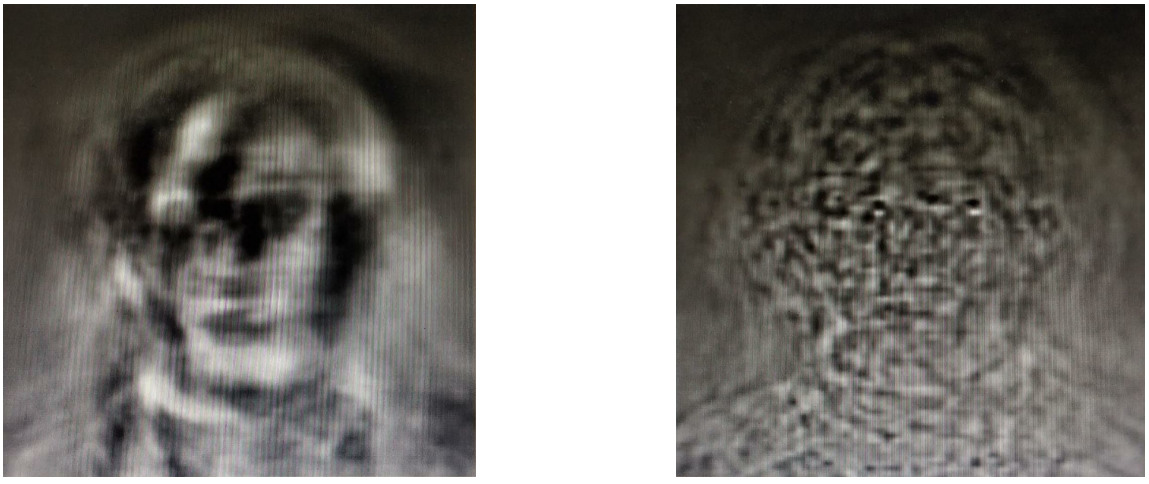
We can approach SVD from a slightly different view using Covariance matrices. Covariance as mentioned earlier, is a measure of how two variables vary between themselves. In our case, finding covariance is extremely useful since it tells us how much a dimension depends on other ones since the covariance matrix measures the dependency of each dimension on every other one.

Covariance matrix : AA^t

The first eigenvector of the covariance matrix C points in the direction of maximum variation, and the corresponding eigenvalue measures the variation in this direction; i.e., it is the variance in this direction. The subsequent eigenvectors point in the directions of maximum variation orthogonal to the previous directions, and the eigenvalues again measure the variations

Eigen vectors of covariance matrices are the same as U from SVD.

Hence we now have the set of basis of calculating all input faces - U.



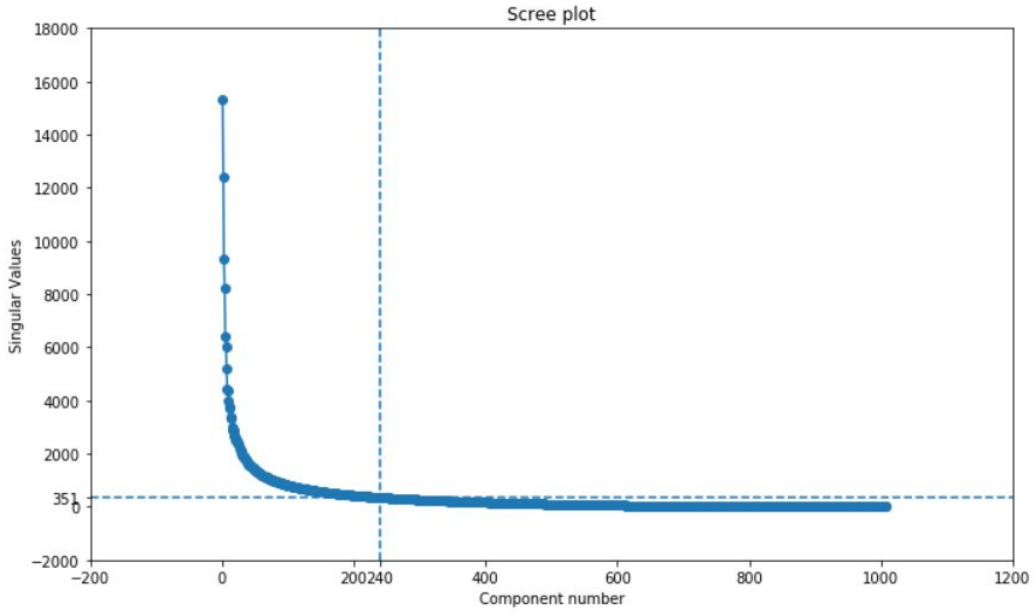
Random eigen faces from face95 dataset

3. Principal Component Analysis

The number of dimensions so far, are so high that it is computationally extremely heavy to find the differences of each vector from all the mxn dimensions. Remember that each dimension, in the "differences of image" vector represents the variation along that dimension. Now, we have to reduce the number of dimensions, along with retaining the information. SVD helps to do this.

Thus, we find the Singular Vector Decomposition of the matrix A found in the previous step. As mentioned earlier, let the singular values of the S matrix be arranged in increasing order. Singular values are the amount of information (variance of the "face points") along that particular dimension.

On plotting the component number, and their singular values, we obtained the following graph:



Carefully note that as the component number increased the amount of features it represented sharply decreased. In fact, most of the information is present in the first 5-6 components.

Hence, using the concept of PCA, we now removed the components with zero or relatively less number of singular values. This no doubt, is going to be a lead to the training process.

As can be seen in the graph, the amount of information a component represents is almost zero, after the 240th component. Hence, in our experiment, we dropped those components after 240, thus decreasing the number of dimensions from more than 1000 to just 240.

Once we found these principal components, we constructed the face space, by projecting each training image on each of these components and summing up.

4. Recognition procedure

Once we identify the principal dimensions, it hardly takes any effort to recognise the images.

As we did earlier, convert the image to grayscale, then a vector and normalise the vectors.

Coming to the main part, this time, we project the sample test image vector along these principal components (face space) found in the step earlier.

Once we have projected every sample test image into the eigenface subspace, we have a bunch of points in R^{240}. We find the smallest distance between the new point and each of the projections of the sample images. The image this smallest distance corresponds to corresponds to the face or class the image belongs to.

Results and Discussions

Dataset	***Size(No. of Images)***	***Accuracy***
Face94	3060	99%
Face95	1440	80%
Face96	3020	94.5%

Face94 dataset having the highest number of images resulted in a higher accuracy than all the other datasets. Face96 having the next higher number of images has a higher accuracy than

Face95 dataset. Face95 dataset yielded a good accuracy of 80% because of its lesser number of images. Dimensionality reduction using PCA highly reduced the computation time of the program. It was reduced to around 240 vectors as compared to dimensions as high as 1010. All the above factors are responsible in yielding the results as indicated by the table.

Following are the results for a set of difference faces in each dataset. The left image is the input test face and the right image is the recognized face with the name of the person. The accuracy for each dataset is reflected in the sample results.

Sample results for dataset Face94. All faces match because of a high accuracy.

Sample results for dataset Face95. One face doesn't match which can be due to the lesser accuracy.

Sample results for dataset Face96. All faces match because of a high accuracy.

Summary and Conclusions

Thus, using PCA, an important concept in Linear Algebra we have reduced the cumbersome task of training a model to recognise faces. A task which would normally be computationally heavy with all dimensions of the image considered, has now been made significantly easier to solve using the concept of Principal Component Analysis. A higher number of training data significantly increases the accuracy of the model.

Linear Algebra and its wonderful results such as SVD, PCA, Eigen Vectors and Eigen values have significantly contributed to the field of machine learning which now finds applications in several tasks, only one of which is facial recognition.

Future Scope

At the present moment, our model is able to recognise a face as belonging to a person whom we had earlier trained on. Hence for a test image, it either identifies the correct person or the

wrong one (thus leading to an accuracy less than 100%). However, we are planning to work on a suitable threshold so that instead of identifying a face as the wrong one, we create another class “None of the faces”, so that images which differ from the faces by a value greater than the threshold could fall under this category. This way, the model won’t be forced to predict a wrong value. This threshold could either be brute forced by varying its value, and measuring the accuracy in each case or another machine learning algorithm could be applied on it to find the ideal threshold value that minimises the error.

This way the face recognizer will be more powerful.

Bibliography

[1] International Journal of Science and Research (IJSR) , “Face Recognition Using Singular Value Decomposition of Facial Colour Image Database”

[2] Shodhganga Chapter 3 , “Singular Value Decomposition and Face Recognition using Different Distance Measure Techniques”

[3] Society for Industrial and Applied Mathematics , “Singular Value Decomposition, Eigenfaces, and 3D Reconstructions” , Neil Muller, Lourenco Magaia, B.M. Herbst

[4] https://jeremykun.com/2011/07/27/eigenfaces , Jeremy Kun

[5] Machine Learning - Singular Value Decomposition (SVD) and Principal Component Analysis (PCA) , Medium blog

[6] “A Step by Step Explanation of Principal Component Analysis” , Zakaria Jaadi , Built-in blog

Books

[7] “Introduction to Linear Algebra - 5th edition” , Gilbert Strang

[8] “Matrix Computations - 4th edition” , Gene H. Golub and Charles F. Van Loan