INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH & DEVELOPMENT

ISSN 2278 – 0211 (Online)

Object Recognition Using Eigen Values

Dheemanth H. N.
Department of Computer Science, National Institute of Engineering, Karnataka, India

Abstract:
Object recognition is a computer application for automatically identifying an image from a set of images in the database. Many recognition systems recognize the images based on some characteristic features of the image. However, this paper helps recognize the images based on some mathematical computations which include the Eigen values and the Eigen vectors. The Eigen face approach is one of the simplest and most efficient methods in recent times for developing a system for Object Recognition. Object recognition is an emerging field with applications varying from video surveillance, forensic use to commercial applications such as in virtual reality, smart cards and information security. Eigen faces are eigenvectors of covariance matrix, representing the given image space. The object images are projected onto a face space (feature space) which best defines the variation of the known images. The face space is defined by the 'Eigen faces' which are the eigenvectors of the set of images. The new image which is projected into this face space is then compared with the available projections of the training set to identify the image. The image with a minimum distance (less than the threshold distance) from the input image in the projection area is most suited to be the result. The framework provides the ability to learn to recognize objects in an unsupervised manner. New images which are fed into the system can be identified with a high rate of success.

Key words: *Eigen values, Eigen vectors, Principal Component Analysis, Covariance matrix*

1. Introduction
Humans are very good at recognizing objects and complex patterns. Even a passage of time does not affect this capability and therefore it would help if computers become as robust as humans in image recognition. Object recognition can be applied for a wide variety of problems like image and film processing, human-computer interaction, criminal identification, etc. This has motivated researchers to develop computational models to identify the faces, which are relatively simple and easy to implement. The model developed for this is simple, fast and accurate in constrained environments. The goal of this research is to implement the model for a recognition system which recognizes an image from a large number of stored images with some real-time variations as well.

The goal is to implement the model for a particular image and distinguish it from a large number of stored images in the database. The scheme is based on an information theory approach that decomposes the images into a small set of characteristic feature images called 'Eigen faces', which are actually the principal components of the initial set of stored images. Recognition is performed by projecting a new image into the subspace spanned by the Eigen faces ('face space') and then recognizing the image by comparing its position in the face space.

2. Problem Definition
The database has a set of images in it. These represent some face space with high dimensionality. For a given new image, we need to recognize this new image among the image classes and check for recognition. Here the new image is matched to one of existing object images in database and perform object recognition, making use of Eigen face Approach and Principal Component Analysis.

3. Eigen Values and Eigenvectors
Eigen values and Eigen vectors are the properties of a matrix. A matrix may act on certain vectors by changing only their magnitude, and leaving their direction unchanged (or possibly reversing it). These vectors are the eigenvectors of the matrix. A matrix may act on an eigenvector by multiplying its magnitude by a factor, which is positive if its direction is unchanged and negative if its direction is reversed. This factor is the Eigen value associated with that eigenvector. In linear algebra, the eigenvectors of a linear operator are non-zero vectors which, when operated on by the operator, result in a scalar multiple of them. The scalar is called the Eigen value associated with the eigenvector.

If A: V-->V is a linear operator on some vector space V, v is a non-zero vector in V and c is a scalar (possibly zero) such that, A v = c v then we say that v is an eigenvector of the operator A, and its associated Eigen value is c.

3.1. To calculate Eigen values and Eigenvectors

Consider the equation,

$(A - \lambda I)X = 0$

Where I, is the n x n identity matrix. This is a homogeneous system of equations, and from fundamental linear algebra, we know that a nontrivial solution exists if and only if

$\det(A - \lambda I) = 0$

Where det() denotes the determinant. When evaluated, it becomes a polynomial of degree n. This is known as the characteristic equation of A, and the corresponding polynomial is the characteristic polynomial. The characteristic polynomial is of degree n. If order of A is n x n, then there are n solutions or n roots of the characteristic polynomial. Thus, there are n Eigen values of A satisfying the equation,

$AX_i = \lambda_i X_i$

If the eigenvalues are all distinct, there are n associated linearly independent eigenvectors whose directions are unique and which span n dimensional Euclidean space.

3.2. Properties of Eigen values and Eigenvectors

Some important properties of Eigen values and Eigenvectors and their significance in case of Symmetric matrices are as follows:

- **Property 1**: If the Eigen values of a symmetric matrix are distinct, then the Eigen vectors are orthogonal.
- **Property 2**: The Eigen values of a symmetric matrix are real.
- **Property 3**: Let A be a matrix with Eigen values λ_i where i = 1 to n and eigenvectors v_i. Then the Eigen values of the matrix (A + s*I*) are λi+s, with corresponding eigenvectors vi, where s is any real number.
- **Property 4**: Let A be an n x n matrix with Eigen values λi where i = 1 to n then the determinant
- $\det(A) = \sum_{i=1}^{n} \lambda_i$
- **Property 5**: If v is an eigenvector of matrix A, then cv is also an eigenvector, where c is any real or complex constant.
- **Property 6**: For the matrices A_{NxM} and A^T_{MxN}, the Eigen values different to zero from AA^T and A^TA are the same ones and they have the same multiplicity. If x is a non trivial eigenvector of AA^T for an Eigen value

$\lambda != 0$, then $y = A^T x$ is a is a non trivial eigenvector of A^TA .

This is very important property, giving relationship between Eigenvectors of AA^T and A^TA.

This can be proved as follows:

The equation is $AA^T x = \lambda x$. Pre-multiplying both sides by A^T, we get

$$A^TAA^T x = \lambda A^T x$$

Substituting $y = A^T x$ in the equation, it arrives to $A^TAy = \lambda y$. The vector x is not trivial if it does not equal zero. Since $y = A^T x$, it is also not equal to zero. This shows that y is eigenvector.

3.3. Variance

The variance is a measure of the spread of data. To use election polls as an example, the population includes all the people in the country, whereas a sample is a subset of the population that the statisticians measure. The good point about statistics is that by measuring only a sample of the population, one can work out what is most likely to be the measurement if we used the entire population. Let's take an example:

X = [1 2 4 6 12 25 45 68 67 65 98]

We could simply use the symbol X to refer to this entire set of numbers. For referring to an individual number in this data set, we will use subscript on the symbol X to indicate a specific number. There are a number of things that we can calculate about a data set. For example, we can calculate the mean of the sample. It can be given by the formulae:-

Mean = sum of all numbers / total no. of numbers

Unfortunately, the mean doesn't tell us a lot about the data except for a sort of middle point. For example, these two data sets have exactly the same mean (10), but are obviously quite different.

[0 8 12 20] and [8 9 11 12]

So what is different about these two sets? It is the spread of the data that is different. The Variance is a measure of how spread out the data is. It is just like Standard Deviation.

3.4. Standard Deviation

The way to calculate standard deviation is to compute the squares of the distance from each data point to the mean of the set, add them all up, divide by n-1 and take the positive square root. As a formula:

$$\sigma^2 = \frac{\Sigma(X - \mu)^2}{N}$$

3.5 Covariance

Variance and SD are purely 1-dimensional. Data sets like this could be: height of all the people in the room, marks for the last CSC378 exam etc. However, many data sets have more than one dimension and the aim of the statistical analysis of these data sets is usually to see if there is any relationship between the dimensions. For example, we might have set, of both the height of all the students in a class and the marks they received in a subject as our data. We could then perform statistical analysis to see if the height of a student has any effect on their marks. It is useful to have measures to find out how much the dimensions vary from the mean with respect to each other. Covariance is a measure that it is always measured between 2 dimensions. If we calculate the covariance between one dimension and itself, you get the variance. So if we had a three dimensional data set (x, y, z), then we could measure the covariance between the x - y dimensions, the x - z dimensions and the y - z dimensions. Measuring the covariance between x and x, or y and y, or z and z would give us the variance of the x, y and z dimensions respectively.
The formula for covariance is very similar to the formula for variance.

$$\text{cov}_{XY} = \frac{\sum (X - M_X)(Y - M_Y)}{n}$$

3.6. Covariance Matrix

A useful way to get all the possible covariance values between all the different dimensions is to calculate them all and put them in a matrix. For example: We will make up the covariance matrix for an imaginary 3 dimensional data set, using the usual dimensions x, y and z. Then the covariance matrix has 3 rows and 3 columns, and the values are this:

$$C = \begin{matrix} \text{cov}(x, x) & \text{cov}(x, y) & \text{cov}(x, z) \\ \text{cov}(y, x) & \text{cov}(y, y) & \text{cov}(y, z) \\ \text{cov}(z, x) & \text{cov}(z, y) & \text{cov}(z, z) \end{matrix}$$

4. Principal Component Analysis

In statistics, principal components analysis (PCA) is a technique that can be used to simplify a dataset; more formally it is a transform that chooses a new coordinate system for the data set such that the greatest variance by any projection of the data set comes to lie on the first axis (then called the first principal component), the second greatest variance on the second axis, and so on. PCA can be used for reducing dimensionality in a dataset while retaining those characteristics of the dataset that contribute most to its variance by eliminating the later principal components.
PCA aims at:

- Reducing the dimensionality of the data set.
- Identifying new meaningful underlying variables.

4.1 Finding principal Components

Principal Component Analysis is traditionally done on a square symmetric covariance or correlation matrix obtained from the given m x n data matrix. A covariance matrix is obtained by mean centering the data across the origin and then taking the dot products.
A correlation matrix is obtained by normalizing the covariance matrix. In statistical data it is very natural to have data spread out over wide ranges. So the normalization is required. If normalization is not done, it would be difficult to assess the contributions of various components to the principal component. Principal components are the eigenvectors of the square symmetric correlation matrix. The eigenvector with the maximum Eigen value is the first principal component, the one with next largest eigen value is the second principal component and so on.

4.2. Image Recognition Problem

The proposed problem consists of verifying if a new object image belongs to one of the stored images in a. A new image is compared with well-known images stored in a database, being classified as a well known image.

4.3 Object Representation Techniques

This can be framed in three different categories: Template-based, Feature-based and Appearance-based.

- **Template-based method:** This represents the object by means of a main two-dimensional template with values representing the facial ellipse borders and all the face organs. Another way is to have multiple templates for the image representation, under several angles and points of view. Another important approach is the use of a group of smaller characteristics models, corresponding to the features like the eyes, nose and mouth. The most attractive advantage of this model is its simplicity. Its disadvantage is the need of a great amount of memory and its inefficient comparison method.
- **Feature-based method**: This considers the positions and sizes of the features of the object such as eyes, nose, mouth, etc., in the image. This method consumes very less computer resource than the template-based method, facilitating larger processing speed, with good acting with database images in varied scales.

- **Appearance-based method**: This intends to project the images in a low dimensional subspace, to obtain the representation. The Eigen faces space is an application of this method. It is built on Principal Component Analysis, from the projection of the images of the training set into the face space with low dimension.

The simplest method for face recognition can be based on comparison approach in which the new image can be compared with each of the existing images in the database to check for a match. This is very simple approach as we need to take dot product of two images that means comparing images pixel by pixel. If the pixel intensity values of both images match, then this new image is said to be a known image. As the database size is large and it contains redundant information in high dimensional space, performance is improved by reducing dimensionality making use of Eigen face approach.

4.4. PCA Applied to Object Recognition

The objective of the Principal Component Analysis (PCA) is to take the total variation on the training set of images and to represent this variation with some variables. When we are working with large sets of images, reduction of space dimension is very important. PCA intends to reduce the dimension of a group or to space it better so that the new base describes the typical model of the group. The image space is highly redundant when it describes similar objects. This happens because each pixel in such objects is highly correlated to the others pixels. The objective of PCA is to reduce the dimension of the work space. The maximum number of principal components is the number of variables in the original space. Even so to reduce the dimension, some principal components should be omitted. This means that some principal components can be discarded because they have a small quantity of data, considering that the larger quantity of information is contained in the other principal components. The Eigen faces are the principal components of the original images, obtained by the decomposition of PCA, forming the face space from these images. So any new image can be expressed as a linear combination of these Eigen faces. Object Recognition is performed from the projection of the input image into the face space and by measuring of the Euclidean distance between the new image and the image classes. If the distance is inside the threshold of a certain class and it is the minimum value, then there is recognition. The face space is described by an Eigen face group.

5. Eigen Face Approach

The information theory approach of encoding and decoding object images, extracts the relevant information in an image, encode it as efficiently as possible and compare it with a database of similarly encoded images. The encoding is done using features which may be different or independent than the distinctly perceived features like eyes, ears, nose, lips, and hair. Mathematically, the principal component analysis approach will treat every image of the training set as a vector in a very high dimensional space. The eigenvectors of the covariance matrix of these vectors would incorporate the variation amongst the face images. Now each image in the training set would have its contribution to the eigenvectors (variations). This can be displayed as an 'Eigen face' representing its contribution in the variation between the images. These Eigen faces look like ghostly images. In each Eigen face some sort of variation can be seen which deviates from the original image. The high dimensional space with all the Eigen faces is called the face space (feature space). Also, each image is actually a linear combination of the Eigen faces. The amount of overall variation that one Eigen face counts for, is actually known by the Eigen value associated with the corresponding eigenvector. If the Eigen face with small eigenvalues is neglected, then an image can be a linear combination of reduced number of these Eigen faces. For example, if there are M images in the training set, we would get M Eigen faces. Out of these, only M' Eigen aces are selected such that they are associated with the largest Eigen values. These would span the M'- dimensional subspace of the face space out of all the possible images. When the object image to be recognized (known or unknown), is projected on this face space, we get the weights associated with the Eigen faces, that linearly approximates the image or can be used to reconstruct the face. Now these weights are compared to the weights of the known images so that it can be recognized as a known image used in the training set. In simpler words, the Euclidean distance between the image projection and known projections is calculated; the face image is then classified as one of the images with minimum Euclidean distance.

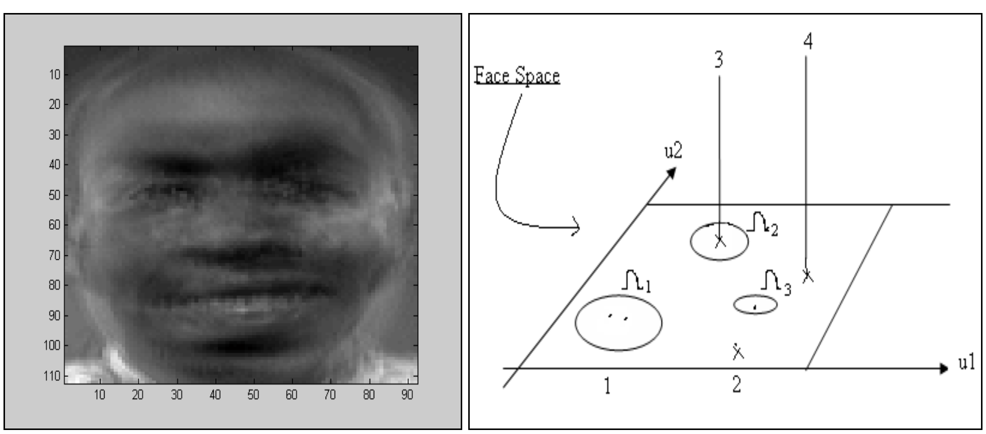

Figure 1

(a) The face space and the three projected images on it. Here u1 and u2 are the eigenfaces.

(b) The projected image from the database

The basic steps involved in Recognition process using Eigen face Approach are as follows:

5.1. Initialization

- Acquire an initial set of object images known as Training Set.
- Calculate Eigen faces from training set keeping only M images that correspond to highest Eigen values. These M images, define the face-space.
- Calculate distribution in this M-dimensional space for each known image by projecting them into this face-space.

5.2. To Recognize New Input Image

- For given input image, calculate a set of weights based on M Eigen faces by projecting this new image onto each of Eigen aces.
- Check if the image is sufficiently close to face-space.
- Now the weight pattern can be compared with known weight patterns to match the images.

6. Representation of Images

The images are represented by intensity values of each pixel. Let the dimensionality of each image be m x n. This means that each image consists of grid of pixels with m rows and n columns. Let I(x, y) represents intensity values for all pixels. So the total number of pixels of each image will be m*n. Let this value be denoted as N. Now this image can also be considered as a vector of dimension N. For example, if the images have dimension 256 x 256 pixels, then the dimension of the image vector will be 65536. So here N = 65536. Normally for all the images, since dimensionality of image is large, the value of N (dimension of image vector) is also large. This maps the image to collection of points in a huge space. Since the images will be similar in overall configuration, these images will be randomly distributed in huge space and thus will lie in relatively low dimensional space.

6.1. Training Set of Images

Let the training set consists of M images representing M image classes. Each of these images can be represented in vector form as stated above. Let these images be represented by: $T_1, T_2, T_3 \ldots T_M$.

The average image of the set is

$$\mathbf{\Psi = (1/M)\, \Sigma_1^M T_i}$$

Each image differs from the average image of the distribution and this distance is calculated by subtracting the average image from each image in the database. This gives us new image space.

$$\mathbf{\Phi = T_i - \Psi \qquad (i = 1,\ldots\ldots, M)}$$

7. Formation of Covariance Matrix

From the above image space, the matrix A is formed with dimension N x M by taking each of image vectors Φ i and placing them in each column of matrix A.

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

Using matrix A, it is important to set up the Covariance matrix C. This can be given by product of matrix A with matrix A^T. The dimension of such covariance matrix will be N x N.

$$C = AA^T$$

7.1. Calculate Eigen vectors and Eigen values

As the dimension of this matrix is N x N, it will result in N Eigen values and N eigenvectors. Since the value of N is very large, say 65536 as in above example, it would be better to reduce this overhead by considering matrix $L = A^TA$. The dimension of this matrix will be M x M.

$$L = AA^T$$

Since the covariance matrix is Symmetric, it holds the Property 5 described earlier. The N Eigen values obtained from C are same as M Eigen values with remaining N - M Eigen values. Also, if x is eigenvector obtained from C then the eigenvectors of L are given by

$$y = A^T x$$

We can make use of this relationship to obtain Eigen values and eigenvectors of AA^T by calculating Eigen values and eigenvectors for A^TA. The eigenvectors of C (Matrix U) are obtained from eigenvectors of L (Matrix V) as given below:

$$U = AV$$

The matrix V, with dimension (M x M), is constituted by the M eigenvectors of L and matrix U, with dimension (N x M), is constituted by all the eigenvectors of C and the matrix A is the image space, with dimension (N x M).

7.2. Eigen Faces

The Eigen faces can be simply defined as the eigenvectors which represent one of the dimensions of image space. The Eigen faces are a group of important characteristics that describe the variation in the group of images. All eigenvectors have an Eigen value associated with it and the eigenvectors with the largest Eigen values provide more information on the variation than the ones with smaller Eigen values.

8. Projection of images into Face space

All the images from training set are projected to the Eigen space. These can be represented by a linear combination of the Eigen faces that have a new descriptor as a point in a great dimensional space. This projection is constructed in the following way:

$\mathbf{\Omega_i = U^T (T_i - \Psi)}$ where (i = 1… M)

As the projection of the faces pace, describes the variation of face distribution, it is possible to use these new descriptors to classify them.

8.1. Object Recognition from Eigen faces

The steps followed for the image recognition using Eigen face approach are as follows:

8.2. Initialization for input image

When a new image is given as input to check for recognition, it can be classified in one of these image classes. Also, it can be compared for a match with any of the existing images in the database. Say the new image is T. This can be represented as a column vector of dimension Nx1. This new image is mean centered by subtracting average face Ψ.

8.3. Calculate Face Key Vector

Each of such new objects submitted to the Object Recognition system is projected into the face space, obtaining the vector, also known as Face Key for this image, by using the following equation.

$\mathbf{\Omega_i = U^T (T - \Psi)}$

8.4. Classification of Input Image

The above vector with dimension (Mx1), is compared with each vector i representing average image for each class of images. If the distance found is minimum for any image i and the distance is inside the threshold of the class then there is a possibility of the image being belonging to image class i. This Euclidean distance between two image vectors can be calculated using the square minimal method given by following equation.

$\mathbf{\varepsilon_k = \| \Omega - \Omega_k \|}$

8.5. Object Recognition

The image can also be checked for a match with one of the existing images in the database. By finding the Euclidean distance between the new image and the images in the database, the match can be checked. This is very efficient technique as databases are of large sizes and checking the Euclidean distance between image vectors is a simple method for object recognition.

8.6. Threshold values

The proposal is to find one threshold for each class, looking for a better result for object recognition. The $\Theta_i$are the thresholds which define the maximum allowed distance between the new image submitted for recognition and for each class image. If the distance found between the new image and one of the classes is inside the class threshold, then the object has a match. This threshold value can be calculated as given below

$\Theta_c = ½ \max_{j,k} \{\|\Omega_j - \Omega_k\|\}$ j,k = 1,…,M

On this approach we use factor k from 1 to 10. If this factor is small (near to 1), we have a big false-positive rate and a little false-negative rate. Otherwise, if this factor is big (near to 10), we have a little false-positive rate and a big false-negative rate.

9. Conclusion

The Eigen face approach for Face Recognition process is fast and simple which works well under constrained environment. It is one of the best practical solutions for the problem of object recognition. Many applications which require image recognition do not require perfect identification, but just low error rate. So instead of searching large databases of faces, it is better to give small set of likely matches. By using Eigen face approach, this small set of likely matches for given images, can be easily obtained. For a given set of images, due to high dimensionality of images, the space spanned is very large. But in reality, all these images are closely related and actually span a lower dimensional space. By using Eigen face approach, high dimensionality is reduced. The Eigen faces are the eigenvectors of the covariance matrix representing the image space. The lower the dimensionality of this image space, the easier it would be for image recognition. Any new image can be expressed as a linear combination of these Eigen faces. This makes it easier to match any two images and thus image recognition. We have also seen that taking eigenvectors with higher M' Eigen values instead of all M eigenvectors enhances the speed of image recognition. The important part is making this choice of M' which will be crucial depending on the type of application and error rate acceptable.

10. References

1. M.Turk and A.Pentland, “Eigen faces for recognition,” Journal of Cognitive Neuroscience, vol. 3, no. 1, pp. 71–86, Mar. 1991.
2. L.I. Smith, “A tutorial on principal components analysis,” February 2002. URL http://www.cs.otago.ac.nz/cosc453/student_tutorials/principal_components.pdf.
3. Dimitri PISSARENKO, “Eigen Face-based Facial recognition,” a document based on Turk and Pentland (1991b), Turk and Pentland (1991a) and Smith (2002),Feb 13, 2003
4. T.F. Cootes, G.J. Edwards, and C.J. Taylor, “Active appearance models,” in Proc. European Conference on Computer Vision, 1998, vol. 2, pp. 484–498.
5. G.J. Edwards, T.F. Cootes, and C.J. Taylor, “Face recognition using active appearance models,” in Proc. European Conference on Computer Vision, 1998, vol. 2, pp. 581–695.
6. Lee Gomes. Can Facial Recognition Help Snag Terrorists? The Wall Street Journal, September 21, 2001.
7. Defense Advanced Research Projects Agency. Human Identification at a Distance, BAA00-29 edition, Feb 2000.
8. T.Kanade, Picture Processing by Computer Complex and Recognition of Human Faces, PhD thesis, Kyoto University, 1973.
9. A.L.Yullie, D.S.Cohen and P.W.Hallind, “Feature Extraction from faces using deformable templates,”Proc CVPR, San Diego, June 1989.
10. T.J Stonham, “Practical face Recognition and verifiaction with WISARD,” in H.Ellis, M.Jeeves, F.NewCombe and A.Young, Aspects of face processing, MartinusNijnoff publishers, Dordchett, 1986.
11. P.Burt, “Smart Sensing with a pyramid Vision Machine,” proc. IEEE Vol.76, No.8, Aug 1988.
12. J.B.Tenenbaum, V.de Silva, and J.C.Langford, “A global geometric framework for nonlinear dimensionality reduction,” Science, vol. 290, pp. 2319–2323, 2000.
13. S.T.Roweis and L.K. Saul, “Nonlinear dimensionality reduction by locally linear embedding,” Science, vol. 290, pp. 2323–2326, 2000.
14. Ming-Hsuan Yang, “Face recognition using extended isomap,” in Proc. IEEE International Conference on Image Processing, Rochester, New York, Sep. 2002, vol. 2, pp. 117–120.
15. P.J. Phillips, P. Grother, R.J Micheals, D.M. Blackburn, E Tabassi, and J.M. Bone, “FRVT 2002: Overview and summary,” March 2003
16. Ralph Gross, Jianbo Shi, and Jeffrey Cohn, “Quo vadis face recognition?,” in Proc. Third Workshop on Empirical Evaluation Methods in Computer Vision, Dec. 2001