World Journal of Science and Technology 2011, 1(12): 01-05
ISSN: 2231 – 2587
www.worldjournalofscience.com

PERFORMANCE AND EVALUATION OF FACE RECOGNITION ALGORITHMS

Arati Kothari[1] and Sampatkumari Maruti Bandagar[2]

[1]Department of Computer Science, Gulbarga University, Gulbarga, Karnataka, India
[2]Department of Computer Science, Govt. First Grade College, Chitguppa, Karnataka, India
Corresponding author e-mail: aratikothari@rediffmail.com

Abstract

Face recognition is a process of identifying or verifying a person from an image and comparing the selected features from the image with a given database. In this project, we implemented eigenface based face recognition and tried to compare the results with fisherface algorithm. The process required preprocessing. The images had to be resized to a consistent size. The database used included cropped faces of various sizes. Hence the need for face detection was eliminated. But the faces being of various sizes, they had to be resized to a smaller and consistent size. The face recognition process works quickly and consistently throughout a range of test images. We tried to compare two of the most frequently used algorithms; eigenface and fisherface. We compared the performance of each algorithm against two constraints, Pose and the size of training data. Our study has shown us that fisherface algorithm is robust in both cases. This leads us conclude that the eigenface algorithm is beneficial when the database is large. But given the robustness of the fisherface algorithm, it would be the algorithm of choice if the resources are not a problem. In this paper we implemented our own resizing algorithm, which resizes the input image to a constant size no matter what the dimensions are, i.e. an algorithm which performs both interpolation and extrapolation. We also tried exploring the possibility of limiting the average image to that of just 3-4 poses in fisherface method. But this gave good results only when the 3 images represented 3 profiles (and hence vary too much).

Keywords: Face recognition, Algorithms

Introduction

Face Recognition becomes one of the most biometrics authentication technologies for past few years. Face recognition is an application area where computer vision research is utilized in both military and commercial products. It is a process of identifying or verifying a person from an image and comparing the selected features from the image with a given database. Most commonly used facial recognition techniques / algorithms include eigenface, fisherface, hidden Markov model & dynamic link matching. Using 3-D facial recognition higher accuracy is being achieved lately. The process of getting the features from the face involves extracting the face from the rest of the image. Then the features (*Nodal Points*) such as the distance between the eyes, the shape of the cheekbones, width of the nose, depth of the eye sockets and other distinguishable features are obtained. These nodal points are then compared to the nodal points computed from a database of pictures in order to find a match.

Fig 1: Example of a Training Set

The original training set includes multiple poses of 50 individuals.
This image only shows 1 pose for 49 individuals.
The rest are not shown due to space constraints

The Eigenface approach

In this approach, the face images are decomposed into a small set of characteristic feature images called "eigenfaces" from the original training set of images by means of principal component analysis. An important feature of PCA is that any original image can be reconstructed from the training set by a linear combination of the eigenfaces. Each eigenface represents only certain features of the face. However, the losses due to omitting some of the eigenfaces can be minimized by choosing only the most important features (eigenfaces).

The eigenface approach involves the following initialization operations:

1. An initial set of images (training set, Figure 1) is acquired.
2. The eigenfaces from the training set are calculated and only M images that correspond to the highest eigenvalues (see Figure 2) define the face space.
3. By projecting the face images onto the face space, the corresponding distribution in M-dimensional weight space for each individual image is found.

With these weights, any image in the database can be reconstructed using the weighted sum of the eigenfaces (see Figure 3).

In order to recognize face images, the following steps are to be followed

1. A set of weights based on the input image and the M eigenfaces are calculated by projecting the input image onto each of the eigenfaces.
2. Nearest neighbor classification is used in order to find out the unknown image in the training set. (see Figure 5a, Figure 5b)

Initialization

Let the training set of face images be T1,T2,T3,….TM. This training data set has to be mean adjusted before calculating the covariance matrix or eigenvectors. The average face is calculated as $\Psi = (1/M)\ \Sigma 1MTi$

Each image in the data set differs from the average face by the vector $\Phi = Ti - \Psi$. This is actually mean adjusted data. The covariance matrix is

1) $C = (1/M)\ \Sigma\ 1M\ \Phi i\ \Phi iT = AAT$

where $A = [\ \Phi 1, \Phi 2, \ldots.\ \Phi M]$. The matrix C is a N2 by N2 matrix and would generate N2 eigenvectors and eigenvalues. With image sizes like 256 by 256, or even lower than that, such a calculation would be impractical to implement.

A computationally feasible method was suggested to find out the eigenvectors. If the number of images in the training set is less than the no of pixels in an image (i.e M < N2), then we can solve an M by M matrix instead of solving a N2 by N2 matrix. Consider the covariance matrix as ATA instead of AAT. Now the eigenvector vi can calculated as follows,

2) $ATAvi = \mu iv$

where μi is the eigenvalue. Here the size of covariance matrix would be M by M. Thus we can have m eigenvectors instead of N2. Premultipying equation 2 by A, we have

3) $AATAvi = \mu i\ Avi$

The right hand side gives us the M eigenfaces of the order N2 by 1.All such vectors would make the image space of dimensionality M. The M' eigenfaces which have the largest associated eigenvalues are

selected. These eigenfaces now span a M'-dimensional subspace instead of N2.

Recognition

A new image T is transformed into its eigenface components (projected into 'face space') by a simple operation,

4) wk = ukT (T – ψ)

where k = 1,2,….M'. The weights obtained as above form a vector ΩT = [w1, w2, w3,…. wM'] that describes the contribution of each eigenface in representing the input face image. The Euclidean distance of the weight vector of the new image from the face class weight vector can be calculated as follows

5) εk = || Ω – Ωk||

where Ωk is a vector describing the kth face class. The face is classified as belonging to class k when the distance εk is minimum.

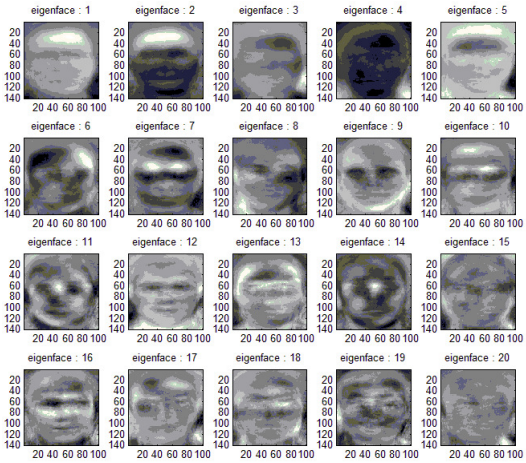

Fig 2. Eigen Faces: As *we can see each face is different from the other.*

Though they may not explicitly signify, in a way each image is storing a distinctive feature.

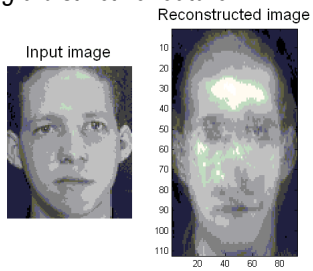

Fig 3. Reconstructed Image.

This does not look the same, but it does include all the significant features of the input image.

Fig 5a. good recognition

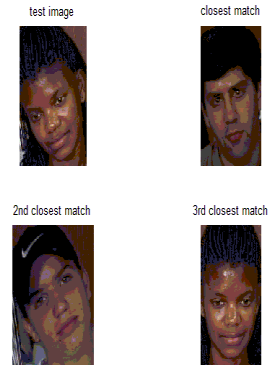

Fig 5b. not too good recognition

The variation in the matched faces can be attributed to pose.

In Figure 5a, the face pose is similar to one in the database; hence a good match is made.

In figure 5b, though the 3rd closest match is the same, the algorithm attributes pose as a more prominent feature. Hence the closest match is one with same pose.

The Fisherface approach

The fisherface approach is a widely used method for feature extraction in face images. This approach tries to find the projection direction in which, images belonging to different classes are separated maximally. Mathematically, it tries to find the projection matrix (the weights) in such a way that the ratio of the between-class scatter matrix and the in-class scatter matrix of projected images is maximized.

For a 'c' class problem, (where c is number of individuals) the between class scatter matrix is

$$S_b = \sum_{i=1}^{c} \Pr(\Omega_i)(\mu_i - \mu)(\mu_i - \mu)^T$$

The with-in class scatter matrix is given by :

$$S_w = \sum_{i=1}^{c} (\Pr(\Omega_i) \times \frac{1}{N_i} \sum_{y_k \in \Omega_i} (y_k - \mu_i)(y_k - \mu_i)^T)$$

for the images to be projected in such a way that they are maximal separated is solved by eigen value equation

$$S_b W = S_w W \Lambda$$

once the weights are obtained, they are used to project the images into the face space. After the weight basis is obtained, the recognition process is the same as in the case of eigenface algorithm.

Comparison between eigenface and fisherface

Comparison by size of training data

We tested both algorithms on 20 images for varying number of poses in training data. As we can see from the graph, the recognition of the images is better for fisherface than eigenface based algorithm when the number of poses is less. But as the number increases, the % of true recognition in both cases is almost the same. The graphical representation of this result can be viewed in Figure 6.

Comparisons by image pose

We tested both algorithms at their optimum working conditions on various poses of same image. At their optimum trained conditions (3 poses for fisherface, 6 for eigenface) the recognition was almost the same. These results can be viewed in Figure 7 as we can see, at their optimum working conditions, they hardly differ in their recognition of faces.

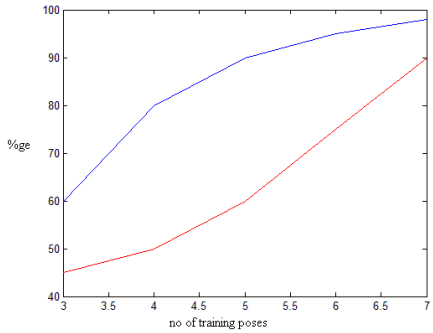

Fig 6. Plot of %ge recognition vs training poses. *Blue : fisherface. Red : eigenface.*

Fig 7a.Fisherface results

Fig 7b. Eigenface results

The matches are all good. But the fisherface was trained with less faces. Hence the closest face images differ.

Conclusion

We did not have the chance of testing the algorithms for light sensitivity which is a major issue in computer vision. The lack of suitable databases with light variance was the problem. In future, implementation to test for light sensitivity would be the obvious step. The main issue in these algorithms would be the one of robustness vs. simplicity. Due to the computational complexity of the fisherface algorithm, it might be hard to implement it in real time. And given a large database, and a dense training set, pose sensitivity of the eigenface algorithm would not be significant. Hence this would be a better choice of algorithm in most cases. But a possible combination of both algorithms which might include the simplicity of eigenfaces and robustness of fisherfaces would be a good step forward.

Acknowledgment

Georgia Technological University: Used their cropped face database for face recognition.

References

1. **M. Turk, A. Pentland,(1991).** Eigenfaces for Recognition, Journal of Cognitive Neurosicence, Vol. 3, No. 1, pp. 71-86

2. **P.N. Belhumeur, J.P. Hespanha, D.J. Kriegman, (1996).** Eigenfaces vs. Fisherfaces: Recognition using Class Specific Linear Projection, Proc. of the 4th EuropeanConference on Computer Vision, ECCV'96, 15-18 Cambridge, UK, pp. 45-58

3. Summary: "Eigenfaces for Recognition" (M. Turk, A. Pentland) – Ed Lawson, cs.gmu.edu/~kosecka/cs803/Eigenfaces.pdf

4. Obtaining the Eigenface Basis -- Jon Krueger, Marshall Robinson, Doug Kochelek, Matthew Escarra, http://cnx.org/content/m12531/latest/

5. **Zhang, J., Yan, Y., and Lades, M., (1997).** "Face Recognition: Eigenfaces, Elastic Matching, and Neural Nets", *Proc. IEEE*, vol.85, no.9, pp.1423-1435.

6. **Rowley, H., Baluja, S., and Kanade, T.,(1998).** "Rotation Invariant Neural Network-Based Face Detection", *Computer Vision and Pattern Recognition*, pp. 38-44.

7. **Gonzalez, R.C. Richard, E.W; (2004).** "Digital Image Processing, II Indian Edition, Pearson Education, New Delhi, India.

8. **Dr. P.S. Hiremath, Dr. B.V. Dhandra, Dr. Mallikarjun Hangarge, (2010).** "Recent Trends in Image Processing and Pattern Recognition", Excel India Publishers, New Delhi, India.

9. **Lin, S.-H., Kung, S.Y., and Lin, L.-J., (1997).** "Face Recognition/Detection by Probabilistic Decision-Based Neural Network", *IEEE Trans. Neural Networks*, vol. 8, No. 1, pp.114-132.

10. **Sung, K. and Poggio, T., (1995).** "Learning a Distribution-Based Face Model for Human Face Detection", Neural Networks for Signal Processing V, pp. 398-406, Cambridge, Mass.

11. **Miros,(1999).** "TrueFace Stands Alone on NT", http://www.miros.com/TrueFace_NT_Standalone_1_99.htm.