ISSN (Online) : 2278-1021
ISSN (Print) : 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 3, Issue 2, February 2014

Reduction in Computational Complexity-A Face Recognition Case Study

Radhika R Nayak[1], Dr. Sunil Surve[2], Prof. Sapna Prabhu[3], Nikita Agwankar[4]

Student, Electronics Engineering, Fr.Conceicao Rodrigues College of Engineering & Technology, Mumbai, India[1,4]

Professor- H.O.D., Computer Engineering, Fr.Conceicao Rodrigues College of Engineering & Technology, Mumbai, India[2]

Associate Professor, Electronics Engineering, Fr.Conceicao Rodrigues College of Engineering & Technology, Mumbai, India [3]

Abstract: Real time application like face recognition consist of loop carried dependencies. It includes a lot of data computation. As a result it leads to computational complexity. Due to computational complexity, execution time of a real time application like face recognition increases. Execution time of face recognition application can be reduced by changing the entire architecture of the system or it could reduce by modifying the existing algorithm. Changing the architecture of the system is not possible but modifications in the existing algorithm can be done. The main objective of this work is to reduce the execution time in a real time application like face recognition, thereby reducing the computational complexity. For this purpose, the existing algorithms used for face recognition is modified.

Keywords:- Eigen Values, Eigen vectors, Threshold, Euclidean distance, PCA, LDA, Computational Complexities, cannonical correlation, Factor Analysis, Covariance, Scaling, Execution time.

I. INTRODUCTION

There are many design constraints in the embedded system like power consumption, parallelism, size, memory etc. But the two design constraint that is of prime importance is power consumption and parallelism. But increase in parallelism without increase in power consumption leads to "computational complexity". Computational complexity is a branch of theory of computation in theoretical computer science and mathematics that focuses on classifying computational problems according to their inherent difficulty, and relating those classes to each other. A computational problem is a task that can be solved by applications of mathematical steps. A problem is difficult if it requires significant resources, whatever the algorithm used. So the solution for this problem is that to quantify the amount of resources needed to solve them, such as time and storage. A distinction between analysis of algorithms and computational complexity theory is that the former one looks into analyzing the amount of resources needed by a particular algorithm to solve the problem whereas the later one looks into all possible algorithms that could be used to solve the same problem.

II. BACKGROUND OVERVIEW

Factors which are contributing to the complexity of embedded systems are resource constraints, safety criticality, concurrency, and the time-dependent functionality required of the artifacts they control.

- The computational complexity arises due to resource limitations and architectural complexity [6,7] arises due to adaptation requirements.
- Representational complexity considers the representations of system model, system interaction and system behaviour [5].
- Cognitive complexity [2, 3] is a part of representational complexity.
- Architectural complexity [1, 5] is checked in terms of the software components interact through message passing mechanism. Internal complexity is the complexity of individual software component.
- The functional complexity [1, 7, 8] characterizes the dynamic performance of the system.

Main purpose of this work is to reduce the execution time in a real time application such as face recognition. The execution time or CPU time of a given task is defined as the time spent by the system executing that task, including the time spent executing run time or system services on its behalf. The mechanism used to measure time is implementation defined.

III. EXISTING ALGORITHMS

There are many existing algorithms which reduces the execution time in a real time application such as face recognition :-

- Factor Analysis
- Correspondence Analysis
- Cannonical Correlation
- Principal Component Analysis
- Linear Discriminant Analysis

Factor analysis:-

Factor Analysis was introduced by Charles Spearman. It is a statistical method used to describe variability among observed, correlated variables in terms of a potentially lower number of unobserved variables called factors. In other words, it is possible, for example, that variations in three or four observed variables mainly reflect the variations in fewer unobserved variables. Factor analysis searches for such joint variations in response to unobserved latent variables. The observed variables are modeled as linear combinations of the potential factors, plus “error” terms. The information gained about the interdependencies between observed variables can be used later to reduce the set of variables in a dataset. Computationally this technique is equivalent to low rank approximation of the matrix of observed variables. Factor Analysis is related to principal component analysis, but the two are not identical. Latent variable models, including factor analysis, use regression modeling techniques to test hypotheses producing error terms, while PCA is a descriptive statistical technique.

Correspondence Analysis (CA)

It is a multivariate statistical technique proposed by Hirschfield and later developed by Jean Paul Benzecri. It is a conceptually similar to principal components analysis, but applies to categorical data rather than continuous data. In a similar manner to principal component analysis, it provides a means of displaying or summarizing a set of data in two dimensional graphical form.All data should be non negative and on a small scale for CA to be applicable, and the method treats rows and column equivalently. It is traditionally applied to contingency tables CA decomposes the chisquared statistic associated with this table into orthogonal factors. Because CA is a descriptive technique, it can be applied to tables whether or not $\mathbf{X}^2$ statistic is appropriate.

Canonical Correlation

In statistics, canonical correlation analysis, introduced Harold Hotelling, is a way of making sense of cross-covariance matrices. If we have two sets of variables $x_1,x_2,\ldots x_n$ and $y_1,y_2\ldots\ldots y_m$ and there are correlations among the variables, then Cannonical correlation analysis finds linear combinations of the x’s and the y’s which have maximum correlation with each other.

Principal Component Analysis(PCA)

Principal Component Analysis (PCA), introduced by karl Pearson, is a mathematical procedure that uses an orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables called principal components. The number of principal components is less than or equal to the number of original variables. The transformation is defined in such a way that the first principal component has the largest possible variance and each succeeding component in turn has the highest variance possible under the constraint that it be orthogonal to the preceding components. Principal components are guaranteed to be independent only if the data set is jointly normally distributed.

PCA can be done by eigen value decomposition of a data covariance matrix or singular value decomposition of a data matrix for each attribute. The results of a PCA are usually discussed in terms of a component scores, called factor scores and loadings.PCA is the simplest of the true eigen vector based multivariate analysis.

Linear Discriminant Analysis (LDA)

LDA, introduced by R.A.Fischer, is also closely related to principal component analysis and factor analysis in that they both look for linear combinations of variables which best explain the data. LDA explicitly attempts to model the difference between the classes of data.PCA on the other hand does not take into account any difference in class, and factor analysis builds the feature combinations based on differences rather than similarities. Discriminant analysis is also different from factor analysis in that it is not an interdependence technique, a distinction between independent variables and dependent variables must be made. LDA works when the measurements made on independent variables for each observation are continuous quantities.

Comparison between Algorithms [11]

Algorithms	Execution time (in seconds)
Factor Analysis	65
Correspondence Analysis	60
Canonical Correlation	56
PCA	53
LDA	53

IV. THE PROPOSED ALGORITHM

A real time application face detection and recognition is taken to indicate that the execution time is reduced.

This Scheme is based on an information theory approach that decomposes face images into a small set of characteristic feature images called “eigenfaces” which may be thought of as principal components of the initial training set of face images. Recognition is performed by projecting a new image into the subspace spanned by eigenfaces i.e. facespace and then classifying the face by comparing its position in facespace with the positions of known individual.

In mathematical terms, the principal components of the distribution of faces or eigenvectors of the covariance matrix of the set of face images, treating an image as a point or vector in a very high dimensional space. The Eigen vectors are ordered, each one accounting for a different amount of the variation among the face images. These Eigen vectors can be thought of as a set of features that together characterize the variation between face images.

Each individual face can be represented exactly in terms of a linear combination of the eigenfaces. Each face can also be approximated using the eigenfaces- those that have the largest eigenvalues and which therefore account for the most variance within the set of face images.

The algorithm is described in following steps:-

Let a face image I(x, y) be a two dimensional N by N array of (8bit) intensity values. The data is arranged in a matrix form. The first step is to take the mean of first row and the subtracting the data and the mean of the first row itself. Then repeat the step for every row. This method is known as row mean method. These steps can be repeated for column also. This gives the matrix ϕ. This data is fairly simple and makes the calculation for the covariance matrix

ISSN (Online) : 2278-1021
ISSN (Print) : 2319-5940
International Journal of Advanced Research in Computer and Communication Engineering
Vol. 3, Issue 2, February 2014

easy. This step is basically done to retain the distinguishing features of the face and to eliminate the common features present in the image.

After performing the first step, calculation of a Covariance matrix is done. Covariance basically denotes the linear relationship between the two variable. Suppose that we have n registered images,. There are n pixels for any given pair of coordinates (i,j), one pixel at that location for each image. These pixels may be arranged in the form of a column vector

$$X=\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ . \\ . \\ x_n \end{bmatrix}$$

If the images are of size MxN, there will be total MN such n-dimensional vectors comprising of all pixels in an images. The mean vector,m_x, of a vector population can be approximated by the sample average:

$$. \; m_x = \frac{1}{K}\sum_{k=1}^{k} x_k \quad \text{with } K=MN$$

Similarly, the nxn covariance matrix,

$$C_x = \frac{1}{K}\sum_{k=1}^{k}(x_k - m_x)(x_k - m_x)^T$$

After obtaining the covariance matrix, the eigen values λ_k and eigen vectors X_k are calculated from the covariance matrix. Eigen values can be calculated as $|C-\lambda I|=0$ where C is a Covariance matrix and λ is a Eigen Value. Eigen vector can be calculated as $[C-\lambda I]X=0$ where C is a Covariance matrix and λ is a Eigen Value and X = $[x_1,x_2\ldots]$ is an Eigen vectors.
Scaling of Eigen vectors will only change the scale and not the direction. And hence, it will produce the same type of result. Scaling basically means to eliminate the least significant data. Scaling basically results into a compressed data. Scaling of Eigen vectors is done using following method:- If Eigen Vector is in a form

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ . \\ . \\ . \end{bmatrix} \text{ then scaled eigen vector is } \begin{bmatrix} \frac{x_1}{\sqrt{x_1^1+x_2^2+x_3^3\ldots}} \\ \frac{x_2}{\sqrt{x_1^1+x_2^2+x_3^3\ldots}} \\ . \\ . \end{bmatrix}$$

After scaling of the Eigen vectors are done, calculation of the feature vector is done. Once the Eigen vectors are found from the covariance matrix, the next step is to order them by Eigen value, highest to lowest. This gives the components in the order of significance. Here the data is compressed producing a lossy compression method, the data lost is deemed to be insignificant.

Then the transpose of the feature vector is multiplied by the tanspose of the data. This gives the transformed data points of the image. This gives the transformation matrix.
The simplest method for determining which face class provides the best description of an input face image is to find the face class k that minimizes the Euclidean distance. If p and q are the two pixels with coordinates (x_1,y_1) and (x_2,y_2), then $D_E=[(x_1-x_2)^2+(y_1-y_2)^2]^{1/2}$ A face is classified as belonging to class k when the minimum D_E is below some chosen threshold T. Otherwise the face is classified as unknown.

Steps for finding a threshold:-
Threshold is always predefined. It is always found on a trial and error basis.
Step1:-Read the given image.
Step2:- Plot the histogram of the given image.
Step3:- Based on the histogram, Chose the threshold T.

V. IMPLEMENTATION

In testing phase, ORL based images are selected from the data base and steps all the steps are carried out for training and recognition purpose.

VI. RESULTS AND DISCUSSION

The ORL database contains 45 different face images. The images vary in terms of lightning, facial expression with and without spectacles during different photo sessions.

VII. SIMULATION RESULTS

STEP	SAMPLE 1	SAMPLE 2
1		
2		
3		
4		
5		
6		
7		
Execution Time	**19 Sec**	**19 Sec**

In order to calculate the threshold, 10 samples were taken and by performing trial and error method on 10 samples, threshold 100 was selected. In order to indicate the results, 10 samples are taken.

Samples	**Threshold (T)**	**Execution time (sec)**
1	100	20
2	100	19
3	100	19
4	100	20
5	100	18
6	100	22
7	100	22
8	100	20
9	100	24
10	100	22
	$\frac{1}{10}\sum Executiont\,ime = 21\sec$	

Following inferences can be made from the results:
The average execution time for face recognition is 21 sec which indicates that the execution time is reduced and thus there is a reduction in computational complexity.

$$Accuracy = \frac{Noofsamplesrecognized}{Tota\ln oofsamples} = \frac{38}{40}x100$$
$$\therefore Accuracy = 95\%$$

VIII. CONCLUSIONS

- In the proposed method, face images are processed.
- The whole face is projected to find the weight vector or feature vector.
- Basically, this algorithm is used to reduce the dimensions of the data.
- And since the no of data is less, execution time is reduced.
- Reduction in the execution time indicates the reduction in computational complexity.
- It gives an accuracy of 95%.

IX. FUTURE WORK

1. The proposed method was found to perform satisfactorily in condition of exposure, illumination and contrast variations and face pose.
2. It can be improved by utilization of additional features like face rotation, gesture geometrical model and it may speed up the execution time.

X. REFERENCES

[1] [1] M. Zheng, and V. S. Alagar- Complexity of component based-development of embedded system, World Academy of Science, Engineering and Technology 8 2007.
[2] R.E.Gonzalez.Xtensa-A configurable and extensible processor. IEEE Micro, 2000.
[3] A. Ibrahim ACT: Adaptive Cellular Telephony Co-processor. Phd Thesis, University of Utah December 2005.
[4] C. Akturan and M. F. Jacome. Caliber: A software pipelining algorithm for clustered embedded VLIW processors. In Proceedings of the International Conference of Computer Aided Design (ICCAD), pages 112 to118, 2001.
[5] R. Gonzalez and M. Horowitz. Energy dissipation in general purpose microprocessors. IEEE Journal of Solid-State Circuits, 31(9):1277{1284, Sept. 1996.
[6] Application Driven Embedded System Design: A Face Recognition Case Study Karthik Ramani, Al Davis School of Computing, University of Utah Salt Lake City, UT 84112, USA.
[7] Alex M. Martinez and Avinash C Kak-Pattern analysis and Machine Intelligence, IEEE vol-23 N0.2 pp.228-233,2001.
[8] E. Waingold, M. Taylor, D. Srikrishna, V. Sarkar, W. Lee, V. Lee, J. Kim, M. Frank, P. Finch, R. Barua, J. Babb, S. Amarasinghe, and A. Agarwal. Baring it all to software: Raw machines. IEEE Computer, 30(9):86, 93, 1997.
[9] Yan Ke and Rahul Sukthankar and PCA-SIFT: A More Distinctive Representation for Local Image Descriptors, IRP-TR-03-15, and November 2003.
[10] Xiaofei He, Shuicheng Yan, Yuxiao Hu, Partha Niyogi, and Hong-Jiang Zhang and Face Recognition Using Laplacianfaces, IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, VOL. 27, NO. 3, MARCH 2005.
[11] Manuel Gunther, Roy Wallace, Sebastian Marcel, An Open Source framework for Standardized comparisons of face recognition algorithm, Idiap-RR-29-october-2012.