Face Recognition Using PCA, LDA and LPP

Krishnan Krishnamoorthy
40089054
Department of Computer Science and Software Engineering(CSSE),
Concordia University
Montreal,Canada.

Chetan Paliwal
40083388
Department of Computer Science and Software Engineering(CSSE),
Concordia University
Montreal,Canada.

Sandeep Siddaramaiah
40087428
Department of Computer Science and Software Engineering(CSSE),
Concordia University
Montreal,Canada.

Yogesh Nimbhorkar
40093384
Department of Computer Science and Software Engineering (CSSE)
Montreal, Canada

Karthik Bekal Pattathana
40094485
Department of Computer Science and Software Engineering(CSSE),
Concordia University
Montreal,Canada.

Shunyu Wang
40043915
Department of Computer Science and Software Engineering(CSSE),
Concordia University
Montreal,Canada
shunyu.wang@mail.concordia.ca

Girish Kumar Kadapa
40083533
Department of Computer Science and Software Engineering (CSSE)
Montreal, Canada

Darwin Anirudh G
40093368
Department of Computer Science and Software Engineering(CSSE),
Concordia University
Montreal,Canada.

***Abstract* —We intend to implement PCA, LDA and LPP for face recognition. We train the system by the training data to obtain feature space "eigenfaces" through PCA, the feature space "fisherfaces" through LDA and the feature space "laplacianfaces" through LPP. Once the feature space FS is obtained, training faces are projected to subspace defined by FS to construct a Face Database. When an unknown face is needed to recognize, this test face is firstly projected onto subspace FS. Afterward, the program finds the K nearest neighbors of the projected data in Face Database. Finally, the class label is assigned to the test face according to the majority vote among the neighbors. This classification algorithm is known as K-nearest neighbor.**

I. INTRODUCTION

There are many successful face recognition techniques have been developed over the past and mainly they can be categories into two groups based on how they represent faces [1]: Appearance-based, which depends on holistic texture features to either whole –face or specific regions in face image. Feature-based, which uses geometric facial features and geometric relationships. In the appearance-based techniques, two popular techniques for this purpose are Principal Component Analysis and Linear Discriminant Analysis.

PCA is a dimension reduction method which projects n dimensional data onto k dimensional subspace where k is smaller than n, and the k dimensional subspace is defined by leading eigenvectors of the data's covariance matrix. In 1991, Turk and Pentland achieved face recognition by projecting face images onto a feature space called "eigenfaces" through PCA [2]. LDA, on the other hand, is a supervised learning algorithm. The aim of LDA is to find a linear combination of features which separate different classes of objects. In other words, LDA is supposed to find a subspace on which objects in different classes are far away from each other while requiring objects in the same class are close to each other. In 1997, Belhumeur, Hespanha and Kriegman applied the linear discriminant analysis to face recognition [3].

However, both PCA and LDA only effectively see the Euclidean Structure. They fail to discover the underlying structure which might be a nonlinear submanifold [4]. In 2005, He, Yan, Hu, Niyogi and Zhang proposed a new approach which considers the manifold structure for face analysis through Locality Preserving Projections (LPP) [4]. They used LPP to find the face subspaces which they called as Laplacianfaces to achieve face recognition.

In this project, the above three methods have been successfully implemented. After the system is trained by the training data, the feature space "eigenfaces" through PCA, the feature space "fisherfaces" through LDA and the feature space "laplacianfaces" through LPP are found using respective methods. Later in this report, W is used to represent the obtained feature space. Once W is obtained, training faces are projected to subspace defined by W to construct FaceDB. When an unknown face is needed to recognize, this test face is firstly projected onto subspace W. Afterward, the program finds the K nearest neighbors of the projected data in FaceDB. Finally, the class label is assigned

to the test face according to the majority vote among the neighbors. This classification algorithm is known as K-nearest neighbor.

The rest of this report is organized as follows: Section II, III, IV describes PCA, LDA and LPP respectively. KNN algorithm is talked about in Section V. In Section VII, the whole process of face recognition is discussed. Section VIII presents the design and architecture of this system.

II. Principal Component Analysis (PCA)

The curse of dimensionality refers to various phenomena that arise when analyzing and organizing data in high dimensional spaces (often with hundreds or thousands of dimensions) that do not occur in low-dimensional settings such as the three-dimensional physical space of everyday experience. One approach to coping with the problem of excessive dimensionality of the image space is to reduce the dimensionality by combining features.The goal of PCA [6] is to reduce the dimensionality of the data while retaining as much as possible of the variation present in the original dataset.

Let X = {x1,x2,x3,...} be the matrix containing face images. Each image matrix is converted into a vector.The subspace "eigenfaces" could be obtained by following the steps. Calculate the mean and covariance using the formulas given below:

$$\mu = \frac{1}{n}\sum_{i=1}^{n} x_i$$

Let S be the covariance matrix.

$$S = \frac{1}{n}\sum_{i=1}^{n} (x_i - \mu)(x_i - \mu)^T$$

Then we proceed to compute the eigenvectors v_i and eigenvalues λ_i as given below :

$$Sv_i = \lambda_i v_i$$

Sort the eigenvectors in descending order by their eigenvalues and choose k eigenvectors corresponding to their k largest eigenvalues.The k eigenvectors form the subspace which is referred as "eigenfaces". In Spite of getting eigenfaces , we are still left a herculean task of applying faces to the covariance matrix S, where we end up with 10304 X 10304 matrix (since each image is database is 92 X 112 meaning that are 10304 dimensions to work with) which is really huge and can not be processed on regular laptops. Therefore we make slight modifications to reduce the computational task by defining a matrix Y as :

$$Y = X - U$$

where $U = \{\mu, \mu, \dots, \mu\}$.

Through this modification, we have transformed the original data set into data set with zero mean Now finding the eigenvectors and eigenvalues of $Y^T Y$:

$$Y^T Y v_i = \lambda_i v_i$$

Then the original eigenvector are calculated by a left multiplication of the data matrix Y:

$$YY^T(Yv_i) = \lambda_i(Yv_i)$$

Compute the normalized eigenvectors. Finally, the k eigenvectors corresponding to k largest eigenvalues are obtained and they form W which is the subspace and column vectors of W is called eigenfaces.Usually, the number of training samples is smaller than the dimensions of each face. As a result, the above method is needed to avoid the computational problem mentioned above. *But how do we find best lower dimensional space ?* The best low-dimensional space can be determined by the "best" eigenvectors of the covariance matrix of x (i.e., the eigenvectors corresponding to the "largest" eigenvalues -- also called "principal components").

III. Linear Discriminant Analysis (LDA)

Linear Discriminant Analysis (LDA) [7] is a generalization of Fisher's linear discriminant, a method used in Statistics, pattern recognition and machine learning to find a linear combination of features which characterizes or separates two or more classes of objects or events. This method projects a dataset onto a lower-dimensional space with good class-separability to avoid overfitting ("curse of dimensionality"), and to reduce computational costs. The resulting combination may be used as a linear classifier or, more commonly, for dimensionality reduction before subsequent classification.In a nutshell, the goal of a LDA is often to project a feature space(a data set n dimensional space) into a smaller sub space k *(where $k <= n-1$)* while maintaining the class-discriminatory information. In general, dimensionality reduction does not only help to reduce computational costs for a given classification task, but it can also be helpful to avoid overfitting by minimizing the error in parameter estimation. Let us briefly look at the methodology of LDA.

Let X be the matrix containing training face images and X_i be the matrix containing face images belonging to class j .

$$X = \{X_1, X_2, \dots, X_c\}$$

$$X_i = \{x_1, x_2, \dots, x_n\}.$$

The scatter matrices S_B and S_w are calculated as follows:

$$S_B = \sum_{i=1}^{c} N_i(\mu_i - \mu)(\mu_i - \mu)^T$$

$$S_w = \sum_{i=1}^{c} \sum_{x_j \epsilon X_i} (x_j - \mu_i)(x_j - \mu_i)^T$$

where N_i represents the number of training samples belonging to class 'i', μ_i represents the mean of training samples belonging to class label i and μ represents the mean of the total samples. In order to find a subspace where S_B is maximized and S_w is minimized, the following general eigenvalue problem needs to be solved.

$$S_B v_i = \lambda_i S_w v_i$$

However, one is confronted with the difficulty that the within-class scatter matrix S_w is always singular. In order to solve this problem, it is suggested that PCA first projects the image set to a lower dimensional space so that the resulting within-class matrix S_w is no longer singular matrix. After this process, the general eigenvalue problem becomes:

$$S_w^{-1} S_B v_i = \lambda_i v_i$$

By solving the above eigenvalue problem, W_{fld} is obtained which defines a subspace. Please note that there are at most c - 1 nonzero generalized eigenvalues for the above formula. Also, note that PCA can directly reduce the dimension of the feature space to N - c because the rank of S_w is at most N - c.

Hence N is the number of samples while c is the number of classes. By combining the transformation matrix of PCA, the below transformation matrix for LDA is obtained:

$$W = W_{pca} W_{fld}$$

The column vectors of W are the so-called fisherfaces.

IV. Locality Preserving Projections (LPP)

Locality Preserving Projections (LPP) are linear projective maps that arise by solving a variational problem that optimally preserves the neighborhood structure of the data set.

The steps to perform LPP are as follows:

A. PCA projection

We first project the training image set to the PCA [8] subspace with ignoring components with relatively less eigenvalues. We make sure that XDX^T is not singular which is important to decide later a generalized eigenvector problem.

B. Constructing the nearest-neighbor graph

If there are n nodes in graph G. The n^{th} node corresponds to the face image x_i and x_j. First we will draw edge between x_i and x_j if x_i and x_j are "nearby". With this we are checking whether two nodes are close enough or not using k-nearest neighbor.. For example, edges will be drawn between x_i and its k neighbors found by KNN.

C. Choosing the weights

We try to get the weight of the edges. If node x_i and x_j are connected, add

$$S_{ij} = 1$$

Else, add $S_{ij} = 0$. Make sure that there can be other more efficient way for measuring the weight of the edges.

D. Eigenmap

For below generalized eigenvector problem, we calculate the eigenvectors and eigenvalues:

$$XLX^T w = \lambda XDX^T w$$

where, X is the training image set, D is a diagonal matrix whose entries are column sums of S and L=D-S.

After solving generalized eigenvalue problem in above step, W_{LPP} is obtained. By combining this with the transformation matrix in PCA part, the following result for the LPP in the form of transformation matrix is obtained :

$$W = W_{pca} W_{LPP}$$

The column vectors of resultant W are so-called Laplacianfaces.

V. K-Nearest Neighbor Classification Algorithm

The k-nearest neighbors (KNN) algorithm is used to solve both classification and regression problems. It is a method for classifying objects based on closest training objects: an object is classified according to a majority vote of its neighbors. In this project, we are using KNN algorithm in the last phase that is recognition phase. Following are the steps of performing KNN:

A. Find K nearest neighbors

We first project the test face into the subspace defined by W , and W is calculated with the help of PCA, LDA and LPP. Then we keep the K nearest neighbors into the collection, then traverse through all the collection faces finding the distance and update the collection by checking if there is a

training face whose distance is smaller than the largest distance in the original collection. At the end of this process, the K nearest neighbors are found. Also, we have already projected the training set in subspace defined by W.

B. Classify according to weights

Once we get K nearest neighbors, a Map collection is created to keep the pairs <label, weight> by going through all the neighbors. If the Map gets new labels, <label, 1 / distance> is put into the map. Otherwise, the new updated version which adds the original weight with 1 / distance is put into the map collection. After the Map collection is populated, this program will traverse the map to search the label associated with the largest weight. This class label is the desired label recognized by this application.

Other thing to mention here is that, we have implemented three different metric methods cosine similarity, L1 distance and Euclidean distance respectively in this application. In further sections, comparisons between these methods are discussed.

VI. Face Recognition Methodology

As most supervised learning methodologies do, there are two phases, which are training and recognizing. The methodology uses PCA, LDA, LPP and KNN in this training and recognizing process [10].

A. Training

The training process involves refining projection matrix W using labelled sample images by using PCA, LPP and LDA algorithm respectively. The inputs are the difference between the mean face and the normalized training face. Each algorithm produces the projection matrix W whose size is restricted by the number of most important components retained.

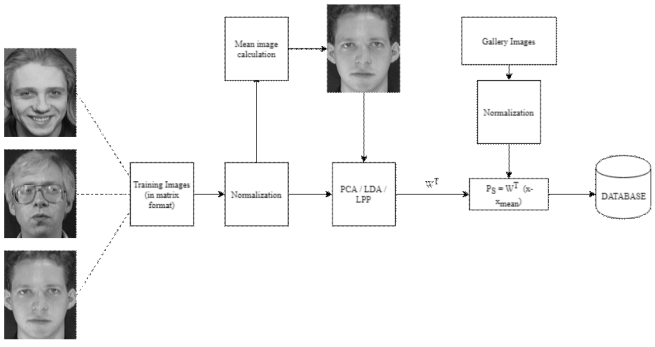

Fig. 1. Training Phase illustration.

Once projection matrix W is learned, gallery images are projected into the subspace defined by the projection matrix, and those coordinate attributes are stored in a database in order for retrieval and comparison in next phase.

B. Recognition

During recognizing a face image, the projection matrix retrieved above is used to project the difference between the mean face and the normalized matrix of the said face image. It enurses the uniformity of inputs between training and recognizing.

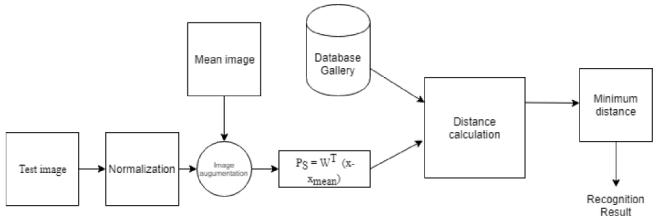

Fig. 2. Recognition phase illustration.

After a recognizing image in projected into the subspace, KNN algorithm takes it as an input to find its N nearest neighbours so that it could be classified into a class according to the majority vote.

More Java Implementation details for the methodology will be provided in the next section.

VII. Implementation

A. Math Operation Class

Since there are heavy matrix operations, we define some math classes to achieve mathematical convenience, such as Matrix, LU decomposition, QR decomposition, Eigenvalue decomposition and etc.

B. Distance Methods

Three implementations, known as Euclidean distance, L1-Distance, and Cosine dissimilarity, implement Metric interface in which getDistance method is contracted. This method calculates the distance between two matrices, **p** and **q**.

a. EuclideanDistance. it is the most common distance calculation seen in two dimensional space. We expand it into multi-dimensional space.

$$d(\boldsymbol{p},\boldsymbol{q}) = \sqrt{\sum_{i=1}^{n}(p_i - q_i)^2}$$

b. L1-Distance. Norm 1 distance.

$$d(p,q) = \sum_{i=1}^{n}|p_i - q_i|$$

c. CosineDissimilarity

$$\cos(\boldsymbol{p},\boldsymbol{q}) = \frac{\boldsymbol{p}^T\boldsymbol{q}}{|\boldsymbol{p}||\boldsymbol{q}|}$$

1/cos(p,q) is returned due to the inverse relationship between the cosine value of two matrices and their distance. The larger the cosine value is, the closer these two matrices are, and also the smaller the return value is.

C. KNN

The class has three methods. The method *assignLabel* assigns labels to projected matrices. The method *findKNN* find its k nearest neighbourhoods in the form of array of projected training matrix. The method *classify* could classify a testing projected matrix into a class indicated by its label.

D. File Manager

The file manager is used to handle all images to or from matrix conversion. Given a picture, the method *convertPGMtoMatrix* is used to present an image in a matrix form. Inversely, given a matrix which in our case is the projected matrix after transformation by PCA, LDA or LPP algorithm, the method *convertMatrixtoPGM* would generate an image file.

E. Feature Extraction

We define an interface called Extraction to extract common operations of PCA, LDA, and LPP algorithms. Thus, we create one class for each algorithm to implement the interface. The benefit of doing this is we can dynamically bind an object of one kind of implementation to a variable. Therefore, the training and testing configuration is easy.

F. Test

There are three parameters to set in order to test an image. They are the number of components retained, the K value for KNN, the feature extraction type. Given the test image as input, the program predicts its class label. We keep doing the testing based on the same trained result, and calculate accuracy rate for each combination of parameters.

VIII. Results and Discussion

8.1 AT&T dataset (ORL Images):

We used the ORL face database composed of 400 images of size 112 x 92. There are 40 persons, 10 images for each face. The images were taken at different times, lighting and facial expressions. The faces are in an upright position in frontal view, with a slight left-right rotation.

The Table.1. illustrates the results which we observed for AT&T dataset. Here, we can see that LPP consistently outperforms both PCA and LDA. Here, we observe very high classification results because of the comparatively small dataset with relatively small dimensions. Do note because we are using cross validations we would get approximately equivalent results but not the same. We performed a 70-30 train-test split for the following dataset.

KNN Value	Accuracy		
	PCA	LDA	LPP
1	93.3%	94.2%	98.3%
2	94.2%	97.5%	99.2%
3	92.5%	91.7%	98.3%

Table. 1. Performance on AT&T Dataset [11] for the project.

8.2 Yale Database

The Yale face database was constructed at the Yale Center for Computational Vision and Control. It contains 165 grayscale images of 15 individuals. The images demonstrate variations in lighting condition (left-light, center-light, rightlight), facial expression (normal, happy, sad, sleepy, surprised, and wink), and with/without glasses.

The Table.2. illustrates the results which we observed for Yale database. Again, we can see that LPP consistently outperforms both PCA and LDA. Here, we observe a better distinction between PCA, LDA and LPP confirming our hypothesis. .

Again, we performed a 70-30 train-test split for the following dataset..

KNN Value	Accuracy		
	PCA	LDA	LPP
1	75.5%	82.2%	88.8%
2	77.7%	77.7%	100%
3	73.3%	82.2%	91.1%

Table. 2. Performance on Yale Database[5] for the project.

The remaining samples of the database were considered to be the testing set for both the dataset. The training samples were used to learn the Laplacianfaces. The testing samples were then projected into the low-dimensional representation subspace. Recognition was performed using a nearest-neighbor classifier.

Some examples we fed into the process show that LPP can have more power to differentiate than PCA and be less susceptible to outliers. Many experiments were performed in this segment to illustrate the efficacy of our proposed method for face representation and recognition of Laplacianfaces.

Three experiments on two dataset have been systematically performed. These experiments reveal a number of interesting points:

A. All these three approaches performed better in the optimal face subspace than in the original image space.

B. Among the experiments we performed Laplacian faces consistently outperformed Eigenfaces [9] and Fisherfaces. The results were better for AT&T (ORL Dataset) compared to Yale dataset as the number of samples and classes were higher in AT&T dataset[5]. Hence, this shows that the accuracies improved with number of training samples. The results shown are shown in Table 1.

C. As compared to PCA (Eigen faces) and LDA(Fisher faces) we find that Laplacian faces is able to encode more finer features for finding distinctions as it preserves the local structure which is more important than the global features. Hence, Laplacianfaces can identify the faces in a variety of different lighting conditions, poses or maybe even different expressions.

D. Also, Laplacianfaces has a major advantage with respect to LDA which requires more than one sample per class. In such a case, the scatter matrix of the S_W becomes a zero matrix. For LPP, we can construct a complete graph and use the inner products as weights similar to PCA.

IX. Conclusions and Summary

This paper incorporates multiple ways of face analysis (representation and recognition) to identify the underlying nonlinear manifold structure in the manner of linear subspace learning. Laplacianfaces approach can be described by the following, the structure of the manifold is approximated by the graph calculated from the data points. Then we calculate a transformation matrix using the graph's Laplacian notion which maps the face images into a face subspace. The Laplacianfaces are obtained by finding the optimal linear approximations to the eigenfunctions of the Laplace Beltrami operator on the face manifold. This linear transformation optimally preserves local manifold structure. The LPP algorithm's conceptual analysis and its ties to PCA and LDA are provided. AT&T and Yale databases experimental results show the effectiveness of LPP method.

One of the key issues in face manifold learning is to estimate the intrinsic dimensionality of the nonlinear face manifold. We know that the dimensionality of the manifold is equal to the dimensionality of the local tangent space. Therefore, estimating the dimensionality of the tangent space is one possible area where we can focus on in the future.

One other possible extension of this work is to consider the unlabelled samples. Discovering the underlying manifold is the principal technique being followed to analyse the facial data. And discovering the underlying manifold is essentially unsupervised learning process. Because of the ease with which we can obtain greater number of samples of unlabelled data, finding the face manifold becomes easier. Since the face images are believed to reside on a submanifold embedded in a high-dimensional ambient space, we believe that the unlabeled samples are of great value.

References

[1] Delac, K., Grgic, M., & Grgic, S. (2005). Independent comparative study of PCA, ICA, and LDA on the FERET data set. International Journal of Imaging Systems and Technology, 15(5), 252-260.

[2] Turk, M., & Pentland, A. (1991). Eigenfaces for recognition. Journal of cognitive neuroscience, 3(1), 71-86.

[3] Belhumeur, P. N., Hespanha, J. P., & Kriegman, D. J. (1997). Eigenfaces vs. fisherfaces: Recognition using class specific linear projection. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 19(7), 711-720.

[4] He, X., Yan, S., Hu, Y., Niyogi, P., & Zhang, H. J. (2005). Face recognition using laplacianfaces. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 27(3), 328-340.

[5] Yale Univ. Face Database, http://cvc.yale.edu/projects/yalefaces/yalefaces.html, 2002.

[6] M. Turk and A.P. Pentland, "Face Recognition Using Eigenfaces," IEEE Conf. Computer Vision and Pattern Recognition, 1991

[7] P.N. Belhumeur, J.P. Hespanha, and D.J. Kriegman, "Eigenfaces vs. Fisherfaces: Recognition Using Class Specific Linear Projection," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 19, no. 7, pp. 711-720, July 1997.

[8] Levin and A. Shashua, "Principal Component Analysis over Continuous Subspaces and Intersection of Half-Spaces," Proc. European Conf. Computer Vision, May 2002.

[9] M.-H. Yang, "Kernel Eigenfaces vs. Kernel Fisherfaces: Face Recognition Using Kernel Methods," Proc. Fifth Int'l Conf. Automatic Face and Gesture Recognition, May 2002.

[10] A.M. Martinez and A.C. Kak, "PCA versus LDA," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 23

[11] AT&T Database of Faces http://www.face-rec.org/databases/