Chapter 3
METHODOLOGY

Over last decades, various feature extraction and pattern classification methods have been proposed for the task of face recognition. For decades, feature-based methods use distances and angles between facial features such as eyes, nose, mouth and chin for face recognition. Despite their economical representation and insensitivity to small variations in pose and illumination, feature-based methods are quite sensitive to the feature extraction and measurement process. It has been argued that the existing facial feature extraction and measurement techniques are not quite reliable [1]. It has been claimed that the existing face recognition methods based on facial feature extraction and establishing identity using geometric relationship between these features are not very effective [2].

Over the last decade, several researchers have introduced face recognition methods those use low-dimensional representations of face images. These methods are called as appearance-based methods. They differ from feature-based methods in the sense that their low-dimensional representation, in a least square sense, is faithful to the original image. Power of appearance-based methods is demonstrated by techniques like eigenfaces in ease of implementation and in accuracy. For classification, these methods use feature vector which is obtained by projecting face images onto lower dimensional subspace. In extreme cases, the feature vector is chosen as the entire image, with each element in the feature vector taken from a pixel in the image [3].

Among the various approaches to face recognition, appearance-based subspace analysis, although one of the oldest, gives the most promising results [4] Examples of appearance-based subspace analysis methods are Principal Component Analysis, Linear Discriminant Analysis and Independent Component Analysis methods.
3.1 Principal Component Analysis

3.1.1 Eigenfaces for Recognition:

In terms of information theory, for the task of face recognition, one wants to extract the relevant information from the face, encode it as efficiently as possible and compare it with the encoding of other faces in the database. Simple approach to extract relevant information from face is to capture the variation in the images of dataset and encode this information to compare with other face images. In mathematical terms, one wishes to find principal components of the distribution of faces, or the eigenvectors of the covariance matrix of all face images. These eigenvectors can be thought of as a set of features which together characterize the variation in the face images. Each image location contributes to each eigenvector and hence the eigenvector can be displayed as a ghostly face, which is called as *eigenface*. Each face image in the database can be represented as a linear combination of all eigenfaces. The number of eigenfaces is equal to the number of images in the training dataset. However, face can be approximated by considering only the “best” eigenfaces i.e. those corresponding to the largest eigenvalues, and thus accounting for the most variance among the set of face images. The main reason for using less eigenfaces is computational efficiency. These best $M'$ eigenfaces span $M'$ dimensional subspace which is called as “facespace” of all images [5,6].

**Initialization operations for face recognition using eigenfaces are as follows:**

1. Acquire the training set of facial images.
2. Calculate the eigenfaces from the given training set. Keep only the $M'$ eigenfaces corresponding to the most significant eigenvalues. These $M'$ eigenfaces define facespace. The eigenfaces can be recalculated or updated on arrival of new face images.
3. Calculate the corresponding distribution in $M'$ dimensional weight space for each person by projecting their face images onto the facespace.

**The following steps are used to recognize new face images:**

1. For each image, calculate set of weights by projecting that image on each of the eigenfaces.
2. Determine if the image is a face at all by checking if it lies sufficiently close to the “facespace”.

3. If it is a facial image, then classify it as either known person or as unknown.

Eigenfaces method decomposes face images into a small set of characteristic feature images called "eigenfaces", which are principal components of all the images in the training set. Recognition is done by projecting a test image into the subspace spanned by the eigenfaces ("facespace") and then the test image is classified by comparing its position in face space with the positions of known individuals.

### 3.1.2 Calculating Eigenfaces:

Let a face image $I(x,y)$ be a 2-dimensional $N$ by $N$ array of intensity values, or a vector of dimension $N^2$. For example, an image of size 256x256 can be considered as a vector of dimension 65,536 or a point in 65,536 dimensional space. An ensemble of images map to a collection of points in this space. Face images being similar in configuration are not distributed randomly in this space and can be described using low-dimensional subspace. The main idea of PCA is to find the vectors which best account for the distribution of face images in this space. These vectors define the subspace of face images which is called as facespace. Each vector is of size $N^2$ and is linear combination of all face images. These vectors are nothing but the eigenvectors of the covariance matrix of all the images in the training set. As they look like face images, they are called as “eigenfaces”.

### 3.1.3 Face Recognition System using PCA Approach:

- A face image in 2-dimension with size $N$ by $N$ can also be considered as one dimensional vector of dimension $N^2$. Let the training set of face images be $\Gamma_1, \Gamma_2, \ldots, \Gamma_M$. The average face is defined by
  \[ \Psi = \frac{1}{M} \sum_{n=1}^{M} \Gamma_n. \]  
  \hspace{1cm} (3.1)

- Each face differs from the average by the vector
  \[ \Phi_l = \Gamma_l - \Psi. \]  
  \hspace{1cm} (3.2)
This set of very large vectors is then subject to PCA, which seeks a set of $M$ orthonormal vectors, $\mathbf{u}_n$, which best describes the distribution of the data. The vectors $\mathbf{u}_n$ are the eigenvectors of the covariance matrix

$$C = \frac{1}{M} \sum_{n=1}^{M} \Phi_n \Phi_n^T = AA^T$$  \hspace{1cm} (3.3)

where the matrix $A = [\Phi_1 \Phi_2 \ldots \Phi_M]$. The matrix $C$ is $N^2$ by $N^2$, and determining the $N^2$ eigenvectors and eigenvalues is an intractable task for typical image sizes. Therefore Turk and Pentland proposed a computationally feasible method to find these eigenvectors.

Consider the eigenvectors $\mathbf{v}_i$ of $A^T A$ such that

$$A^T A \mathbf{v}_i = \mu_i \mathbf{v}_i.$$  \hspace{1cm} (3.4)

Pre multiplying both sides by $A$, we have

$$A A^T A \mathbf{v}_i = \mu_i A \mathbf{v}_i$$  \hspace{1cm} (3.5)

where $A \mathbf{v}_i$ are the eigenvectors and $\mu_i$ are the eigenvalues of $C = AA^T$. Then construct the $M$ by $M$ matrix $L = A^T A$, where $L_{mn} = \mathbf{\Phi}_m^T \mathbf{\Phi}_n$ and find the $M$ eigenvectors, $\mathbf{v}_i$, of $L$. These vectors determine linear combinations of the $M$ training set face images to form the eigenfaces $\mathbf{u}_l$.

$$\mathbf{u}_l = \sum_{k=1}^{M} v_{lk} \mathbf{\Phi}_k \hspace{1cm} l = 1, \ldots, M$$  \hspace{1cm} (3.6)

This greatly reduces calculations from the order of the number of pixels in the images ($N^2$) to the order of the number of images in the training set ($M$).

Using associated eigenvalues, eigenvectors can be ranked according to their usefulness in characterizing the variation among the images.

**Classification of a Face Image using Eigenfaces**

- The eigenvectors of matrix $L$ form the basis set which describes the face images. Practically, smaller number of $M'$ eigenvectors are enough for performing identification, since exact reconstruction of face image is not required. Most significant eigenfaces form $M'$ dimensional subspace of original $N^2$ space.
Methodology

- Choose $M'$ significant eigenvectors of the $L$ matrix corresponding to largest eigenvalues. A new face image ($\Gamma$) is projected onto "face space" by using following operation,

$$w_k = u_k^T (\Gamma - \Psi) \text{ for } k = 1,...,M'.$$

(3.7)

- The weights form following projection vector,

$$\Omega^T = [w_1, w_2, ..., w_{M'}].$$

(3.8)

It describes the contribution of each eigenface in representing the input face image. Eigenfaces are treated as a basis set for face images.

- This projection vector is used to find which of the class describes the face best. Classification is done by comparing projection vector of test image with projection vectors of all known classes using some similarity measure. For example, if Euclidean distance is used as a similarity measure than class $k$ is found which minimizes the Euclidean distance.

$$\varepsilon_k^2 = \| \Omega - \Omega_k \|^2$$

(3.9)

where $\Omega_k$ is a projection vector of $k^{th}$ face class. The $\Omega_i$ of a class is found by averaging the eigenfaces of all images of a particular individual. Test face image is classified as belonging to class $k$ when minimum $\varepsilon_k$ is below some threshold $\theta_k$. Otherwise the face is classified as "unknown".

3.2 Linear Discriminant Analysis or Fisherfaces Method

This method overcomes the limitations of eigenfaces method by applying the Fisher’s linear discriminant criterion. Fisher discriminants group images of the same class and separates images of different classes [7,8].

Consider two sets of points in two-dimensional space that are projected onto a single line. Depending on the direction of the line, the points can either be mixed together (fig. 3.1(a)) or separated (fig. 3.1(b)). Fisher discriminants find the line that best separates the points as in fig. 3.1(b).
Development of an Efficient Face Recognition System by Fusion of Appearance-based Classifiers

Figure 3.1. (a) Points mixed when projected onto a line; (b) Points separated when projected onto another line.

Figure 3.2. (a) Good class separation; (b) Bad class separation.

Fisher Linear Discriminant (FLD) is a pattern recognition [9] technique developed by R. Fisher [10] in 1936 for taxonomic classification. It has been successfully applied in computer vision and even in face recognition. FLD is a class-specific technique i.e. it tries to shape the scatter so as to make it reliable for classification. It attempts to maximize the ratio of the between-class scatter ($S_b$) to within-class ($S_w$) scatter. $S_w$ and $S_b$ are

$$S_w = \sum_{j=1}^{c} \sum_{i=1}^{N_j} (\Gamma_i^j - \mu_j) (\Gamma_i^j - \mu_j)^T$$

(3.10)

where $\Gamma_i^j$ is the $i^{th}$ sample of class $j$, $\mu_j$ is the mean of class $j$, $C$ is the number of classes, $N_j$ is the number of samples in class $j$.

$$S_b = \sum_{j=1}^{c} (\mu_j - \mu)(\mu_j - \mu)^T$$

(3.11)

where $\mu$ represents the mean of all classes.

If $S_w$ is nonsingular, the optimal projection $W_{opt}$ is chosen as the matrix which maximizes the ratio of the determinant of the between-class scatter matrix of the
projected samples to the determinant of the within-class scatter matrix of the projected samples, i.e.

$$W_{opt} = \arg \max_w w^T S_b w = \left[ w_1 \ w_2 \ldots \ w_m \right]$$

(3.12)

where \{w_i| i = 1, 2, \ldots, m\} is the set of generalized eigenvectors of \(S_b\) and \(S_w\) corresponding to set of decreasing generalized eigenvalues \{\lambda_i| i = 1, 2, \ldots, m\} i.e.

$$S_b w_i = \lambda_i S_w w_i, i = 1, 2, \ldots, m$$

(3.13)

As there are at the most \(C-1\) non-zero eigenvalues, upper bond on \(m\) is \(C - 1\), where \(C\) is the number of classes.

In face recognition using LDA, the problem is that \(S_w\) is always singular. It’s because the rank of \(S_w\) is at most \(M - C\) and number of pixels in each image \(N^2\) is much larger than number of images in training set \(M\).

To overcome the complications of singular \(S_w\), an alternative to criterion in eq. (3.12) is proposed. This method is called as *fisherfaces* method. This method projects images to a lower dimensional subspace so as to make resulting \(S_w\) nonsingular. It can be done by first applying PCA to the set of feature set so as to reduce its dimensions to \(M - C\), and then applying standard FLD given by eq. (3.12) to further reduce its dimensions to \(C - 1\). Now \(W_{opt}\) can be given as:

$$W_{opt}^T = W_{fld}^T W_{pca}^T$$

(3.14)

where

$$W_{pca} = \arg \max_w |w^T S_f W|$$

(3.15)

$$W_{fld} = \arg \max_w |w^T W_{pca}^T S_b W_{pca} w| / |w^T W_{pca}^T S_w W_{pca} w|$$

(3.16)

In short, images are projected from \(N^2\) dimensional space (where \(N^2\) is the number of pixels in the image) to \(C - 1\) dimensional space (where \(C\) is the number of classes of images).

### 3.3 Independent Component Analysis

ICA is a highly researched area for blind source separation. It has been developed a lot over the years and several algorithms have been proposed till date. In real life situation, ICA can be used for recognizing faces.
3.3.1 Motivation:

Suppose two people are speaking simultaneously in a room having two microphones at different locations. The microphones give two recorded signals, \( x_1(t) \) and \( x_2(t) \), where \( x_1 \) and \( x_2 \) are amplitudes and \( t \) is the time index. Each recorded signal is a weighted sum of the speech signals, \( s_1(t) \) and \( s_2(t) \), emitted by the two speakers. This can be expressed as linear equations:

\[
x_1(t) = a_{11}s_1 + a_{12}s_2 \\
x_2(t) = a_{21}s_1 + a_{22}s_2
\]

where parameters \( a_{11}, a_{12}, a_{21}, a_{22} \) depends on the distances between microphones and the speakers. Here only recorded signals \( x_1(t) \) and \( x_2(t) \) are known. The goal is to estimate the two original speech signals \( s_1(t) \) and \( s_2(t) \). This is called the cocktail-party problem [11]. This problem can be solved by assuming that \( s_1(t) \) and \( s_2(t) \), at each time instant \( t \), are statistically independent. With this assumption, ICA can separate the two original source signals \( s_1(t) \) and \( s_2(t) \) from their mixtures \( x_1(t) \) and \( x_2(t) \). The algorithm requires equal number of sensors and source signals.

3.3.2 ICA Basic Model:

In general, for \( n \) mixtures \( x_1, \ldots, x_n \) of \( n \) independent sources, linear equation is:

\[
x_j = a_{j1}s_1 + a_{j2}s_2 + \cdots + a_{jn}s_n, \text{ for all } j.
\]

It is assumed that all mixtures and source signals are random variables instead of time signals. Another assumption made is that the mixtures and the independent components have zero mean. If not, then sample mean should be subtracted from mixture variables, which make it a zero-mean model. It is convenient to denote above equation in sum using vector-matrix notation. Let us use random vector \( \mathbf{x} \) whose components are mixtures \( x_1, \ldots, x_n \), vector \( \mathbf{s} \) with elements \( s_1, \ldots, s_n \) and \( \mathbf{A} \) be the matrix with components \( a_{ij} \). Let denote vectors using bold lower case letters, matrices using bold upper-case letters and all vectors be column vectors. Above equation can be written using vector-matrix notation as:

\[
\mathbf{x} = \mathbf{A}\mathbf{s}.
\]

Let \( a_i \) be the columns of matrix \( \mathbf{A} \). The model can also be written as

\[
\mathbf{x} = \sum_{i=1}^{n} a_i s_i
\]
The statistical model in eq. 3.20 is called as ICA model. Here mixing matrix \( A \) is unknown. Only known variable is random vector \( x \) and goal is to estimate both \( A \) and \( s \) using it. For ICA, it is assumed that source signals \( s_i \) are statistically independent of each other i.e. value of one does not depend on value of other and unknown mixing matrix \( A \) is square and invertible. Once matrix \( A \) is obtained, its inverse, say \( W \), can be estimated and independent sources can be found simply by:

\[
    s = Wx
\]

ICA is one of the most widely used methods for performing blind source separation (BSS) or blind signal separation. A "source" means an original independent signal, e.g. speaker in cocktail party problem. "Blind" means very little is known about mixing matrix.

### 3.3.3 Ambiguities and Limitations of ICA Algorithm:

1. **Indeterminate Energy:** The variances (energies) of the independent components cannot be determined: As both \( s \) and \( A \) are unknown, any scalar multiplier in one of the sources \( s_i \) gets always cancelled by dividing the corresponding column \( a_i \) of \( A \) by same scalar (refer eq. 3.21) i.e. scalar multiplier could be pulled out of \( s \) and multiplied to \( A \) with no change in the equation. So it is assumed that each has unit variance: \( E\{s_i^2\} = 1 \). This still leaves the ambiguity of the sign. An independent component can be multiplied by \(-1\) without affecting the model. Therefore the output signals can be inversions of the original signals. Fortunately, this ambiguity is not significant in many cases.

2. **Order ambiguity:** ICA algorithm cannot determine the order of the independent components. Both \( s \) and \( A \) being unknown, the order of the terms in the sum in eq. (3.21) can be freely changed, and any of the independent components can be called as the first one.

3. **Under-Determination:** There must be as many sensors as number of sources in order to properly isolate the sources. If sensors are not enough, the resulting signals will be mixtures of multiple sources.
4. Linearity: ICA can only handle linear mixtures that can be represented in the form \( x = A s \). It fails to find the independent sources if the sources are out of phase in the mixtures or if the mixtures have other nonlinear features.

### 3.3.4 Principles of ICA Estimation

#### 1. Statistical Independence:

The variables \( y_1 \) and \( y_2 \) are independent if the conditional probability of \( y_1 \) with respect to \( y_2 \) is just the probability of \( y_1 \). This can be expressed as the equation:

\[
P(y_1 | y_2) = P(y_1)
\]

Since \( P(y_1 | y_2) = \frac{P(y_1, y_2)}{P(y_2)} \), where \( P(y_1, y_2) \) is the joint density of \( y_1 \) and \( y_2 \).

\[
P(y_1, y_2) = P(y_1) * P(y_2)
\]

* i.e. the two variables \( y_1 \) and \( y_2 \) are said to be independent if information on the value of \( y_1 \) does not give any information on the value of \( y_2 \), and vice versa.

Statistical independence is much stronger property as compared to uncorrelatedness since it takes into account only second order statistics. Two variables \( y_1 \) and \( y_2 \) are uncorrelated, if their covariance is zero. If two variables are independent, they are uncorrelated, but the vice versa is not true.

#### 2. Non-gaussianity

Non-gaussianity is the key factor to estimate ICA. Without non-gaussianity, ICA cannot be performed. According to Central Limit Theorem, the sum of several independent variables tend to become more gaussian than individual variable. For example, \( x_i = a_1s_1 + a_2s_2 \) is more gaussian than either \( s_1 \) or \( s_2 \).

To find one of the independent components, consider a linear combination of the \( x_i \) (see eq. 3.22). Let this be denoted by \( y = w^T x = \sum_i w_i x_i \), where \( w \) is a vector to be determined. Let us make a change of variables, defining \( z = A^T w \). Therefore \( y = w^T x = w^T As = z^T s \). Thus \( y \) is a linear combination of \( s_i \), with weights given by \( z_i \). Since sum of even two independent variables tends to be more gaussian than individual variables, \( z^T s \) is more gaussian than any of the \( s_i \) and becomes least gaussian when it is equal to one of the \( s_i \). In this case, only one of the elements \( z_i \) of \( z \) is nonzero. Therefore, it can be said
that $w$ is a vector that maximizes the non-gaussianity of $w^T x$. Such a vector would correspond to a $z$ (in the transformed coordinate system) which has only one nonzero component. This implies that $w^T x = z^T s$ equals one of the independent components. Thus one of the independent components can be estimated by maximizing the non-gaussianity of $w^T x$.

### 3.3.5 Approaches to ICA:

**Mutual Information:**

Based on information theory, one of the approaches for ICA estimation is mutual information. It also finds the most non-gaussian directions.

Mutual information $I$ between $m$ variables $y_i, i = 1,..., m$ is:

$$I(y_1, y_2, ..., y_m) = \sum_{i=1}^{m} H(y_i) - H(y).$$

(3.25)

It is the measure of the dependence between random variables. If all variables are statistically independent then mutual information between them is always positive and zero. Thus mutual information considers whole dependence structure of the variables and not only covariances as in case of PCA. The relation between negentropy and mutual information is given by:

$$I(y_1, y_2, ..., y_m) = C - \sum_i J(y_i)$$

(3.26)

where $C$ is a constant.

### 3.3.6 Pre-processing for ICA:

Preprocessing is done in ICA to make sure that the observed mixture signals have zero mean, unit variance and are de-correlated. Decorrelation makes sure that second order dependencies between observed signals are removed. Following preprocessing steps are performed on data.

1. **Centering:** To simplify the ICA algorithm, the most basic preprocessing step is to subtract mean from $x$ to make it a zero mean variable. This will make $s$ also zero-mean. After finding mixing matrix $A$ with centered data, mean vector of $s$ is added back to the centered $s$. 
2. Whitening: Whitening is another pre-processing strategy. Before applying ICA, the centered data $\mathbf{x}$ is transformed linearly so as to obtain new vector $\mathbf{\tilde{x}}$ which is whitened. i.e. its components are uncorrelated and have unit variance. In other words, covariance matrix of $\mathbf{\tilde{x}}$ is equal to identity matrix i.e. $E\{\mathbf{\tilde{x}}\mathbf{\tilde{x}}^T\} = \mathbf{I}$. Eigenvalue decomposition (EVD) of the covariance matrix $E\{\mathbf{\tilde{x}}\mathbf{\tilde{x}}^T\} = \mathbf{E}\mathbf{D}\mathbf{E}^T$ can be used for whitening, where $\mathbf{E}$ is the orthogonal matrix of eigenvectors of $E\{\mathbf{x}\mathbf{x}^T\}$ and $\mathbf{D}$ is the diagonal matrix of its eigenvalues. Whitening can be performed using following equation.

$$\mathbf{\tilde{x}} = \mathbf{ED}^{-1/2}\mathbf{E}^T\mathbf{x} \tag{3.27}$$

where $\mathbf{D}^{-1/2} = \text{diag}(d_1^{-\frac{1}{2}}, ..., d_n^{-\frac{1}{2}})$. Now $E\{\mathbf{\tilde{x}}\mathbf{\tilde{x}}^T\} = \mathbf{I}$. Whitening transforms the mixing matrix $\mathbf{A}$ into $\mathbf{\tilde{A}}$ as we have following eq:

$$\mathbf{\tilde{x}} = \mathbf{E}\mathbf{D}^{-1/2}\mathbf{E}^T\mathbf{As} = \mathbf{\tilde{A}}\mathbf{s} \tag{3.28}$$

It can be seen from the following equation that the new mixing matrix $\mathbf{\tilde{A}}$ is orthogonal.

$$E\{\mathbf{\tilde{x}}\mathbf{\tilde{x}}^T\} = \mathbf{\tilde{A}}E\{\mathbf{s}\mathbf{s}^T\}\mathbf{\tilde{A}}^T = \mathbf{\tilde{A}}\mathbf{A}^T = \mathbf{I}. \tag{3.29}$$

Whitening reduces complexity of the ICA estimation by reducing number of parameters to be estimated. Now instead of estimating $n^2$ parameters (which are the elements of $\mathbf{A}$), only the new, orthogonal $\mathbf{\tilde{A}}$ need to be estimated (which has $n(n-1)/2$ degrees of freedom). Whitening is good way to reduce complexity of the problem.

3.3.7 InfoMax Algorithm:

For face recognition, various ICA algorithms can be used like InfoMax, FastICA and JADE. Bell and Sejnowski proposed InfoMax algorithm [12]. It is used to separate statistically independent components of a dataset using unsupervised learning. It has been successfully applied for separation of mixed auditory signals (cocktail party problem), for separating electroencephalogram (EEG) signals and functional magnetic resonance imaging (fMRI) signals. InfoMax algorithm is derived from the principle of optimal information transfer in neurons with sigmoidal transfer functions [13].
Consider an \( n \)-dimensional input vector \( \mathbf{X} \). Here, let variables be denoted by boldface capitals and matrices by plaintext capitals. Let \( \mathbf{W} \) be an invertible \( n \times n \) matrix, \( \mathbf{U} = \mathbf{WX} \) and \( \mathbf{Y} = \mathbf{f(U)} \) an \( n \)-dimensional vector representing outputs of \( n \)-neurons. Each component of \( \mathbf{f} = (f_1, \ldots, f_n) \) is an invertible squashing function which maps real numbers to interval \([0,1]\). Logistic function used is:

\[
f_i(u) = \frac{1}{1 + e^{-u}}
\]

(3.30)

The goal is to maximize the mutual information between the input \( \mathbf{X} \) and the output of the neural network \( \mathbf{Y} \). It can be achieved by performing gradient ascent on the entropy of the output with respect to the weight matrix \( \mathbf{W} \). The gradient update rule for the weight matrix, \( \mathbf{W} \) is:

\[
\Delta \mathbf{W} \propto \nabla_{\mathbf{W}} H(\mathbf{Y}) = (\mathbf{W}^T)^{-1} + E(\mathbf{Y}'\mathbf{X}^T)
\]

(3.31)

where \( \mathbf{Y}' = f''(\mathbf{U})/f'(\mathbf{U}) \) is the ratio of second and first partial derivatives of the activation function, \( E \) is expected value, \( T \) is transpose, \( H(\mathbf{Y}) \) is the entropy of the vector \( \mathbf{Y} \) and \( \nabla_{\mathbf{W}} H(\mathbf{Y}) \) is the gradient of the entropy in matrix form. To avoid computing matrix inverse, natural gradient is employed by multiplying absolute gradient by \( \mathbf{W}^T \mathbf{W} \).

\[
\Delta \mathbf{W} \propto \nabla_{\mathbf{W}} H(\mathbf{Y}) \mathbf{W}^T \mathbf{W} = (I + \mathbf{Y}'\mathbf{U}^T) \mathbf{W}
\]

(3.32)

where \( I \) is the identity matrix. \( \mathbf{Y}' = (1 - 2\mathbf{Y}) \) is obtained from logistic function. It is shown that, if the joint entropy of the outputs in \( \mathbf{Y} \) is maximized, then mutual information between the outputs in \( \mathbf{U} \) gets minimized and they become statistically independent.

The algorithm can be speeded by performing “sphering”. The centered \( \mathbf{X} \) is passed through whitening matrix \( \mathbf{W}_z \) which is twice the inverse square root of the covariance matrix.

\[
\mathbf{W}_z = 2 \ast (\text{Cov(\mathbf{X})})^{-\frac{1}{2}}.
\]

(3.33)

This sets mean and covariances to zero, equalizes the variances and removes first and second-order statistics of data. Here the full transform matrix is the product of the matrix learned by ICA and the sphering matrix.

\[
\mathbf{W}_t = \mathbf{W} \mathbf{W}_z
\]

(3.34)
Theorem: ICA can be performed by finding the maximum, with respect to $W$, of the joint entropy, $H(y)$, of an output vector, $y$, which is the vector $u$, except that each element is transformed by a sigmoidal function which is a cumulative density function (c.d.f.) of sources [14].

3.3.8 ICA for Face Recognition:

Independent Component Representations of Face Images:

3.3.8.1 Architecture-1. (Statistically Independent Basis Images):

![Figure 3.3. ICA (InfoMax algorithm) - Image synthesis model for Architecture-1.](image)

Let $X$ be a vector representing inputs i.e. all images of training set are loaded in matrix $X$ as shown in Fig. 3.3. Each face is made up of linear combination of independent source (i.e. basis) images from $S$. Let $W$ be $n$ by $n$ invertible weight matrix. As seen earlier from eq. 3.22,

$$U = WX. \quad (3.35)$$

Goal is to find matrix $W$ such that $I(u_i, u_j) = 0$ i.e. mutual information between rows of $U$ is zero.
Methodology

Figure 3.4. Optimal information flow in sigmoidal neurons. Right: $f_y(y)$ is plotted for different values of the weight, $w$. The optimal weight, $w_{opt}$, transmits the most information.

If each $u_i$ is passed through sigmoidal transfer (i.e. logistic) function as (see Fig. 3.4):

$$y_i = f_i(u) = 1 / (1 + e^{-u}),$$

then output $y_i$ has maximum entropy. Joint entropy $H(y_i, y_j)$ is:

$$H(y_i, y_j) = H(y_i) + H(y_j) - I(y_i, y_j).$$

Joint entropy is maximum when $I(y_i, y_j) = 0$. Joint entropy can be maximized by performing gradient ascent on it w.r.t. the weight matrix $W$. The gradient update rule for the $W$ is:

$$\Delta W = (I + Y^T) W,$$

where $I$ is identity matrix.

Now find $W$ for which joint entropy is maximum. When there are multiple inputs and outputs, maximizing the joint entropy of the output $Y$ encourages the individual outputs to move toward statistical independence. Once $W$ is known, $U$ can be found as seen previously:

$$U = WX.$$ 

$U$ gives independent basis vectors.

For face recognition, two architectures of ICA are proposed by Barlette et al. [15] as: Architecture-1: statistically independent basis images (see Fig. 3.3) and Architecture-2: factorial code representation. Architecture-1 treats the images as random variables and the pixels as outcomes and Architecture-2 treats the pixels as random variables and the images as outcomes.
We have implemented ICA using InfoMax algorithm which is proposed by Bell and Sejnowski. Architecture-1 finds a set of independent basis images. All training images are stored in rows of matrix $X$ and $U = WX$. ICA algorithm finds a weight matrix $W$ such that the rows of $U$ are as statistically independent as possible. As seen earlier (Fig. 3.3), rows of $U$ will have independent basis images which will represent faces. The number of Independent Components found by the ICA algorithm corresponds to the dimensionality of the input ($n$). This is an intractable task. Therefore first dimensionality reduction is done using PCA and most significant $m$ eigenvectors of training set are found. Then ICA is performed on these most significant $m$ eigenvectors (instead of performing on all $n$ training images). Here $m < n$. This produces $m$ independent source (i.e. basis) images in the rows of $U$. Let $P_m$ be matrix containing first $m$ eigenvectors in its columns. PC representation of set of images in $X$ based on $P_m$ is:

$$R_m = X P_m$$  \hspace{1cm} (3.40)

ICA algorithm produces matrix:

$$W_i = W W_z,$$  \hspace{1cm} (3.41)

where $W_z$ is a sphering matrix as defined in eq. 3.33. The IC representation of face images is given by rows of the matrix:

$$B = R_m W_i^{-1}.$$  \hspace{1cm} (3.42)

These $b$’s are some scalar values. Basis vectors are in rows of $U$. Each image in training set $X$ can be represented as weighted linear combination of all independent basis images, $u$, as shown over here.

![ICA representation](image)

Figure 3.5. InfoMax algorithm (Architecture-1): The independent basis image representation.

**Recognition:** For recognition, ICA representation of test image i.e. $b_{test}$ is compared with ICA representations of all images in training database using distance measure (say cos) as:
Test image is then assigned identity of closest training image.

### 3.3.8.2 Architecture-2. (A Factorial Code):

Architecture-1 produced basis images which are statistically independent. ICA representation consists of coefficients which in combination with basis images reconstructs face images. Here though basis images are statistically independent, the coefficients are not. Barlett et al. altered the architecture of ICA and introduced second representation which makes the coefficients independent i.e. new ICA outputs formed factorial code for the face images. Instead of separating the face images, as in case of Architecture-1, elements of the face representation are separated in Architecture-2. In Architecture-2, input matrix $X$ is transposed such that images are in columns and the pixels in rows as seen in Fig. 3.6.

Figure 3.6. Left: performing source separation on face images produced independent basis images in the rows of $u$. Right: performing source separation on the pixels produced a factorial code in the columns of $u$.

**Figure 3.6.** Left: performing source separation on face images produced independent basis images in the rows of $u$. Right: performing source separation on the pixels produced a factorial code in the columns of $u$.

Figure 3.7. ICA (Infomax algorithm): Image synthesis model for Architecture-2.
The factorial code for the training images in $X$ is provided by the columns of output matrix, $U = W_i X$. The coefficients of the basis images in $A$ are contained in columns of $U$ which are used for reconstructing each face image in $X$.

$$= u_1^* + u_2^* + ... + u_n^*$$

ICA factorial representation = ( $u_1$, $u_2$, ..., $u_n$ )

Figure 3.8. ICA factorial representation.

**Recognition:** The representational code for test images is given by $W_i X_{test} = U_{test}$, where weight matrix, $W_i$, is found by performing ICA on the training set.

### 3.4 General Subspace Appearance-based Face Recognition System

A face image in 2-dimension with size $N \times N$ can also be considered as one dimensional vector of dimension $N^2$. Since space derived this way is highly dimensional, recognition in it is unfeasible. Therefore, recognition algorithms derive lower dimensional spaces to do the actual recognition while retaining as much information from the original images as possible. An illustration of building a general subspace appearance-based face recognition system is shown in Fig. 3.9. Training of the system is shown in the left part of the figure and the procedure for projecting gallery images onto a subspace (projection matrix $W^T$) is shown in the right part of the figure; $X$ is a matrix containing the images expressed as vectors in its columns. $\psi$ – mean image (as a vector), $\psi$ – mean image (as a vector), $\psi$ – mean image (as a vector)}
\( \bar{X} \) – matrix containing mean-subtracted images in its columns, \( W^T \) – projection matrix, \( X_g \) – gallery image (as a vector). During the training phase, the projection matrix (containing the basis vectors of the subspace) is calculated and then the gallery images (the images of known persons) are projected onto that subspace and their projections are stored in a database. Later, in the recognition phase (Fig. 3.10), new image is normalized, mean-subtracted, projected onto the same subspace as the gallery image was and its projection is then compared to the stored gallery projections. The nearest neighbor is determined by calculating the distances \( (d) \) from a probe image projection to all gallery images projections. Then the minimum distance is chosen as a similarity measure. The identity of the most similar gallery image is then chosen to be the result of recognition and the unknown probe image is identified [16].

![Figure 3.10. The recognition phase of a general subspace face recognition system.](image)

### 3.5 Face Databases

In last decades, due to major advances in face recognition, researchers were successful in achieving recognition rate of more than 90%. However in real-world scenarios, face acquisition remains a challenge due to several variations. Following factors affect the performance of face recognition systems [17]:

1. **Illumination** variations can occur due to camera settings and due to skin reflectance properties. Several face recognition systems perform well only under
moderate illumination variations while their performances drop if large illumination or pose changes occur.

2. *Pose* changes affect the recognition process as they introduce projection deformations and self-occlusion. Though there exist systems dealing with 32° head rotation, they can’t handle situations where security cameras cause pose variation beyond this range. On contrary, there are systems robust to *facial expressions* which can handle all situations except extreme expressions like scream.

3. Other key factor is *time delay*, as faces change over time. This problem is relatively hard to solve compared to others and not much has been done in the area of age variations.

4. *Occlusions* can drastically affect face recognition performances especially if they occur in the upper part of the face.

To access how well face recognition systems work in the presence of one or combination of these variations, several face databases have been developed. Many researchers have built face databases which provides as many variations as possible on the face images. Table 3.1 shows the most popular 2D face image databases along with their main characteristics like name, RGB/gray, image size, number of persons, number of images per person, available variations e.g. pose (p), expression (e), illumination (i), occlusions (o), indoor/outdoor (i/o), time delay (t). Here FERET represents good testing framework if large gallery and probe sets is the requirement. AR is the only database which provides naturally occluded (with sunglasses/scarves) face images.

Though significant efforts have been made in creating very large databases, there is no unique standard protocol that can evaluate performances. Face recognition system performance can be characterized using recognition rate, false acceptance rate and false rejection rate. There is not a common standard database to test face recognition algorithms and FERET is an excellent attempt in this direction [18].
Table 3.1. Most important face databases

<table>
<thead>
<tr>
<th>Name</th>
<th>RGB/ Gray</th>
<th>Image size</th>
<th>Number of people</th>
<th>Images/person</th>
<th>Number of conditions</th>
</tr>
</thead>
<tbody>
<tr>
<td>AR Face Database* [19]</td>
<td>RGB</td>
<td>576x768</td>
<td>126</td>
<td>26</td>
<td>i, e, o, t</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>70 Male</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>56 Female</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Richard’s MIT database [20]</td>
<td>RGB</td>
<td>480x640</td>
<td>154</td>
<td>6</td>
<td>p, o</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>82 Male</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>74 Female</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CVL Database [21]</td>
<td>RGB</td>
<td>640x480</td>
<td>114</td>
<td>7</td>
<td>p, e</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>108 Male</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>6 Female</td>
<td></td>
<td></td>
</tr>
<tr>
<td>The Yale Face Database B* [22]</td>
<td>gray</td>
<td>640x480</td>
<td>10</td>
<td>576</td>
<td>p, i</td>
</tr>
<tr>
<td>The Yale Face Database [23]</td>
<td>gray</td>
<td>320x243</td>
<td>15</td>
<td>11</td>
<td>i, e</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>14 Male</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1 Female</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PIE Database* [24]</td>
<td>RGB</td>
<td>640x486</td>
<td>68</td>
<td>608</td>
<td>p, i, e</td>
</tr>
<tr>
<td>The UMIST Face Database [25]</td>
<td>gray</td>
<td>220x220</td>
<td>20</td>
<td>19-36</td>
<td>p</td>
</tr>
<tr>
<td>AT&amp;T Database of Faces (ORL)* [26]</td>
<td>gray</td>
<td>92x112</td>
<td>40</td>
<td>10</td>
<td>p, e</td>
</tr>
<tr>
<td>(JAFEE) Database [27]</td>
<td>gray</td>
<td>256x256</td>
<td>10</td>
<td>7</td>
<td>e</td>
</tr>
<tr>
<td>The Human Scan Database [28]</td>
<td>gray</td>
<td>384x286</td>
<td>23</td>
<td>66</td>
<td></td>
</tr>
<tr>
<td>The University of Oulu Physics-based Face Database [29]</td>
<td>gray</td>
<td>428x569</td>
<td>125</td>
<td>16</td>
<td>i</td>
</tr>
<tr>
<td>XM2VTS database [30]</td>
<td>RGB</td>
<td>576x720</td>
<td>295</td>
<td></td>
<td>p</td>
</tr>
<tr>
<td>FERET* [31]</td>
<td>gray, RGB</td>
<td>256x384</td>
<td>30000</td>
<td></td>
<td>p, i, e, i/o, t</td>
</tr>
</tbody>
</table>

The * points most used databases. Image variations are indicated by (i) illumination, (p) pose, (e) expression, (o) occlusion, (i/o) indoor/outdoor conditions and (t) time delay.
REFERENCES


[19] AR Face Database. Available: 
http://rvl1.ecn.purdue.edu/~aleix/aleix_face_DB.html

http://web.mit.edu/emeyers/www/face_databases.html#richard

[22] The Yale face database B. Available:
   http://cvc.yale.edu/projects/yalefacesB/yalefacesB.html


   http://www.ri.cmu.edu/projects/project_418.html

[25] The UMIST Face Database. Available:
   http://images.ee.umist.ac.uk/danny/database.html

[26] The AT&T database of faces (ORL). Available:
   http://www.uk.research.att.com/facedatabase.html

[27] The Japanese Female Facial Expression (JAFFE) Database. Available:
   http://www.mis.atr.co.jp/~mlyons/jaffe.html

[28] The Human Scan Database. Available:
   http://www.humanscan.de/support/downloads/facedb.php

[29] The University of Oulu Physics-based Face Database. Available:
   http://www.ee.oulu.fi/research/imag/color/pbfd.html

   http://www.ee.surrey.ac.uk/Reseach/VSSP/xm2vtsdb/