Presented at *Computer Vision and Pattern Recognition 98*, Santa Barbara, CA, June 1998.

Mixtures of Local Linear Subspaces for Face Recognition

Brendan J. Frey (www.cs.utoronto.ca/~frey), Antonio Colmenarez, Thomas S. Huang
Beckman Institute for Advanced Science and Technology
University of Illinois at Urbana-Champaign

Abstract

Traditional subspace methods for face recognition compute a measure of similarity between images after projecting them onto a fixed linear subspace that is spanned by some principal component vectors (a.k.a. "eigenfaces") of a training set of images. By supposing a parametric Gaussian distribution over the subspace and a symmetric Gaussian noise model for the image given a point in the subspace, we can endow this framework with a probabilistic interpretation so that Bayes-optimal decisions can be made. However, we expect that different image clusters (corresponding, say, to different poses and expressions) will be best represented by different subspaces. In this paper, we study the recognition performance of a *mixture of local linear subspaces* model that can be fit to training data using the expectation maximization algorithm. The mixture model outperforms a nearest-neighbor classifier that operates in a PCA subspace.

1 Introduction

This paper is about modeling face images of the sort shown in Fig. 1 for the purpose, say, of robust face recognition. In one approach to visual face modeling, normalized N-pixel face images are projected onto a subset of D eigenvectors or "eigenfaces" of the covariance matrix estimated from a training set of images [1]. The D-dimensional subspace spanned by these orthogonal eigenfaces is the subspace in which the training data has the greatest variance. In fact, these eigenfaces are equal to the first D principal components obtained from principal components analysis [2]. The distance of a new input image from this linear subspace has been used quite successfully to detect faces [3]. Moghaddam and Pentland [4] recently extended the eigenface framework to include a distance measure *within* the eigenspace.

Although the eigenface method is currently one of the best algorithms for face recognition, it seems natural that different types of images ought to be better represented by different subspaces. As an extreme example, if some of the training images consist of vertically inverted faces, then the appropriate subspace for the inverted images is spanned by a set of "eigen-inverted-faces" obtained by inverting the eigenfaces for the uninverted images. In such a model, the orientation of the subspace depends on the input image. *Local dimensionality reduction* of this type has been considered by Bregler and Omohundro [5], Kambhatla and Leen [6], Sung and Poggio [7] and Hinton *et al.* [8]. These models can represent input images in locally linear, globally nonlinear subspaces, but they do not include a distance within the subspace.

The use of distances has progressively led to a more probabilistic view in which we define a probability density function over the input image $\mathbf{x}$ in terms of some *latent* variables $\mathbf{z}$:

$$p(\mathbf{x}) = \int_{\mathbf{z}} p(\mathbf{x}|\mathbf{z})p(\mathbf{z})d\mathbf{z} \tag{1}$$

For example, in the linear subspace model described above, $\mathbf{z}$ is a position on the surface of a D-dimensional hyperplane. If we build one such latent variable model for each class of data, the probabilities of the different classes $C_1, C_2, \ldots$ can be computed using Bayes' rule:

$$P(C_i|\mathbf{x}) = \frac{p(\mathbf{x}|C_i)P(C_i)}{\sum_j p(\mathbf{x}|C_j)P(C_j)}, \tag{2}$$

where $P(C_j)$ is the *a priori* probability of class j.

Using this approach, a variety of nonlinear latent variable models have been successfully applied to the task of pattern classification [9]. Recently, Hinton *et al.* [10] applied a mixture of linear subspaces to the task of handwritten character recognition.

In this paper, we begin with a description of the eigenface model and its shortcomings, and then develop a mixture of local subspaces model for face recognition. We review a variation on the expectation maximization algorithm described in [11] and then give results on a new database that is being compiled at the Beckman Institute, University of Illinois at Urbana-Champaign. In the appendix, we describe some linear algebra tricks that can be used to speed up the algorithm by 2 or 3 orders of magnitude.

Figure 1: Example video frames plus normalized faces.

2 Eigenfaces: The Principal Components

For a training set of image vectors $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(T)}$, after the sample mean and variance for each pixel, the simplest statistic to investigate is the sample covariance. In particular, the unit vector $\boldsymbol{\lambda}_1$ that leads to maximum variance when the training vectors are projected upon it is called the first *principal component.* That is, $\sum_t(\boldsymbol{\lambda}_1'\mathbf{x}^{(t)})^2 \geq \sum_t(\mathbf{u}'\mathbf{x}^{(t)})^2$ for all unit vectors $\mathbf{u}$ (we use "$'$" to denote vector transpose). If we subtract this component from each training image, we may then seek the second principal component, and so on. The first D principal components give the D-dimensional subspace in which the training set has maximum variance (energy).

The principal components can be determined by solving an eigenvalue equation:

$$\boldsymbol{\Sigma}\boldsymbol{\Lambda} = \mathbf{S}\boldsymbol{\Lambda}, \tag{3}$$

where $\boldsymbol{\Sigma} = \frac{1}{T}\sum_t(\mathbf{x}^{(t)} - \boldsymbol{\mu})(\mathbf{x}^{(t)} - \boldsymbol{\mu})'$ is the sample covariance matrix, $\boldsymbol{\Lambda}$ is the matrix of eigenvectors (principal components), and $\mathbf{S}$ is the diagonal matrix of eigenvalues (sample variances in the directions of the principal components). The method of principal components analysis (PCA) finds the first D components having the D largest eigenvalues. Because of this eigenvalue formulation of PCA, the principal components of face images were dubbed "eigenfaces" in [1].

2.1 Sensitivity of Eigenfaces to Variation in Pixel Noise

We expect different regions of input images to have different levels of pixel noise that cannot be explained by our model. Even if the sample variance for each pixel is the *same* we hope our model will *explain* the variability caused by structure and ignore the variability caused by noise.

Fig. 2a shows a scatter plot of some 2-dimensional data that was generated as follows. $T = 1000$ values were randomly picked from a Gaussian distribution with mean 0.0 and standard deviation 0.2. Then, each value $v^{(t)}$ was mapped to a 2-dimensional data point using $x_1^{(t)} = x_2^{(t)} = v^{(t)}$, producing "noise-free" data which was then made noisy by adding independent zero-mean Gaussian noise to each dimension. The standard deviations of the noise for x_1 and x_2 were equal to 0.5 and 0.01 respectively. Two 1-dimensional subspaces were estimated from the training data: one (PC) is the first principal component, and the other (FA) was determined using the expectation maximization algorithm to fit the *factor analysis* model described in Scn. 3. As shown in Fig. 2b, the principal component is *not* aligned with subspace in which the noise-free data (NFD) lies. The other subspace is more closely aligned with the correct subspace and is thus able to more clearly differentiate between structure and noise.

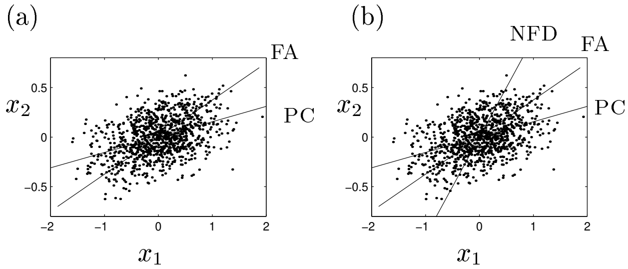

Figure 2: Scatter plots of some data with unequal noise in each dimension

2.2 Insensitivity of Eigenfaces to the Manifold of Faces

Imagine a vast convoluted manifold in the space of pixel intensities, in which all "noise-free" faces lie. This manifold accounts for all sorts of local variations in faces, such as pupil dilation, lip posture, *etc.*, but does not account for "noise" such as whisker detail, eyelash detail, *etc.* It seems plausible that locally this manifold is low-dimensional (relative to the number of pixels), since the class of faces is a very small subset of all possible images. The method of principal components approximates this highly non-linear low-dimensional manifold with one linear subspace. Since this linear subspace must account for all significant variation, we expect its dimensionality will tend to be higher than the local dimensionality. In this way, eigenfaces are not sensitive to the local structure in the manifold of faces.

Fig. 3a shows a 2-dimensional scatter plot of some 3-dimensional data that happens to lie in a two-dimensional linear subspace. (We dispense with 3-dimensional rendering for the sake of visual clarity.) A single principal component (long line) fails to capture the curvature of the data within the 2-dimensional subspace. We can add a second principal component (short line) and even try to model the data within the 2-dimensional subspace using a mixture of Gaussians (*e.g.*, [4]). Although this will work for the toy data, we expect the manifold of faces to twist and turn so that each of many directions in pixel-space is significant somewhere on the manifold. As a result, PCA must ignore some directions of local variability.

Fig. 3b shows how a mixture of 2 1-dimensional linear subspaces can be fit to the same data. In general, a

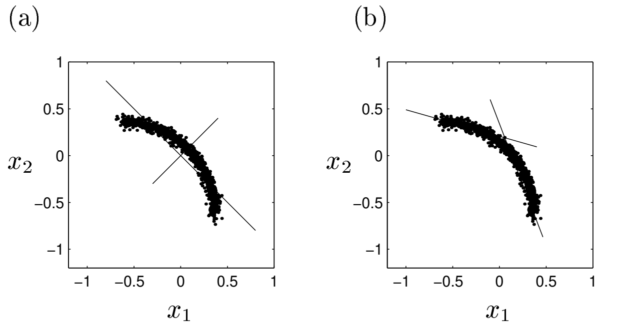

Figure 3: (a) A 2-dimensional scatter plot of some 3-dimensional data that lies on a 2-dimensional subspace. The data actually comes from a curved 1-dimensional manifold. The principal components fail to capture the curvature. (b) A mixture of 1-dimensional subspaces fits the data much better.

model manifold of this sort can be low-dimensional locally and high-dimensional globally, just as we expect the manifold of faces to be. Results in Scn. 4 show that local variability can be nicely modeled using a mixture of local linear subspaces.

3 Factor Analysis: A Probability Model for Globally Linear Subspaces

PCA extracts a linear subspace from a training set, but does not model the off-subspace noise nor the in-subspace variability. Moghaddam and Pentland [4] append a Gaussian in-subspace model and a single off-subspace noise variance parameter to account for these deficiencies in PCA. Although this model can be estimated directly from a training set, the "noise" is actually structured since it lies in the null space of the linear subspace. In contrast, factor analysis (FA) [12] is a probabilistic model where the noise on each pixel is independent. FA models the joint density of the input and a vector of D latent variables (or *factors*) $\mathbf{z}$, meant to capture input covariance:

$$p(\mathbf{x}, \mathbf{z}) = p(\mathbf{z})p(\mathbf{x}|\mathbf{z}) = \mathrm{N}(\mathbf{z}; \mathbf{0}, \mathbf{I})\mathrm{N}(\mathbf{x}; \mathbf{\Lambda}\mathbf{z}, \mathbf{\Psi}), \quad (4)$$

where $\mathrm{N}(\mathbf{x}; \boldsymbol{\mu}, \mathbf{C})$ is the normal density function with mean $\boldsymbol{\mu}$ and covariance $\mathbf{C}$, $\mathbf{I}$ is the $D \times D$ identity matrix, $\mathbf{\Lambda}$ is the *factor loading matrix* that relates the latent variables to the means of the inputs linearly, and $\mathbf{\Psi}$ is a diagonal matrix of input pixel variances.

The principal components of PCA are roughly analogous to the factors in FA. The kth factor models variability in the input space in the direction given by the kth column of the loading matrix. Unlike PCA, each input pixel i has its own noise variance parameter ψ_{ii}.

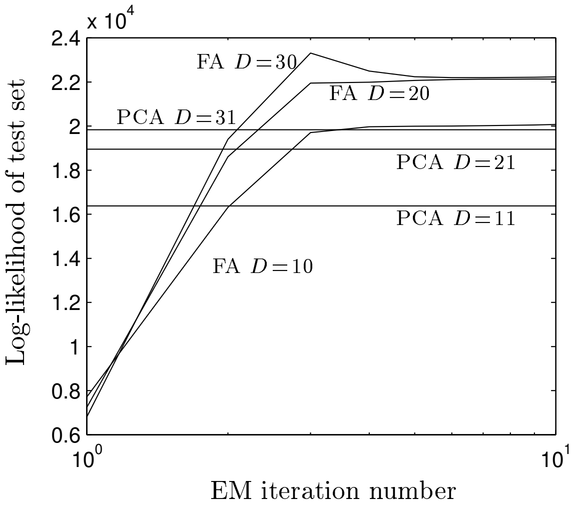

Figure 4: The log-likelihood of a test set during learning, for three different sizes of FA model ($D = 10, 20$, and 30). Also shown are the test set likelihoods produced by a PCA-type model with the same number of parameters.

This allows FA to model variation in pixel noise across the image.

Notice that the columns of $\mathbf{\Lambda}$ are not required to be orthogonal. That is, the factors may introduce variability in non-orthogonal directions in input space. This property is useful in nonlinear extensions of FA [13].

We fit a FA model with $D = 10, 20$, and 30 factors to a training set of 181 normalized front-view FERET face images, using 10 iterations of the EM algorithm [14]. Such a model with D factors for N-dimensional data has $(D+1)N$ parameters. By modeling the in-subspace variability of PCA using an axis-aligned Gaussian and the off-subspace noise of PCA using a single variance parameter as described above, we are able to compare these two methods as density estimators. Since such a PCA model with D components effectively has only ND parameters, in order to make a fair comparison between the performance of FA and PCA, we extracted the first $11, 21$, and 31 principal components. The log-likelihood of a test set of 60 images versus EM iteration number is shown in Fig. 4. For all three model sizes the FA model gives a higher density to the training set, indicating that it is a superior density estimator.

4 Mixtures of Local Linear Subspaces

In this section, we consider a mixture of K local linear subspaces as a mixture of K factor analyzers, where each factor analyzer has the same number D of factors. Let $\mathbf{\Lambda}_k$ be the factor loading matrix for analyzer k. Each analyzer will also have its own image mean $\boldsymbol{\mu}_k$ and its own diagonal pixel noise covariance matrix $\mathbf{\Psi}_k$. The mixture model can be written

$$\begin{aligned} p(\mathbf{x}, \mathbf{z}, k) &= P(k)p(\mathbf{z}|k)p(\mathbf{x}|\mathbf{z}, k) \\ &= \pi_k \mathrm{N}(\mathbf{z}; \mathbf{0}, \mathbf{I})\mathrm{N}(\mathbf{x}; \boldsymbol{\mu}_k + \mathbf{\Lambda}_k\mathbf{z}, \mathbf{\Psi}_k), \end{aligned} \tag{5}$$

where π_k is the *mixing proportion* of component k.

After integrating out $\mathbf{z}$, we have

$$p(\mathbf{x}|k) = \frac{\exp[-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_k)'(\mathbf{\Lambda}_k\mathbf{\Lambda}_k' + \mathbf{\Psi}_k)^{-1}(\mathbf{x}-\boldsymbol{\mu}_k)]}{(2\pi)^{N/2}|\mathbf{\Lambda}_k\mathbf{\Lambda}_k' + \mathbf{\Psi}_k|^{1/2}}. \tag{6}$$

This can be computed efficiently using the techniques described in the appendix, and then we can compute $p(\mathbf{x}) = \sum_k \pi_k p(\mathbf{x}|k)$. If we have one such mixture model for each class of data (*e.g.*, individual to be recognized), this procedure gives us $p(\mathbf{x}|C_i)$ for the different classes $C_1, C_2, \ldots$. We then apply Bayes' rule (2) to make a recognition decision.

4.1 Maximum Likelihood Parameter Estimation via the EM Algorithm

The EM algorithm for this mixture model is similar to the EM algorithm for a single factor analysis model [14], except that the E-step must now also fill in the subspace model identity k for each input image. The identity k of the subspace can be represented as a K-element binary "subspace indicator" vector $\mathbf{s}$ that has a 1 in the kth position and zeros in all other positions. In this case, the latent variables $\mathbf{s}$ and $\mathbf{z}$ can be probabilistically filled in using

$$\begin{gathered} \mathrm{E}[\mathbf{s}|\mathbf{x}^{(t)}], \\ \mathrm{E}[\mathbf{z}|\mathbf{x}^{(t)}, k], \quad \text{and} \\ \mathrm{E}[\mathbf{z}\mathbf{z}'|\mathbf{x}^{(t)}, k]. \end{gathered} \tag{7}$$

These expectations over the posterior distribution $p(\mathbf{s}, \mathbf{z}|\mathbf{x})$ are sufficient for computing the maximum likelihood model using EM. They can be computed easily using linear algebra (all likelihoods are Gaussian). See [11] for details.

10 iterations of this algorithm were used to fit a mixture of 2 1-dimensional subspaces to the scatter-plot data shown in Fig. 3. The mixture model is clearly a better fit than the single 1-dimensional PCA model. Although a mixture of Gaussians could be used to model the data in the subspace spanned by the first 2 principal components, such a model would ignore the locally-linear distribution of the data.

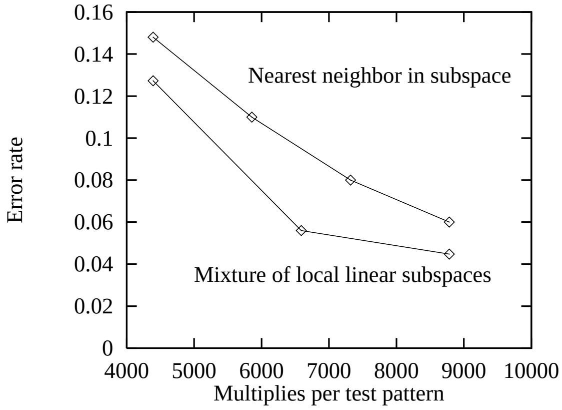

Figure 5: Comparison of recognition performance as a function of computational requirement.

4.2 Results on Robust Face Recognition

We are in the process of compiling a database of video sequences of a relatively small number of individuals (100), but with wide variation in facial expression and pose. Fig. 1 shows some of the more tame video frames from this database. The application we have in mind is highly robust face recognition in a relatively closed environment, such as an office area.

In this section, we compare the recognition performances of the mixture of local linear subspaces model and a method that performs nearest neighbor classification in a PCA subspace [4]. In these experiments, temporal structure is not modeled. Of course, treating the video as a time-series is expected to greatly improve performance and we are currently investigating temporal mixture models. The training set is completely separate from the test set, and each contains a total of 4000 images of the sort shown in Fig. 1. In this figure, the picture-in-picture images show the intensity-normalized output of our real-time face tracking system [15].

Fig. 5 shows the error rates for the nearest neighbor method and the mixture model as a function of recognition algorithm complexity (the number of multiplies required to recognize each input pattern). Each mixture component has $D = 3$ dimensions and the performances of models with $K = 2$, 3 and 4 clusters are given. For the nearest neighbor classifier, subspaces with 11, 15, 19 and 23 dimensions are used.

5 Discussion

We have presented results that indicate the mixture of linear subspaces model is more powerful than classification within a single linear subspace, *on a particular data set*, for low computation rates. It remains to be seen how significant the difference is statistically and how well the method works for different types of face data. The mixture model is capable of automatically extracting pose, but it is probably unnecessarily complex in cases where there is little variation in pose, lighting, expression, *etc.* On the other hand, for data sets that have a wide variation in these attributes, we feel the mixture model is much more suitable.

6 Acknowledgements

We appreciate helpful conversations we had with Zoubin Ghahramani and Geoffrey Hinton and we thank Karla Miller for comments on a draft of this paper.

Appendix

Directly computing $(\mathbf{x}-\boldsymbol{\mu})'(\boldsymbol{\Lambda}\boldsymbol{\Lambda}'+\boldsymbol{\Psi})^{-1}(\mathbf{x}-\boldsymbol{\mu})$ in (6) requires $\mathcal{O}(N^2)$ operations, where N is the length of $\mathbf{x}$. Here, we show how it can be done using $\mathcal{O}(DN)$ operations, where D is the dimensionality of the linear subspace (the number of columns in $\boldsymbol{\Lambda}$). Beforehand, SVD is used to compute $\mathbf{U}$ and a diagonal matrix $\mathbf{S}$ such that $\mathbf{U}\mathbf{S}\mathbf{U}' = (\mathbf{I}+\boldsymbol{\Lambda}'\boldsymbol{\Psi}^{-1}\boldsymbol{\Lambda})^{-1}$. Then, we create a $D \times N$ matrix $\mathbf{B} = \mathbf{S}^{1/2}\mathbf{U}'\boldsymbol{\Lambda}'\boldsymbol{\Psi}^{-1}$. Using a matrix inversion identity, it turns out that the quantity we are seeking can be expressed as

$$(\mathbf{x}-\boldsymbol{\mu})'\boldsymbol{\Psi}^{-1}(\mathbf{x}-\boldsymbol{\mu}) - ||\mathbf{B}(\mathbf{x}-\boldsymbol{\mu})||^2, \tag{8}$$

which can be computed using $\mathcal{O}(DN)$ operations.

We do not know of an efficient way to compute $|\boldsymbol{\Lambda}\boldsymbol{\Lambda}'+\boldsymbol{\Psi}|$ exactly. (If you do, please let us know.) However, by leaving out relatively small values during the LU decomposition of $\boldsymbol{\Lambda}\boldsymbol{\Lambda}'+\boldsymbol{\Psi}$, we can compute $\mathbf{L}$ and $\mathbf{U}$ much more quickly and then set

$$\log|\boldsymbol{\Lambda}\boldsymbol{\Lambda}'+\boldsymbol{\Psi}| \approx \sum_i \log|\ell_{ii}|, \tag{9}$$

where ℓ_{ii} is the ith diagonal element of $\mathbf{L}$.

References

[1] M. Turk and A Pentland, “Eigenfaces for recognition,” *Journal of Cognitive Neuroscience*, vol. 3, no. 1, 1991.

[2] I. T. Jolliffe, *Principal Component Analysis*, Springer-Verlag, New York NY., 1986.

[3] A. Pentland, B. Moghaddam, and T. Starner, “View-based and modular eigenspaces for face recognition,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. IEEE Press, June 1994.

[4] B. Moghaddam and A. Pentland, “Probabilistic visual learning for object recognition,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 7, pp. 696–710, July 1997.

[5] C. Bregler and S. M. Omohundro, “Surface learning with applications to lip-reading,” in *Advances in Neural Information Processing Systems 6*, J. D. Cowan, G. Tesauro, and J. Alspector, Eds., pp. 43–50. Morgan Kaufmann, San Francisco CA., 1994.

[6] N. Kambhatla and T. K. Leen, “Fast non-linear dimension reduction,” in *Advances in Neural Information Processing Systems 6*, J. D. Cowan, G. Tesauro, and J. Alspector, Eds., pp. 152–159. Morgan Kaufmann, San Francisco CA., 1994.

[7] K.-K. Sung and T. Poggio, “Example-based learning for view-based human face detection,” MIT AI Memo 1521, CBCL Paper 112, 1994.

[8] G. E. Hinton, M. Revow, and P. Dayan, “Recognizing handwritten digits using mixtures of linear models,” in *Advances in Neural Information Processing Systems 7*, G. Tesauro, D. Touretzky, and T. Leen, Eds. 1995, pp. 1015–1022, MIT Press.

[9] B. J. Frey, *Graphical Models for Machine Learning and Digital Communication*, MIT Press, Cambridge MA., 1998, See `http://www.cs.utoronto.ca/~frey`.

[10] G. E. Hinton, P. Dayan, and M. Revow, “Modeling the manifolds of images of handwritten digits,” Submitted for publication, 1997.

[11] Z. Ghahramani and G. E. Hinton, “The EM algorithm for mixtures of factor analyzers,” University of Toronto Technical Report CRG-TR-96-1, 1997.

[12] B. S. Everitt, *An Introduction to Latent Variable Models*, Chapman and Hall, New York NY., 1984.

[13] B. J. Frey and G. E. Hinton, “Variational learning in non-linear Gaussian belief networks,” Submitted to *Neural Computation*, 1998.

[14] D. Rubin and D. Thayer, “EM algorithms for ML factor analysis,” *Psychometrika*, vol. 47, no. 1, pp. 69–76, 1982.

[15] A. Colmenarez, B. J. Frey, and T. S. Huang, “Face detection and tracking using probability trees,” in preparation, April 1998.