CLASSIFICATION MODULO INVARIANCE, WITH APPLICATION TO FACE RECOGNITION

ANDREW M. FRASER, NICOLAS W. HENGARTNER, KEVIN R. VIXIE,
AND BRENDT E. WOHLBERG

Abstract. We present techniques for constructing classifiers that combine statistical information from training data with tangent approximations to known transformations, and we demonstrate the techniques by applying them to a face recognition task. Our approach is to build Bayes classifiers with approximate class-conditional probability densities for measured data. The high dimension of the measurements in modern classification problems such as speech or image recognition makes inferring probability densities from feasibly sized training data sets difficult. We address the difficulty by imposing severely simplifying assumptions and exploiting *a priori* information about transformations to which classification should be invariant. For the face recognition task, we used a five parameter group of such transformations consisting of rotation, shifts, and scalings. On the face recognition task, a classifier based on our techniques has an error rate that is 20% lower than that of the best algorithm in a reference software distribution.

Key Words: "tangent space", "nuisance parameters", "dimension reduction"

1. Introduction

The task of identifying objects and features from image data is central in many active research fields. In this paper we address the inherent problem that a single object may give rise to many possible images, depending on factors such as the lighting conditions, and the location and orientation of the object relative to the camera. Ideally, the classification should be *invariant* with respect to such changes, because recent empirical studies [4, 5] have shown that the variation induced by these sources in images of a single object is often as large as the variation induced by varying objects.

Inspired by the work of Simard et al. [8, 9], we think of each object as generating a low dimensional manifold in image space by a group of transformations corresponding to changes in nuisance parameters. Given the functional form of the transformation group corresponding to a subset of the nuisance parameters, we could in principle calculate the corresponding manifold associated with a given object from a single image of it. Classification based on the entire manifold leads to procedures that will be invariant to changes induced by that group of transformations. The procedures we describe here approximate such a classification of equivalence classes of images. They are quite general, and we expect them to be useful in the many data analysis problems involving transformations to which classification should be invariant. Although the procedures are general, in the remainder of the paper, we will use terms such as *faces*, *objects*, and *image classification* for concreteness.

Figure 1 illustrates our intuition about equivalence classes of images. The following problems make it difficult to build a classifier that exactly implements this intuition: (1) since these manifolds (i.e., equivalence classes) are highly nonlinear, finding the manifold to which a new point belongs is computationally expensive and (2) for noisy data, the computational problem is further compounded by the uncertainty in the assigned manifold. To address these problems, we use tangents to the manifolds at selected points in image space and model limited excursions by a random effects model with variance related to the curvature of the manifold. Using first and second derivatives of the transformations, our procedures provide substantial improvements to current image classification methods.

2. Structure of our model

We approach the classification problem as a problem of learning from examples. For this paper, an image $I \in \mathcal{I}$ of an object $Y \in \mathcal{Y}$ consists of a raster of N pixels that we choose to represent by a single vector in $\mathbb{R}^N$, with $\mathcal{I} \subset \mathbb{R}^N$. We think of the examples as independent realizations from a joint distribution of images

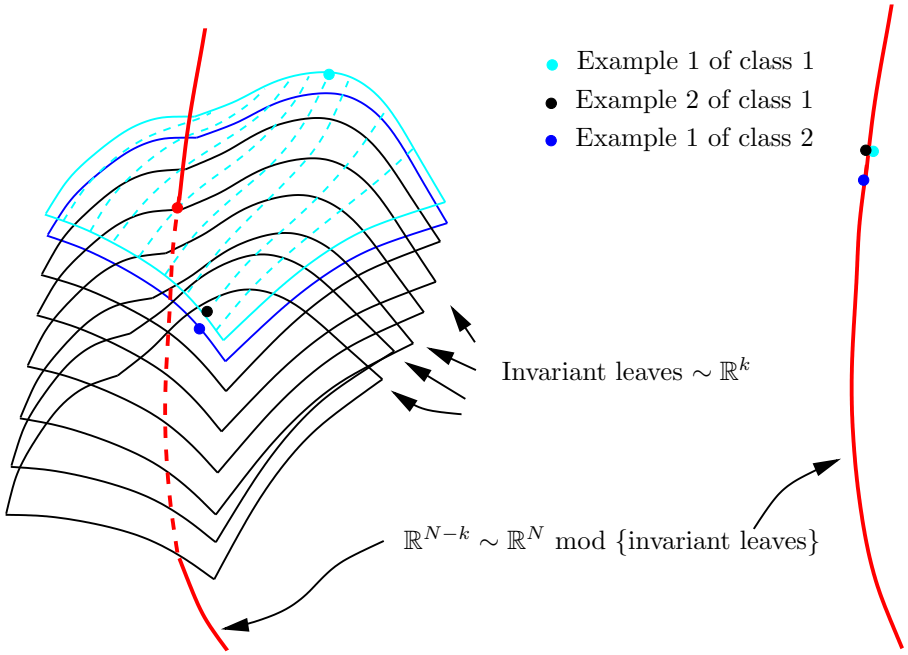

FIGURE 1. Equivalence classes in image space: The cartoon on the left depicts orbits generated by a k-dimensional nuisance parameter ψ. Roughly, the orbits form the leaves of a foliation of the data subset that we are taking to be $\mathcal{I} \subset \mathbb{R}^N$. Data points on these invariant leaves are considered identical for the purposes of our classification. In the quotient space ($\sim \mathbb{R}^{N-k}$) we find that points which seemed far apart in $\mathbb{R}^N$ (but are images of the same object) are identified, whereas points that were close in $\mathbb{R}^N$ are distinguished — see the three example points colored cyan, black, and blue. To implement a Bayes classifier, we could either work in the quotient space, with class-conditional densities defined there, or work in $\mathbb{R}^N$, with each class-conditional density spread uniformly over the corresponding invariant leaf.

and objects $\mathbb{P}(I, Y) = \mathbb{P}(I|Y)\pi(Y)$. Given a set of examples of images of objects $\{(I_i, Y_i) : 1 \le i \le n\}$, we want to estimate a classification rule $\hat{Y} : \mathcal{I} \mapsto \mathcal{Y}$ that assigns to each image an object identifier.

We would like to assess the performance of classification rules by the expected value of the classification error. Given a classification rule $\hat{Y}$ and a pair of values I and Y, the classification error is

$$c(I, Y, \hat{Y}) \equiv \begin{cases} 0 & \text{if } \hat{Y}(I) = Y \\ 1 & \text{otherwise} , \end{cases} \tag{1}$$

and its expected value is

$$\bar{c}(\hat{Y}) \equiv \mathbb{E}_{(Y,I)} c(I, Y, \hat{Y}) . \tag{2}$$

The Bayes classifier

$$\hat{Y}(I) = \operatorname*{argmax}_k \mathbb{P}(Y = k|I) \tag{3a}$$

$$= \operatorname*{argmax}_k \mathbb{P}(I|Y = k)\pi(k) \tag{3b}$$

minimizes the expected classification error and is therefore the standard against which all classification rules are compared.

Many traditional classification procedures can be cast as Bayes classifiers for specific choices of *class-conditional* probability distributions $\mathbb{P}(I|Y)$. For example, if the $\mathbb{P}(I|Y)$ are Gaussian distributions with class specific means $\mu_k = \mathbb{E}_{I|Y_k}(I)$ but common within-class covariance $C_w = \mathbb{E}_{I|Y_k}(I - \mu_k)(I - \mu_k)^t \ \forall k$, then

for the best classifier, the decision boundaries are hyperplanes in $\mathcal{I}$ (see page 39 of Duda, Hart, and Stork [3]). Departures from these model assumptions degrade the performance of the classifier. In particular, the above assumptions do not hold for many image classification tasks. This paper presents a framework for modeling known sources of image variability which in turn leads to better approximations of the within-class data-generating distributions. As a result, classification rules using these distributions have smaller classification errors.

The high dimension of the measurements in modern classification problems such as speech or image recognition makes inferring statistical structure from feasibly sized training data sets difficult. A common practice is to use principal component analysis (PCA) as a preprocessing step to reduce the effective dimensionality of the data before applying classification algorithms. For our work on faces, we followed the default in the code we obtained from the Colorado State University (CSU) archive [1, 2] and used 60% of the number of training images as the number of eigenvectors to retain. This approach works well in practice.

To motivate our proposed modeling of the class-conditional probability distribution, note that images from an object Y will vary with nuisance parameters ψ which characterize effects such as the location of the camera, the orientation of the object, pose, lighting conditions, and facial expression. The variation leads to a set of images that lie on a low dimensional manifold in image space through a function $\tau : \mathcal{Y} \times \Psi \mapsto \mathcal{I}$. Thus, we represent the j^{th} image of object Y_i by

$$I_{ij} = \tau(Y_i, \psi_{ij}) + \varepsilon_{ij}, \tag{4}$$

in which ε_{ij} denotes the measurement error. As the variations along the manifold are usually not well controlled, we shall also view ψ_{ij} as a random displacement that is independent of ε_{ij}. If enough images of the same object were available, one might discover each manifold $\tau(Y_i, \cdot)$ from data using nonparametric estimation techniques such as local linear embedding (LLE) [7] or the ISOMAP algorithm [10]. Our data set was not large enough for these methods.

Instead, we propose to approximate the manifolds $\tau(Y_i, \cdot)$ by exploiting known mathematical properties in the problem. To this end, we decompose the nuisance parameters into two components, $\psi = (\theta, \eta)$. We will use analytic expressions to describe the effects of the first component, the *analytic nuisance parameters*, denoted by θ. The second component, denoted by η, parameterizes variations that we will handle with a stochastic model to be fitted to the training data. For face images, we let θ describe changes in position, orientation, and scale of the object relative to the camera, and we imagine η being a vector of parameters for characteristics such as lighting and the contractions of specific facial muscles. Changes in θ move images on a submanifold $\tau(Y_i, \cdot, \eta)$.

If we consider only small excursions of ψ_{ij} from its mean $\bar{\psi}_i \equiv (\bar{\theta}_i, \bar{\eta}_i)$, we may linearize the model

$$\tau(Y_i, \psi_{ij}) \approx \mu_i + V_i(\theta_{ij} - \bar{\theta}_i) + U_i(\eta_{ij} - \bar{\eta}_i) \tag{5a}$$

where

$$\mu_i \equiv \tau(Y_i, \bar{\psi}_i), \tag{5b}$$

$$V_i \equiv \left.\frac{\partial \tau(Y_i, \psi_{ij})}{\partial \theta_{ij}}\right|_{\psi_{ij}=\bar{\psi}_i}, \tag{5c}$$

and

$$U_i \equiv \left.\frac{\partial \tau(Y_i, \psi_{ij})}{\partial \eta_{ij}}\right|_{\psi_{ij}=\bar{\psi}_i}. \tag{5d}$$

Note that for each i in Eq. (5a), the derivative matrices V_i are known, whereas the derivative matrices U_i are not. In analogy to random-effect models, we respond to ignorance about and inability to estimate the U_i by treating them as random. Thus we group $U_i(\eta_{ij} - \bar{\eta}_i)$ with ε_{ij} into an error term. Making the further simplifying assumption that θ_{ij}, $U_i(\eta_{ij} - \bar{\eta}_i)$, and ε_{ij} are independent and Gaussian, yields

$$I_{ij} \sim \mathcal{N}\left(\mu_i, V_i C_{\theta,i} V_i^t + C_w\right). \tag{6}$$

For each individual i, we calculate $C_{\theta,i}$, the analytic nuisance-parameter covariance, using the techniques of section 3, and we estimate the pooled within-class covariance C_w from the training data.

3. Deriving model parameters from data and second-order approximations to manifolds

According to Eq. (6), the class-conditional densities are Gaussian, i.e.,

$$\mathbb{P}(I|Y_i) = \frac{1}{\sqrt{(2\pi)^N |C_i|}} \exp\left(-\frac{1}{2}(I-\mu_i)^t C_i^{-1}(I-\mu_i)\right), \tag{7}$$

where N is the dimension of $\mathcal{I}$. All that remains before we can implement a Bayes classifier, Eq. (3), is to specify the parameter values, which we do as follows:

The class mean, μ_i: Use the mean of the images available for individual Y_i.

The class-conditional covariance matrix C_i:

$$C_i \equiv V_i C_{\theta,i} V_i^t + C_w \tag{8}$$

The prior probabilities for the classes, $\pi(i)$: For simplicity, we suppose that $\pi(i) \propto \sqrt{|C_i|}$.

The pooled within-class covariance, C_w: Fit to the training data.

Tangent to the manifold, V_i: Differentiate the function τ with respect to the analytic nuisance parameters θ and evaluate at μ_i (see Appendix B).

The covariance of the analytic nuisance parameters, $C_{\theta,i}$: We use a formula, Eq. (17), for this matrix that depends on the second derivative of the function τ and the within-class covariance matrix C_w. We justify Eq. (17) in Subsection 3.1, and we explain the calculation of second derivatives in Appendix B.

With these parameters, the Bayes classifier simply minimizes Mahalanobis distance,

$$\hat{Y}(I) = \operatorname*{argmax}_i \mathbb{P}(I|Y=i)\pi(i) \tag{9a}$$

$$= \operatorname*{argmin}_i (I-\mu_i)^t C_i^{-1}(I-\mu_i)\ . \tag{9b}$$

3.1. Covariance of the analytic nuisance parameters.

We choose C_θ for each individual Y_i (we drop the second subscript on $C_{\theta,i}$) as a compromise between two contradictory goals. We want to allow large excursions in the tangent directions, and at the same time we want to limit the error of approximating the manifold with its tangent. Figure 2 illustrates the situation. The intuition is that in order to constrain the distance from points on the tangent to the true manifold, the bounds on displacements along the tangent should be inversely proportional to the second derivative. If everything were scalar, we could write $H \equiv \frac{\partial^2 \tau(\theta)}{\partial \theta^2}$ and

$$\sigma_\theta \propto \frac{1}{H}\ . \tag{10}$$

Since all of the variables are vectors, we must generalize Eq. (10). In terms of the Taylor series

$$\tau(Y,\theta) = \tau(Y,0) + V\theta + \theta^t H \theta + R,$$

where we have set $\bar{\psi} = 0$ and

$$V_i = \left.\frac{\partial \tau(Y,\theta)}{\partial \theta_i}\right|_{\theta=0} \text{ and } H_{i,j} = \left.\frac{\partial^2 \tau(Y,\theta)}{\partial \theta_i \partial \theta_j}\right|_{\theta=0}, \tag{11}$$

there are two conflicting goals:

Big: We want to allow θ to be large so that we can classify images with large displacements in the invariant directions.

Small: We want $\theta^t H \theta \in \mathcal{I}$ to be small so that the truncated Taylor series will be a good approximation.

We have assumed that the distribution of θ is Gaussian. Now we search for a resolution of these conflicting goals in terms of a norm on Θ and the covariance of its distribution, C_θ.

If, for a particular image component d, the Hessian H_d has both a positive eigenvalue λ_1 and a negative eigenvalue λ_2, then the quadratic term is zero along a direction e_0 that is a linear combination of the corresponding eigenvectors, i.e., $(\gamma e_0)^t H_d (\gamma e_0) = 0\ \forall \gamma$. We suspect that higher-order terms will contribute to significant errors when $\gamma \geq \min\left(|\lambda_1|^{\frac{1}{2}}, |\lambda_2|^{\frac{1}{2}}\right)$, so we eliminate the canceling effect by replacing H_d with

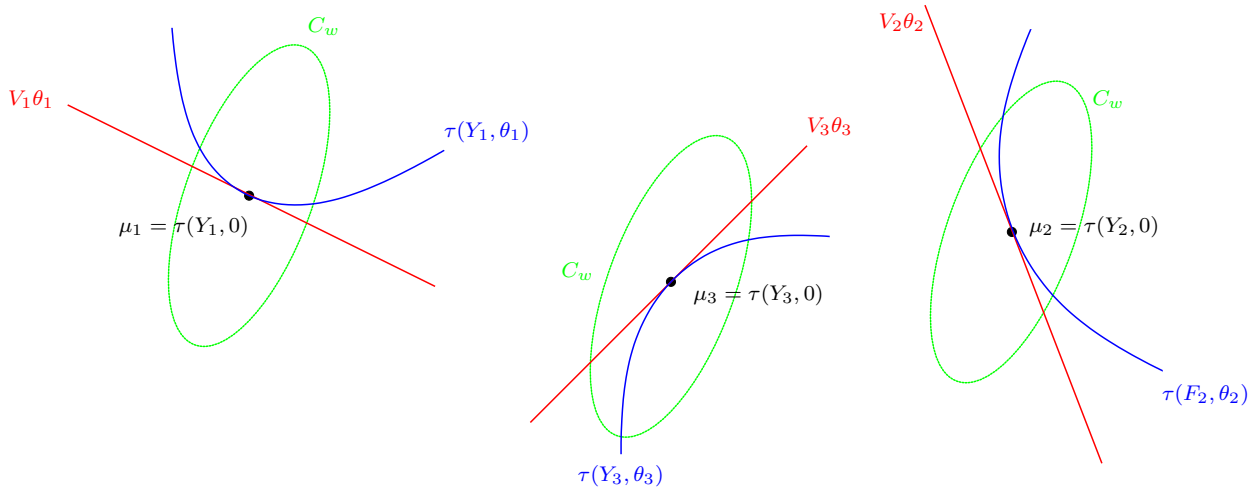

FIGURE 2. A geometric view of the model. For each class $k \in \{1,2,3\}$, the black dot labeled μ_k represents the class mean, the blue curve labeled $\tau(Y_k,\theta_k)$ represents the manifold generated by the analytic nuisance parameters, the red line labeled $V_k\theta_k$ represents the tangent to the manifold at μ_k, and the green ellipse labeled C_w represents level sets of $(I-\mu_k)^t C_w^{-1}(I-\mu_k)$. To obtain class-specific covariance matrices C_k, we augment the pooled covariance with a term whose direction is given by the tangent to the manifold and whose magnitude is proportional to the inverse of the curvature of the manifold.

the *positive square root* of $(H_d H_d)$, i.e., if an eigenvalue λ of H_d is negative, replace it with $-\lambda$. This suggests the following *mean root square* norm

$$|\theta|_{H_{\text{mrs}}} \equiv \sqrt{\sum_{d=1}^{N} \theta^t \sqrt{H_d H_d}\,\theta} \quad . \tag{12}$$

The last modification we make to the norm addresses the following objection. The norm of Eq. (12) would suppress useful parameter excursions in any direction d for which H_d was large, even if that direction were unimportant for recognition. For example, this effect would suppress horizontal shifts of a face image if the background had vertical stripes with sharp edges. Our modification uses the eigenvalues, λ_d, of the pooled within-class covariance matrix C_w to quantify the importance each component of an image. We suppose that larger values of λ_d correspond to directions that are less important for recognition. If there is a large within-class variance in the direction of component d, we allow large parameter excursions in that direction even if they cause large errors in that component.

Using the eigen-decomposition

$$C_w = \sum_d \lambda_d e_d e_d^t$$

we break the $\dim(\Theta) \times \dim(\mathcal{I}) \times \dim(\Theta)$ tensor H into components

$$H_d \equiv e_d^t H \ . \tag{13}$$

For each component, we define the $\dim(\Theta) \times \dim(\Theta)$ matrix

$$H_d^+ \equiv \sqrt{H_d H_d} \ , \tag{14}$$

and take the average to get

$$\bar{H} \equiv \sum_d H_d^+ \lambda_d^{-\frac{1}{2}} \ . \tag{15}$$

Then we define the norm

$$|\theta|_{\bar{H}} \equiv \sqrt{\theta^t \bar{H} \theta}.$$

Given H and C_w, one can calculate $\bar{H}$ using Eqs. (13), (14), and (15). Then by using the determinant $|C_\theta|$ to quantify goal **Big:** (allow θ to be large) and using $\mathbb{E}\left(|\theta|_{\bar{H}}\right)^2$ to quantify goal **Small:** (keep $\theta^t H \theta \in \mathcal{I}$ small), we get the constrained optimization problem:

Maximize: the determinant $|C_\theta|$
Subject to:

$$\mathbb{E}\left(|\theta|_{\bar{H}}\right)^2 \leq \gamma \,, \tag{16}$$

where γ is a constant.

The solution to the problem is

$$C_\theta = \alpha \left(\bar{H}\right)^{-1} , \tag{17}$$

where α, which is a function of γ, is a constant that balances the competing goals.

To verify that Eq. (17) indeed solves the optimization problem, note the following:

$$\begin{aligned} \mathbb{E}\left(|\theta|_{\bar{H}}\right)^2 &= \mathbb{E}\left(\sum_{i,j} \theta_i \bar{H}_{i,j} \theta_j\right) \\ &= \sum_{i,j} \bar{H}_{i,j} \mathbb{E}\left(\theta_i \theta_j\right) \\ &= \mathrm{Tr}\left(\bar{H} C_\theta\right) . \end{aligned}$$

In the coordinates that diagonalize $\bar{H}$, Eq. (16) only constrains the diagonal entries of C_θ. Of the symmetric positive definite matrices with specific diagonal entries, the matrix that has the largest determinant is simply diagonal. So C_θ and $\bar{H}$ must be simultaneously diagonalizable, and the problem reduces to

$$\begin{aligned} \text{Maximize:} \quad & \prod_{i=1}^{\dim(\Theta)} \sigma_i \\ \text{Subject to:} \quad & \sum_{i=1}^{\dim(\Theta)} \sigma_i h_i = \gamma \,. \end{aligned}$$

The Lagrange multiplier method yields Eq. (17).

Summary. Given a new image I, we estimate its class with

$$\hat{Y}(I) \equiv \underset{k}{\mathrm{argmin}} (I - \mu_k)^t C_k^{-1} (I - \mu_k) \,,$$

where Eq. (8) gives C_k in terms of $C_{\theta,k}$ which in turn is given by Eq. (17). We have derived the parameters of this classifier by synthesizing statistics from training data with analytic knowledge about transformations we wish to ignore.

4. Application to Face Recognition

We tested our techniques by applying them to a face recognition task and found that they reduce the error rate by more than 20% (from an error rate of 26.7% to an error rate of 20.6%). We used an analytic expression for transformations in image space and developed procedures for evaluating first and second derivatives of the transformations. The transformations have the following five degrees of freedom:

- Horizontal translation
- Vertical translation
- Horizontal scaling
- Vertical scaling
- Rotation

To implement the test, we relied on the FERET data set [6] and a source code package from Beveridge et al. [1, 2] at CSU for evaluating face recognition algorithms.

4.1. **The test task.** Version 4.0 (October 2002) of the CSU package contains source code that implements 13 different face recognition algorithms, scripts for applying those algorithms to images from the FERET data set, and source code for Monte Carlo studies of the distribution of the performance of the recognition algorithms. For each algorithm tested, the CSU evaluation procedure reports a distribution of performance levels. The specific task is defined in terms of a single *probe* image and a *gallery* of N_G images. The images in the gallery are photographs of N_G distinct individuals. The gallery contains a single *target* image, which is another photograph of the individual represented in the probe image. Using distances reported by the algorithm under test, the evaluation procedure sorts the gallery into a list, placing the target image as close to the top as it can. The algorithm scores a success at rank n if the target is in the first n entries of the sorted list. The CSU evaluation procedure randomly selects $N_G \times 10,000$ gallery-probe pairs and reports the distribution of successful recognition rates as a function of rank. By applying the evaluation procedure to two of the recognition algorithms distributed by CSU, we produced Fig. 3.

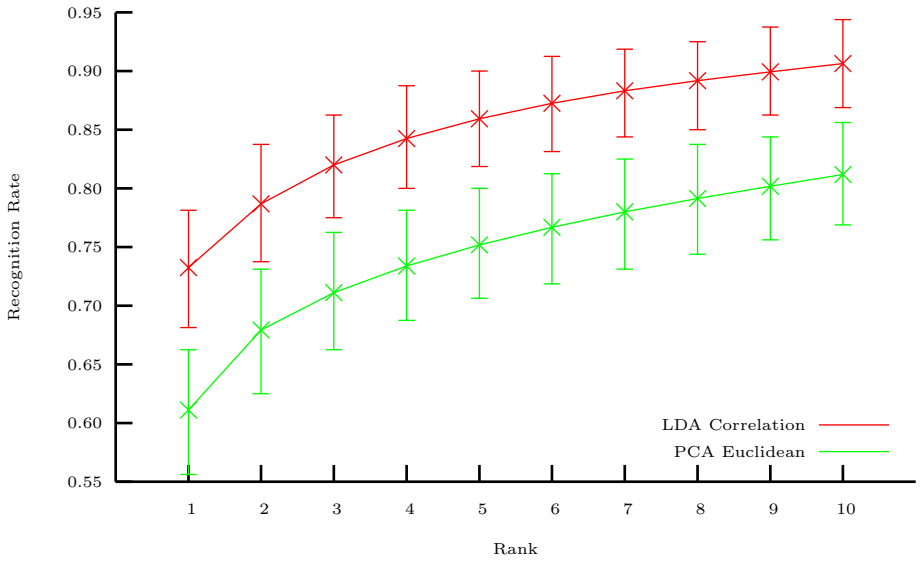

FIGURE 3. Recognition rate versus rank for two face recognition algorithms: *Linear discriminant analysis with correlation distance* (LDA Correlation) and *principal component analysis with Euclidean distance* (PCA Euclidean). For detailed descriptions of these recognition algorithms see [1, 2]. Of the recognition algorithms in the CSU package, linear discriminant analysis with correlation distance has the best recognition rate and principal component analysis with Euclidean distance has the worst recognition rate. For each algorithm, estimates of the mean recognition rate and a 95% confidence interval appear.

Restricting the test data set to those images in the FERET data that satisfy the following criteria:

- Coordinates of the eyes have been measured and are part of the FERET data.
- There are at least four images of each individual.
- The photographs of each individual were taken on at least two separate occasions.

yields a set of 640 images consisting of 4 images each of 160 individuals. Thus we use $N_G = 160$. Of the remaining images for which eye coordinates are given, we used a training set of 591 images consisting of 3 images per individual for 197 individuals. The testing and training images were uniformly preprocessed by code from the CSU package. In [1] the authors describe the preprocessing as,

> "All our FERET imagery has been preprocessed using code originally developed at NIST and used in the FERET evaluations. We have taken this code and converted it ...
>
> Spatial normalization rotates, translates and scales the images so that the eyes are placed at fixed points in the imagery based on a ground truth file of eye coordinates supplied with the FERET data. The images are cropped to a standard size, 150 by 130 pixels. The NIST

code also masks out pixels not lying within an oval shaped face region and scales the pixel data range of each image within the face region. In the source imagery, grey level values are integers in the range 0 to 255. These pixel values are first histogram equalized and then shifted and scaled such that the mean value of all pixels in the face region is zero and the standard deviation is one."

Each recognition algorithm calculates subspaces and fits parameters using the preprocessed training images and knowledge of the identity of the individuals in the images. Then, using those parameters, each algorithm constructs a matrix consisting of the distances between each pair of images in the testing set of 640 images. Thus, in the training phase, one can calculate the mean image, μ_i, of an individual, but in the testing phase, the algorithm has no information about the identity of the individuals in the images. Distributions for the Rank 1 recognition performance of 13 algorithms distributed in the CSU package appear in Fig. 4. Following Turk and Pentland [11], all of the CSU algorithms use principal component analysis as a first step. Those with the best recognition rates also follow Zhao et al. [12] and use a discriminant analysis.

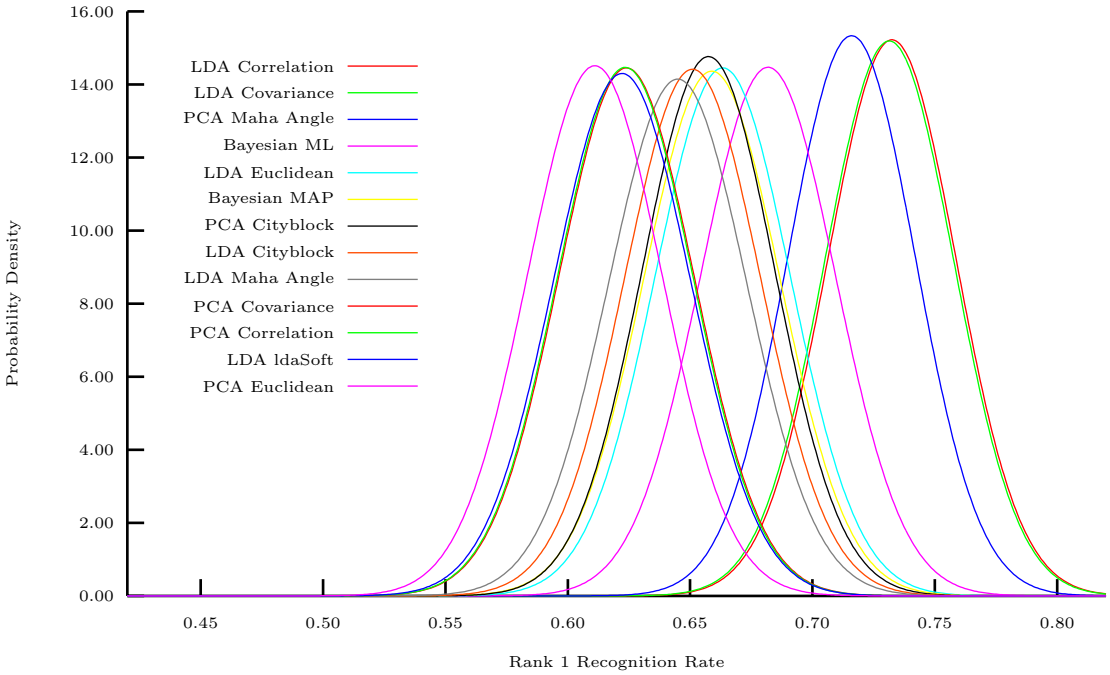

FIGURE 4. Approximate distributions for the Rank 1 recognition performance of thirteen algorithms. For each algorithm, a Gaussian is plotted with a mean and variance estimated by a Monte-Carlo study — the key lists the algorithms in order of decreasing mean. Code for both the recognition algorithms and the Monte-Carlo study came from Beveridge et al. of Colorado State University [2].

4.2. **Modifications to our approach.** We developed three recognition algorithms for the test task of Subsection 4.1. The first consists of the general techniques of Section 3 combined with minor modifications to fit the test task. In Section 3 we supposed that we would have several examples of each class, making an estimate of each class mean μ_k plausible, but for the task defined by the CSU evaluation procedure, we must simply provide 640×640 interimage distances. We developed the second two algorithms after observing that the CSU algorithms based on angular distance perform best (see Fig. 4).

Each of our algorithms operates in a subspace learned from the training data and uses an estimated covariance,

$$C_k = C_w + \alpha V_k \bar{H}_k^{-1} V_k^t,$$

associated with each image I_k (see Eq. (8)). While the details of our estimation of C_k for each test image appear in Appendix A, we list the key ideas here:

- Use the training data (which includes image identities) to calculate raw within-class sample covariances, $\tilde{C}_w$. Regularize the raw covariances as follows: (1) Do an eigen-decomposition to find $\tilde{C}_w = Q\tilde{\Lambda}Q^t$. (2) Sum the eigenvalues, $S = \sum_i \tilde{\lambda}_i$. (3) Using a small parameter δ, put a floor on the regularized eigenvalues, $\lambda_i = \tilde{\lambda}_i + \delta S$. (4) Multiply to get the estimated matrix $C_w = Q\Lambda Q^t$, which has no eigenvalues less than δS.
- Conceptually convolve the test image with a Gaussian kernel that has mean zero and variance

$$\begin{bmatrix} \left(\frac{h-1}{8}\right)^2 & 0 \\ 0 & \left(\frac{h-1}{8}\right)^2 \end{bmatrix},$$

  where h is an adjustable parameter in the code that must be an odd integer. Change variables to transfer differentiation from the image to the kernel. Evaluate the matrices V_k and $\bar{H}_k$ (see Eqs. (11) and (17)) by convolving (using FFT methods) differentiated kernels with the image.

Thus α, δ, and h are three adjustable parameters in the estimate of C_k.

Given two images, I_1 and I_2, our three algorithms report the following three distances:

- **Symmetrization of Original Idea**

$$D_1(I_1, I_2) = (I_1 - I_2)^t \left(C_1^{-1} + C_2^{-1}\right)(I_1 - I_2) \tag{18}$$

- **Mahalanobis Angle then Symmetrize**

$$D_2(I_1, I_2) = 2 - \frac{I_1^t C_1^{-1} I_2}{\sqrt{\left(I_1^t C_1^{-1} I_1\right)\left(I_2^t C_1^{-1} I_2\right)}} - \frac{I_1^t C_2^{-1} I_2}{\sqrt{\left(I_1^t C_2^{-1} I_1\right)\left(I_2^t C_2^{-1} I_2\right)}} \tag{19}$$

- **Symmetrize Covariance then Mahalanobis Angle**

$$D_3(I_1, I_2) = 1 - \frac{I_1^t A I_2}{\sqrt{\left(I_1^t A I_1\right)\left(I_2^t A I_2\right)}}\ , \quad \text{where } A = (C_1 + C_2)^{-1} \tag{20}$$

Evaluating each of the first two distances on the test set of 640 images takes about 30 minutes on a 2.2 GHz Pentium III. We found that the second distance performed better than the first. Because we estimated that evaluating the third distance would take about 160 hours, we instead implemented a hybrid of D_2 and D_3, with the idea of using the faster D_2 to identify a subgroup of images that probably contains the match and then applying D_3 to that subgroup to make better guesses. In more detail, we do the following steps.

(1) For all image pairs calculate $D_2(I_i, I_j)$.
(2) If $D_2(I_i, I_j)$ is less than a threshold ϵ, calculate $D_3(I_i, I_j)$.
(3) Find a scaling factor s such that if $D_2(I_i, I_j) \geq \epsilon$, then $s \cdot D_2(I_i, I_j) > D_3(I_k, I_l)\ \forall (k, l)$.
(4) For all pairs (i, j) report the following:

$$\tilde{D}_3 = \begin{cases} D_3(I_i, I_j) & \text{if } D_2(I_i, I_j) < \epsilon \\ s \cdot D_2(I_i, I_j) & \text{otherwise} \end{cases}. \tag{21}$$

The idea of the hybrid is that the key to better Rank 1 recognition is doing well with images that are close to each other. There are $205,120 = \frac{(640)(640+1)}{2}$ interimage distances, but of those, only $1,600 = 160 \cdot \frac{(4)(4+1)}{2}$ are distances between images of the same individual. The parametric plot in Fig. 5 indicates that there is a value of ϵ for which 75% of the 1,600 within-class distances will be calculated using $D_3(I_i, I_j)$ while 98.7% of *all* interimage distances will be calculated using only D_2.

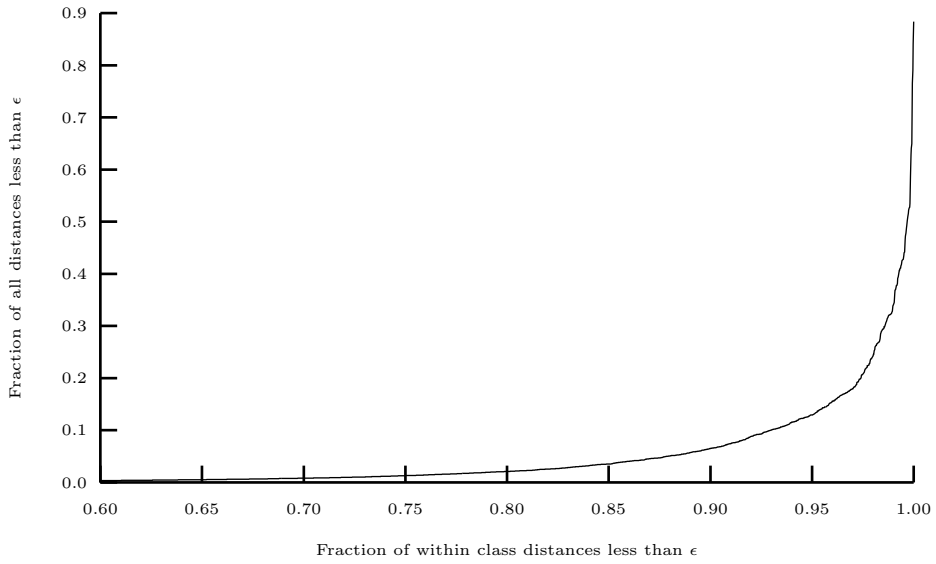

FIGURE 5. A parametric plot, with independent parameter ϵ, comparing the fraction of all distances less than ϵ in the distance matrix against the fraction of distances less than ϵ that correspond to within-class distances. We used a value of ϵ that is greater than 75% of all within-class distances and less than 98.7% of all distances.

4.3. **Results.** We are pleased that, as depicted in Fig. 6, each of our algorithms performs better than all of the algorithms in the CSU package. The figure displays results for the following algorithms (in order of Rank 1 performance):

$\tilde{\mathbf{D}}_3$: The hybrid defined in Eq. (21) that uses symmetrized covariance and then Mahalanobis angle for fine distinctions. The parameter values were set as follows: α (which balances second-order errors with extent of tangent approximation) set to 100, δ (which characterizes the regularization of the estimated within-class covariance matrix) set to 0.0003, and h (the width of the kernel used for calculating derivatives) set to 11.

$\mathbf{D}_2$: Mahalanobis angle then symmetrize, defined in Eq. (19). Parameters α, δ, and h set as for $\tilde{D}_3$.

$\mathbf{D}_2(\alpha = \mathbf{0})$: The same as D_2 above, except that $\alpha = 0$. Note that this simple algorithm, which does not use any tangent information, outperforms D_1 and all of the algorithms in the CSU package.

$\mathbf{D}_1$: Symmetrization of the ideas described in Section 3, defined in Eq. (18). Parameters α, δ, and h set as for $\tilde{D}_3$. We are surprised that the *angular distance* algorithms are better than this one.

Linear discriminant analysis (LDA) Correlation: Fisher LDA followed by a correlation measure of distance. This is the best of the algorithms in the CSU package.

In choosing parameters and subspaces, we either followed the CSU defaults or optimized performance of D_2 on the test set. For the initial PCA projection, the CSU default is to retain 60% of the degrees of freedom. Thus, if the principal component analysis starts with N images and N is less than the number of pixels in each image, then the dimension of the retained subspace is $0.6(N-1)$. Since we used 591 training images, we projected to a PCA subspace of dimension 354. The singular value decomposition (SVD) spectrum of the preprocessed training images appears in Fig. 7.

Figure 8 describes the sensitivity of the Rank 1 recognition rate for D_2 to changes in the adjustable parameters in our estimates of C_k for each image I_k. Level sets of recognition rate are roughly elliptical in the $(h, \log(\delta))$ planes at fixed α values, and they extend indefinitely in the positive α direction. This indefinite extent suggests that rather than carefully choosing C_θ in terms of $\bar{H}$ with all of the discussion in Section 3, we could obtain the same performance by simply projecting out the directions tangent to the invariant transformations. We believe that this curious result follows from the data being so sparse that the

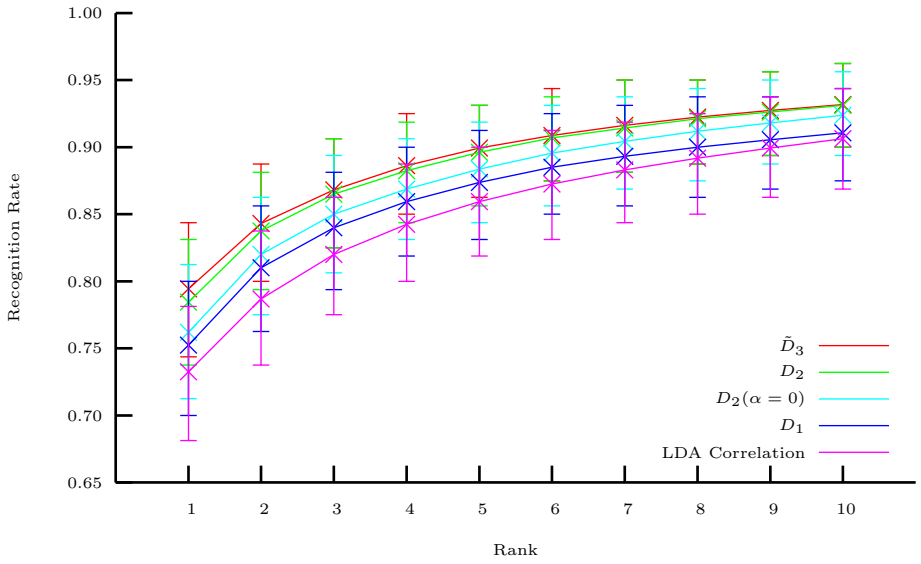

FIGURE 6. The mean recognition rate and 95% confidence intervals as a function of rank for the following algorithms: $\tilde{D}_3$ (the hybrid algorithm); D_2 (Mahalanobis angle then symmetrize); $D_2(\alpha = 0)$ (D_2 with α set to 0, which uses the pooled within-class covariance, C_w, but no tangent information); D_1 (symmetrization of original idea); and the best of the algorithms in the CSU package (linear discriminant analysis, LDA, with a correlation measure of distance).

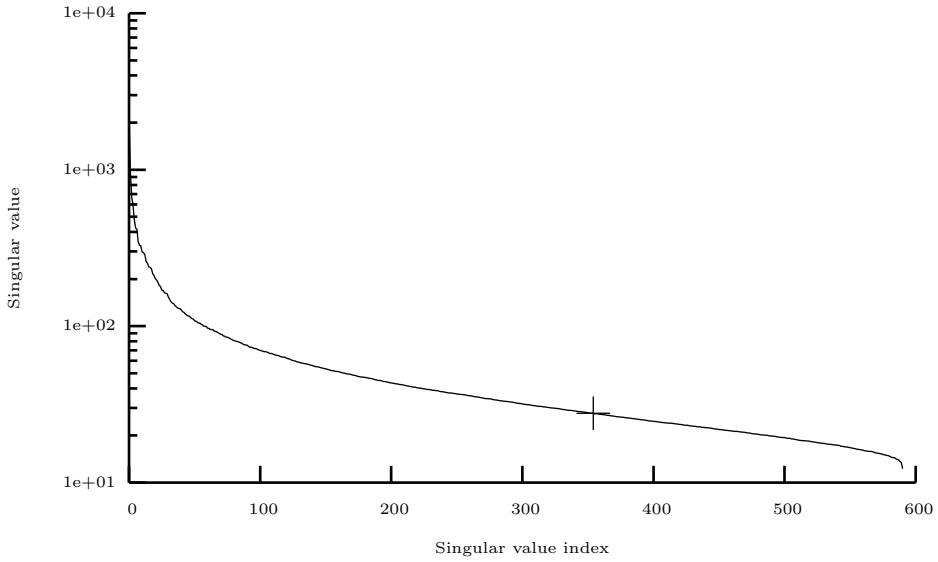

FIGURE 7. The singular spectrum of the training data. Following the CSU default, we use a projection that retains the first 354 directions (the cut-off point is indicated on the plot).

tangent spaces at each image do not include or even come close to other images and that for denser data sets, finite values of α would be optimal.

On the basis of the plots in Fig. 8, we chose the values $\alpha = 100$, $h = 11$, and $\delta = 0.0003$. We chose values of α and h that yielded slightly suboptimal performance because of a prejudice for smaller α values

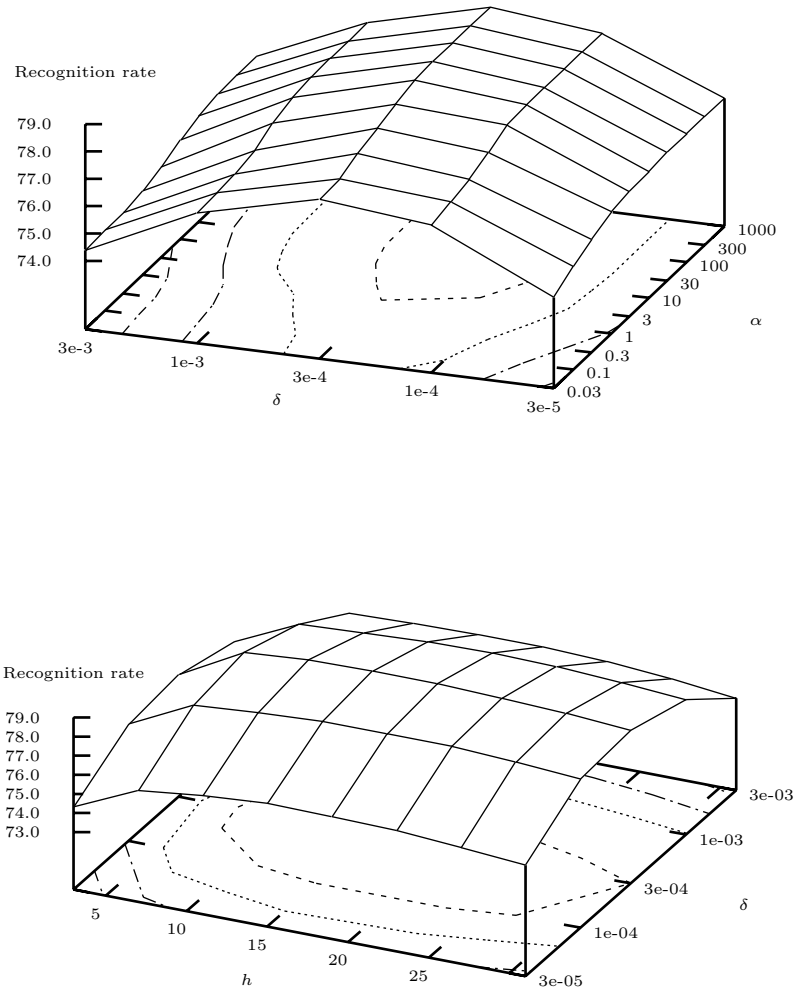

FIGURE 8. Dependence of the Rank 1 recognition rate for D_2 on α, h, and δ. Recall that α balances control of second-order errors with the ability to accommodate large excursions on the invariant manifolds, h is the width of the smoothing kernel used in calculating derivatives, and δ is the parameter that characterizes the regularization of the within-class covariance matrix, C_w. In the upper plot, $h = 11$, and in the lower plot, $\alpha = 100$.

and because the algorithms run faster with smaller h values. The flatness of the plots in Fig. 8 suggests that if we had chosen the parameter values on the basis of performance on a set of training images that our performance on the test set would not change much.

The 354×354 within-class covariance matrix C_w has 62,385 degrees of freedom, and we estimate it using 591 images of 197 individuals, i.e., 394 degrees of freedom. We are surprised that the optimal regularization is not more severe. Figure 9 illustrates the magnitude of the regularization that we used.

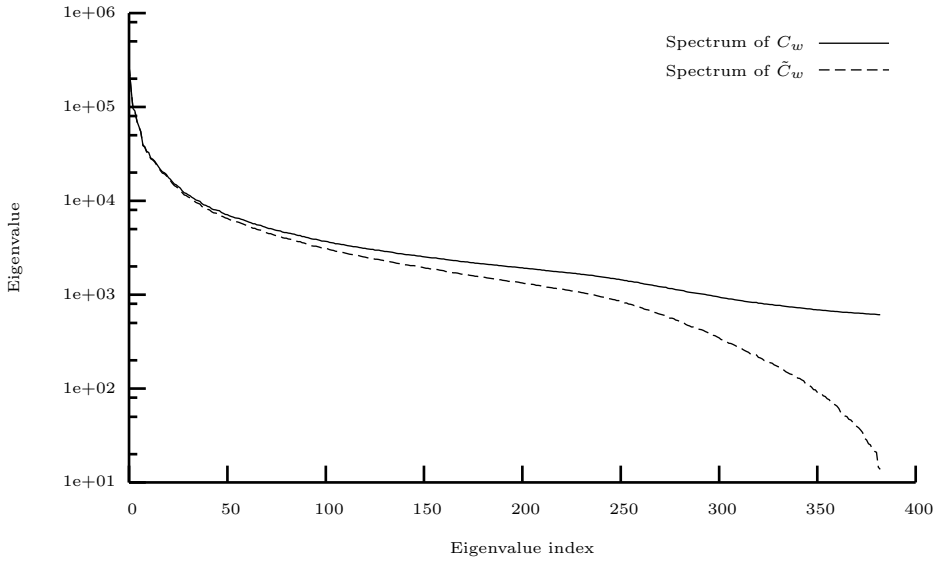

FIGURE 9. Plots of the eigenvalue spectrum of the within-class covariance before and after regularization ($\tilde{C}_w$ and C_w respectively).

5. Conclusions

We have presented techniques for constructing classifiers that combine statistical information from training data with tangent approximations to known transformations, and we have demonstrated the techniques by applying them to a face recognition task. Our techniques improve on the work of Simard et al. by using a curvature term to control approximation errors. For the face recognition task we used a five parameter group of invariant transformations consisting of rotation, shifts, and scalings. On the face test case, a classifier based on our techniques has an error rate more than 20% lower than that of the best algorithm in a reference software distribution.

The improvement we obtained is surprising because our techniques handle rotation, shifts, and scalings, but we also preprocessed the FERET data with a program from CSU that centers, rotates, and scales each image based on measured eye coordinates. While our techniques may compensate for errors in the measured eye coordinates or weaknesses in the preprocessing algorithms, we suspect that much of the improvement is due to similarities between the transformations we handle and differences between images. For example, a smile is probably something like a dilation in the horizontal direction.

We expect competitive results on other data classification tasks such as the recognition of near field sonar signatures perturbed by time varying Doppler shifts, the removal of atmospheric haze from hyperspectral data and compensation for channel characteristics in speaker identification.

Appendix A. Algorithm

In describing the algorithms, when we refer to *our example*, we mean the face recognition work described in Section 4.

A.1. **Algorithm training.**

1: Compute the mean of the training images $X_i, i \in \{1, ..., N_{\mathrm{Tr}}\}$ (for our example $N_{\mathrm{Tr}} = 591$.)

$$\mu \equiv \frac{1}{N_{\mathrm{Tr}}} \sum_{i=1}^{N_{\mathrm{Tr}}} X_i$$

2: Compute the SVD of the matrix whose columns are the training images with mean removed:

$$[U, S, W] = \mathrm{SVD}(\hat{\mathbf{X}}) \ ,$$

where

$$\hat{\mathbf{X}} \equiv [\hat{X}_1, ..., \hat{X}_{N_{\rm Tr}}] = [X_1 - \mu, ..., X_{N_{\rm Tr}} - \mu]$$

and S is diagonal, and we have that $\hat{\mathbf{X}} = USW^t$.

3: Pick $N_{\rm proj}$, the reduced dimension, by simply setting $N_{\rm proj} = \lfloor .6N_{\rm Tr} \rfloor$. Now create the corresponding projection matrix $\mathbf{P}$:

$$\mathbf{P} = [W_1 W_2 ... W_{N_{\rm proj}}]^t$$

4: Define $X_{i,k}$ to be the ith instance of the kth individual: this is simply a re-labeling of the $N_{\rm Tr}$ training images. Define N_k to be the number of instances of the kth individual. For each individual k, compute the mean μ_k, by

$$\mu_k \equiv \frac{1}{N_k} \sum_{i=1}^{N_k} X_{i,k}$$

and use this to create the reduced (dimensional) population, $X_{i,k}$ defined by:

$$\hat{R}_{i,k} \equiv \mathbf{P}\hat{X}_{i,k} = \mathbf{P}X_{i,k} - \mathbf{P}\mu_k.$$

Note that we are working in the reduced dimensional description space now.

5: Compute the within-class covariance:

$$\tilde{C}_w = \frac{1}{N_{\rm Tr}} \hat{R}\hat{R}^t \ ,$$

where $\hat{R}$ is the matrix with columns $\hat{R}_{i,k}$ defined in the previous step.

6: Compute the eigenvalue decomposition of $\tilde{C}_w$:

$$\tilde{C}_w \tilde{\Lambda} = \tilde{\Lambda} E \ ,$$

where E is the matrix whose columns e_k are the eigenvectors of $\tilde{C}_w$ and $\tilde{\Lambda} = \{\tilde{\lambda}_k\}_1^{N_{\rm Tr}}$ is the vector of eigenvalues of $\tilde{C}_w$.

7: Regularize $\tilde{C}_w$. Set $\Lambda = \tilde{\Lambda} + \delta S \mathbb{I}$, where $S = \sum_k \tilde{\lambda}_k$ and $\mathbb{I}$ is the vector of 1s the same length as $\tilde{\Lambda}$ for our example we set $\delta = 0.0003$. Now define $C_w \equiv E\Lambda E^t$.

A.2. **Algorithm testing.**

1: For each image Z_k, $1 \le k \le N_{\rm Te}$ in the test set (for our example, $N_{\rm Te} = 640$), compute the derivatives V_k and the Hessian matrix $\bar{H}_k$:

$$C_k \equiv C_w + \alpha P V_k \bar{H}_k V_k^t P^t \ ,$$

where

$$\bar{H}_k = \sum_d \sqrt{\frac{H_d H_d}{\lambda_d}} \ ,$$

where $\sqrt{A}$ is the principal square root of the matrix A and

$$\bar{H}_d \equiv H(P^t e_d, \cdot, \cdot) \ ,$$

where $H(P^t e_d, \cdot, \cdot)$ is the matrix that results from applying the *projected* second derivative tensor to the dth eigenvector of C_w and λ_d is the eigenvalue corresponding to the dth eigenvector (see Eq. (14)).

2: Remove the mean from the test images, project

$$\hat{I}_k = PZ_k - P\mu$$

and compute the distance matrix D' with $\{i,j\}$th component. For example, in the case of D_1 (see Eq. (18)), we calculate

$$d'_{i,j} = (\hat{I}_i - \hat{I}_j)^t C_i^{-1} (\hat{I}_i - \hat{I}_j) \text{ for } 1 \le i,j \le N_{\rm Te} \ .$$

3: Symmetrize the resulting matrix D' to get D:

$$D \equiv \frac{1}{2}\left(D' + D'^{t}\right) \ .$$

4: Do the permutation experiments to obtain the histograms and rank curves by a re-sampling of the distances in the matrix D computed in the previous step.

A.3. **Algorithm summary.** The algorithm uses training data to select the reduced dimensional description of the data (training and test data). The training data are also used to generate the first global metric which is then modified locally through computations performed on the test data to yield a locally varying metric which is used to generate a distance matrix which contains all the pairwise distances between all pairs of the test data.

Appendix B. Derivatives

In this appendix we give explicit formulas for the calculation of tangents and second-derivative tensors for the five-parameter transformation that we used in our example. The action of any particular element of this five-parameter transformation produces another image of the *same* individual. We represent this action as

$$I \to \tau(\boldsymbol{\theta})I \ ,$$

where we have that $\boldsymbol{\theta} \equiv \{\theta_i\}_1^5$ and

$$\tau(\mathbf{0})I = I.$$

If we define

$$\tilde{I}_{\boldsymbol{\theta}}(x,y) \equiv \int (\tau(\boldsymbol{\theta})I)(u,v)K(x-u,y-v)\ dudv \ ,$$

then we can express the tangents we are interested in as

$$D_{\theta_i} I \equiv \lim_{\boldsymbol{\epsilon}\to 0} \Delta_{\theta_i,\boldsymbol{\epsilon}} I \ ,$$

where

$$\Delta_{\theta_i,\boldsymbol{\epsilon}} I \equiv \frac{1}{||\boldsymbol{\epsilon}||}\left(\tilde{I}_{\boldsymbol{\theta}=\boldsymbol{\epsilon}}(x,y) - \tilde{I}_{\boldsymbol{\theta}=\mathbf{0}}(x,y)\right) \ ,$$

where $\boldsymbol{\epsilon} = \{\epsilon_j\}_1^5 = \epsilon\delta_j^i$.

B.1. **Our Five-Parameter Group.** Essentially, the game is to transfer the derivatives to the kernels, thereby obtaining a new kernel whose convolution with the image generates the desired tangent vector (image).

B.1.1. *Translation.* We will let θ_1 and θ_2 be the parameters that translate the image in the x and y directions respectively. Then we find that, after a change of variables,

$$\begin{aligned}
\Delta_{\theta_1,\boldsymbol{\epsilon}} I &= \frac{1}{|\theta_1|}\int I(u+\theta_1,v)K(x-u,y-v) - I(u,v)K(x-u,y-v)dudv \\
&= \frac{1}{|\theta_1|}\int I(u,v)\left(K(x-u+\theta_1,y-v) - K(x-u,y-v)\right)dudv
\end{aligned}$$

$$\begin{aligned}
\Delta_{\theta_2,\boldsymbol{\epsilon}} I &= \frac{1}{|\theta_2|}\int I(u,v+\theta_2)K(x-u,y-v) - I(u,v)K(x-u,y-v)dudv \\
&= \frac{1}{|\theta_2|}\int I(u,v)\left(K(x-u,y-v+\theta_2) - K(x-u,y-v)\right)dudv \ .
\end{aligned}$$

B.1.2. *Scaling.* We let $1+\theta_3$ and $1+\theta_4$ be the parameters that scale the x and y directions respectively.

$$\begin{aligned}\Delta_{\theta_3,\boldsymbol{\epsilon}} I &= \frac{1}{|\theta_3|}\int I(\frac{u}{1+\theta_3},v)K(x-u,y-v) - I(u,v)K(x-u,y-v)dudv \\ &= \frac{1}{|\theta_3|}\int I(u,v)\left((1+\theta_3)K(x-u-\theta_3 u,y-v) - K(x-u,y-v)\right)dudv\end{aligned}$$

$$\begin{aligned}\Delta_{\theta_4,\boldsymbol{\epsilon}} I &= \frac{1}{|\theta_4|}\int I(u,\frac{v}{1+\theta_4})K(x-u,y-v) - I(u,v)K(x-u,y-v)dudv \\ &= \frac{1}{|\theta_4|}\int I(u,v)\left((1+\theta_4)K(x-u,y-v-\theta_4 v) - K(x-u,y-v)\right)dudv\end{aligned}$$

B.1.3. *Rotation.* If we let θ_5 be the angle of rotation around the origin $u=0, v=0$, then we find that

$$\begin{aligned}\Delta_{\theta_5,\boldsymbol{\epsilon}} I &= \frac{1}{|\theta_5|}\int I(c_5 u + s_5 v, -s_5 u + c_5 v)K(x-u,y-v) - I(u,v)K(x-u,y-v)dudv \\ &= \frac{1}{|\theta_3|}\int I(u,v)\left(K(x-c_5 u + s_5 v, y - s_5 u - c_5 v) - K(x-u,y-v)\right)dudv\ .\end{aligned}$$

where c_5 and s_5 are $\cos(\theta_5)$ and $\sin(\theta_5)$ respectively.

B.2. **The kernel.** Combining the results above *to first order in the θ_is*, we get the combined kernel

$$(1+\theta_3)(1+\theta_4)K(x-u+\theta_1-\theta_3 u+\theta_5 v, y-v+\theta_2-\theta_4 v-\theta_5 u)\ ,$$

which is valid for the calculation of first derivatives (only).

B.3. **The derivatives.** Taking the limit as the θ_is go to 0, we get

$$\begin{aligned}K_{\theta_1} &= K_1 \\ K_{\theta_2} &= K_2 \\ K_{\theta_3} &= K - uK_1 \\ K_{\theta_4} &= K - vK_2 \\ K_{\theta_5} &= vK_1 - uK_2\ .\end{aligned}$$

B.4. **Second-order terms.** For the calculations of the second derivatives *independent of order*, we need the transformations to commute. Since they do not, we form a specific composition which we can then differentiate twice to get the second order term (the Hessian tensor). We consider the five-parameter transformation formed by first rotating, then scaling, then translating. We get

$$\begin{aligned}(1+\theta_3+\theta_4+\theta_3\theta_4)K(\quad & x-u+\theta_1-\theta_3 u+\theta_5 v+\theta_1\theta_3-\theta_2\theta_5+\theta_5\theta_4 v+\frac{\theta_5^2}{2}u, \\ & y-v+\theta_2-\theta_4 v-\theta_5 u+\theta_2\theta_4+\theta_1\theta_5-\theta_5\theta_3 u+\frac{\theta_5^2}{2}v)\ ,\end{aligned}$$

which we can write as

$$p_1(\boldsymbol{\theta})K(x-u+p_2(\boldsymbol{\theta},u,v), y-v+p_3(\boldsymbol{\theta},u,v))\ ,$$

which can now be differentiated twice to get the kernels that give the components of the second derivative tensor.

B.5. **The terms.** Taking the second derivatives and evaluating at $\boldsymbol{\theta} = \mathbf{0}$, we get

$$\begin{aligned}
K_{\theta_1,\theta_1} &= K_{1,1} \\
K_{\theta_1,\theta_2} &= K_{1,2} \\
K_{\theta_1,\theta_3} &= 2K_1 - uK_{1,1} \\
K_{\theta_1,\theta_4} &= K_1 - vK_{1,2} \\
K_{\theta_1,\theta_5} &= K_2 + vK_{1,1} - uK_{1,2} \\
K_{\theta_2,\theta_2} &= K_{2,2} \\
K_{\theta_2,\theta_3} &= K_2 - uK_{2,1} \\
K_{\theta_2,\theta_4} &= 2K_2 - vK_{2,2} \\
K_{\theta_2,\theta_5} &= -K_1 + vK_{2,1} - uK_{2,2} \\
K_{\theta_3,\theta_3} &= -2uK_1 + u^2K_{1,1} \\
K_{\theta_3,\theta_4} &= K - uK_1 - vK_2 + uvK_{1,2} \\
K_{\theta_3,\theta_5} &= vK_1 - uK_2 - uvK_{1,1} + u^2K_{1,2} \\
K_{\theta_4,\theta_4} &= -2vK_2 + v^2K_{2,2} \\
K_{\theta_4,\theta_5} &= vK_1 - uK_2 + (uv - v^2)K_{1,2} + uvK_{2,2} \\
K_{\theta_5,\theta_5} &= uK_1 + vK_2 + v^2K_{1,1} - 2uvK_{1,2} + u^2K_{2,2} \ .
\end{aligned}$$

B.6. **Calculation of $K_1, K_2, K_{1,1}, K_{1,2}$, and $K_{2,2}$.** If we use a Gaussian kernel

$$K(s,t) \equiv \frac{1}{2\pi\sigma^2} \exp(-(s^2+t^2)/2\sigma^2) \ ,$$

we get

$$\begin{aligned}
K_1 \quad = K_s \quad &= -\frac{1}{2\pi\sigma^4} s \exp(-(s^2+t^2)/2\sigma^2) \\
K_2 \quad = K_t \quad &= -\frac{1}{2\pi\sigma^4} t \exp(-(s^2+t^2)/2\sigma^2) \\
K_{1,1} \quad = K_{s,s} \quad &= -\frac{1}{2\pi\sigma^4} \exp(-(s^2+t^2)/2\sigma^2) + \frac{1}{2\pi\sigma^6} s^2 \exp(-(s^2+t^2)/2\sigma^2) \\
K_{1,2} \quad = K_{s,t} \quad &= \frac{1}{2\pi\sigma^6} st \exp(-(s^2+t^2)/2\sigma^2) \\
K_{2,2} \quad = K_{t,t} \quad &= -\frac{1}{2\pi\sigma^4} \exp(-(s^2+t^2)/2\sigma^2) + \frac{1}{2\pi\sigma^6} t^2 \exp(-(s^2+t^2)/2\sigma^2) \ .
\end{aligned}$$

B.7. **Example convolution.** Finally, we may put the results above together to get any derivative by convolution in the following way. As an example, we compute the entry of the second derivative tensor corresponding to the kernel K_{θ_3,θ_5}. **Remark:** *Note that even though the kernels are not shift invariant, we can still use FFTs to do the convolution by moving some terms in the kernel over to the image (effectively premultiplying the image by some function – in our case, polynomial terms).*

$$
\begin{aligned}
D_{\theta_3,\theta_4} I &= \int I(u,v)(K(x-u,y-v) - uK_1(x-u,y-v) \\
&\qquad -vK_2(x-u,y-v) + uvK_{1,2}(x-u,y-v))dudv \\
&= \int I(u,v)K(x-u,y-v)dudv \\
&+ \int (-uI(u,v))\, K_1(x-u,y-v)dudv \\
&+ \int (-vI(u,v))\, K_2(x-u,y-v)dudv \\
&+ \int (uvI(u,v))\, K_{1,2}(x-u,y-v)dudv \\
&= \int I(u,v)\frac{1}{2\pi\sigma^2}\exp(-((x-u)^2+(y-v)^2)/2\sigma^2)dudv \\
&+ \int (uI(u,v))\,\frac{1}{2\pi\sigma^4}(x-u)\exp(-((x-u)^2+(y-v)^2)/2\sigma^2)dudv \\
&+ \int (vI(u,v))\,\frac{1}{2\pi\sigma^4}(y-v)\exp(-((x-u)^2+(y-v)^2)/2\sigma^2)dudv \\
&+ \int (uvI(u,v))\,\frac{1}{2\pi\sigma^6}(x-u)(y-v)\exp(-((x-u)^2+(y-v)^2)/2\sigma^2)dudv
\end{aligned}
$$

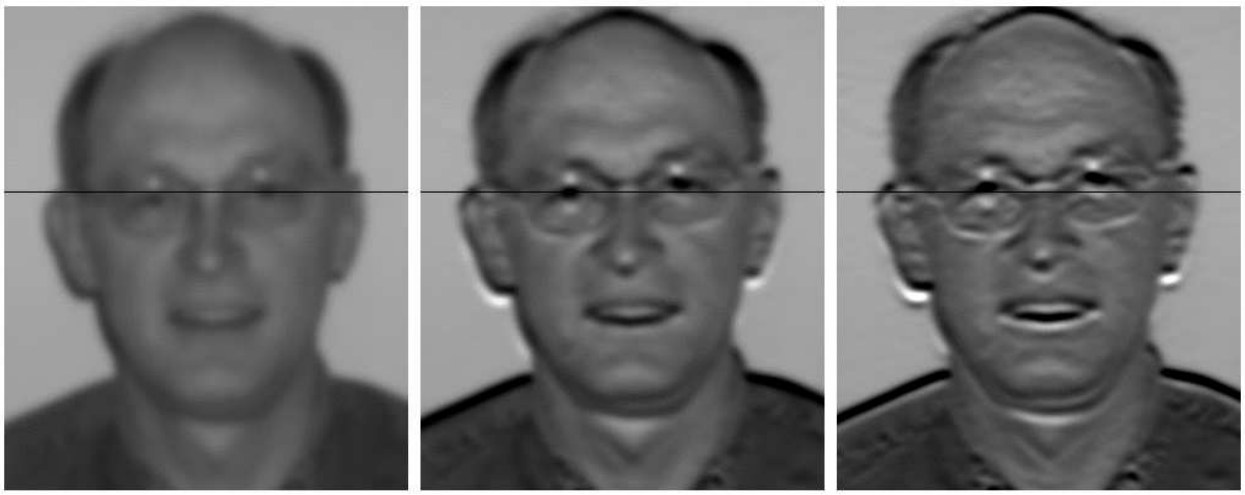

FIGURE 10. Depicted in this figure: left most we see a smoothed figure, center the figure shifted up with the first order approximation to the shift and on the right, the shift including the first and second order terms. The shifted figures are visibly, though slightly, higher, with the second order correction slightly higher than the first order shift. The figure is 300 by 300 pixels and the shift is 12 pixels.

B.8. **Example Application of first and second order terms.**

References

[1] Beveridge, J. R., She, K., Draper, B., and Givens, G. H. A nonparametric statistical comparison of principal component and linear discriminant subspaces for face recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2001). http://www.cs.colostate.edu/evalfacerec/index.html.

[2] Beveridge, R. Evaluation of face recognition algorithms web site. http://www.cs.colostate.edu/evalfacerec/, Oct. 2002.

[3] Duda, R. O., Hart, P. E., and Stork, D. G. *Pattern Classification*, second ed. Wiley, 2001.

[4] Georghiades, A. S., Belhumeur, P. N., and Kriegman, D. J. From few to many: Illumination cone models for face recognition under variable lighting and pose. *IEEE Transactions on Pattern Analysis and Machine Intelligence 23*, 6 (June 2001), 643–660.

[5] Moses, Y., Adini, Y., and Ullman, S. Face recognition: The problem of compensating for changes in illumination direction. In *Proc. European Conf. Computer Vision* (1994), pp. 286–296.

[6] Phillips, P. J., Moon, H., Rauss, P. J., and Rizvi, S. The feret evaluation methodology for face recognition algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence 22*, 10 (Oct. 2000). available as report NISTR 6264.

[7] Roweis, S., and Saul, L. Nonlinear dimensionality reduction by locally linear embedding. *Science 290* (2000), 2323–2326.

[8] Simard, P. Y., Cun, Y. A., Denker, J. S., and Victorri, B. Transformation invariance in pattern recognition: Tangent distance and propagation. *International Journal of Imaging Systems and Technology 11*, 3 (2000), 181–197.

[9] Simard, P. Y., Cun, Y. A. L., Denker, J. S., and Victorri, B. Transformation invariance in pattern recognition - tangent distance and tangent propagation. In *Neural Networks: Tricks of the Trade*, G. B. Orr and K.-R. Muller, Eds. Springer, 1998, ch. 12.

[10] Tenenbaum, J. B., Silva, V., and Langford, J. C. A global geometric framework for nonlinear dimensionality reduction. *Science 290* (2000), 2319–2323.

[11] Turk, M., and Pentland, A. Face recognition using eigenfaces. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition* (Maui, HI, USA, 1991).

[12] Zhao, W., Chellappa, R., and Krishnaswamy, A. Discriminant analysis of principal components for face recognition. In *Face Recognition: From Theory to Applications* (1998), Wechsler, Phillips, Bruce, Fogelman-Soulie, and Huang, Eds., pp. 73–85.

This work was supported by a LANL 2002 Homeland defense LDRD-ER (PI K. Vixie), a LANL 2003 LDRD-DR (PI J. Kamm), and Portland State University's sabbatical leave program. It is available as LAUR-03-3409.