Local Discriminant Embedding and Its Variants

Hwann-Tzong Chen Huang-Wei Chang Tyng-Luh Liu
Institute of Information Science, Academia Sinica
Nankang, Taipei 115, Taiwan
{*pras, hwchang, liutyng*}*@iis.sinica.edu.tw*

Abstract

We present a new approach, called local discriminant embedding (LDE), to manifold learning and pattern classification. In our framework, the neighbor and class relations of data are used to construct the embedding for classification problems. The proposed algorithm learns the embedding for the submanifold of each class by solving an optimization problem. After being embedded into a low-dimensional subspace, data points of the same class maintain their intrinsic neighbor relations, whereas neighboring points of different classes no longer stick to one another. Via embedding, new test data are thus more reliably classified by the nearest neighbor rule, owing to the locally discriminating nature. We also describe two useful variants: two-dimensional LDE and kernel LDE. Comprehensive comparisons and extensive experiments on face recognition are included to demonstrate the effectiveness of our method.

1. Introduction

Mapping data from the input space to a low-dimensional space is inevitable in solving many computer vision and pattern recognition problems. For this purpose, dimensionality reduction techniques that work with subspace properties are perhaps the most popular choices. Nevertheless, it is often the case that Euclidean distance is incapable of capturing the intrinsic similarities between data points, and as a result, traditional *subspace methods* can no longer approximate a credible data distribution in the low-dimensional space.

On the other hand, take, for example, the face images varying in rotation, pose, or expression as the data. Such images reside on a manifold of their original space, as discussed in recent work, e.g., [10], [17], [21]. Indeed when data are densely distributed on a manifold, it is possible to apply *manifold learning* to reveal their intrinsic distribution in a lower-dimensional space. The seminal works of Isomap [23] and LLE [17] address data representation, and mainly consider how to preserve the local or global property of training data. Because it remains a difficult issue to map new test data to the low-dimensional space, their algorithms cannot be easily extended for classification problems. Some algorithms resolve this kind of difficulty by finding a mapping for the whole data space, not just for the training data, e.g., [1], [4], [9], [12], [18]. However, these methods are designed in a way to best preserve data localities or similarities in the embedding space, and consequently cannot promise good discriminating capability. Only a few manifold learning algorithms explicitly address classification problems, including [26], [28]. In [26], the authors propose to group the training data into clusters, and combine local discriminative features into a global Fisher criterion. Yang [28] considers the geodesic distances between data points as the features, and uses such features in linear discriminant analysis (LDA) for classification.

In view of the foregoing discussions, we are motivated to propose a new embedding framework for pattern classification on high-dimensional data. Particularly, our aim is to develop an algorithm that its discriminating efficiency does not strongly depend on the data distribution, and is directly applicable for dealing with new test data.

1.1. Related Work

Principle component analysis (PCA), or equivalently the Karhunen-Loeve transform, is widely used for linear dimensionality reduction. For example, Kirby and Sirovich [11] use PCA to model human faces, and Turk and Pentland introduce the Eigenface method for face recognition [24]. Techniques originated from manifold learning such as Isomap [23], LLE [17], and Laplacian Eigenmap [3] consider nonlinear dimensionality reduction by investigating the local geometry of data. Such embeddings are good for representation, but only concern with the training data. To facilitate nearest-neighbor searches for new test data, locality preserving projection (LPP) [8] and BoostMap [1] attempt to reconstruct data localities or similarities in the low-dimensional Euclidean space.

The above-mentioned linear or nonlinear dimensionality reduction approaches are generally not devised for classification use. On the contrary, in the context of pattern classification, LDA seeks the best projection subspace for separating data, and is shown to be a useful tool for feature extraction and classification, e.g., the Fisherface method [2].

Often, without getting rid of those nice linear properties behind techniques like PCA and LDA, their discriminating power can be enhanced by incorporating nonlinearities via *kernel methods* [20], [14]. Yang [29] applies Kernel PCA and Kernel LDA to face recognition, and shows that such modifications improve the classical Eigenface and Fisherface methods. Yet another possibility to boost the classification efficiency has to do with the data representation. Most subspace methods deal with vectors, and require that data such as images must be first transformed into vectors. Yang et al. [27] and Ye et al. [30] propose to view an image as a matrix, and present subspace algorithms that work directly on 2-D image data. These methods can find principal features in the rows or columns of an image, and result in a much smaller dimensionality of the corresponding eigenvalue problem than that in the conventional formulation.

1.2. Our Approach

Focusing on manifold learning and pattern classification, our embedding method achieves good discriminating performance by integrating the information of *neighbor* and *class* relations between data points. The crux of our approach is to learn the embedding by taking account of the respective submanifold of each class. While maintaining the original neighbor relations for neighboring data points of the same class is important, it is also crucial to differentiate and to *keep away* neighboring points of different classes after the embedding. With that, the class of a new test point can be more reliably predicted by the nearest neighbor criterion, owing to the locally discriminating nature.

In what follows, we begin by presenting our embedding algorithm in its generic form and then its two generalizations, including the two-dimensional and the kernel formulation. Finally, we verify the effectiveness of our method through a variety of experiments on face recognition and extensive comparisons with other approaches.

2. Local Discriminant Embedding

Let $\mathcal{M}$ be a manifold embedded in $\mathbb{R}^n$. Now suppose we have m data points $\{\mathbf{x}_i|\mathbf{x}_i \in \mathbb{R}^n\}_{i=1}^m \subset \mathcal{M}$ and the corresponding class labels $\{y_i|y_i \in \{1, 2, \ldots, P\}\}_{i=1}^m$. Also, any subset of data points that belong to the same class is assumed to lie in a submanifold of $\mathcal{M}$. In practice we are interested in the case that the dimensionality n is high; for example, if each data point represents an image, then n is equal to the number of image pixels.

For multi-class classification, a simple but effective way to determine the class label of a new test point is to use the nearest neighbor criterion [6]. Since the data points are on the manifold, one would expect that if the geometrical structure of the underlying manifold is unraveled, then the computation of finding the nearest neighbor can be done efficiently in a lower-dimensional space. Alas, manifold learning algorithms like Isomap [23] and LLE [17] are defined simply on the training data points, and therefore not suitable for classification problems. Some recent studies have tried to address this issue by explicitly learning the mapping functions [1] or projections [8] for nearest-neighbor search in low-dimensional Euclidean space. Although the learned embedding in [1] or [8] is applicable to new test points, the performance of classification relies heavily on how well the nearest neighbor criterion works in the original high-dimensional space. In fact, since their main goal is to preserve localities or similarity rankings after embedding, these algorithms are more appropriate for retrieval or clustering rather than for classification.

Our formulation directly incorporates the class information into the construction of embedding. In effect, the proposed *local discriminant embedding* (LDE) framework seeks to dissociate the submanifold of each class from one another, and specifically derives the embedding for nearest neighbor classification in a low-dimensional Euclidean space. The structure of LDE can be characterized by three key ideas: 1) similarities are locally measured based on manifold assumptions; 2) the embedding is in the form of linear projection, which is obtained by finding generalized eigenvectors; 3) the algorithm itself solves an optimization problem, which is defined to discriminate submanifolds.

2.1. LDE Algorithm

For convenience of presentation, we first describe the steps of the LDE algorithm, and then justify them in detail. Recall that the data points $\{\mathbf{x}_i\}_{i=1}^m$ are in $\mathbb{R}^n$, and each $\mathbf{x}_i$ is labeled by some class label y_i. We also write the data matrix as $X = [\mathbf{x}_1\, \mathbf{x}_2\, \cdots\, \mathbf{x}_m] \in \mathbb{R}^{n\times m}$. Then, the proposed LDE can be realized by the following three steps.

1. *Construct neighborhood graphs.* Let G and G' denote two (undirected) graphs both over all data points. To construct G, we consider each pair of points $\mathbf{x}_i$ and $\mathbf{x}_j$ from *the same* class, i.e., $y_i = y_j$. An edge is added between $\mathbf{x}_i$ and $\mathbf{x}_j$ if $\mathbf{x}_j$ is one of $\mathbf{x}_i$'s k-nearest neighbors. (The other possibility is to consider the ϵ-ball implementation.) For G', we instead consider each pair of $\mathbf{x}_i$ and $\mathbf{x}_j$ with $y_i \neq y_j$, and likewise, connect $\mathbf{x}_i$ and $\mathbf{x}_j$ if $\mathbf{x}_j$ is one of $\mathbf{x}_i$'s k'-nearest neighbors.

2. *Compute affinity weights.* Specify the affinity matrix W of G, where each element w_{ij} refers to the weight of the edge between $\mathbf{x}_i$ and $\mathbf{x}_j$, and is given by

$$w_{ij} = \exp[-\|\mathbf{x}_i - \mathbf{x}_j\|^2/t]. \tag{1}$$

   By default, $w_{ij} = 0$ if $\mathbf{x}_i$ and $\mathbf{x}_j$ are not connected. It is clear that W so defined is an $m \times m$, sparse, and symmetric matrix. The other affinity matrix W' of G'

can be computed in the same way. Of note, the affinity weights defined in (1) are derived from the heat kernel [16]. Another possible choice called "simple-minded" is also suggested in [3]; namely, $w_{ij} = 1$ if $\mathbf{x}_i$ and $\mathbf{x}_j$ are connected, and $w_{ij} = 0$ otherwise.

3. *Complete the embedding.* Find the generalized eigenvectors $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_\ell$ that correspond to the ℓ largest eigenvalues in

$$X(D' - W')X^T\mathbf{v} = \lambda X(D - W)X^T\mathbf{v}\,, \quad (2)$$

where D and D' are diagonal matrices with diagonal elements $d_{ii} = \sum_j w_{ij}$ and $d'_{ii} = \sum_j w'_{ij}$. The embedding of $\mathbf{x}_i$ is accomplished by $\mathbf{z}_i = V^T\mathbf{x}_i$, where $V = [\mathbf{v}_1\, \mathbf{v}_2\, \cdots\, \mathbf{v}_\ell]$.

We now give justifications for the above steps of LDE. Specifically, the aim of the algorithm is to construct the embedding $\mathbf{z} = V^T\mathbf{x}$ based on linear projections, where V is an $n \times \ell$ matrix with $\ell \ll n$. Unlike PCA and LDA, our method explores the local relations between neighboring data points. Since the class information is given, it is reasonable to require that, after embedding, the neighbor relations can better reflect the class relations. That is, in the low-dimensional embedding subspace, we want to keep neighboring points close if they have the same label, whereas prevent points of other classes from entering the neighborhood. With these two aspects of consideration, we arrive at the following constrained optimization problem:

$$\begin{aligned} &\text{Maximize } J(V) = \sum_{i,j} \|V^T\mathbf{x}_i - V^T\mathbf{x}_j\|^2 w'_{ij} \\ &\text{subject to } \sum_{i,j} \|V^T\mathbf{x}_i - V^T\mathbf{x}_j\|^2 w_{ij} = 1\,. \end{aligned} \quad (3)$$

The optimization formulation essentially uses the class and neighbor information through the two affinity matrices W (for $y_i = y_j$ in neighborhoods) and W' (for $y_i \neq y_j$ in neighborhoods), computed in Step 2 of LDE.

To gain more insight into (3), we write the square of norm in the form of trace

$$\begin{aligned} J &= \sum_{i,j} \|V^T\mathbf{x}_i - V^T\mathbf{x}_j\|^2 w'_{ij} \\ &= \sum_{i,j} \mathrm{tr}\big\{(V^T\mathbf{x}_i - V^T\mathbf{x}_j)(V^T\mathbf{x}_i - V^T\mathbf{x}_j)^T\big\} w'_{ij} \\ &= \sum_{i,j} \mathrm{tr}\big\{V^T(\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^T V\big\} w'_{ij}\,. \end{aligned} \quad (4)$$

Since the operation of trace is linear and w'_{ij} is a scalar, we can move the summation and w'_{ij} inside the trace:

$$\begin{aligned} J &= \mathrm{tr}\big\{V^T \sum_{i,j} \big((\mathbf{x}_i - \mathbf{x}_j) w'_{ij} (\mathbf{x}_i - \mathbf{x}_j)^T\big) V\big\} \\ &= \mathrm{tr}\big\{V^T(2XD'X^T - 2XW'X^T)V\big\} \\ &= 2\,\mathrm{tr}\big\{V^T X(D' - W')X^T V\big\}\,, \end{aligned} \quad (5)$$

where X is the data matrix and D' is a diagonal matrix with $d'_{ii} = \sum_j w'_{ij}$. The objective function and the constraint in (3) can be reformulated as

$$\begin{aligned} &\text{Maximize } J(V) = 2\,\mathrm{tr}\big\{V^T X(D' - W')X^T V\big\} \\ &\text{subject to } 2\,\mathrm{tr}\big\{V^T X(D - W)X^T V\big\} = 1\,. \end{aligned} \quad (6)$$

Thus, we can conclude (also see [7], p.447) that the columns of an optimal V are the generalized eigenvectors corresponding to the ℓ largest eigenvalues in

$$X(D' - W')X^T\mathbf{v} = \lambda X(D - W)X^T\mathbf{v}\,. \quad (7)$$

Once we have learned the projection matrix V using the LDE algorithm, nearest neighbor classifications become straightforward. For any test point $\bar{\mathbf{x}} \in \mathbb{R}^n$, we compute $\bar{\mathbf{z}} = V^T\bar{\mathbf{x}}$. Its label is then predicted as y_{i^*} provided that $\mathbf{z}_{i^*} = V^T\mathbf{x}_{i^*}$ minimizes $\|\mathbf{z}_i - \bar{\mathbf{z}}\|$.

3. Generalizations for LDE

The previous section introduces LDE in its basic form. In general, it is impractical to argue the superiority of a learning algorithm over others, since the amount and the prior distribution of training data, and the type of problem all have direct bearings on the classification performance. For this reason, we introduce two useful generalizations for local discriminant embedding, the *two-dimensional* LDE (2DLDE) and the *kernel* LDE. As we shall see later, the two variants of LDE have their own advantages for different circumstances. Hence, the overall LDE framework is thereby capable of resolving a wide range of problems.

3.1. Two-Dimensional LDE

Sometimes we are limited to work with a small number of training data, and have to face the problem that their underlying manifold cannot be accurately approximated. We address this issue by employing the two-dimensional based approaches [27], [30], under the assumption that the data are images. Then, instead of raster-scanning an image to produce a vector, a 2-D based scheme treats an image as a *matrix*, and solves a subspace problem according to *matrix norms*. It follows that the columns and rows of image data implicitly emerge as points on a manifold. A handy example is that such manifold properties of row and column

hold for facial images, owing to the abundance of regular horizontal and vertical patterns of human faces. Especially when the variations in images are known to be caused by translation, pitch, or yaw, 2DLDE has significant advantages over the vector-based formulation.

Let $\{A_i | A_i \in \mathbb{R}^{n_1 \times n_2}\}_{i=1}^m$ be the training data, where each A_i is now a matrix representation of an image of size n_1 by n_2. To generalize the formulation of standard LDE, processes for vectors need to be modified for matrices. First, the matrix-vector multiplication $\mathbf{z}_i = V^T \mathbf{x}_i$ should be changed to the two-sided form as $B_i = L^T A_i R$, where $L \in \mathbb{R}^{n_1 \times \ell_1}$ and $R \in \mathbb{R}^{n_2 \times \ell_2}$ transform A_i into a smaller matrix $B_i \in \mathbb{R}^{\ell_1 \times \ell_2}$. Second, the vector 2-norm used in the analysis is replaced by the Frobenius matrix norm, given by $\|A\|_F = (\sum_{j,k} a_{jk}^2)^{1/2}$. Keeping in mind these adjustments, we rewrite (3) in two-sided form:

$$\begin{aligned} \text{Maximize } Q(L,R) &= \sum_{i,j} \|L^T A_i R - L^T A_j R\|_F^2 w'_{ij} \\ \text{subject to } & \sum_{i,j} \|L^T A_i R - L^T A_j R\|_F^2 w_{ij} = 1\,. \end{aligned} \tag{8}$$

We solve the two-sided optimization problem numerically, in a flip-flop manner—one of L and R is solved while the other is fixed [30]. The iterative procedure is repeated until convergence. For the sake of brevity, we shall not go into the details of calculations, but simply remark that they involve the property $\|A\|_F^2 = \mathrm{tr}(AA^T)$ and the derivatives of traces. To this end, we have the following iterations for the maximization in (8):

1. Given L, solve for R by finding the generalized eigenvectors $\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_{\ell_2}$ corresponding to the ℓ_2 largest eigenvalues in

$$\begin{aligned} &\Big(\sum_{i,j} w'_{ij}(A_i - A_j)^T L L^T (A_i - A_j)\Big)\mathbf{r} \\ &= \lambda_R \Big(\sum_{i,j} w_{ij}(A_i - A_j)^T L L^T (A_i - A_j)\Big)\mathbf{r}\,, \end{aligned} \tag{9}$$

   and $R = [\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_{\ell_2}]$.

2. As R is given, L can be obtained by finding the generalized eigenvectors $\mathbf{l}_1, \mathbf{l}_2, \ldots, \mathbf{l}_{\ell_1}$ corresponding to the ℓ_1 largest eigenvalues in

$$\begin{aligned} &\Big(\sum_{i,j} w'_{ij}(A_i - A_j) R R^T (A_i - A_j)^T\Big)\mathbf{l} \\ &= \lambda_L \Big(\sum_{i,j} w_{ij}(A_i - A_j) R R^T (A_i - A_j)^T\Big)\mathbf{l}\,, \end{aligned} \tag{10}$$

   and $L = [\mathbf{l}_1, \mathbf{l}_2, \ldots, \mathbf{l}_{\ell_1}]$.

After L and R are derived, nearest neighbor classifications with 2DLDE can be carried out as follows. For a test point $\bar{A}$ (in matrix form), we first compute $\bar{B} = L^T \bar{A} R$. Then, we find its nearest neighbor B_{i^*}, which minimizes $\|B_i - \bar{B}\|_F$, and assign the label as $\bar{y} = y_{i^*}$.

3.2. Kernel LDE

Historically, the classification power of linear learning algorithms is considered limited, and is often insufficient to deal with complicated problems [15], [25]. One possible attempt to elevate the classification performance is to transform input data to a higher-dimensional space via a non-linear mapping. Among the many efforts, the *kernel trick* [19], best known for its use in *support vector machines* [25], provides an effective way for that purpose. Thus, for our second generalization to LDE, we investigate its kernel representation, and establish a new algorithm incorporating nonlinearity, namely the kernel LDE.

Suppose that we map the input data $\{\mathbf{x}_i\}_{i=1}^m$ to some high-dimensional feature space $\mathcal{F}$ via a nonlinear mapping $\phi : \mathbb{R}^n \to \mathcal{F}$. We thus focus on $\{\phi(\mathbf{x}_i) \,|\, \phi(\mathbf{x}_i) \in \mathcal{F}\}_{i=1}^m$. Also, a dot product in $\mathcal{F}$ can be computed by the kernel function $\mathbf{k}(\mathbf{x}_1, \mathbf{x}_2) = \phi(\mathbf{x}_1)^T \phi(\mathbf{x}_2)$. To simplify the discussion, now assume that we are to find only one projection direction $\mathbf{v}$ in $\mathcal{F}$ for local discriminant embedding, where $\mathbf{v}$ can be expressed in terms of a combination of mapped data, i.e., $\mathbf{v} = \sum_i^m \alpha_i \phi(\mathbf{x}_i)$, and is therefore determined by the expansion coefficients α_i.

With some effort, the optimization problem in (3) can be *kernelized* as

$$\begin{aligned} \text{Maximize } U(\boldsymbol{\alpha}) &= \boldsymbol{\alpha}^T K(D' - W')K\boldsymbol{\alpha} \\ \text{subject to } & \boldsymbol{\alpha}^T K(D - W)K\boldsymbol{\alpha} = 1, \end{aligned} \tag{11}$$

where K is a kernel matrix with $K_{ij} = \mathbf{k}(\mathbf{x}_i, \mathbf{x}_j)$, and $\boldsymbol{\alpha} = [\alpha_1 \alpha_2 \ldots \alpha_m]^T$ consists of the expansion coefficients. It follows that the corresponding generalized eigenvalue problem to find $\boldsymbol{\alpha}$ of (11) is

$$K(D' - W')K\boldsymbol{\alpha} = \lambda K(D - W)K\boldsymbol{\alpha}\,, \tag{12}$$

where the generalized eigenvector associated with the largest eigenvalue is of our interest.

To test a new point $\bar{\mathbf{x}}$ by nearest neighbor classification with kernel LDE, we need to compute the dot product via kernel, that is, $\bar{z} = \mathbf{v}^T \phi(\bar{\mathbf{x}}) = \sum_i^m \alpha_i \mathbf{k}(\mathbf{x}_i, \bar{\mathbf{x}})$, and find its nearest neighbor in the one-dimensional embedding space. Note that although the construction of embedding presented above is based on 1-D projection, it is equally valid for kernel embedding in a multi-dimensional subspace (where $\boldsymbol{\alpha}$ forms a matrix), and indeed this is the case that we use for the face recognition experiments.

4. Face Recognition and LDE

Face images that vary in lighting, orientation, pose, or facial expression are good real-world examples to illustrate the concept of manifold learning. In this section we discuss the results of applying local discriminant embedding to face recognition problems. Various experiments are conducted to demonstrate the performances of the LDE algorithms on different types of variations, including pose, orientation, translation, lighting, and expression. A face recognition task is handled as a multi-class classification problem—we map each test image to a low-dimensional subspace via the embedding learned from training data, and then classify the test data by the nearest neighbor criterion.

4.1. The Datasets

Three databases are used in our experiments: the AR face database [13], the CMU PIE database [22], and the AT&T face database [5]. AR and PIE are rather large databases. We use them to build several face datasets for different kinds of experiments, and adopt five-fold cross validation for evaluating the performances of assorted algorithms. The AT&T database is smaller; we test on the whole database by the leave-one-out strategy, as well as five-fold cross validation. These experiments are referred to as AR_14, PIE_27, PIE_5, PIE_R, PIE_T, AT&T_LOO, and AT&T_CV. We give below a brief description for each of them.

AR_14: The AR database contains more than 4,000 images featuring frontal faces with different expressions, lighting conditions, and occlusions. We remove those images of occluded faces and manually crop the remaining images to generate a dataset of 118 people. The dataset consists of 1,652 ($= 118 \times 14$) images, where each subject has 14 facial images taken in two sessions separated by two weeks, as illustrated in Fig. 1. Furthermore, two formats of data representations are used according to the formulations of algorithms. By 2D formulation, an image is stored as a matrix of size 115 by 115. For vector-based formulation (including kernel LDE), we resize each image to 28 by 28 pixels and transform it into a vector of 784 pixels through raster scan. The dataset is randomly divided into five subsets of approximately equal size for five-fold cross validation.

Figure 1: AR_14. The dataset is generated from the AR face database. AR_14 includes 1,652 cropped images of 118 people. Each person has 14 facial images taken in two sessions separated by two weeks.

Figure 2: PIE_27. This dataset is produced from the PIE database. PIE_27 includes 1,836 images of 68 people. Each person has 27 facial images (9 poses $\times$ 3 expressions).

PIE_27: The PIE database includes over 40,000 facial images of 68 people. We generate a dataset from it by selecting 27 images for each person. The selected 27 images feature nine poses with three expression variations, as shown in Fig. 2. Images are cropped and resized to 100 by 100 pixels for 2D formulation. For vector-based formulation, we further downsample the images and represent them as vectors of 625 ($= 25 \times 25$) pixels. The 1,836 images in the dataset are randomly divided into five subsets for cross validation.

PIE_5: The PIE_5 experiment is to test on pose variations. From the dataset of PIE_27 we select five near-frontal facial images of each person putting on a neutral expression, as illustrated by Fig. 3a. The dataset comprises 340 face images of 68 people. We divide the dataset into five subsets *according to the pose*. Therefore, during the process of five-fold cross validation, a test set comprises the images of all subjects in the same pose, and we want to recognize them based on the training images with the other four poses. This experiment is more difficult: first, we have only a small number of training samples; second, test data are not very close to the training data; third, the algorithms are required to handle interpolation and extrapolation.

PIE_R and PIE_T: In the PIE database we pick 37 people who do not wear glasses, and use their frontal face images (with a neutral expression) to produce two datasets for experiments on rotation (PIE_R) and translation (PIE_T). For PIE_R, we rotate a frontal face image in 10° steps between $\pm 40^\circ$ to generate nine rotated images (including the original one with 0° rotation). For PIE_T, we shift a frontal face image by the combination of $-5, 0, 5$ pixels in horizontal and vertical coordinates, and consequently we have nine images: the original image plus the eight images with

Figure 3: (a) PIE_5. This dataset contains 340 near-frontal facial images of 68 people. (b) PIE_R. From left to right, the rotation angle increases by 10° from 0° to 40°. The other four images of the same subject, not shown here, feature the rotations in the clockwise direction. (c) PIE_T. The first image is shifted $\downarrow$, $\uparrow$, $\leftarrow$, $\rightarrow$, respectively, by 5 pixels to generate the following four images. The other four images not shown here are generated by translations with $(\pm 5, \pm 5)$ pixels in the four diagonal directions.

Figure 4: AT&T database. We use the whole database for the leave-one-out evaluation and the five-fold cross validation. (a) The ten facial images of a subject. (b) An example of look-alikes. The test face on top is prone to be misclassified as the one below.

translations to the eight directions. Fig. 3b and Fig. 3c illustrate two examples of PIE_R and PIE_T. Each dataset is randomly divided into five subsets for cross validation.

AT&T_LOO and AT&T_CV: The AT&T face database contains 400 images of 40 subjects, as shown in Fig. 4a. Since the database is small and the images are already cropped, we use the whole database for the leave-one-out test (referred to as AT&T_LOO), in accordance with the setups of previous works [27], [29]. Note that, for vector-based formulation, images are down-sampled to vectors of length 644 ($= 28 \times 23$) pixels, as in [29]; for 2D formulation, images are trained and tested with their original size as 112-by-92 matrices. In addition, we also carry out five-fold cross validation (referred to as AT&T_CV) on this database.

4.2. Experimental Results

Before we go through the elaborate experimental results, it is insightful to first look at a concrete example of the proposed embedding. In Fig. 5, the data of PIE_R are embedded in two-dimensional Euclidean space by the LDE algorithm. It can be seen that the rotated face images of the same person mostly lie on a one-dimensional submanifold (along the z_2 direction). Nevertheless, the overlaps of different classes indicate that the two-dimensional embedding space is still not sufficient to differentiate the whole dataset. And as we will see later, it indeed takes a four-dimensional LDE for producing faultless recognition testing on PIE_R.

Five-fold cross validation. Table 1 summarizes the results of the aforementioned experiments with five-fold cross validation. Each reported error rate is an average over the five runs of cross validation, and is empirically optimized by choosing within a wide range the best parameters (e.g. ℓ and k) for each method, based on its performance in five-fold cross validation. We compare LDE with the nearest neighbor classification in input space (NN), Eigenface [24], Fisherface [2], locality preserving projections (LPP) [9], and GPCA [30]. (RBF kernels are used in the experiments of kernel LDE.) For numerical stability concern, we adopt the techniques suggested by [2] and [9] to avoid singularities encountered in those methods that require solving generalized eigenvalue problems. More specifically, we use PCA to pre-process the singular matrix and keep 98% information in the sense of low rank approximation. Overall, LDE achieves superior performances in all aspects of testing. We now highlight some observations about these experiments:

- Among the experiments, the AT&T_CV is the easiest one. (The performance of NN largely reflects the difficulty of a problem.) Each image in the AT&T database includes the entire face and the head contour, which provide useful information for classification.
- For more difficult problems, the PCA-based methods, including Eigenface and GPCA, have high error rates, and their performances are similar to those of NN. The phenomenon suggests that their performances mainly depend on how the data distribute on the manifolds. If the neighborhoods contain mostly the data points of the same class, such as the case in AT&T_CV, PCA-based methods can perform satisfactorily; otherwise one should not expect good classification accuracy with PCA-based dimensionality reduction.
- The locality preserving projection algorithm performs better than the PCA-based methods, but still not well enough as the discriminant-based ones. Although LPP seeks to preserve neighbor relations, it does not exploit the class information for classification. The at-

Table 1: Evaluations by five-fold cross validation.

Method	Error Rate % (and Reduced Space)											
	AR_14		PIE_27		PIE_5		PIE_R		PIE_T		AT&T_CV	
NN	42.49	(784)	9.37	(625)	79.71	(625)	37.54	(625)	76.88	(625)	2.50	(644)
Eigenface	42.43	(210)	9.10	(170)	80.00	(150)	29.73	(50)	73.57	(20)	2.25	(60)
Fisherface	7.93	(57)	6.75	(33)	47.06	(25)	**0.00**	(4)	15.92	(8)	2.25	(19)
LPP	12.59	(27)	5.90	(33)	52.35	(13)	*0.90*	(8)	32.13	(8)	3.75	(20)
GPCA	42.86	(12, 12)	8.93	(9, 9)	78.82	(15, 15)	25.23	(6, 6)	63.36	(3, 3)	2.25	(25, 25)
LDE	*7.38*	(43)	*3.81*	(31)	*45.29*	(13)	**0.00**	(4)	*10.51*	(10)	1.50	(21)
2DLDE	15.98	(10, 10)	4.79	(9, 9)	**37.35**	(10, 10)	2.40	(10, 10)	**1.50**	(8, 8)	*1.25*	(7, 7)
Kernel LDE	**7.02**	(31)	**3.38**	(36)	45.59	(36)	**0.00**	(9)	11.71	(10)	**0.75**	(21)

The numbers in parentheses denote the dimensionality ℓ of reduced space. For 2D formulation, they are the values of ℓ_1 and ℓ_2.

tempt to keep neighbors of the same class closer via projections might also attract unwanted neighbors of different classes.

- Fisherface generally has low classification error rates. However, it models only the global structure of data, since the data distribution is assumed to be a Gaussian mixture. Lack of neighborhood properties makes the Fisherface method less appealing, especially when the nearest neighbor criterion is used for classification.
- When a face image is viewed as a vector, some useful spatial features might also be lost. For instance, in Fig. 3a and Fig. 3c, data points of the same class will look *dissimilar* if they are represented in the form of raster-scan vector. It is difficult for a test vector to find a neighbor of the correct class among the training vectors in PIE_5 and PIE_T. We note that 2DLDE is more capable of resolving these problems. In the 2D formulation, an image is viewed as a matrix, and its column and row vectors are analyzed as data points on manifolds. As can be observed in Fig. 3a and Fig. 3c, despite pose variations and translations, it is still easy to find similar pixel columns or rows in the images of the same class. The regularities in the columns and rows of translated faces are particularly beneficial to nearest neighbor classification, and give rise to the impressive result of 2DLDE in PIE_T. On the other hand, 2DLDE is not so suitable for rotated faces, in which the horizontal and vertical patterns are more irregular.

Leave-one-out. The results of leave-one-out evaluations on the AT&T face database are shown in Table 2. All error rates are empirically optimized through parameter selection. Every method gets just few misclassifications among the 400 tests. In fact, by applying nearest-neighbor searches in the input space, we already have a very low error rate (8/400). Even though the performance is seemingly hard to improve, the three proposed algorithms achieve lower error rates than previous methods. In particular, kernel LDE has only one error in the 400 tests. Fig. 4b depicts the sole misclassified face (top) and its nearest neighbor (bottom).

Table 2: Results of leave-one-out tests on AT&T database.

Method	Reduced Space	Error Rate (%)
NN (vector-based)	644	2.00
NN (2D)	112, 92	2.50
Eigenface [29]	40	2.50
Fisherface [29]	39	1.50
Kernel Eigenface [29]	40	2.00
Kernel Fisherface [29]	39	1.25
LPP, $k = 5$	16	2.25
GPCA	24, 24	1.75
LDE, $k = 7, k' = 4$	27	*1.00*
2DLDE, $k = 2, k' = 5$	7, 7	*1.00*
Kernel LDE, $k = 4, k' = 3$	27	**0.25**

5. Conclusion

We have presented a new manifold embedding framework for pattern classification. Based on the class information, our approach achieves good accuracy by realigning the submanifolds and rectifying the neighbor relations in the embedding space. The promising experimental results on face recognition further suggest that local discriminant embedding and its generalizations are convincingly applicable for multi-class classification problems whose data have underlying manifold structures. For future work, we are now exploring the connection between our method and large margin classifiers. Applications to multi-class and multi-view object detection are also of our main interests.

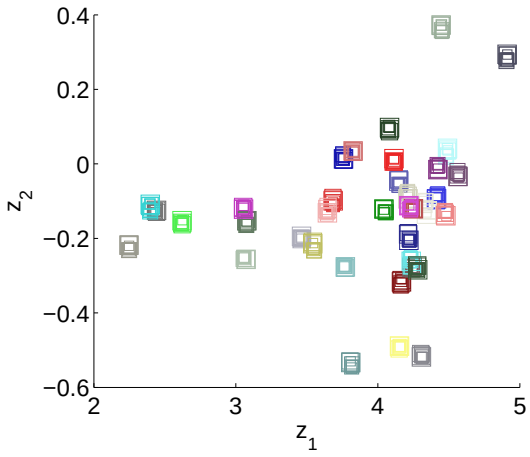

Figure 5: The local discriminant embedding of PIE_R face images (with variations in rotation) in two-dimensional Euclidean space.

Acknowledgments. This work was supported in part by an NSC grant 93-2213-E-001-010.

References

[1] V. Athitsos, J. Alon, S. Sclaroff, and G. Kollios, "BoostMap: a Method for Efficient Approximate Similarity Rankings," *CVPR*, 2004.

[2] P.N. Belhumeur, J.P. Hespanha, and D.J. Kriegman, "Eigenfaces vs. Fisherfaces: Recognition Using Class-Specific Linear Projection," *PAMI*, vol. 19, no. 7, pp. 711–720, July 1997.

[3] M. Belkin and P. Niyogi, "Laplacian Eigenmaps and Spectral Techniques for Embedding and Clustering," *NIPS 14*, 2001.

[4] C.M. Bishop, M. Svensén, and C.K.I. Williams, "The Generative Topographic Mapping," *Neural Computation*, vol. 10, no. 1, pp. 215–234, 1998.

[5] AT&T Laboratories Cambridge, "The AT&T Database of Faces, http://www.uk.research.att.com/facedatabase.html," 2002.

[6] T.M. Cover, "Estimation by the Nearest Neighbor Rule," *IEEE Trans. Information Theory*, vol. IT-14, no. 1, pp. 50–55, Jan. 1968.

[7] K. Fukunaga, *Introduction to Statistical Pattern Recognition*, Academic Press, 1990.

[8] X. He and P. Niyogi, "Locality Preserving Projections," *NIPS 16*, 2003.

[9] X. He, S. Yan, Y. Hu, and H.-J. Zhang, "Learning a Locality Preserving Subspace for Visual Recognition," *ICCV*, 2003.

[10] C. Hu, Chang Y., R. Feris, and M. Turk, "Manifold Based Analysis of Facial Expression," *IEEE Workshop on Face Processing in Video*, 2004.

[11] M. Kirby and L. Sirovich, "Application of the Karhunen-Loeve Procedure for the Characterization of Human Faces," *PAMI*, vol. 12, no. 1, pp. 103–108, Jan. 1990.

[12] D. Lowe and M.E. Tipping, "Neuroscale: Novel Topographic Feature Extraction with Radial Basis Function Networks," *NIPS 9*, pp. 543–549, 1996.

[13] A.M. Martinez and R. Benavente, "The AR Face Database," *CVC Technical Report #24*, Computer Vision Center at the U.A.B, June 1998.

[14] S. Mika, G. Rätsch, J. Weston, B. Schölkopf, and K.-R. Müller, "Fisher Discriminant Analysis with Kernels," *Neural Networks for Signal Processing IX*, pp. 41–48, 1999.

[15] M.L. Minsky and S.A. Papert, *Perceptrons*, MIT Press, 1969.

[16] S. Rosenberg, *The Laplacian on a Riemannian Manifold*, Cambridge University Press, 1997.

[17] S.T. Roweis and L.K. Saul, "Nonlinear Dimensionality Reduction by Locally Linear Embedding," *Science*, vol. 290, pp. 2323–2326, 2000.

[18] S.T. Roweis, L.K. Saul, and G. Hinton, "Global Coordination of Local Linear Models," *NIPS 14*, 2001.

[19] B. Schölkopf and A.J. Smola, *Learning with Kernels*, MIT Press, 2002.

[20] B. Schölkopf, A.J. Smola, and K.-R. Müller, "Nonlinear Component Analysis as a Kernel Eigenvalue Problem," *Neural Computation*, vol. 10, no. 5, pp. 1299–1319, 1998.

[21] H. Seung and D. Lee, "The Manifold Ways of Perception," *Science*, vol. 290, pp. 2268–2269, 2000.

[22] T. Sim, S. Baker, and M. Bsat, "The CMU Pose, Illumination, and Expression Database," *PAMI*, vol. 25, no. 12, pp. 1615–1618, 2003.

[23] J.B. Tenenbaum, V. de Silva, and J.C. Langford, "A Global Geometric Framework for Nonlinear Dimensionality Reduction," *Science*, vol. 290, pp. 2319–2323, 2000.

[24] M. Turk and A. Pentland, "Eigenfaces for Recognition," *Journal of Cognitive Neuroscience*, vol. 3, no. 1, pp. 71–86, 1991.

[25] V.N. Vapnik, *Statistical Learning Theory*, Wiley, 1998.

[26] S. Yan, H. Zhang, Y. Hu, B. Zhang, and Q. Cheng, "Discriminant Analysis on Embedded Manifold," *ECCV*, vol. 1, pp. 121–132, 2004.

[27] J. Yang, D. Zhang, A.F. Frangi, and J.-Y. Yang, "Two-Dimensional PCA: a New Approach to Appearance-Based Face Representation and Recognition," *PAMI*, vol. 26, no. 1, pp. 131–137, Jan. 2004.

[28] M.-H. Yang, "Face Recognition Using Extended Isomap," *ICIP*, vol. 2, pp. 117–120, 2002.

[29] M.-H. Yang, "Kernel Eigenfaces vs. Kernel Fisherfaces: Face Recognition Using Kernel Methods," *AFGR*, pp. 205–211, 2002.

[30] J. Ye, R. Janardan, and Q. Li, "GPCA: An Efficient Dimension Reduction Scheme for Image Compression and Retrieval," *ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining*, pp. 354–363, 2004.