Local Learning Projections

Mingrui Wu MINGRUI.WU@TUEBINGEN.MPG.DE
Max Planck Institute for Biological Cybernetics, Tübingen, Germany

Kai Yu KYU@SV.NEC-LABS.COM
NEC Labs America, Cupertino CA, USA

Shipeng Yu SHIPENG.YU@SIEMENS.COM
Siemens Medical Solutions, Malvern PA, USA

Bernhard Schölkopf BERNHARD.SCHOELKOPF@TUEBINGEN.MPG.DE
Max Planck Institute for Biological Cybernetics, Tübingen, Germany

Abstract

This paper presents a *Local Learning Projection* (LLP) approach for linear dimensionality reduction. We first point out that the well known *Principal Component Analysis* (PCA) essentially seeks the projection that has the minimal *global* estimation error. Then we propose a dimensionality reduction algorithm that leads to the projection with the minimal *local* estimation error, and elucidate its advantages for classification tasks. We also indicate that LLP keeps the local information in the sense that the projection value of each point can be well estimated based on its neighbors and their projection values. Experimental results are provided to validate the effectiveness of the proposed algorithm.

1. Introduction

In the c-class classification problem, we are given a set of training data $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$, where $\mathbf{x}_i \in \mathcal{X} \subset \mathbb{R}^d$ is the input data, $\mathcal{X}$ is the input space, and $y_i \in \{1, 2, \ldots, c\}$ is the class label. The goal is to build a classifier which can correctly classify the unseen test data.

Given the classification problem, our goal in this paper is to find a low dimensional subspace of $\mathcal{X}$, which retains the discriminating information for classification. In particular, we want to find a linear transformation matrix $\mathbf{P} \in \mathbb{R}^{d \times p}$, $p < d$, which transforms each $\mathbf{x}_i$ to a p-dimensional vector by $\mathbf{P}^\top \mathbf{x}_i$. In many pattern classification problem, such as face recognition, the dimensionality d of the input data is very high. Mapping the input data into a low dimensional subspace of $\mathcal{X}$ can reduce the effect of noise and remove redundant information in the data that is irrelevant to the classification task. Therefore dimensionality reduction is a widely used pre-processing step for classification tasks.

Principal Component Analysis (PCA) is a widely used method for dimensionality reduction. It looks for directions along which the data variance is the largest. PCA is an unsupervised algorithm as it ignores the class labels of the given data, which implies that the subspace obtained by PCA may not be effective for classification problems.

Hence, many supervised projection algorithms have been proposed, which make use of both the input data and the target labels. A popular method is the *Linear Discriminant Analysis* (LDA), which seeks the subspace where the within-class variance is minimized while the between-class variance is maximized. LDA has been successfully applied for face recognition, which is known as the Fisherface method (Belhumeur et al., 1997). However, the dimensionality of the subspace obtained by LDA is upper bounded by $c - 1$, where c is the number of classes. And it is known that LDA assumes a Gaussian distribution for each class of data, which is not necessarily true in practice.

Recently, several manifold embedding algorithms have been proposed, such as the *Locally Linear Embedding* (LLE) (Roweis & Saul, 2000), Laplacian eigenmap (Belkin & Niyogi, 2002), and the *Locality Preserving*

Appearing in *Proceedings of the 24th International Conference on Machine Learning*, Corvallis, OR, 2007. Copyright 2007 by the author(s)/owner(s).

Projection (LPP) (He & Niyogi, 2004; He et al., 2005). These methods look for subspaces that optimally preserve local neighborhood information in some sense.

In this paper, we present a *Local Learning Projection* (LLP) method. Motivated by a new interpretation of PCA that its projections minimize the *global* estimation error, our method seeks projections which minimize the *local* estimation error. As can be seen later, compared with PCA, our method can utilize the supervised information which is important for classification. From another point of view, analogous to manifold embedding algorithms, our method also aims to preserve the local relationship among neighboring points, but in the sense that the projection value of each point can be well estimated based on its neighbors and their projection values.

The rest of this paper is organized as follows. In section 2, we briefly review the *Kernel Ridge Regression* (KRR) algorithm that will be used in later sections. In section 3, a new explanation of PCA is proposed, based on which we introduce the basic idea of our LLP approach. Then the details of the LLP algorithm are presented in section 4. Some comparisons with related methods are given in section 5. Experimental results are provided in section 6. Finally we conclude the paper in the last section.

2. Kernel Ridge Regression

In this section, we briefly review the *Kernel Ridge Regression* (KRR) (Shawe-Taylor & Cristianini, 2004) algorithm that will be used later. Given n labeled points $\{(\mathbf{x}_i, t_i)\}_{i=1}^n$, where $\mathbf{x}_i \in \mathcal{X}$ is the input data and $t_i \in \mathbb{R}$ is the real valued label, the KRR algorithm aims to learn a function that maps from $\mathcal{X}$ to $\mathbb{R}$.

Applying KRR to the training data $\{(\mathbf{x}_i, t_i)\}_{i=1}^n$ results in a *Kernel Machine* (KM) whose output function can be expressed as

$$g(\mathbf{x}) = \sum_{i=1}^{n} \beta_i K(\mathbf{x}, \mathbf{x}_i) \tag{1}$$

where $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a positive definite kernel function (Schölkopf & Smola, 2002), and β_i are the expansion coefficients. To compute the coefficients β_i, we need to solve the following KRR training problem:

$$\min_{\boldsymbol{\beta} \in \mathbb{R}^n} \lambda \boldsymbol{\beta}^\top \mathbf{K} \boldsymbol{\beta} + \|\mathbf{K}\boldsymbol{\beta} - \mathbf{t}\|^2 \tag{2}$$

where $\boldsymbol{\beta} = [\beta_1, \ldots, \beta_n]^\top \in \mathbb{R}^n$ is the vector of the expansion coefficients, $\lambda > 0$ is the regularization parameter, $\mathbf{t} = [t_1, \ldots, t_n]^\top \in \mathbb{R}^n$ denotes the vector of real valued labels, and $\mathbf{K} = [K(\mathbf{x}_i, \mathbf{x}_j)] \in \mathbb{R}^{n \times n}$ is the kernel matrix over all the training points.

The solution of (2) is

$$\boldsymbol{\beta} = (\mathbf{K} + \lambda \mathbf{I})^{-1} \mathbf{t} \tag{3}$$

where $\mathbf{I}$ is the unit matrix. Substituting (3) into (1) leads to

$$g(\mathbf{x}) = \mathbf{k}_x^\top \boldsymbol{\beta} = \mathbf{k}_x^\top (\mathbf{K} + \lambda \mathbf{I})^{-1} \mathbf{t} \tag{4}$$

where $\mathbf{k}_x = [K(\mathbf{x}, \mathbf{x}_1), \ldots, K(\mathbf{x}, \mathbf{x}_n)]^\top \in \mathbb{R}^n$.

3. Projection via Minimizing the Estimation Error

For the ease of description, we start from the simple case where $p = 1$. Namely, the projection result is a vector $\mathbf{f} = [f_1, \ldots, f_n]^\top$, with f_i to be the projection value of $\mathbf{x}_i$.

3.1. Minimizing the Global Estimation Error

Before constructing our dimensionality reduction algorithm, we investigate the projections given by the popular *Kernel Principal Component Analysis* (KPCA) (Schölkopf & Smola, 2002).[1] It is known that the KPCA seeks the projection values $f_1, \ldots, f_n$ that have a large variance. Here we explain KPCA from the point of view of supervised learning. For clarity, we first define the *global estimation error* as follows:

Definition 1. *Given the input data* $\{\mathbf{x}_i\}_{i=1}^n$*, a positive definite kernel function* K*, a positive real number* λ*, and a vector* $\mathbf{f} = [f_1, \ldots, f_n]^\top \in \mathbb{R}^n$*, the global estimation error of* $\mathbf{f}$ *is defined as follows:*

$$E_{global}(\mathbf{f}) = \sum_{i=1}^{n} (f_i - o_{all}(\mathbf{x}_i))^2 \tag{5}$$

where $o_{all}(\cdot)$ *denotes the output function of a Kernel Machine (KM) trained with all the data* $\{(\mathbf{x}_i, f_i)\}_{i=1}^n$*, using the KRR algorithm with the kernel function* K *and the regularization parameter* $\lambda > 0$ *(cf. (2)).*

We can see that the E_{global} is simply the total squared *training* error for the KRR algorithm. The following Proposition indicates that KPCA searches for the projections with the minimal global estimation error.[2]

[1] Although we focus on linear projections in this paper, the content in this section can be applied for both linear and nonlinear cases. Therefore, for generality, we describe our results for KPCA. This can be easily applied for PCA by using the linear kernel in KPCA.

[2] Note that the "estimation error" in definition 1 addresses the relationship among the projection values of different points. This is different from the widely known objective of KPCA, i.e. the "reconstruction error", which describes the relationship between $\mathbf{x}_i$ and their projections.

Proposition 1. *Let $\bar{\mathbf{f}} = [\bar{f}_1, \ldots, \bar{f}_n]^\top \in \mathbb{R}^n$, where $\bar{f}_i$ denotes the projection value of $\mathbf{x}_i$ given by KPCA algorithm. Then among all the unit length vectors, $\bar{\mathbf{f}}/\left\|\bar{\mathbf{f}}\right\|$ is the one with the minimal global estimation error. Namely,*

$$\bar{\mathbf{f}}/\left\|\bar{\mathbf{f}}\right\| = \arg\min_{\mathbf{f}^\top\mathbf{f}=1} E_{global}(\mathbf{f}) \tag{6}$$

Proof. As stated in Definition 1, $o_{all}(\cdot)$ in (5) is the output function of a KM trained with the data $\{(\mathbf{x}_i, f_i)\}_{i=1}^n$, using the KRR algorithm. According to (4), we have

$$o_{all}(\mathbf{x}_i) = \mathbf{k}_{x_i}^\top(\mathbf{K} + \lambda\mathbf{I})^{-1}\mathbf{f}$$

where $\mathbf{k}_{x_i} = [K(\mathbf{x}_i, \mathbf{x}_1), \ldots, K(\mathbf{x}_i, \mathbf{x}_n)]^\top \in \mathbb{R}^n$, and $\mathbf{K} = [K(\mathbf{x}_i, \mathbf{x}_j)] \in \mathbb{R}^{n\times n}$ is the kernel matrix over the input data. Let $\mathbf{o}_{all} = [o_{all}(\mathbf{x}_1), \ldots, o_{all}(\mathbf{x}_n)]^\top \in \mathbb{R}^n$, then

$$\mathbf{o}_{all} = \mathbf{K}(\mathbf{K} + \lambda\mathbf{I})^{-1}\mathbf{f} \tag{7}$$

Substituting (7) into (5) leads to

$$\begin{aligned} E_{global}(\mathbf{f}) &= \sum_{i=1}^n (f_i - o_{all}(\mathbf{x}_i))^2 = \left\|\mathbf{f} - \mathbf{o}_{all}\right\|^2 \\ &= \left\|\mathbf{f} - \mathbf{K}(\mathbf{K} + \lambda\mathbf{I})^{-1}\mathbf{f}\right\|^2 \\ &= \mathbf{f}^\top(\mathbf{I} - \mathbf{B})^\top(\mathbf{I} - \mathbf{B})\mathbf{f} \end{aligned} \tag{8}$$

where

$$\mathbf{B} = \mathbf{K}(\mathbf{K} + \lambda\mathbf{I})^{-1} \tag{9}$$

Performing eigen-decomposition for the kernel matrix $\mathbf{K}$ results in

$$\mathbf{K} = \mathbf{V}\boldsymbol{\Lambda}\mathbf{V}^\top \tag{10}$$

where $\boldsymbol{\Lambda}$ is a diagonal matrix of the eigenvalues of $\mathbf{K}$, and $\mathbf{V}$ is an orthogonal matrix whose columns are eigenvectors of $\mathbf{K}$, hence

$$\mathbf{V}^\top\mathbf{V} = \mathbf{V}\mathbf{V}^\top = \mathbf{I} \tag{11}$$

Substituting (10) into (9) and using (11), we have,

$$\begin{aligned} \mathbf{B} &= \mathbf{K}(\mathbf{K} + \lambda\mathbf{I})^{-1} \\ &= \mathbf{V}\boldsymbol{\Lambda}\mathbf{V}^\top(\mathbf{V}\boldsymbol{\Lambda}\mathbf{V}^\top + \lambda\mathbf{V}\mathbf{V}^\top)^{-1} \\ &= \mathbf{V}\boldsymbol{\Lambda}_1\mathbf{V}^\top \end{aligned} \tag{12}$$

where

$$\boldsymbol{\Lambda}_1 = \boldsymbol{\Lambda}(\boldsymbol{\Lambda} + \lambda\mathbf{I})^{-1} \tag{13}$$

is a diagonal matrix. Based on (12) and (11), we have,

$$\mathbf{I} - \mathbf{B} = \mathbf{V}\mathbf{V}^\top - \mathbf{V}\boldsymbol{\Lambda}_1\mathbf{V}^\top = \mathbf{V}(\mathbf{I} - \boldsymbol{\Lambda}_1)\mathbf{V}^\top \tag{14}$$

Substituting (14) into (8) and using (11) leads to,

$$E_{global}(\mathbf{f}) = \mathbf{f}^\top\mathbf{V}(\mathbf{I} - \boldsymbol{\Lambda}_1)^2\mathbf{V}^\top\mathbf{f} \tag{15}$$

Let λ_i and $\hat{\lambda}_i$ denote the i-th eigenvalue of matrix $\mathbf{K}$ and $\mathbf{V}(\mathbf{I} - \boldsymbol{\Lambda}_1)^2\mathbf{V}^\top$ respectively, then $\lambda_i \geq 0$ since $K(\cdot, \cdot)$ is positive definite, and the following equation follows from (13),

$$\hat{\lambda}_i = \left(\frac{\lambda}{\lambda + \lambda_i}\right)^2 \tag{16}$$

Let $\mathbf{f}_o = \arg\min_{\mathbf{f}^\top\mathbf{f}=1} E_{global}(\mathbf{f})$. From (15) it can be seen that $\mathbf{f}_o$ is the eigenvector that corresponds to the minimal eigenvalue of matrix $\mathbf{V}(\mathbf{I} - \boldsymbol{\Lambda}_1)^2\mathbf{V}^\top$. Based on (16) and (10), we can conclude that,

$$\mathbf{f}_o = \mathbf{v}_o \tag{17}$$

where $\mathbf{v}_o$ is the eigenvector of $\mathbf{K}$ corresponding to the maximal eigenvalue of $\mathbf{K}$, which is denoted by λ_o in the following. Therefore,

$$\mathbf{K}\mathbf{v}_o = \lambda_o\mathbf{v}_o \tag{18}$$

According to (Schölkopf & Smola, 2002), for a point $\mathbf{x} \in \mathbb{R}^d$, its projection given by KPCA is computed as: $\frac{1}{\sqrt{\lambda_o}}\mathbf{k}_x^\top\mathbf{v}_o$, where $\mathbf{k}_x = [K(\mathbf{x}, \mathbf{x}_1), \ldots, K(\mathbf{x}, \mathbf{x}_n)]^\top \in \mathbb{R}^n$.[3] Hence, for the given points $\mathbf{x}_1, \ldots, \mathbf{x}_n$, their projections given by KPCA can be computed as

$$\bar{\mathbf{f}} = \frac{1}{\sqrt{\lambda_o}}\mathbf{K}\mathbf{v}_o = \sqrt{\lambda_o}\mathbf{v}_o = \sqrt{\lambda_o}\mathbf{f}_o \tag{19}$$

where we have used (18) and (17) in the second and third equation above respectively.

Therefore, equation $\bar{\mathbf{f}}/\left\|\bar{\mathbf{f}}\right\| = \mathbf{f}_o$ follows from (19). And the proposition is proven. □

Note that in Proposition 1, we consider the unit length vectors, because otherwise the trivial solution $\mathbf{0} = [0, \ldots, 0]^\top \in \mathbb{R}^n$ is the optimal one that minimizes the global estimation error (5).

3.2. Minimizing the Local Estimation Error

In the following, $\mathcal{N}_i$ denotes the set of neighboring points of $\mathbf{x}_i$, *not* including $\mathbf{x}_i$ itself. And n_i denotes $|\mathcal{N}_i|$, i.e. the number of points in $\mathcal{N}_i$. Here, “neighboring points” or “neighbors” of $\mathbf{x}_i$ simply means the nearest neighbors of $\mathbf{x}_i$ according to some distance metric, such as a the Euclidean distance.

Imagine that given the input data $\mathbf{x}_i$ and the projection value f_i, $1 \leq i \leq n$, we want to use the KRR algorithm to learn a function which maps from $\mathbf{x}_i$ to

[3] Here we assume that the input data have been centralized in the feature space of the kernel function K. However for the projection algorithm that will be proposed later, we do not need to make this assumption.

f_i. Then Proposition 1 tells that the projection obtained from KPCA is the easiest one to learn in the sense that it has the minimal global estimation error.

Starting from this point of view, we also search for the projection result that is easy to learn. However, in contrast to KPCA, we propose that the quality of learning is measured by the *local estimation error*, which is computed as

$$E_{local}(\mathbf{f}) = \sum_{i=1}^{n} (f_i - o_i(\mathbf{x}_i))^2 \tag{20}$$

where $o_i(\cdot)$ denotes the output function of a KM trained with the KRR algorithm, using the training data $\{(\mathbf{x}_j, f_j)\}_{\mathbf{x}_j \in \mathcal{N}_i}$. For the function $o_i(\cdot)$, the subscript i means the KM is trained with the neighbors of $\mathbf{x}_i$.

By minimizing the local error rather than the global one, we can gain more flexibility than KPCA by varying the number of neighbors of each point. More importantly, a real advantage of this local learning approach is that in classification problems, where the class label of each point is available, we can require that the neighboring points belong to the same class. This way, we can get a projection result where the projection value of each point can be well estimated based on its neighbors in the same class. This implies that for the neighboring points in the same class, their projection values are similar to each other, which is helpful for classification.

4. Local Learning Projections

In this section, we describe our LLP algorithm that can produce projections with small local estimation errors.

In LLP algorithm, we want to find a linear projection matrix $\mathbf{P} = [\mathbf{p}_1, \ldots, \mathbf{p}_p] \in \mathbb{R}^{d\times p}$, where $\mathbf{p}_l \in \mathbb{R}^d$ $(1 \le l \le p)$ is the l-th column of $\mathbf{P}$. And we require the columns of $\mathbf{P}$ to constitute an orthogonal basis of a p-dimensional subspace of $\mathcal{X}$, namely $\mathbf{P}^\top\mathbf{P} = \mathbf{I}$.

In the following, $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_n] \in \mathbb{R}^{d\times n}$ denotes the matrix of the input data, $\mathbf{F} = \mathbf{P}^\top\mathbf{X} \in \mathbb{R}^{p\times n}$ denotes the matrix of the projection result, and $\mathbf{f}^l = \mathbf{X}^\top\mathbf{p}_l = [f_1^l, \ldots, f_n^l]^\top \in \mathbb{R}^n$ is the l-th column of $\mathbf{F}^\top$, i.e. the vector containing the l-th projection values of the input data.

4.1. Objective Function

Following the idea of minimizing the local estimation error, in order to find the projection matrix $\mathbf{P}$, we can solve the following optimization problem:

$$\min_{\mathbf{P}\in\mathbb{R}^{d\times p}, \mathbf{F}\in\mathbb{R}^{p\times n}} \quad \sum_{l=1}^{p} E_{local}(\mathbf{f}^l) \tag{21}$$

$$\text{subject to} \quad \mathbf{F} = \mathbf{P}^\top\mathbf{X} \tag{22}$$

$$\left\|\mathbf{f}^l\right\| = 1, \quad 1 \le l \le p \tag{23}$$

$$\mathbf{p}_i^\top\mathbf{p}_j = 0, \quad 1 \le i < j \le p \tag{24}$$

The objective function (21) to be minimized is the sum over the local estimation errors of all $\mathbf{f}^l$, $1 \le l \le p$. In (21) $E_{local}(\cdot)$ is defined as (20), hence (21) can be rewritten as

$$\sum_{l=1}^{p} E_{local}(\mathbf{f}^l) = \sum_{l=1}^{p}\sum_{i=1}^{n} (f_i^l - o_i^l(\mathbf{x}_i))^2 = \sum_{l=1}^{p} \left\|\mathbf{f}^l - \mathbf{o}^l\right\|^2 \tag{25}$$

where $o_i^l(\cdot)$ denotes the output function of a KM, trained with KRR algorithm, using the training data $\{(\mathbf{x}_j, f_j^l)\}_{\mathbf{x}_j\in\mathcal{N}_i}$. In (25), $\mathbf{o}^l = [o_1^l(\mathbf{x}_1), \ldots, o_n^l(\mathbf{x}_n)]^\top \in \mathbb{R}^n$. For the function $o_i^l(\cdot)$, the superscript l indicates that it is for the l-th projection value $\mathbf{f}^l$, and the subscript i means the KM is trained with the neighbors of $\mathbf{x}_i$.[4]

Similarly as in Proposition 1, we put the constraint (23) to avoid the trivial projection result of $\mathbf{f}^l = \mathbf{X}^\top\mathbf{p}_l = \mathbf{0}$, i.e. $\mathbf{p}_l$ is orthogonal to all the data points, which can happen when the data dimensionality d is larger than the number of data n. After solving the problem (21)–(24), we can normalize each $\mathbf{p}_l$ $(1 \le l \le p)$ to unit length such that $\mathbf{P}$ can satisfy $\mathbf{P}^\top\mathbf{P} = \mathbf{I}$.

Unfortunately, the problem (21)–(24) is difficult to solve. We have to modify it and find that the following simple approach can result in both a easy-to-solve optimization problem and good classification results: We just replace the constraint (23) with $\|\mathbf{p}_l\| = 1$ $(1 \le l \le p)$ and substitute (25) into (21), leading to the following optimization problem,

$$\min_{\mathbf{P}\in\mathbb{R}^{d\times p}, \mathbf{F}\in\mathbb{R}^{p\times n}} \quad \sum_{l=1}^{p} \left\|\mathbf{f}^l - \mathbf{o}^l\right\|^2 \tag{26}$$

$$\text{subject to} \quad \mathbf{F} = \mathbf{P}^\top\mathbf{X} \tag{27}$$

$$\mathbf{P}^\top\mathbf{P} = \mathbf{I} \tag{28}$$

In order to prevent the undesirable result where $\mathbf{f}^l = \mathbf{X}^\top\mathbf{p}_l = \mathbf{0}$, we fist project the input data into the subspace obtained by PCA, with all the zero principal

[4] Note that although we are constructing a linear projection algorithm, we are free to choose the kernel function $K(\cdot,\cdot)$ for computing the local estimation error. However, in order to compare with PCA, we will just use linear kernel for LLP in the experiments.

components removed. This way, we keep all the information and make the data dimensionality d smaller than the number of data n. After PCA transformation, the projection matrix $\mathbf{P}$ for the transformed data can be obtained by solving the problem (26)–(28). Then the final projection matrix is computed as $\mathbf{P}_{PCA}\mathbf{P}$, where $\mathbf{P}_{PCA}$ denotes the projection matrix of PCA. In fact, using PCA as a preprocessing step has already been adopted by other linear projection methods, such as LDA (Belhumeur et al., 1997) and LPP (He et al., 2005).

4.2. Solution

Now we consider how to solve the problem (26)–(28). Recall that $o_i^l(\cdot)$ denotes the output function of a KM, trained with KRR algorithm, using the training data $\{(\mathbf{x}_j, f_j^l)\}_{\mathbf{x}_j \in \mathcal{N}_i}$. Based on equation (4), we have

$$o_i^l(\mathbf{x}_i) = \mathbf{k}_i^\top (\mathbf{K}_i + \lambda \mathbf{I})^{-1} \mathbf{f}_i^l \tag{29}$$

where $\mathbf{k}_i \in \mathbb{R}^{n_i}$ denotes the vector $[K(\mathbf{x}_i, \mathbf{x}_j)]^\top$ for $\mathbf{x}_j \in \mathcal{N}_i$, $\mathbf{f}_i^l \in \mathbb{R}^{n_i}$ denotes the vector $\left[f_j^l\right]^\top$ for $\mathbf{x}_j \in \mathcal{N}_i$, and $\mathbf{K}_i \in \mathbb{R}^{n_i \times n_i}$ is the kernel matrix over $\mathbf{x}_j \in \mathcal{N}_i$. Equation (29) can be written as a linear equation:

$$o_i^l(\mathbf{x}_i) = \boldsymbol{\alpha}_i^\top \mathbf{f}_i^l \tag{30}$$

where $\boldsymbol{\alpha}_i \in \mathbb{R}^{n_i}$ is computed as

$$\boldsymbol{\alpha}_i^\top = \mathbf{k}_i^\top (\mathbf{K}_i + \lambda \mathbf{I})^{-1} \tag{31}$$

It can be seen that $\boldsymbol{\alpha}_i$ is independent of $\mathbf{f}_i^l$ and the projection index l, and it is different for different $\mathbf{x}_i$. Note that $\mathbf{f}_i^l$ is a sub-vector of $\mathbf{f}^l$, so equation (30) can be written in a compact form as:

$$\mathbf{o}^l = \mathbf{A}\mathbf{f}^l \tag{32}$$

where $\mathbf{o}^l$ and $\mathbf{f}^l$ are the same as in (26), while the matrix $\mathbf{A} = [a_{ij}] \in \mathbb{R}^{n \times n}$ is constructed as follows: $\forall \mathbf{x}_i$ and $\mathbf{x}_j$, $1 \le i, j \le n$, if $\mathbf{x}_j \in \mathcal{N}_i$, then a_{ij} equals the corresponding element of $\boldsymbol{\alpha}_i$ in (31), otherwise a_{ij} equals 0. Similar as $\boldsymbol{\alpha}_i$, the matrix $\mathbf{A}$ is also independent of $\mathbf{f}^l$ and the projection index l.

Substituting (32) into (26) results in the following optimization problem,

$$\min_{\mathbf{P} \in \mathbb{R}^{d \times p}, \mathbf{F} \in \mathbb{R}^{p \times n}} \quad \sum_{l=1}^{p} \left\| \mathbf{f}^l - \mathbf{A}\mathbf{f}^l \right\|^2 \tag{33}$$

$$\text{subject to} \quad \mathbf{F} = \mathbf{P}^\top \mathbf{X} \tag{34}$$

$$\mathbf{P}^\top \mathbf{P} = \mathbf{I} \tag{35}$$

Let $\mathbf{T} = (\mathbf{I} - \mathbf{A})^\top (\mathbf{I} - \mathbf{A})$, then the objective function (33) can be re-written as:

$$\sum_{l=1}^{p} \left\| \mathbf{f}^l - \mathbf{A}\mathbf{f}^l \right\|^2 = \sum_{l=1}^{p} (\mathbf{f}^l)^\top \mathbf{T} \mathbf{f}^l = trace(\mathbf{F}\mathbf{T}\mathbf{F}^\top) \tag{36}$$

Substituting (34) into (36), the problem (33)–(35) is transformed into the following,

$$\min_{\mathbf{P} \in \mathbb{R}^{d \times p}} \quad trace(\mathbf{P}^\top \mathbf{X}\mathbf{T}\mathbf{X}^\top \mathbf{P}) \tag{37}$$

$$\text{subject to} \quad \mathbf{P}^\top \mathbf{P} = \mathbf{I} \tag{38}$$

Let $\mathbf{P}_0 \in \mathbb{R}^{d \times p}$ denote the matrix whose columns consist of p eigenvectors associated with the p smallest eigenvalues of $\mathbf{X}\mathbf{T}\mathbf{X}^\top$, then for any orthogonal matrix $\mathbf{R} \in \mathbb{R}^{p \times p}$, $\mathbf{R}^\top \mathbf{R} = \mathbf{I}$, the matrix $\mathbf{P}_0 \mathbf{R}$ is a global optimum of the problem (37)–(38). In this paper, we just take $\mathbf{P}_0$ as the solution for problem (37)–(38). In the experiments, for simplicity we will apply the nearest neighbor algorithm to classify the projected data $\mathbf{P}_0^\top \mathbf{x}_i$ $(1 \le i \le n)$, whose classification result is not affected by the orthogonal matrix $\mathbf{R}$.

5. Comparison with Related Approaches

5.1. Comparison with Local PCA

Although LLP is derived from PCA and the idea of minimizing the local estimation error, it still seeks a *global* linear projection without partitioning the data into subgroups. This is different from the *local PCA* approach (Tipping & Bishop, 1999; Meinicke & Ritter, 1999), which uses the mixture model framework and the EM algorithm to combine the PCA results computed locally for different subgroups of data. Therefore in local PCA, we have to choose the number of mixtures properly and there exists the local optimum problem since the EM algorithm is used.

5.2. Comparison with LPP

The recently proposed LPP algorithm (He & Niyogi, 2004; He et al., 2005) is a linear projection method that also preserves the local relationship among neighboring points. Similarly as Laplacian Eigenmap (Belkin & Niyogi, 2002), its basic idea is to seek the projection result $\mathbf{f}$ that minimizes the following objective function,[5]

$$E_{LPP}(\mathbf{f}) = \frac{1}{2} \sum_{i,j} (f_i - f_j)^2 w_{ij} \tag{39}$$

where w_{ij} measures the similarity between $\mathbf{x}_i$ and $\mathbf{x}_j$. According to (39), mapping $\mathbf{x}_i$ and $\mathbf{x}_j$ far apart will cause a heavy penalty if w_{ij} is large. This implies that if $\mathbf{x}_i$ and $\mathbf{x}_j$ are similar to each other, then f_i and f_j should also be close.

In classification problems, we can construct w_{ij} to be positive if $\mathbf{x}_i$ and $\mathbf{x}_j$ are close and belong to the same

[5] We only describe the case of $p = 1$ for simplicity.

class, otherwise $w_{ij} = 0$. Thus in LPP, the projections of nearby points belonging to the same class tend to be grouped together, which is beneficial for classification.

By setting the gradient $\frac{\partial}{\partial \mathbf{f}} E_{LPP}(\mathbf{f})$ to $\mathbf{0}$, it can be seen that the optimal $\mathbf{f}$ minimizing (39) must satisfy the following equation:

$$f_i = \frac{\sum_{\mathbf{x}_j \in \mathcal{N}_i} w_{ij} f_j}{\sum_{\mathbf{x}_j \in \mathcal{N}_i} w_{ij}} \tag{40}$$

where we have used the property that w_{ij} is nonzero only for the neighboring points. Equation (40) indicates that LPP *implicitly* expects that f_i equals the local weighted average of f_j for $\mathbf{x}_j \in \mathcal{N}_i$, and the weight of f_j is proportional to w_{ij}, which measures the similarity between $\mathbf{x}_i$ and $\mathbf{x}_j$.

Therefore we can see the connections between LLP and LPP from the local learning point of view. LLP *explicitly* requires that f_i can be well estimated based on its neighbors, while LPP specifies this *implicitly*. In LLP, f_i is estimated by $o_i(\mathbf{x}_i)$, which is trained with a well established regression approach, while in LPP, f_i is estimated with the local average, which is identical to the regression solution given by the classical Nadaraya-Watson algorithm (Nadaraya, 1989).

5.3. Comparison with Manifold Embedding

There exist several other dimensionality reduction methods that aim to preserve the local information. Such as LLE (Roweis & Saul, 2000) and Laplacian eigenmap (Belkin & Niyogi, 2002). But these methods are mainly for data representation and can not compute the projection values for new test data.

5.4. Local learning for Classification and Clustering

The idea of local learning, estimating the label of a point based on its neighbors, is applied for classification in (Bottou & Vapnik, 1992). It is also adapted into a clustering approach in (Wu & Schölkopf, 2007), which tries to minimize the local estimation error for the cluster labels of neighboring points. Good classification and clustering results obtained in (Bottou & Vapnik, 1992) and (Wu & Schölkopf, 2007) respectively illustrate that the label of a point can be well estimated based on its neighbors, and minimizing the local estimation error is effective in keeping the local relationship. In this paper, we exploit this fact and develop the LLP algorithm. Good classification results can be expected for LLP since the subspace found by LLP can well preserve the local information among the neighboring points in the same class.

5.5. Other Related Methods

In both statistics and machine learning literature, many projection algorithms have been proposed that consider simultaneously dimensionality reduction and the target labels for supervised learning problems. For instance, the *Sliced Inverse Regression* (Li, 1991), the *Sliced Average Variance Estimation* (Cook & Weisberg, 1991) and the *Minimum Average Variance Estimation* (MAVE) (Xia et al., 2002) are feature extraction approaches for regression. Furthermore, based on the MAVE framework, a spectral method for projection is provided in (Mukherjee et al., 2006). Similarly, the algorithms presented in (Sugiyama, 2006; Goldberger et al., 2005) are supervised dimensionality reduction methods for classification. The algorithm presented in (Globerson & Roweis, 2006) addresses the metric learning problem for classification, which is also related to the topic of this paper. Compared with these algorithms, the distinct features of LLP are: It is closely related to a new explanation of PCA, and the regression technique is adopted to investigate whether the projection of a point can be well estimated based on its neighbors in the same class.

6. Experimental Results

In this section, we empirically compare the proposed LLP algorithm with PCA, LDA and LPP.

6.1. Datasets and Experimental Settings

Both LDA and LPP have been successfully applied for face recognition, which are known as Fisherface (Belhumeur et al., 1997) and Laplacianface (He et al., 2005) respectively. PCA, or Eigenface, is also a well known method for face recognition. Therefore five face image datasets are considered in the experiments: Yale, ORL, YaleB, PIE and UMist.[6] All the face images are resized to 32×32 pixels. So each image can be represented by a 1024 dimensional vector. Further descriptions of these datasets are provided in Table 1.

On each dataset, m images per class are randomly selected as training samples, while the remaining are used for test. The linear projection matrix $\mathbf{P}$ is learned with the training data. Both the training and test data are transformed by the learned matrix $\mathbf{P}$. Then following the scheme in (He & Niyogi, 2004; He et al., 2005), the transformed data are input to a *1-Nearest neighbor* (1-NN) classifier, and the classification performance on the test data is used to evaluate the projection algo-

[6] The first four are obtained from http://ews.uiuc.edu/~dengcai2/Data/data.html. While UMist is available at http://www.cs.toronto.edu/~roweis/data.html.

rithms.

Table 1. The number of data n and the number of classes c for the datasets used in the experiments.

Dataset	Yale	ORL	YaleB	PIE	UMist
n	165	400	2414	11554	575
c	15	40	38	68	20

On each dataset, three different values of m are tried to investigate the performance of different algorithms with different number of training data. For each given m, 20 training/test splits are randomly generated and the average test error over these splits is used to evaluate the classification performance. Furthermore, the Wilcoxon rank sum test is conducted to examine whether the performance difference between different approaches are statistically significant.

6.2. Parameter Selection

Five fold cross validation is performed on the training data for parameter selection.

As stated before, in LPP and LLP, the neighboring points belong to the same class. For these two algorithms, the number of neighbors for each point is searched from: $\{5,\ 10,\ 20,\ m-1\}$, where m is the number of training samples in each class.[7]

Proposition 1 tells that PCA gives the projection with the minimal global estimation error that is computed based on the linear ridge regression. In order to compare with PCA, we simply use the linear kernel in the local KRR for LLP. The regularization parameter λ (cf. (2)) is searched in: $\{0.1,\ 1,\ 10\}$.

For LPP, as suggested in (He et al., 2005), for two neighboring points $\mathbf{x}_i$ and $\mathbf{x}_j$, w_{ij} (cf. (39)) is computed with a Gaussian kernel: $w_{ij} = \exp(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{\gamma})$, where the kernel parameter γ is searched from the grid: $\{\frac{\sigma_0^2}{16},\ \frac{\sigma_0^2}{8},\ \frac{\sigma_0^2}{4},\ \frac{\sigma_0^2}{2},\ \sigma_0^2,\ 2\sigma_0^2,\ 4\sigma_0^2,\ 8\sigma_0^2,\ 16\sigma_0^2\}$, where σ_0 is the mean norm of the training data.

For all projection algorithms, the number of projections p varies from 1 to 300, also selected with cross validation.[8]

6.3. Numerical Results

Numerical Results are summarized in Table 2. As a baseline, we also report the classification result of the 1-NN algorithm using the raw input data directly without data projection.

Consistent with the results in (Belhumeur et al., 1997; He et al., 2005), PCA performs much worse than LDA. Its result is identical to the baseline on the datasets that are used in the experiments.

As can be seen from Table 2, LLP compares favorably to the other projection algorithms. In particular, it outperforms PCA in all cases. This illustrates that by minimizing the local estimation error rather than the global estimation error, the classification performance can be significantly improved.

7. Conclusions

A new explanation for the widely used PCA algorithm has been presented, which tells that PCA seeks projections with the minimal global estimation error. Inspired by this discovery and the local learning idea, we have proposed a *local learning projection* (LLP) approach for linear dimensionality reduction. By minimizing the local estimation error for the neighboring points in the same class, LLP can utilize the class labels that are ignored by PCA, and the solution given by LLP has the property that the projection value of each point can be well estimated based on its neighbors. This is effective for preserving the local information among neighboring points and hence can lead to good classification results for the projected data. Experimental results indicate that our approach often performs better than the related algorithms to which we compared.

References

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

Belkin, M., & Niyogi, P. (2002). Laplacian eigenmaps and spectral techniques for embedding and clustering. In T. G. Dietterich, S. Becker and Z. Ghahramani (Eds.), *Advances in neural information processing systems 14*. Cambridge, MA: MIT Press.

Bottou, L., & Vapnik, V. (1992). Local learning algorithms. *Neural Computation*, *4*, 888–900.

Cook, R., & Weisberg, S. (1991). Discussion of "sliced inverse regression for dimension reduction". *Journal of the American Statistical Association*, *86*, 328–332.

Globerson, A., & Roweis, S. (2006). Metric learning by collapsing classes. In Y. Weiss, B. Schölkopf and

[7] Only the values that are smaller than m are considered for the number of neighbors of each point.

[8] For LDA, p can not be larger than $c-1$ in a c-class problem. For the other projection algorithms, p should be smaller than the total number of training data.

Table 2. Average test error rate (%) and the standard deviations (%) of different methods on the five face image datasets. For PCA, LDA, LPP and LLP, the numbers in parentheses denote the mean optimal number of projections obtained via cross validation over 20 training/test splits. In each group, the results shown in boldface is significantly better than the others, judged by Wilcoxon rank sum test, with a significance level of 0.01.

Dataset	m	1-NN	PCA	LDA	LPP	LLP
Yale	5	42.1±4.0	42.1±4.0 (31.0)	24.3±2.7 (12.9)	22.4±3.6 (19.5)	**19.8±3.5** (20.4)
	6	40.3±4.1	40.3±4.1 (35.7)	21.5±3.8 (12.9)	20.2±4.7 (17.7)	**16.9±3.5** (24.9)
	7	38.8±4.3	38.8±4.3 (38.2)	19.8±4.1 (13.1)	19.7±4.4 (20.5)	**14.8±3.7** (25.7)
ORL	5	11.9±1.2	11.9±1.2 (76.8)	5.7±1.0 (35.8)	6.7±1.1 (37.4)	**3.1±1.2** (82.6)
	6	9.1±2.0	9.1±2.0 (113.9)	4.3±1.2 (36.9)	4.5±1.7 (38.1)	**2.6±1.1** (87.6)
	7	6.9±2.4	6.9±2.4 (97.3)	3.5±1.2 (35.2)	3.8±1.3 (40.7)	**2.0±1.0** (89.1)
YaleB	10	55.2±1.0	55.2±1.0 (232.1)	21.6±1.1 (36.8)	19.6±3.1 (85.0)	**16.6±1.0** (61.0)
	20	41.7±0.8	41.7±0.8 (269.7)	13.8±0.9 (36.9)	17.6±2.9 (114.9)	**11.2±1.5** (83.8)
	30	34.5±1.3	34.5±1.3 (278.6)	12.8±1.1 (37.0)	13.6±1.1 (161.7)	**8.7±1.4** (247.1)
PIE	10	65.0±0.5	65.0±0.5 (253.4)	29.7±1.3 (66.6)	28.8±1.4 (157.3)	**18.7±0.9** (107.2)
	20	48.8±0.7	48.8±0.7 (283.6)	21.1±0.7 (65.9)	20.7±1.1 (270.2)	**16.6±0.6** (276.2)
	30	37.9±0.8	37.9±0.8 (287.3)	10.8±0.6 (66.4)	**9.7±0.8** (67.7)	14.3±0.7 (291.9)
UMist	5	15.2±3.8	15.2±3.8 (33.2)	10.2±2.8 (13.2)	14.5±3.7 (19.8)	**6.2±2.7** (44.2)
	6	12.0±3.1	12.0±3.1 (33.2)	7.2±2.3 (15.0)	12.0±2.3 (19.6)	**4.5±2.3** (54.7)
	7	9.7±2.3	9.7±2.3 (43.4)	5.9±2.2 (13.3)	9.8±2.4 (18.9)	**3.7±1.8** (56.3)

J. Platt (Eds.), *Advances in neural information processing systems 18*. Cambridge, MA: MIT Press.

Goldberger, J., Roweis, S., Hinton, G., & Salakhutdinov, R. (2005). Neighbourhood components analysis. In L. K. Saul, Y. Weiss and L. Bottou (Eds.), *Advances in neural information processing systems 17*. Cambridge, MA: MIT Press.

He, X., & Niyogi, P. (2004). Locality preserving projections. In S. Thrun, L. Saul and B. Schölkopf (Eds.), *Advances in neural information processing systems 16*. Cambridge, MA: MIT Press.

He, X., Yan, S., Hu, Y., Niyogi, P., & Zhang, H. (2005). Face recognition using laplacianfaces. *IEEE Transactions on Pattern Analysis and machine Intelligence*, *27*, 328–340.

Li, K.-C. (1991). Sliced inverse regression for dimension reduction. *Journal of the American Statistical Association*, *86*, 316–327.

Meinicke, P., & Ritter, H. (1999). Local PCA learning with resolution-dependent mixtures of gaussians. *International Conference on Artifical Neural Networks*. Edinburgh, UK: MIT Press.

Mukherjee, S., Wu, Q., & Zhou, D.-X. (2006). Learning gradients and feature selection on manifolds. *submitted to Annals of Statistics*.

Nadaraya, E. A. (1989). *Nonparametric estimation of probabiliry densities and regression curves*. Dordrecht: Kluwer Academic.

Roweis, S. T., & Saul, L. K. (2000). Nonlinear dimensionality reduction by locally linear embedding. *Science*, *290*, 2323–2326.

Schölkopf, B., & Smola, A. J. (2002). *Learning with kernels*. Cambridge, MA: The MIT Press.

Shawe-Taylor, J., & Cristianini, N. (2004). *Kernel methods for pattern analysis*. Cambridge, UK: Cambridge University Press.

Sugiyama, M. (2006). Local fisher discriminant analysis for supervised dimensionality reduction. In W. Cohen and A. Moore (Eds.), *Proc. 23th international conference on machine learning*. ACM Press.

Tipping, M. E., & Bishop, C. M. (1999). Mixtures of probabilistic principle component analysers. *Neural Computation*, *11*, 443–482.

Wu, M., & Schölkopf, B. (2007). A local learning approach for clustering. In B. Schölkopf, J. Platt and T. Hoffman (Eds.), *Advances in neural information processing systems 19*. Cambridge, MA: MIT press.

Xia, Y., Tong, H., Li, W. K., & Zhu, L.-X. (2002). An adaptive estimation of dimension reduction space. *Journal of Royal Statistical Society*, *64*, 363–410.