

Kernel-based Feature Extraction under Maximum Margin Criterion

Jiangping Wang, Jieyan Fan, Huanghuang Li, and Dapeng Wu¹

Department of Electrical and Computer Engineering, University of Florida, Gainesville, FL 32611, USA

Abstract

In this paper, we study the problem of feature extraction for pattern classification applications. RELIEF is considered as one of the best-performed algorithms for assessing the quality of features for pattern classification. Its extension, local feature extraction (LFE), was proposed recently and was shown to outperform RELIEF. In this paper, we extend LFE to the nonlinear case, and develop a new algorithm called kernel LFE (KLFE). Compared with other feature extraction algorithms, KLFE enjoys nice properties such as low computational complexity, and high probability of identifying relevant features; this is because KLFE is a nonlinear wrapper feature extraction method and consists of solving a simple convex optimization problem. The experimental results have shown the superiority of KLFE over the existing algorithms.

Keywords: Feature extraction, kernel method, pattern classification, RELIEF, maximum margin criterion.

1. Introduction

In this paper, we study the problem of feature extraction for pattern classification applications. As shown in Fig. 1, a typical pattern classification system consists of two parts: one for the training phase and one for the classification phase. In the training phase, the system is given a training data set,

$$\mathcal{D} \triangleq \{(\mathbf{x}_n, y_n)\}_{n=1}^N \subset \mathcal{X} \times \mathcal{Y}, \quad (1)$$

where N is the number of samples in the training data set, $\mathcal{X} \subset \mathbb{R}^I$ is the I -dimensional feature space, and $\mathcal{Y} = \{\pm 1\}$ ² is the label space. At the end of the training phase, the system obtains a parameter set, which provides information needed by the feature extraction module and the classifier in the classification phase. Although in many papers, the feature extraction module for a classification system is not explicitly specified, it is a significant component. A good feature extraction technique not only reduces system complexity and processing time, but also improves classification accuracy by eliminating irrelevant features.

¹Correspondence author: Huanghuang Li, lihh@ufl.edu.

²In this paper, we focus on two-category pattern classification problem.

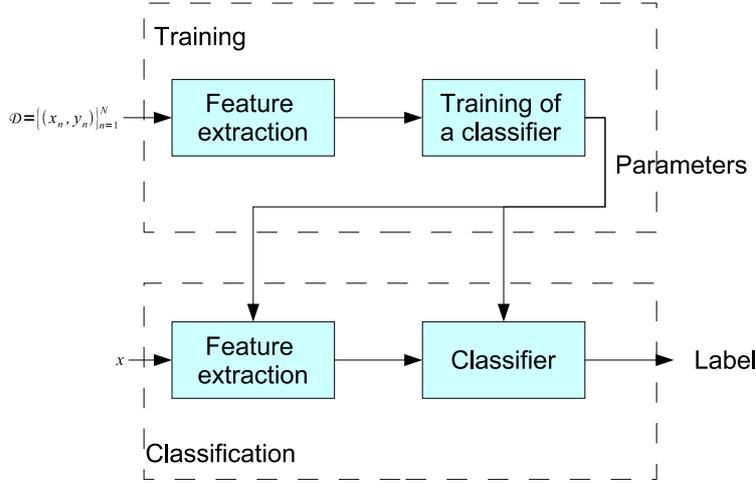


Figure 1: A typical pattern classification system.

In this paper, we propose a feature extraction algorithm, called kernel local feature extraction (KLFE), which has low computational complexity, and achieves high probability of identifying irrelevant features for removal (dimension reduction).

To begin with, we define feature extraction as a mapping

$$f : \mathcal{X} \subset \mathbb{R}^I \rightarrow \mathcal{X}' \subset \mathbb{R}^{I'}, \quad (2)$$

which maps patterns in input feature space \mathcal{X} to output feature space \mathcal{X}' , in order to optimize some pre-defined criterion. Usually, we have $I' \leq I$ so that features are mapped from a high-dimensional space to a lower one, which reduces system complexity.

A feature extraction method can be categorized according to the following criteria.

First, a feature extraction method can be linear or nonlinear. A linear feature extraction method has the form

$$f(\mathbf{x}) = \mathcal{A} \cdot \mathbf{x}, \quad \mathbf{x} \in \mathcal{X}, \quad (3)$$

where \mathcal{A} is a $I' \times I$ matrix. Otherwise, it is nonlinear. Usually nonlinear methods outperform linear ones as nonlinear methods are able to capture the true pattern, which is usually nonlinear. In this paper, we propose a nonlinear feature extraction method, called KLFE, which has the form of

$$f(\mathbf{x}) = \mathcal{A} \cdot \varphi(\mathbf{x}), \quad (4)$$

where $\varphi : \mathcal{X} \subset \mathbb{R}^I \rightarrow \bar{\mathcal{X}} \subset \mathbb{R}^{\bar{I}}$ is a nonlinear function and \mathcal{A} is an $I' \times \bar{I}$ matrix.

Second, feature extraction has two special cases, namely, *feature selection* and *feature weighting*. If \mathcal{A} in Eqs. (3) and (4) is a diagonal matrix whose diagonal elements are restricted to either 0 or 1, the feature extraction method is also called *feature selection*; if the diagonal elements of diagonal matrix \mathcal{A} can take any real-valued number between 0 and 1, the feature extraction method is also called *feature weighting*.

Allowing diagonal elements of \mathcal{A} to take real-valued numbers, instead of binary ones, enables feature weighting to employ some well-established optimization techniques and thus allows for more efficient algorithm implementation than feature selection. A celebrated feature weighting method is the RELIEF algorithm [1].

One major shortcoming of feature selection and feature weighting is their inability to remove correlation among different feature dimensions so as to achieve a sparser representation of data samples [2]. In some applications, such as object recognition, where there is no need to preserve the physical meaning of individual features, feature extraction is more appropriate than feature selection and feature weighting. For example, Sun and Wu [3] [4] proposed a linear feature extraction method, called local feature extraction (LFE), which outperforms feature selection and feature weighting. In this paper, we extend LFE to a nonlinear one called Kernel LFE.

Third, a feature extraction method can be further categorized as a wrapper method or a filter method [5]. A wrapper method determines the mapping in Eq. (2) by minimizing the classification error rate of a classifier, whereas a filter method does not. Therefore, filter methods are computationally more efficient but usually perform worse than wrapper methods. The famous principal component analysis (PCA) [6, page 115] is a filter method, whereas RELIEF [1] and LFE [3] are wrapper methods, as they both optimize a 1-nearest-neighbor (1-NN) classifier [6, page 174]. Our KLFE algorithm is also a wrapper method in that it optimizes a 1-NN classifier in the nonlinear-transformed space.

Last, a feature extraction method can be obtained by solving a convex optimization problem or a non-convex optimization problem. A convex optimization problem formulation is preferred since it can be solved efficiently, compared to a non-convex optimization problem. PCA, RELIEF, and LFE are all based on convex optimization formulation. Our KLFE algorithm is also based on a convex optimization formulation, which admits a closed-form solution.

In summary, in this paper, we propose a nonlinear wrapper feature extraction method, which is based on a convex optimization formulation.

The remainder of the paper is organized as follows. In Section 2, related work is briefly reviewed. In Section 3, we describe two existing feature extraction techniques, namely, RELIEF and LFE; our proposed KLFE is a generalization of these two algorithms. In Section 4, we present a novel feature extraction algorithm, called KLFE. Section 5 presents experimental results, which demonstrate that KLFE outperforms the existing feature extraction algorithms. Section 6 concludes the paper.

2. Related Work

In this section, we briefly review some feature extraction algorithms, which will be compared to our KLFE algorithm. Principal component analysis (PCA) [6, page 115] is probably one of the most commonly used algorithms for feature extraction. One major drawback of PCA, however, is that it is targeted at minimizing mean squared error for data compression or efficient data representation, rather than minimizing the classification error probability for pattern classification. Other PCA-type algorithms, e.g., kernel PCA (KPCA) [7], usually perform better than PCA in representing nonlinear

relationship among different feature dimensions, but suffer from the same limitation as PCA since they do not use class labels in the training phase; i.e., they are unsupervised algorithms. The KLFE algorithm proposed in this paper, like its predecessors LFE and RELIEF, utilizes the class label information in the training phase; i.e., KLFE is an supervised algorithm.

Among the existing feature weighting techniques, RELIEF [1] is considered as one of the best-performed ones due to its simplicity and effectiveness [8]. RELIEF determines the parameters of diagonal matrix \mathcal{A} in Eq. (3) by solving a convex optimization problem, which maximizes a margin-based criterion [9]. The LFE algorithm, proposed in Ref. [3], is an extension to RELIEF; LFE removes the constraint of matrix \mathcal{A} in Eq. (3) being diagonal, which is required by RELIEF. Both LFE and RELIEF are linear methods. In contrast, our KLFE method is a nonlinear extension to LFE; experimental results show that KLFE performs better than LFE.

In Ref. [10], the authors extended RELIEF to a kernel space, which is the space that contains the image of the nonlinear-transformation used in a kernel method; their approach is to identify an orthonormal basis of the kernel space and perform RELIEF in this kernel space; the resulting schemes are called Feature Space KPCA (FSKPCA) and Feature Space Kernel Gram-Schmidt Process (FSKGP). They showed that FSKPCA and FSKGP achieve similar performance to that of the state-of-the-art algorithms. Our KLFE adopts a similar strategy, i.e., our KLFE first computes an orthonormal basis of the kernel space and then performs LFE in the kernel space. Since LFE achieves improved performance over RELIEF, KLFE is expected to outperform FSKPCA and FSKGP, which are kernel-based versions of RELIEF.

In the next section, we briefly review RELIEF and LFE before we present our KLFE in Section 4.

3. Review of RELIEF and LFE

In this section, we briefly review RELIEF and LFE since LFE is an extension of RELIEF and KLFE is an extension of LFE. We first define two terms, nearest hit (NH) and nearest miss (NM) in Section 3.1, which will be used in all of the three algorithms, i.e., RELIEF, LFE, and KLFE. Then we introduce RELIEF and LFE in Sections 3.2 and 3.3, respectively.

3.1. Nearest Hit (NH) and Nearest Miss (NM)

Suppose we are given a training data set, as shown in Eq. (1). For any pattern $(\mathbf{x}, y) \in \mathcal{D}$, we define its nearest hit (NH) as

$$NH(\mathbf{x}, y) \triangleq \arg \min_{\mathbf{x}'} \|\mathbf{x}' - \mathbf{x}\|_p \quad (5)$$

$$s.t. (\mathbf{x}', y') \in \mathcal{D}, \quad (6)$$

$$y' = y, \quad (7)$$

and its nearest miss (NM) as

$$NM(\mathbf{x}, y) \triangleq \arg \min_{\mathbf{x}'} \|\mathbf{x}' - \mathbf{x}\|_p \quad (8)$$

$$s.t. (\mathbf{x}', y') \in \mathcal{D}, \quad (9)$$

$$y' \neq y, \quad (10)$$

where $\|\mathbf{x}\|_p$ is L^p -norm of vector \mathbf{x} . In this paper, we let $p = 1$, i.e., we choose L^1 norm.

Using Eqs. (5) and (8), we further denote

$$\mathbf{m}_n \triangleq \mathbf{x}_n - NM(\mathbf{x}_n, y_n), \quad (11)$$

$$\mathbf{h}_n \triangleq \mathbf{x}_n - NH(\mathbf{x}_n, y_n), \quad (12)$$

for $n = 1, \dots, N$.

3.2. RELIEF

Now we briefly introduce RELIEF. Denote by $\mathbf{w} = [w_1, \dots, w_I]^T$ the weight vector, where w_i is the weight of the i -th dimension of $\mathbf{x}_n \in \mathbb{R}^I$. RELIEF defines the margin of a pattern $(\mathbf{x}_n, y_n) \in \mathcal{D}$ as

$$\rho_n \triangleq \|\mathbf{m}_n\|_1 - \|\mathbf{h}_n\|_1, \quad n = 1, \dots, N. \quad (13)$$

Then the objective of RELIEF is to maximize the overall margin over weight vector, i.e.,

$$\max_{\mathbf{w}} \sum_{n=1}^N \rho_n(\mathbf{w}) = \sum_{n=1}^N (\mathbf{w}^T |\mathbf{m}_n| - \mathbf{w}^T |\mathbf{h}_n|), \quad (14a)$$

$$s.t. \quad \|\mathbf{w}\|_2^2 = 1, \mathbf{w} \geq 0, \quad (14b)$$

where $|\cdot|$ denotes element-wise absolute operator. Let

$$\mathbf{z} \triangleq \sum_{n=1}^N (|\mathbf{m}_n| - |\mathbf{h}_n|), \quad (15)$$

we simplify Eq. (14) to

$$\max_{\mathbf{w}} \quad \mathbf{w}^T \mathbf{z}, \quad s.t. \quad \|\mathbf{w}\|_2^2 = 1, \mathbf{w} \geq 0. \quad (16)$$

Applying Lagrangian multipliers λ and θ to Eq. (16), one obtains

$$L = -\mathbf{w}^T \mathbf{z} + \lambda (\|\mathbf{w}\|_2^2 - 1) + \mathbf{w}^T \theta. \quad (17)$$

Taking derivative with respect to \mathbf{w} at both sides of Eq. (17) and setting it to zero results in

$$\begin{aligned} \frac{\partial L}{\partial \mathbf{w}} &= -\mathbf{z} + 2\lambda \mathbf{w} - \theta = 0 \\ \Rightarrow \quad \mathbf{w} &= \frac{1}{2\lambda} (\mathbf{z} + \theta). \end{aligned} \quad (18)$$

The closed-form solution to Eq. (14) is [9]

$$\mathbf{w} = \frac{(\mathbf{z})^+}{\|(\mathbf{z})^+\|_2}, \quad (19)$$

where $(\mathbf{z})^+ = [(z_1)^+, \dots, (z_I)^+]^T$ and $(\cdot)^+ = \max(\cdot, 0)$. The RELIEF algorithm specifies the projection matrix

$$\mathcal{A} = \begin{bmatrix} w_1 & & 0 \\ & \ddots & \\ 0 & & w_I \end{bmatrix}.$$

3.3. LFE

A natural extension of RELIEF is to use a full matrix instead of a diagonal matrix, which results in LFE [3]. In LFE, the following optimization problem is considered:

$$\max_{\mathbf{W}} \sum_{n=1}^N \rho_n(\mathbf{W}) = \sum_{n=1}^N \mathbf{m}_n^T \mathbf{W} \mathbf{m}_n - \sum_{n=1}^N \mathbf{h}_n^T \mathbf{W} \mathbf{h}_n, \quad (20a)$$

$$\text{s.t.} \quad \|\mathbf{W}\|_F^2 = 1, \mathbf{W} \geq 0, \quad (20b)$$

where $\|\mathbf{W}\|_F$ is the Frobenius norm of \mathbf{W} , i.e., $\|\mathbf{W}\|_F = \sqrt{\sum_{i,j} w_{i,j}^2} = \sqrt{\sum_i \lambda_i^2}$, where $\{\lambda_i\}_{i=1}^I$ are the eigenvalues of \mathbf{W} .

Sun and Wu [3] proved Theorem 1, which provides a solution to Eq. (20).

Theorem 1. *Let*

$$\Sigma_{mh} \triangleq \sum_{n=1}^N \mathbf{m}_n \mathbf{m}_n^T - \sum_{n=1}^N \mathbf{h}_n \mathbf{h}_n^T, \quad (21)$$

and let $\{(\sigma_i, \mathbf{a}_i)\}_{i=1}^I$ be the eigen-system of Σ_{mh} such that $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_I$. The solution for Eq. (20), up to the difference of a constant, is

$$\mathbf{W} = \sum_{\{i:\sigma_i>0\}} \sigma_i \mathbf{a}_i \mathbf{a}_i^T. \quad (22)$$

According to Theorem 1, the LFE algorithm produces a projection matrix by maintaining the dimensions specified by eigenvectors $\{\mathbf{a}_i\}_{i=1}^{I'}$, which correspond to the largest I' eigenvalues of Σ_{mh} where $I' \leq I$ is the target dimension size as defined in Eq. (3) and I' should be chosen such that $\sigma_1 \geq \dots \geq \sigma_{I'} \geq 0$; in other words, LFE is defined by

$$f(\mathbf{x}) = \mathcal{A} \mathbf{x}, \mathbf{x} \in \mathcal{X}, \quad (23)$$

where

$$\mathcal{A} = [\sqrt{\sigma_1} \mathbf{a}_1, \dots, \sqrt{\sigma_{I'}} \mathbf{a}_{I'}]^T. \quad (24)$$

4. Kernel LFE

In this section, we propose KLFE algorithm. This section is organized as follows. In Section 4.1, we present the extension of LFE to a high-dimensional space by introducing a nonlinear mapping. In Section 4.2, we prove that both LFE and KLFE are basis rotation invariant and thus KLFE can be considered to perform LFE under an orthonormal basis in the kernel space. In Section 4.3, we present an algorithm for implementing KLFE by using KPCA to find an orthonormal basis in the kernel space. We summarize KLFE algorithm and describe the implementation details in Section 4.4. Computational complexity is also analyzed in this section.

4.1. Nonlinear LFE in High-Dimensional Space

As presented in Section 3.3, the LFE algorithm is a linear feature extraction method as in Eq. (3). Our idea of extending LFE is to introduce a nonlinear function,

$$\varphi : \mathcal{X} \subset \mathbb{R}^I \rightarrow \bar{\mathcal{X}} \subset \mathbb{R}^{\bar{I}}, \quad (25)$$

where usually $\bar{I} \gg I$, which maps patterns from a low-dimensional space to a high-dimensional one. We call $\bar{\mathcal{X}}$ or $\mathbb{R}^{\bar{I}}$ *kernel space*, which contains the image of φ . Then we apply the LFE algorithm to kernel space $\bar{\mathcal{X}}$; the resulting algorithm is called KLFE. We further assume $\bar{I} \gg N$, as is always the case when using a kernel method. Similar to Eqs. (11) and (12), we let

$$\bar{\mathbf{m}}_n \triangleq \varphi(\mathbf{x}_n) - \varphi(NM(\mathbf{x}_n, y_n)), \quad (26)$$

$$\bar{\mathbf{h}}_n \triangleq \varphi(\mathbf{x}_n) - \varphi(NH(\mathbf{x}_n, y_n)), \quad (27)$$

$n = 1, \dots, N$. KLFE can be obtained by solving the following optimization problem,

$$\max_{\bar{\mathbf{W}}} \sum_{n=1}^N \bar{\mathbf{m}}_n^T \bar{\mathbf{W}} \bar{\mathbf{m}}_n - \sum_{n=1}^N \bar{\mathbf{h}}_n^T \bar{\mathbf{W}} \bar{\mathbf{h}}_n, \quad (28a)$$

$$\text{s.t.} \quad \|\bar{\mathbf{W}}\|_F^2 = 1, \bar{\mathbf{W}} \geq 0. \quad (28b)$$

Since the only difference between Eq. (28) and Eq. (20) is the use of mapping φ , one can directly use Theorem 1 to solve Eq. (28). Corollary 1.1 summarizes the solution to Eq. (28).

Corollary 1.1. *Let*

$$\bar{\Sigma}_{mh} \triangleq \sum_{n=1}^N \bar{\mathbf{m}}_n \bar{\mathbf{m}}_n^T - \sum_{n=1}^N \bar{\mathbf{h}}_n \bar{\mathbf{h}}_n^T, \quad (29)$$

and let $\{(\bar{\sigma}_i, \bar{\mathbf{a}}_i)\}_{i=1}^{\bar{I}}$ be the eigen-system of $\bar{\Sigma}_{mh}$, such that $\bar{\sigma}_1 \geq \bar{\sigma}_2 \geq \dots \geq \bar{\sigma}_{\bar{I}}$. The solution to Eq. (28), up to the difference of a constant, is

$$\bar{\mathbf{W}} = \sum_{\{i:\bar{\sigma}_i>0\}} \bar{\sigma}_i \bar{\mathbf{a}}_i \bar{\mathbf{a}}_i^T.$$

From Corollary 1.1, the KLFE algorithm produces a projection matrix by maintaining the dimensions specified by eigenvectors $\{\bar{\sigma}_i\}_{i=1}^{I'}$, which correspond to the largest I' eigenvalues of Σ_{mh} where $I' \leq \bar{I}$ is the target dimension size as defined in Eq. (3) and I' should be chosen such that $\bar{\sigma}_1 \geq \dots \geq \bar{\sigma}_{I'} \geq 0$; in other words, KLFE is defined by

$$f(\mathbf{x}) = \bar{\mathcal{A}}\varphi(\mathbf{x}), \mathbf{x} \in \mathcal{X}, \quad (30)$$

where

$$\bar{\mathcal{A}} = [\sqrt{\bar{\sigma}_1}\bar{\mathbf{a}}_1, \dots, \sqrt{\bar{\sigma}_{I'}}\bar{\mathbf{a}}_{I'}]^T. \quad (31)$$

4.2. Basis Rotation Invariant Property of KLFE

In this section, we study an important property of KLFE: basis rotation invariance. Before we show the basis rotation invariant property of KLFE in Proposition 1, we present Lemma 1.

Lemma 1 states that LFE is basis rotation invariant.

Lemma 1. *Let $\{\mathbf{e}_1^{(i)}\}_{i=1}^I$ and $\{\mathbf{e}_2^{(i)}\}_{i=1}^I$ be two different orthonormal bases in input feature space \mathbb{R}^I . Assume that $\{\mathbf{e}_2^{(i)}\}_{i=1}^I$ can be obtained by counterclockwise rotating $\{\mathbf{e}_1^{(i)}\}_{i=1}^I$, and the rotation matrix is denoted by Q . Then, LFE is basis rotation invariant, i.e., a feature vector extracted by LFE under $\{\mathbf{e}_1^{(i)}\}_{i=1}^I$ is the same as the feature vector extracted by LFE under $\{\mathbf{e}_2^{(i)}\}_{i=1}^I$.*

Proof. Assume training samples are given under basis $\{\mathbf{e}_1^{(i)}\}_{i=1}^I$, i.e.,

$$\mathcal{D} \triangleq \{(\mathbf{x}_n, y_n)\}_{n=1}^N \subset \mathcal{X} \times \mathcal{Y},$$

where $\mathcal{X} \subset \mathbb{R}^I$ is the I -dimensional feature space and $\mathcal{Y} = \{\pm 1\}$. Under basis $\{\mathbf{e}_1^{(i)}\}_{i=1}^I$, LFE is formulated as follows:

$$\max_{\mathbf{W}_1} \sum_{n=1}^N \rho_n(\mathbf{W}_1) = \sum_{n=1}^N \mathbf{m}_n^T \mathbf{W}_1 \mathbf{m}_n - \sum_{n=1}^N \mathbf{h}_n^T \mathbf{W}_1 \mathbf{h}_n, \quad (32a)$$

$$\text{s.t.} \quad \|\mathbf{W}_1\|_F^2 = 1, \mathbf{W}_1 \geq 0. \quad (32b)$$

From Theorem 1, the projection matrix of LFE under basis $\{\mathbf{e}_1^{(i)}\}_{i=1}^I$ is given by

$$\mathcal{A}_1 = [\sqrt{\sigma_1}\mathbf{a}_1, \dots, \sqrt{\sigma_{I'}}\mathbf{a}_{I'}]^T, \quad (33)$$

where $I' \leq I$ is the target dimension size and $\{\mathbf{a}_i\}$ are the eigenvectors of Σ_{mh} corresponding to the largest I' eigenvalues.

Under basis $\{\mathbf{e}_2^{(i)}\}_{i=1}^I$, the training samples become

$$\mathcal{D} \triangleq \{(Q\mathbf{x}_n, y_n)\}$$

The nearest miss and nearest hit remain the same for each data sample since the Euclidean distance does not change under different orthonormal bases. Under basis $\{\mathbf{e}_2^{(i)}\}_{i=1}^I$, LFE is formulated as below:

$$\begin{aligned} \max_{\mathbf{W}_2} \quad & \sum_{n=1}^N \rho_n(\mathbf{W}_2) = \sum_{n=1}^N (\mathbf{Q}\mathbf{m}_n)^T \mathbf{W}_2 \mathbf{Q}\mathbf{m}_n - \sum_{n=1}^N (\mathbf{Q}\mathbf{h}_n)^T \mathbf{W}_2 \mathbf{Q}\mathbf{h}_n, \quad (34a) \\ \text{s.t.} \quad & \|\mathbf{W}_2\|_F^2 = 1, \mathbf{W}_2 \geq 0. \quad (34b) \end{aligned}$$

Comparing (32) and (34), we have $\mathbf{W}_1 = \mathbf{Q}^T \mathbf{W}_2 \mathbf{Q}$. Then we have

$$\mathbf{W}_2 = \mathbf{Q} \mathbf{W}_1 \mathbf{Q}^T \quad (35)$$

$$\stackrel{(a)}{=} \mathbf{Q} \left(\sum_{\{i:\sigma_i>0\}} \sigma_i \mathbf{a}_i \mathbf{a}_i^T \right) \mathbf{Q}^T \quad (36)$$

$$= \sum_{\{i:\sigma_i>0\}} \sigma_i \mathbf{Q} \mathbf{a}_i \mathbf{a}_i^T \mathbf{Q}^T \quad (37)$$

where (a) is due to (22). Let the projection matrix of LFE under basis $\{\mathbf{e}_2^{(i)}\}_{i=1}^I$, be \mathcal{A}_2 . Then, from (37), (22), and (24), we have

$$\mathcal{A}_2 = [\sqrt{\sigma_1} \mathbf{Q} \mathbf{a}_1, \dots, \sqrt{\sigma_{I'}} \mathbf{Q} \mathbf{a}_{I'}]^T \quad (38)$$

$$\stackrel{(a)}{=} \mathcal{A}_1 \mathbf{Q}^T \quad (39)$$

where (a) is due to (33). Then, the feature vector extracted by LFE under $\{\mathbf{e}_2^{(i)}\}_{i=1}^I$ is given by

$$\mathcal{A}_2 \mathbf{Q} \mathbf{x} \stackrel{(a)}{=} \mathcal{A}_1 \mathbf{Q}^T \mathbf{Q} \mathbf{x} \quad (40)$$

$$\stackrel{(b)}{=} \mathcal{A}_1 \mathbf{x} \quad (41)$$

where (a) is due to (39); (b) is due to the fact that \mathbf{Q} is an orthogonal matrix. Eq. (41) means that a feature vector extracted by LFE under $\{\mathbf{e}_1^{(i)}\}_{i=1}^I$, i.e., $\mathcal{A}_1 \mathbf{x}$ is the same as the feature vector extracted by LFE under $\{\mathbf{e}_2^{(i)}\}_{i=1}^I$, i.e., $\mathcal{A}_2 \mathbf{Q} \mathbf{x}$. This completes the proof. \square

Usually we do not know the dimension of the kernel subspace that contains the mapped data samples (including training and test data samples), but given sufficient number of training samples, we can estimate the dimension of this kernel subspace. Assume the rank of the mapped training data samples $\{\varphi(\mathbf{x}_n)\}_{n=1}^N$ is N_t , i.e., the mapped training data samples are contained in an N_t -dimensional kernel subspace denoted by \mathcal{S} . Suppose the kernel space $\bar{\mathcal{X}}$ has an orthonormal basis $\{\mathbf{e}^{(i)}\}_{i=1}^{\bar{I}}$.

Proposition 1 states that KLFE is basis rotation invariant for bases in the kernel space.

Proposition 1. *Let $\{\mathbf{e}_1^{(i)}\}_{i=1}^{\bar{I}}$ and $\{\mathbf{e}_2^{(i)}\}_{i=1}^{\bar{I}}$ be two orthonormal bases in kernel space. Assume that $\{\mathbf{e}_2^{(i)}\}_{i=1}^{\bar{I}}$ can be obtained by counterclockwise rotating $\{\mathbf{e}_1^{(i)}\}_{i=1}^{\bar{I}}$, and the*

rotation matrix is denoted by Q . Then, KLFE is basis rotation invariant for all samples \mathbf{x} where $\varphi(\mathbf{x}) \in \mathcal{S}$, i.e., a feature vector extracted by KLFE under $\{\mathbf{e}_1^{(i)}\}_{i=1}^{\bar{I}}$ for input sample \mathbf{x} is the same as the feature vector extracted by KLFE under $\{\mathbf{e}_2^{(i)}\}_{i=1}^{\bar{I}}$.

Proof. Assume the training sample set is

$$\mathcal{D} \triangleq \{(\mathbf{x}_n, y_n)\}_{n=1}^N \subset \mathcal{X} \times \mathcal{Y},$$

where $\mathcal{X} \subset \mathbb{R}^I$ is the I -dimensional feature space and $\mathcal{Y} = \{\pm 1\}$.

In KLFE, a sample is first mapped from a low-dimensional space to a high-dimensional space by the following nonlinear transformation

$$\varphi : \mathcal{X} \subset \mathbb{R}^I \rightarrow \bar{\mathcal{X}} \subset \mathbb{R}^{\bar{I}}. \quad (42)$$

The training sample set in the kernel space under basis $\{\mathbf{e}_1^{(i)}\}_{i=1}^{\bar{I}}$ is denoted by

$$\mathcal{D}_1 \triangleq \{(\varphi(\mathbf{x}_n), y_n)\}_{n=1}^N \quad (43)$$

Under basis $\{\mathbf{e}_2^{(i)}\}_{i=1}^{\bar{I}}$, the training sample set in the kernel space becomes

$$\mathcal{D}_2 \triangleq \{(Q\varphi(\mathbf{x}_n), y_n)\}_{n=1}^N$$

Denote the projection matrix of KLFE in (30) under basis $\{\mathbf{e}_1^{(i)}\}_{i=1}^{\bar{I}}$ and $\{\mathbf{e}_2^{(i)}\}_{i=1}^{\bar{I}}$ by $\bar{\mathcal{A}}_1$ and $\bar{\mathcal{A}}_2$, respectively. Note that the dimension of the feature vector extracted by KLFE is I' , where $I' < \bar{I}$. Since KLFE is equivalent to applying LFE in the kernel space, from Lemma 1, we have

$$\bar{\mathcal{A}}_1 \varphi(\mathbf{x}) = \bar{\mathcal{A}}_2 Q \varphi(\mathbf{x}) \quad (44)$$

This completes the proof. \square

Proposition 2. Let $\{\mathbf{e}_1^{(i)}\}_{i=1}^{\bar{I}}$ be an orthonormal basis in kernel space. Denote by \mathcal{S} the kernel subspace spanned by training data $\{\varphi(\mathbf{x}_n)\}_{n=1}^N$; the dimension of \mathcal{S} is N_t . Then an \bar{I} -dimensional feature vector $f(x)$ extracted by KLFE under $\{\mathbf{e}_1^{(i)}\}_{i=1}^{\bar{I}}$ for input sample \mathbf{x} (where $\varphi(\mathbf{x}) \in \mathcal{S}$) must be in the form of

$$f(x) = \begin{bmatrix} f_1(x) \\ \mathbf{0}_{\bar{I}-N_t, 1} \end{bmatrix} \quad (45)$$

where $f_1(x)$ is an N_t -dimensional vector and $\mathbf{0}_{\bar{I}-N_t, 1}$ is an $(\bar{I} - N_t)$ -dimensional vector whose entries are all zero. In other words, the last $\bar{I} - N_t$ entries of the extracted feature vector $f(x)$ are all zero.

Proof. Assume the training sample set is

$$\mathcal{D} \triangleq \{(\mathbf{x}_n, y_n)\}_{n=1}^N \subset \mathcal{X} \times \mathcal{Y},$$

where $\mathcal{X} \subset \mathbb{R}^I$ is the I -dimensional feature space and $\mathcal{Y} = \{\pm 1\}$.

After nonlinear mapping, the training sample set in the kernel space under basis $\{\mathbf{e}_1^{(i)}\}_{i=1}^{\bar{I}}$ becomes

$$\mathcal{D}_1 \triangleq \{(\varphi(\mathbf{x}_n), y_n)\}_{n=1}^N \quad (46)$$

Let $\{\mathbf{e}_2^{(i)}\}_{i=1}^{N_t}$ be an orthonormal basis for \mathcal{S} . Denote by \mathcal{S}^\perp the complementary subspace of \mathcal{S} . Let $\{\mathbf{e}_3^{(i)}\}_{i=N_t+1}^{\bar{I}}$ be an orthonormal basis for \mathcal{S}^\perp . Thus $\{\mathbf{e}_2^{(i)}\}_{i=1}^{N_t} \cup \{\mathbf{e}_3^{(i)}\}_{i=N_t+1}^{\bar{I}}$ form an orthonormal basis for kernel space.

From Proposition 1, the feature vector extracted by KLFE algorithm under basis $\{\mathbf{e}_1^{(i)}\}_{i=1}^{\bar{I}}$ is the same as that extracted by KLFE under a rotated basis $\{\mathbf{e}_2^{(i)}\}_{i=1}^{N_t} \cup \{\mathbf{e}_3^{(i)}\}_{i=N_t+1}^{\bar{I}}$. So we can compute the extracted feature vector under basis $\{\mathbf{e}_2^{(i)}\}_{i=1}^{N_t} \cup \{\mathbf{e}_3^{(i)}\}_{i=N_t+1}^{\bar{I}}$ for simplicity.

Since $\varphi(\mathbf{x}) \in \mathcal{S}$, hence the last $\bar{I} - N_t$ entries of the coordinates of $\varphi(\mathbf{x})$ under basis $\{\mathbf{e}_2^{(i)}\}_{i=1}^{N_t} \cup \{\mathbf{e}_3^{(i)}\}_{i=N_t+1}^{\bar{I}}$ are all zero. From the definition of $\bar{\Sigma}_{mh}$ in Eq. (29), we have

$$\bar{\Sigma}_{mh} = \begin{bmatrix} \bar{\Sigma}_{mh}^* & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \quad (47)$$

where $\bar{\Sigma}_{mh}$ is an $\bar{I} \times \bar{I}$ matrix, and $\bar{\Sigma}_{mh}^*$ is an $N_t \times N_t$ matrix.

Let $\bar{\mathcal{A}}$ denote the projection matrix of KLFE in (30) under basis $\{\mathbf{e}_2^{(i)}\}_{i=1}^{N_t} \cup \{\mathbf{e}_3^{(i)}\}_{i=N_t+1}^{\bar{I}}$. From Corollary 1.1, Eq. (30) and Eq. (47), we have

$$\bar{\mathcal{A}} = \begin{bmatrix} \bar{\mathcal{A}}_1 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \quad (48)$$

where

$$\bar{\mathcal{A}}_1 = [\sqrt{\bar{\sigma}_1} \bar{\mathbf{a}}_1, \dots, \sqrt{\bar{\sigma}_{N_t}} \bar{\mathbf{a}}_{N_t}]^T, \quad (49)$$

and $\{(\bar{\sigma}_i, \bar{\mathbf{a}}_i)\}_{i=1}^{N_t}$ is the eigen-system of $\bar{\Sigma}_{mh}^*$.

From (30), the extracted feature vector $f(x)$ is given by

$$f(x) = \bar{\mathcal{A}}\varphi(\mathbf{x}) \quad (50)$$

$$= \begin{bmatrix} \bar{\mathcal{A}}_1 \varphi^*(\mathbf{x}) \\ \mathbf{0} \end{bmatrix} \quad (51)$$

where

$$\varphi(\mathbf{x}) = \begin{bmatrix} \varphi^*(\mathbf{x}) \\ \mathbf{0} \end{bmatrix} \quad (52)$$

This completes the proof. \square

It is worth mentioning that if $\varphi(\mathbf{x}) \notin \mathcal{S}$, then the distance $d(\varphi(\mathbf{x}), \varphi'(\mathbf{x}))$ is negligible where $\varphi'(\mathbf{x})$ is the projection of $\varphi(\mathbf{x})$ onto \mathcal{S} . The reason is that if the training data $\{\mathbf{x}_n\}_{n=1}^N$ and the test data \mathbf{x} are sampled from the same distribution, $\|\varphi(\mathbf{x}) - \varphi'(\mathbf{x})\|$ is mainly caused by irrelevant features or measurement noise [10].

From Proposition 2, we can perform feature extraction in kernel subspace \mathcal{S} in which the basis can be expressed by linear combinations of mapped data samples in

the kernel space. In this way, we can simplify the computation involved in KLFE. Proposition 2 shows that KLFE can extract at most N_t dimensional nonzero feature vector for arbitrary input sample which lies in \mathcal{S} .

Based on the two propositions, KLFE can be computed in three steps. First, we find a basis in kernel subspace. This can be done by using KPCA or Kernel Gram-Schmidt Procedure (KGP) [10]; the dimension of the basis is equal to the rank of the mapped data set in kernel space. Second, the data in the kernel space can be mapped onto the basis, each basis vector of which is a linear combination of the mapped data $\{\varphi(\mathbf{x})\}$, i.e., $v^{(i)} = \sum_j \alpha_{ij} \varphi(\mathbf{x}^{(j)})$. Note that the kernel method can be used to obtain the kernel feature under the basis. Third, we perform LFE on the resulting kernel features which have dimension of N_t .

Next, we present the final KLFE algorithm by using KPCA to find a basis in kernel subspace.

4.3. KLFE using KPCA

For a given φ , its kernel function, $\mathcal{K} : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$, is defined as

$$\mathcal{K}(\mathbf{x}_1, \mathbf{x}_2) = \langle \varphi(\mathbf{x}_1), \varphi(\mathbf{x}_2) \rangle, \quad (53)$$

where $\langle \cdot, \cdot \rangle$ represents inner-product operator. It is known that \mathcal{K} and φ have 1-to-1 mapping [11]. In other words, we can ignore the explicit form of φ by using a given \mathcal{K} directly, as long as all computations are conducted through inner product.

Without loss of generality, assume the average of the data samples in kernel space is zero. Let $K \triangleq \bar{X}^T \bar{X}$ be the kernel matrix, where matrix $\bar{X} \triangleq [\varphi(\mathbf{x}_1), \varphi(\mathbf{x}_2), \dots, \varphi(\mathbf{x}_N)]$. Hence the entry of i -th row, j -th column in K is given by

$$K_{i,j} = \langle \varphi(\mathbf{x}_i), \varphi(\mathbf{x}_j) \rangle = \mathcal{K}(\mathbf{x}_i, \mathbf{x}_j), \quad i = 1, \dots, N; j = 1, \dots, N. \quad (54)$$

Let $\{\gamma_n, \mathbf{v}_n\}_{n=1}^N$ be the eigen-system of K , where the eigenvalues are sorted in decreasing order, i.e., $\gamma_1 \geq \gamma_2 \geq \dots \geq \gamma_N$. Then, by definition of eigenvalue decomposition,

$$\bar{X}^T \bar{X} \mathbf{v}_n = \gamma_n \mathbf{v}_n, \quad (55)$$

$$\bar{X} \bar{X}^T (\bar{X} \mathbf{v}_n) = \gamma_n (\bar{X} \mathbf{v}_n), \quad (56)$$

for $n = 1, \dots, N$. As a result, $\{\gamma_n\}_{n=1}^N$ are the N largest eigenvalues of $\bar{X} \bar{X}^T$, whose corresponding eigenvectors are $\{\bar{X} \mathbf{v}_n\}_{n=1}^N$. Denote the dimension of matrix K by N' . If $N' = N$, then $\gamma_i > 0$ for $i = 1, 2, \dots, N$. If $N' < N$, we perform LFE in the kernel subspace of dimension N' .

Normalizing eigenvectors $\{\bar{X} \mathbf{v}_n\}_{n=1}^N$ produces an orthonormal basis

$$\Psi = \bar{X} \begin{bmatrix} \frac{\mathbf{v}_1}{\sqrt{\gamma_1}}, \dots, \frac{\mathbf{v}_N}{\sqrt{\gamma_N}} \end{bmatrix} \triangleq \bar{X} \mathbf{V}'. \quad (57)$$

Thus for input vector \mathbf{x}_n ($n = 1, \dots, N$), the feature vector extracted by KLFE under the basis in (57) is given by

$$\begin{aligned}\tilde{\mathbf{x}}_n &= \mathbf{\Psi}^T \varphi(\mathbf{x}_n) = \begin{bmatrix} \mathbf{v}_1^T / \sqrt{\gamma_1} \\ \vdots \\ \mathbf{v}_N^T / \sqrt{\gamma_N} \end{bmatrix} \bar{X}^T \varphi(\mathbf{x}_n) \\ &= \mathbf{V}'^T \bar{X}^T \varphi(\mathbf{x}_n) = \mathbf{V}'^T K^{(n)},\end{aligned}\quad (58)$$

where $K^{(n)}$ denotes the n -th column of kernel matrix K . More generally, for any $\mathbf{x} \in \mathcal{X}$, we have

$$\tilde{\mathbf{x}} = \mathbf{V}'^T \begin{bmatrix} \mathcal{K}(\mathbf{x}_1, \mathbf{x}) \\ \vdots \\ \mathcal{K}(\mathbf{x}_N, \mathbf{x}) \end{bmatrix} \triangleq \tilde{\varphi}(\mathbf{x}).\quad (59)$$

Eq. (59) actually specifies an N -dimensional kernel space,

$$\tilde{\mathcal{X}} = \{\tilde{\varphi}(\mathbf{x}) : \mathbf{x} \in \mathcal{X}\}\quad (60)$$

We summarize KPCA-based KLFE algorithm as follows. Let

$$\tilde{\mathbf{m}}_n = \tilde{\varphi}(\mathbf{x}_n) - \tilde{\varphi}(NM(\mathbf{x}_n, y_n)),\quad (61)$$

$$\tilde{\mathbf{h}}_n = \tilde{\varphi}(\mathbf{x}_n) - \tilde{\varphi}(NH(\mathbf{x}_n, y_n)),\quad (62)$$

$n = 1, \dots, N$. The KLFE algorithm solves the following optimization problem,

$$\max_{\tilde{\mathbf{W}}} \sum_{n=1}^N \tilde{\mathbf{m}}_n^T \tilde{\mathbf{W}} \tilde{\mathbf{m}}_n - \sum_{n=1}^N \tilde{\mathbf{h}}_n^T \tilde{\mathbf{W}} \tilde{\mathbf{h}}_n,\quad (63a)$$

$$\text{s.t.} \quad \|\tilde{\mathbf{W}}\|_F^2 = 1, \tilde{\mathbf{W}} \geq 0.\quad (63b)$$

Using the result in Theorem 1, the solution to Eq. (63) is given in Theorem 2.

Theorem 2. *Let*

$$\tilde{\Sigma}_{mh} = \sum_{n=1}^N \tilde{\mathbf{m}}_n \tilde{\mathbf{m}}_n^T - \sum_{n=1}^N \tilde{\mathbf{h}}_n \tilde{\mathbf{h}}_n^T\quad (64)$$

and let $\{(\tilde{\sigma}_i, \tilde{\mathbf{a}}_i)\}_{i=1}^N$ be the eigen-system of $\tilde{\Sigma}_{mh}$, such that $\tilde{\sigma}_1 \geq \dots \geq \tilde{\sigma}_N$. The solution to Eq. (63), up to the difference of a constant, is

$$\tilde{\mathbf{W}} = \sum_{\{n: \tilde{\sigma}_n > 0\}} \tilde{\sigma}_n \tilde{\mathbf{a}}_n \tilde{\mathbf{a}}_n^T.$$

Accordingly, the projection matrix is

$$\tilde{\mathbf{A}} = [\sqrt{\tilde{\sigma}_1} \tilde{\mathbf{a}}_1, \dots, \sqrt{\tilde{\sigma}_T} \tilde{\mathbf{a}}_T]^T.\quad (65)$$

For an input \mathbf{x} , the extracted feature is given by

$$f(\mathbf{x}) = \tilde{\mathbf{A}} \tilde{\varphi}(\mathbf{x}) = \tilde{\mathbf{A}} \mathbf{V}'^T \begin{bmatrix} \mathcal{K}(\mathbf{x}_1, \mathbf{x}) \\ \vdots \\ \mathcal{K}(\mathbf{x}_N, \mathbf{x}) \end{bmatrix}.\quad (66)$$

where \mathbf{V}' is defined in Eq. (57).

■

KLFE is superior to LFE in that it performs LFE in a high-dimensional space, where discriminant information is much easier to extract. From the above analysis, KLFE can be considered as KPCA followed by LFE. Therefore, KLFE is superior over KPCA since KLFE takes into account the label information; KLFE also outperforms kernel RELIEF, i.e., FSKPCA and FSKGP [10], where FSKPCA is KPCA followed by RELIEF.

4.4. KLFE Algorithm

Now the pseudo-code of KLFE is shown. In the initialization step, we need some parameters, like the number of neighbors for computing Σ_{mh} . Assuming that we use the RBF kernel and K-nearest-neighbors as classifier, we need the width of RBF kernel and the number of neighbors for KNN. In our experiments, we use 10-fold cross validation to find these parameters.

If we use complex tuning method to find better parameter set, the performance of KLFE will be improved. In this paper, we do not focus on complex tuning method or classification method. Therefore we use simple tuning method: 10-fold cross validation to tune all parameters needed. For each parameter, we use only 5-10 candidate points.

Algorithm KLFE

Input: Training samples $X = [x_1 \dots x_N]$ and labels $Y = [y_1 \dots y_N]$

1) Initialization

 Normalize X , give kernel parameter, number of neighbors L .

2) Mapping to kernel space

 2.1) $K = \text{kernel}(X)$, K is the kernel of X .

 2.2) $[V, D] = \text{EigenDecomposition}(K)$,

V 's column contains one principal component, D is a diagonal matrix with eigenvalues.
 All the zero values are removed.

 2.3) $\bar{X} = DV^T K$

3) LFE

 3.1) for $n = 1 : N$

ζ_i is the i^{th} nearest \bar{x}_i labeled the same class with \bar{x}

η_i is the i^{th} nearest \bar{x}_i labeled different class with \bar{x}

 Then $H_n = [\bar{x}_n - \zeta_1 \dots \bar{x}_n - \zeta_L]$, $M_n = [\bar{x}_n - \eta_1 \dots \bar{x}_n - \zeta_L]$

 3.2) $\Sigma_{mh} = \sum_{n=1}^N M_n M_n^T - \sum_{n=1}^N H_n H_n^T$

 3.3) $[\bar{V}, \bar{D}] = \text{EigenDecomposition}(\Sigma_{mh})$, the same as 2.2.

 3.4) $\tilde{X} = \bar{D}^{1/2} \bar{V}^T \bar{X}$

4) **Output:** \tilde{X} .

The complexity of KLFE depends on the kernel function. For example, consider the radial basis function (RBF) kernel [11, page 77], which is given by

$$\mathcal{K}(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}'\|^2}{2\rho^2}\right). \quad (67)$$



Figure 2: USPS handwritten digits 3(top row) and 5(bottom row).

Table 1: UCI and USPS data sets used in the experiments

data set	training sample size	testing sample size	number of features
UCI-diabetes	468	300	8
UCI-ringnorm	400	7000	20
USPS-(3vs.5)	326	1214	256

The complexity of computing kernel matrix, i.e., Eq. (54), is $\mathcal{O}(N^2I)$. The complexity of eigenvalue decomposition for kernel matrix and the complexity of LFE in the N -dimensional kernel space are both $\mathcal{O}(N^3)$. As a result, the overall complexity of KLFE using RBF kernel is

$$\mathcal{O}(N^2I) + \mathcal{O}(N^3). \quad (68)$$

It is comparable to the complexity of LFE, which is also $\mathcal{O}(N^2I) + \mathcal{O}(N^3)$ [3]. To reduce the computational complexity of KLFE, we can use Kernel Gram-Schmidt Procedure [10] to find a basis instead of using KPCA.

5. Experimental Results

In this section, we conduct experiments on pattern classification to show the performance of our KLFE algorithm and compare it with existing feature extraction schemes. This section is organized as follows. In Section 5.1, we describe the experimental setting. In Sections 5.2 and 5.3, we show the experimental results for simulated data sets and real-world data sets, respectively.

5.1. Experimental Setting

We conduct classification experiments on two types of data sets, namely, simulated data sets (sine-surface and Swiss roll) and real-world data sets (UCI Machine Learning Repository [12] and USPS digit handwriting data). In our experiments, we use two data sets from UCI Machine Learning Repository, i.e., data sets for diabetes, and ringnorm. For USPS data, we choose only two digits, namely "3 versus 5", since they are the most challenging digits for recognition. (See Fig. 2) We exchange the "traditional" training sets and testing sets as shown in Table 1.

To make a fair comparison, we compare KLFE with the following feature extraction schemes, which are also based on kernel. We also compare KLFE with origin LFE.

1. Generalized Discriminant Analysis (GDA) using a kernel approach [13]
2. KPCA

3. FSKPCA, which is one type of algorithm for kernel RELIEF
4. Kernel K-Nearest Neighbor (KKNN).

GDA can generate at most $n-1$ features where n is the number of categories/classes. In this paper, we only study feature extraction methods for binary classification problem. KKNN is kernelized K-nearest-neighbor (KNN) [6, page 174] based on a distance function induced by the kernel function.

Note that KLFE, GDA, KPCA and FSKPCA are feature extraction algorithms and we are interested in their classification capability, i.e., how well a given classifier performs if the classifier uses the features obtained from these feature extraction algorithms. In our experiments, we choose KNN as the classifier because of two reasons. First, KNN is a simple yet effective classifier, which often yields competitive results, compared to some advanced machine learning algorithms [14]. Second, the focus of this paper is not on an optimal classifier for each dataset. KNN is surely not an optimal classifier in many cases but it provides a platform where we can compare different feature extraction algorithms with a reasonable computational cost. Actually, for fair comparison, we let $K = 1$. Then we can explicitly see how these feature extraction algorithms improve classification ability of KNN in kernel space, i.e., KKNN.

In the experiments, we use RBF kernel function defined by Eq. (67) with $\sigma = 1$. Our comparison strategy is to use the same kernel function with the same width σ and the same classifier KNN where $K = 1$. To eliminate statistical variations, each algorithm is run several times for each data set. In each run, a data set is split into training data subset and testing data subset randomly. Then the testing error rate is obtained by averaging over all the runs.

5.2. Experimental Results for Simulated Data

In this section, we conduct experiments on two simulated data sets: twin sine and Swiss roll, which are both in the following form:

$$\mathcal{D} = \{(\mathbf{x}_n, y_n)\}_{n=1}^N,$$

where

$$y_n \in \{-1, 1\}, \quad (69)$$

$$\mathbf{x}_n = R_{I \times 3} \times \begin{bmatrix} \mathbf{x}_n^{(1)} \\ \mathbf{x}_n^{(2)} \\ \mathbf{x}_n^{(3)} \end{bmatrix}, \quad (70)$$

$$(71)$$

where $R_{I \times 3}$ denotes a random $I \times 3$ matrix.

For twin sine data, $\mathbf{x}_n^{(1)}$ is a random variable uniformly distributed in $[0, 2\pi]$; $\mathbf{x}_n^{(2)}$ is a random variable and $\mathbf{x}_n^{(2)} = \sin(\mathbf{x}_n^{(1)}) + \mathcal{I}(y_n = 1) \times D + \beta_N$, where $\mathcal{I}(\cdot)$ denotes an indicator function, D is a constant and β_N denotes a Gaussian random variable with zero mean and variance σ^2 ; $\mathbf{x}_n^{(3)}$ is a random variable uniformly distributed in $[0, 1]$. Actually, the data set is composed of two 3-dimensional sine surfaces, labeled

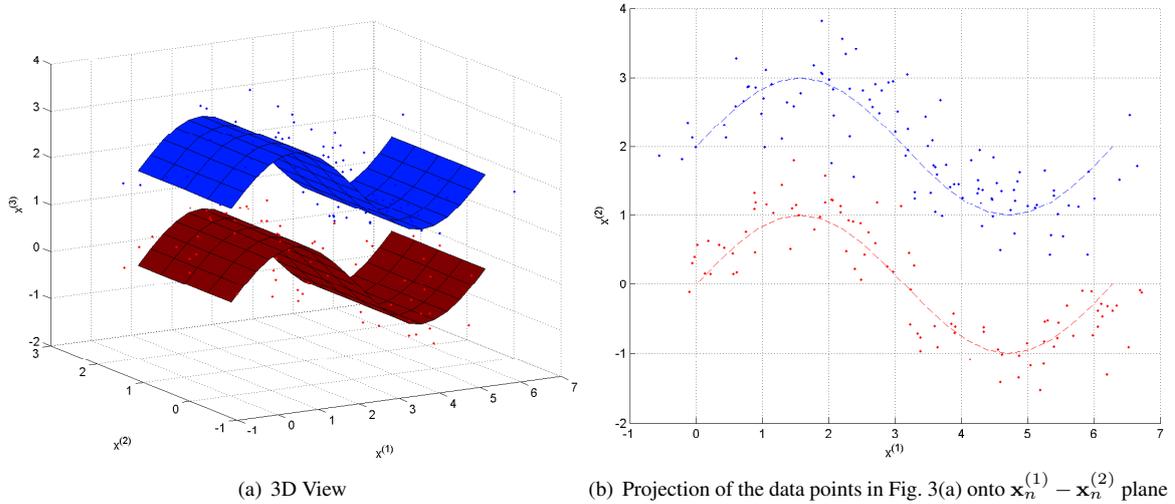


Figure 3: Simulated data set containing sine surfaces.

as -1 and 1 , with additive Gaussian noise β_N , and the two sine surfaces are separated apart by a distance D . Then the 3-dimensional vector $[\mathbf{x}_n^{(1)}, \mathbf{x}_n^{(2)}, \mathbf{x}_n^{(3)}]^T$ is mapped to a I -dimensional vector by matrix $R_{I \times 3}$.

Fig. 3(a) shows simulated 3-dimensional sine-surfaces and the data set of $[\mathbf{x}_n^{(1)}, \mathbf{x}_n^{(2)}, \mathbf{x}_n^{(3)}]^T$. Fig. 3(b) shows the projection of the data points in Fig. 3(a) onto $\mathbf{x}_n^{(1)} - \mathbf{x}_n^{(2)}$ plane.

We further consider the relationship among classification error rate, separation distance D , and noise variance σ^2 . It is obvious that, as D increases, the two sine-surfaces are further away from each other, resulting in better classification accuracy. Similarly, as σ^2 decreases, the probability that samples from two classes overlap decreases, which also increases the classification accuracy. Hence, we define signal-noise-rate (SNR) as

$$\text{SNR} = \frac{D^2}{\sigma^2} \quad (72)$$

$$\text{SNR (dB)} = 20 \log_{10} \left(\frac{D}{\sigma} \right). \quad (73)$$

Figs. 5(a) and 5(b) show classification error rate vs. target feature dimension I' for different schemes under $\text{SNR} = 0\text{dB}$ and -5dB , respectively.

For the Swiss roll data, we let $\mathbf{x}_n^{(1)} = \theta \times \cos(\theta)$ and $\mathbf{x}_n^{(2)} = \theta \times \sin(\theta)$, where θ is a random variable uniformly distributed in $[0, 4\pi]$. Actually the $\mathbf{x}_n^{(1)} - \mathbf{x}_n^{(2)}$ curve is a helix. $\mathbf{x}_n^{(3)}$ is a random variable uniformly distributed in $[0, 2]$; then the data set is a 3-D helix surfaces. We label samples with $\theta \in [0, 2\pi]$ as -1 and those with $\theta \in (2\pi, 4\pi]$ as 1 . Then the 3-dimensional vector $[\mathbf{x}_n^{(1)}, \mathbf{x}_n^{(2)}, \mathbf{x}_n^{(3)}]^T$ is mapped to a I -dimensional vector by matrix $R_{I \times 3}$.

Fig. 4(a) shows simulated 3-dimensional Swiss roll and the data set of $[\mathbf{x}_n^{(1)}, \mathbf{x}_n^{(2)}, \mathbf{x}_n^{(3)}]^T$. Fig. 3(b) shows the projection of the data points in Fig. 3(a) onto $\mathbf{x}_n^{(1)} - \mathbf{x}_n^{(2)}$ plane.

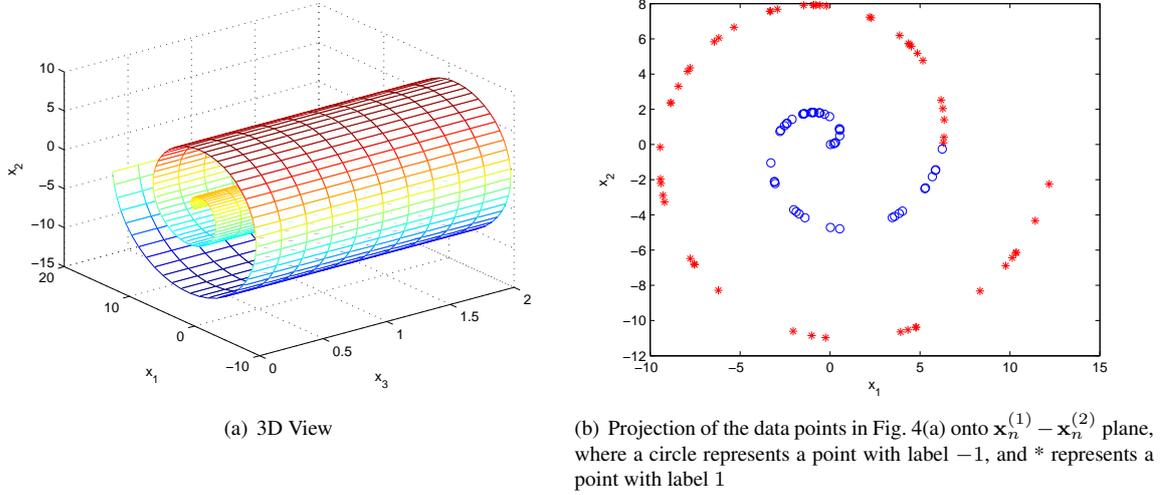


Figure 4: Simulated data set containing Swiss roll.

The classification error rates are all averaged over 10 simulation runs. From Figs. 5(a) and 5(b), it is observed that KLFE+KNN achieves the minimum classification error rate among all the schemes, for $I' \geq 10$; KLFE+KNN is able to reduce the classification error rate by more than 10%, compared to other four schemes. From Fig. 6, we can see that KLFE is similar with LFE. Both KLFE and LFE are better than PCA. When dimension equals to only 1, KLFE can achieve good results while the other two perform much worse. In addition, our KLFE is quite robust against the change of target feature dimension I' ; this is because KLFE has an explicit mechanism to eliminate irrelevant features.

5.3. Experimental Results for Real-World Datasets

In this section, we conduct experiments on three real-world data sets: UCI-diabetes, UCI-ringnorm, and USPS-3vs5 (digit “3” vs. “5”).

Fig. 7(a) shows classification error rate vs. target feature dimension I' for different schemes on diabetes dataset. It is observed that KLFE+KNN achieves the minimum classification error rate among all the schemes, for $I' \geq 30$.

Fig. 7(b) shows classification error rate vs. target feature dimension I' for different schemes on ringnorm dataset. It is observed that KLFE+KNN improves the classification ability of KKNN dramatically, by reducing the classification error rate by more than 50%. GDA also yields good performance but FSKPCA and KPCA give quite poor results, which are even worse than KKNN.

Fig. 8 shows classification error rate vs. target feature dimension I' for three different schemes: KLFE+KNN, LFE+KNN, and PCA+KNN. In this experiment, KLFE performs better than PCA, and achieves performance similar to that of LFE.

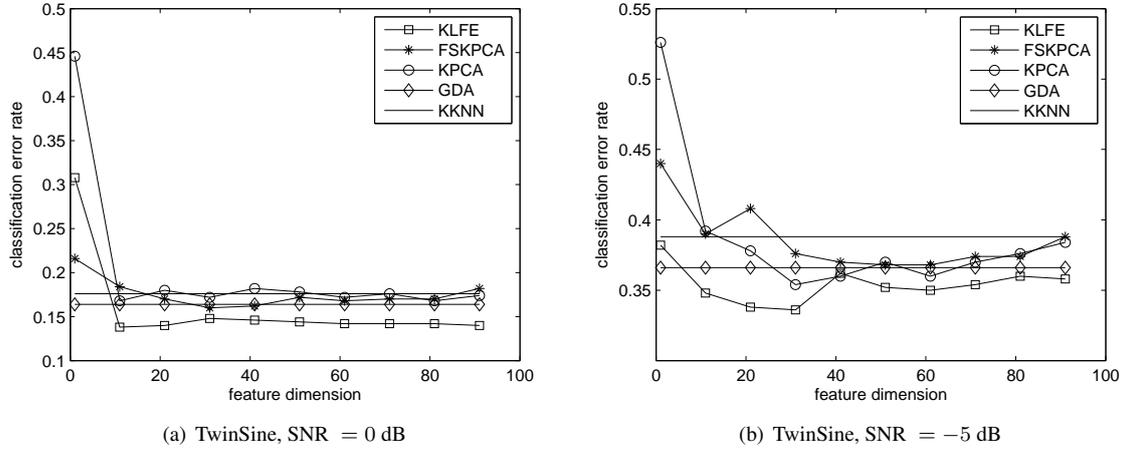


Figure 5: Classification error rate vs. target feature dimension of simulated data.

In summary, our proposed KLFE achieves superior performance over the existing algorithms in most cases. Note that there are strong relationship among KPCA, KLFE, and FSKPCA. Both KLFE and FSKPCA find a basis in kernel subspace, differing in that KLFE uses feature extraction matrix to maximize the average margin whereas FSKPCA uses feature weighting to maximize the average margin. Compared to KLFE and FSKPCA, which use supervised learning, KPCA uses unsupervised learning (i.e., without using label information).

It is worth mentioning that our experiments focus on comparison of various feature extraction methods rather than optimal classifier design. In fact, in order to achieve best classification performance using KLFE+KNN, we should select the optimal K for KNN under KLFE, and the best kernel function. But for fair comparison, we just use the same classifier and the same parameter setting for all the feature extraction methods.

6. Conclusion

This paper is concerned with feature extraction techniques for pattern classification applications. A good feature extraction algorithm is critical in a pattern classification system as it helps reduce system complexity and enhance classification accuracy by eliminating irrelevant features.

In this paper, we proposed a novel feature extraction algorithm, referred to as KLFE, which is a generalization of LFE. The power of KLFE lies in the fact that KLFE has the good properties of a feature extraction technique, i.e., it is a nonlinear wrapper feature extraction method that solves a convex optimization problem. Although nonlinearly mapping a pattern to a high-dimensional space followed by LFE, seems to incur extremely high computation complexity, we theoretically proved that LFE and KLFE

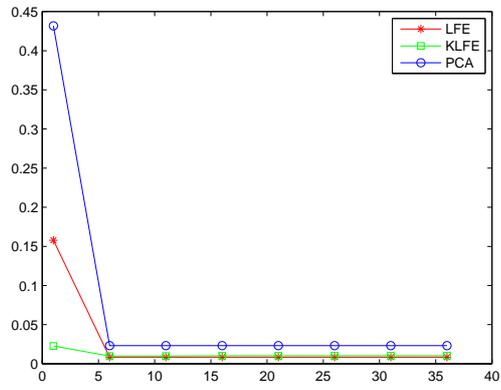


Figure 6: Classification error rate vs. target feature dimension on Swiss Roll

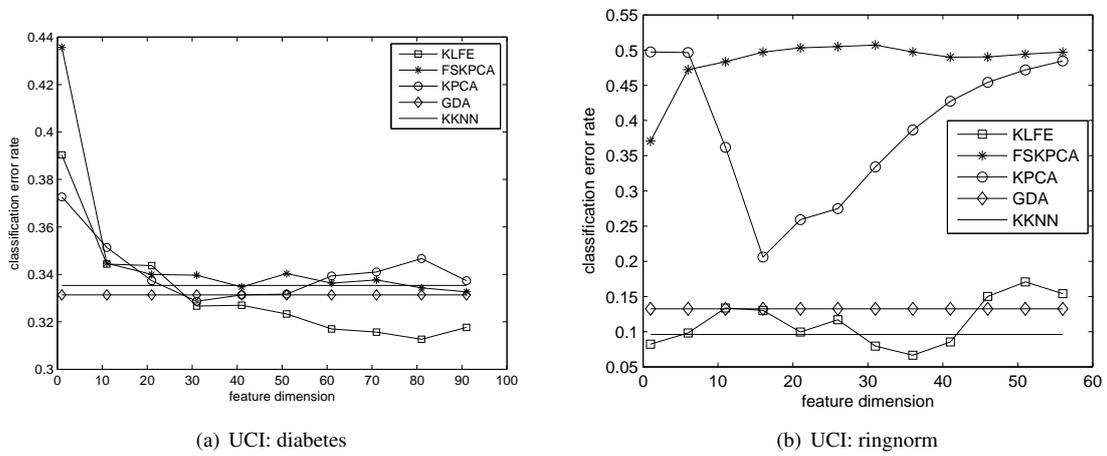


Figure 7: Classification error rate vs. target feature dimension of UCI data.

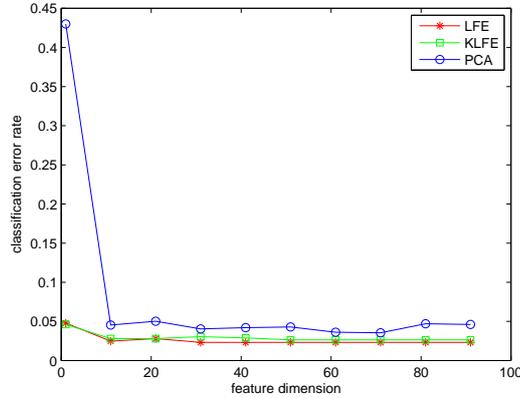


Figure 8: Classification error rate vs. target feature dimension on usps 3 vs. 5.

are both basis rotation invariant, which allows us to implement KLFE via KPCA or KGP followed by LFE.

As shown in Eq. (68), the overall computation complexity of KLFE using RBF kernel function is $\mathcal{O}(N^2I) + \mathcal{O}(N^3)$, comparable to LFE in original feature space. In other words, KLFE has the advantage of better discriminant information extraction in high-dimensional space, while preserving a comparable computation complexity as LFE in low-dimensional space. The experiments conducted on both simulated data set and three real-world data sets demonstrate the effectiveness and robustness of our KLFE algorithm.

7. Acknowledgment

This work was supported in part by the US Air Force Office of Scientific Research under grant FA9550-09-1-0132.

References

- [1] K. Kira, L. A. Rendell, A practical approach to feature selection, in: *ML92: Proceedings of the ninth international workshop on Machine learning*, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1992, pp. 249–256.
- [2] D. Wettschereck, D. W. Aha, T. Mohri, A review and empirical evaluation of feature weighting methods for a class of lazy learning algorithms, *Artificial Intelligence Review* 11 (1–5) (1997) 273–314.
- [3] Y. Sun, D. Wu, A relief based feature extraction algorithm, in: *Proceedings of SIAM International Conference on Data Mining*, 2008.
- [4] Y. Sun, D. Wu, Feature extraction through local learning, *Statistical Analysis and Data Mining* 2 (1) (2009) 34–47.

- [5] R. Kohavi, G. H. John, Wrappers for feature subset selection, *Artificial Intelligence* 97 (1997) 273–324.
- [6] R. O. Duda, P. E. Hart, D. G. Stork, *Pattern Classification*, 2nd Edition, Wiley-Interscience, 2000.
- [7] B. Schölkopf, A. Smola, K.-R. Müller, Nonlinear component analysis as a kernel eigenvalue problem, *Neural Computation* 10 (1998) 1299–1319.
- [8] T. G. Dietterich, Machine-learning research: Four current directions, *AI Magazine* 18 (4) (1997) 97–136.
- [9] Y. Sun, J. Li, Iterative relief for feature weighting, in: *ICML '06: Proceedings of the 23rd international conference on Machine learning*, ACM Press, New York, NY, USA, 2006, pp. 913–920. doi:<http://doi.acm.org/10.1145/1143844.1143959>.
- [10] B. Cao, D. Shen, J. Sun, Q. Yang, Z. Chen, Feature selection in a kernel space, in: *Proceedings of the 24th international conference on Machine learning*, ACM New York, NY, USA, 2007, pp. 121–128.
- [11] J. Shawe-Taylor, N. Cristianini, *Kernel Methods for Pattern Analysis*, Cambridge University Press, 2004.
- [12] D. Newman, S. Hettich, C. Blake, C. Merz, *UCI repository of machine learning databases* (1998).
URL <http://www.ics.uci.edu/~lms/mllearn/MLRepository.html>
- [13] G. Baudat, F. Anouar, Generalized discriminant analysis using a kernel approach, *Neural computation* 12 (10) (2000) 2385–2404.
- [14] B. V. Dasarthy, *Nearest Neighbor: Pattern Classification Techniques*, IEEE Computer Society, 1990.