Kernel Sparse Representation-Based Classifier

Li Zhang, *Member, IEEE*, Wei-Da Zhou, *Member, IEEE*, Pei-Chann Chang, Jing Liu, *Member, IEEE*, Zhe Yan, Ting Wang, and Fan-Zhang Li

***Abstract*—Sparse representation-based classifier (SRC), a combined result of machine learning and compressed sensing, shows its good classification performance on face image data. However, SRC could not well classify the data with the same direction distribution. The same direction distribution means that the sample vectors belonging to different classes distribute on the same vector direction. This paper presents a new classifier, kernel sparse representation-based classifier (KSRC), based on SRC and the kernel trick which is a usual technique in machine learning. KSRC is a nonlinear extension of SRC and can remedy the drawback of SRC. To make the data in an input space separable, we implicitly map these data into a high-dimensional kernel feature space by using some nonlinear mapping associated with a kernel function. Since this kernel feature space has a very high (or possibly infinite) dimensionality, or is unknown, we have to avoid working in this space explicitly. Fortunately, we can indeed reduce the dimensionality of the kernel feature space by exploiting kernel-based dimensionality reduction methods. In the reduced subspace, we need to find sparse combination coefficients for a test sample and assign a class label to it. Similar to SRC, KSRC is also cast into an ℓ_1-minimization problem or a quadratically constrained ℓ_1-minimization problem. Extensive experimental results on UCI and face data sets show KSRC improves the performance of SRC.**

***Index Terms*—ℓ_1-norm, compressed sensing, kernel method, machine learning, sparse representation.**

I. Introduction

SPARSITY could be a useful principle when developing a classification algorithm in machine learning, signal and image processing, and computer vision. The term sparsity has different meanings depending on which context it is used in. For instance, sparsity means that a signal or an image is compressible and can be efficiently represented on an appropriate basis such as wavelets in compressed sensing (CS) which is a new area of signal processing [1]–[7]. In CS's framework, a sparse signal can be exactly reconstructed from far less samples (measurements) than those required by the Shannon–Nyquist Theorem. Thus, sparsity is one important fundamental principle in CS [5], [8] . In machine learning, sparsity usually refers to the extent to which a representation model contains null values, and can be measured by the number of nonzero coefficients in a decision function $f(\boldsymbol{\alpha}, \mathbf{x})$, that is, the number of nonzero elements in the parameter vector $\boldsymbol{\alpha}$, where $\mathbf{x} \in \mathbb{R}^m$ is a sample [9]. The less the number of nonzero elements is, the better sparsity we get. A sparse model representation in machine learning is expected to improve the generalization performance and computational efficiency [9]–[11]. The mechanism of maximizing the sparsity of a model representation can be regarded as an approximative form of the minimum description length principle which can be used to improve the generalization performance [12].

Presently, three main techniques or their combinations are available to obtain a sparse model representation in machine learning [9], including zero-trapped loss functions, sparse regularization and matching pursuits. Two of them, sparse regularization and matching pursuits are used to recover the sparse signal/image in CS. Here we focus on the sparse regularization technique, such as the ℓ_0-norm regularization and the ℓ_1-norm regularization. The ℓ_0-norm regularization is the desirable one to obtain sparseness, but the ℓ_0-norm regularization is so discontinuous that it is very difficult to optimize the objective function containing it. As an approximation of the ℓ_0-norm regularization, the ℓ_1-norm regularization can also induce sparseness and is segment-wise differentiable to make the optimization possible. The ℓ_1-norm regularization is widely used to recover a compressible signal in compressed sensing domain [1]–[4], [6], [7]. Some typical methods using the ℓ_1-norm regularization are basis pursuit [13], least absolute shrinkage and selection operator (LASSO) [14], forward stage-wise regression [15], least angle regression [16], and gradient projection for sparse reconstruction (GPSR) [17].

Recently, sparse representation-based classifier (SRC) is proposed in [18], [19]. SRC is a nonparametric learning method which does not need a training process but does need training data, and can directly assign a class label to a test sample. SRC can also be thought of as a perfect combination of machine learning and compressed sensing. SRC implements sparse representation of data by using the methods for sparse signal reconstruction in CS and classifies data in terms of reconstruction errors. Usually, SRC takes the sample matrix as a sparse representation matrix, and adopts some transformation matrix to reduce dimensionality of the input space. Each column of the

Manuscript received January 05, 2011; revised May 10, 2011, July 30, 2011, and November 22, 2011; accepted November 25, 2011. Date of publication December 13, 2011; date of current version March 06, 2012. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Andrea Cavallaro. This work was supported in part by the National Natural Science Foundation of China under Grants 60970060, 61033013, and 60872135, and by the Natural Science Foundation of Jiangsu Province of China under Grant BK2011284.

L. Zhang and F.-Z. Li are with the Research Center of Machine Learning and Data Analysis, School of Computer Science and Technology, Soochow University, Suzhou 215006, Jiangsu, China (e-mail: zhangliml@suda.edu.cn; lfzh@suda.edu.cn).

W.-D. Zhou is with the AI Speech Ltd., Suzhou 215123, Jiangsu, China (e-mail: zhou.machinelearning@gmail.com).

P.-C. Chang is with the Department of Information Management, Yuan Ze University, Taoyuan 32026, Taiwan, China (e-mail: iepchang@saturn.yzu.edu.tw).

J. Liu, Z. Yan, and T. Wang are with the Key Laboratory of Intelligent Perception and Image Understanding of Ministry of Education, Xidian University, Xi'an 710071, China (e-mail: neouma@mail.xidian.edu.cn; shuiqiong1839@gmail.com; wangtinglaohuang@163.com).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSP.2011.2179539

sample matrix consists of a training sample. Random projection (RP) is a good choice for reducing dimensionality, since Wright *et al.* state that "the precise choice of feature space is no longer critical, even random features contain enough information to recover the sparse representation and hence correctly classify any test image" [19]. The experimental results show that SRC has better classification performance than nearest neighbor (NN) [12] and nearest subspace (NS) [20], [21], and compares to linear SVM on two face databases [18], [19]. However, SRC would lose its classification ability on data with the same direction distribution. In other words, SRC could not classify a test sample if it has the same vector direction as training samples belonging to two or more classes.

In machine learning, the kernel trick is originally used to construct nonlinear support vector machines (SVMs) [22]–[24]. A Mercer kernel implicitly defines a nonlinear mapping which maps data in the input space into a high or even infinite dimensional kernel feature space [25], [26]. Then a linear processing in the kernel feature space, such as a linear classification or a linear regression, gives a nonlinear machine learning method with respect to the input space [23], [24]. So far, almost all linear learning methods can be generalized to the corresponding nonlinear ones by using kernel tricks [27], [28]. Kernel principal component analysis (KPCA) [29] and kernel Fisher discriminant analysis (KFDA) [30] are generated by applying the kernel method to principal component analysis (PCA) and Fisher discriminant analysis (FDA), respectively.

SVMs are one of typical techniques which combine the sparsity-induced techniques and kernel tricks. There have proposed various methods for combining the sparsity-induced techniques and kernel tricks, such as 1-norm SVM [31], sparse kernel regressor [32], kernel sparse representation (KSR) [33] and so on, where KSR is very relative to our work. There are two cases in KSR, or a codebook is given or not given. The objective of KSR is not convex when a codebook is not given. KSR is to solve a quadratic programming (QP) problem when the codebook is given. In both cases, KSR can not use the algorithms available for sparse signal reconstruction, such as methods proposed in [14], [16], [17], [34], and [35], some of which are very fast and efficient. Thus, the optimization speed of KSR is relatively slow. Here we only consider the case of given the codebook, or the sample matrix. In addition, the dimensionality reduction is not performed in a kernel feature space, but in the input space. In other words, KSR only uses the linear dimensionality reduction methods which can not get nonlinear features.

This paper also proposes a kernel sparse representation-based classifier (KSRC) by introducing the kernel trick, which can deal with the problem occurred in SRC, employ the algorithms available for sparse signal reconstruction, and perform nonlinear dimensionality reduction in a kernel feature space. In KSRC, the data in the input sample space are implicitly mapped into a high or even infinite dimensional kernel feature space by using some nonlinear mapping associated with a kernel function. Since we can only access the kernel feature space in terms of the kernel function, the kernel-based dimensionality reduction method is required to reduce the dimensionality of the feature space. By the representation of projection matrices in KFDA and KPCA, we design two projection matrices for KSRC. The sparse combination coefficients can be obtained by solving an ℓ_1-minimization problem or a quadratically constrained ℓ_1-minimization problem in the reduced subspace. Both of them are typical formulations for sparse signal reconstruction in CS [17], [34], [35].

The rest of this paper is organized as follows. In Section II, we briefly review SRC and simply introduce the kernel trick. Section III presents KSRC, describes the construction of a transformation matrix in the kernel feature space for KSRC, and discusses the connection between KSRC and KSR. KSRC is compared with other classifiers on synthetic and real-world public data in Section IV. Section V concludes this paper.

II. Related Work

In this section, we introduce the SRC proposed in [18], [19]. Finally, the kernel trick is briefly reviewed.

A. Sparse Representation-Based Classifier

Actually, SRC is a nonparametric learning method similar to nearest neighbor (NN) [12] and nearest subspace [20], [21]. These methods can directly assign a class label to a test sample without a training process. In other words, they do not need a set of hypothesis functions and to learn the parameters (e.g., weight vector) of the hypothesis function. SRC can be viewed as a learning machine in which the classification process is implemented by using signal reconstruction methods. From this viewpoint, SRC is a perfect combination of machine learning and compressed sensing. In SRC, all possible projection methods, such as PCA [12], FDA [12], [36], and even random projection (RP) [18], [19] could be used. SRC has been successfully applied to human frontal face recognition in [18], [19]. They experimentally show that SRC has better classification performance than NN and NS on face data.

In the following, we simply introduce SRC. Note that notations used here slightly differ from those in [18], [19]. We prefer to exploit notations in the field of machine learning. Assume that there are a set of training samples $\{(\mathbf{x}_i, y_i) | (\mathbf{x}_i \in \mathcal{X} \subseteq \mathbb{R}^m, y_i \in \{1, 2, \ldots, c\}, i = 1, 2, \ldots, n\}$, where c is the number of classes, m is the dimensionality of the input space $\mathcal{X}$, and y_i is label corresponding to $\mathbf{x}_i$. Given a test sample $\mathbf{x} \in \mathcal{X}$, the goal is exactly to predict the label y of $\mathbf{x}$ from the given c-class training samples. Now we arrange the jth class training samples as columns of a matrix $\mathbf{X}_j = [\mathbf{x}_{j,1}, \ldots, \mathbf{x}_{j,n_j}] \in \mathbb{R}^{m \times n_j}, j = 1, \ldots, c$, where $\mathbf{x}_{j,i}$ denotes the sample belonging to the jth class, and n_j is the number of the jth class training samples. Define a new sample matrix $\mathbf{X}$ for all training samples

$$\mathbf{X} = [\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_c] \in \mathbb{R}^{m \times n} \tag{1}$$

where $n = \sum_{j=1}^{c} n_j$.

According to SRC, the test sample $\mathbf{x}$ can be linearly represented by all training samples:

$$\mathbf{x} = \mathbf{X}\boldsymbol{\alpha} \tag{2}$$

where $\boldsymbol{\alpha} \in \mathbb{R}^n$ is the vector of coefficients. If the test sample $\mathbf{x}$ belongs to the jth class, then the entries of $\boldsymbol{\alpha}$ are expected to be zero except some of those associated with this class. Namely

$$\boldsymbol{\alpha} = [0, \ldots, 0, \alpha_{j,1}, \ldots, \alpha_{j,n_j}, 0, \ldots, 0]^T \tag{3}$$

where $\alpha_{j,i} \in \mathbb{R}$ is the coefficient corresponding to the training sample $\mathbf{x}_{j,i}$. Thus, the coefficient vector $\boldsymbol{\alpha}$ is expected to be sparse. In SRC, the problem of finding the coefficient vector is formulated as a convex programming problem

$$\begin{aligned} \min_{\boldsymbol{\alpha}} \quad & \|\boldsymbol{\alpha}\|_1 \\ \text{subject to} \quad & \mathbf{x} = \mathbf{X}\boldsymbol{\alpha} \end{aligned} \tag{4}$$

where $\|\cdot\|_1$ denotes the ℓ_1-norm. Equation (4) is also called the ℓ_1-minimization problem [18], [19]. Consider the case of noisy data, the model (2) can be modified as

$$\mathbf{x} = \mathbf{X}\boldsymbol{\alpha} + \boldsymbol{\xi} \tag{5}$$

where $\boldsymbol{\xi} \in \mathbb{R}^m$ is a noise vector with bounded energy $\|\boldsymbol{\xi}\|_2 < \varepsilon$, where $\|\cdot\|_2$ denotes the ℓ_2-norm. So the programming (4) can be modified as

$$\begin{aligned} \min_{\boldsymbol{\alpha}} \quad & \|\boldsymbol{\alpha}\|_1 \\ \text{subject to} \quad & \|\mathbf{x} - \mathbf{X}\boldsymbol{\alpha}\|_2 \leq \varepsilon \end{aligned} \tag{6}$$

which is one standard formulation for sparse reconstruction problems in CS, called the quadratically constrained ℓ_1-minimization problem [17], [35].

Both convex problems (4) and (6) can be efficiently solved [18], [19], [37]. For the problem (6), there propose some new methods, such as spectral projected gradient method (SPGL1) [34] and NESTA (a shorthand for Nesterov's algorithm) [35]. In [19], ℓ_1-magic software package [38] is used to solve these problems. Once we get the coefficient vector, we can classify $\mathbf{x}$ according to reconstruction errors (residuals) between $\mathbf{x}$ and its approximations. The ith approximation is obtained by using only the coefficients associated with the ith class. Definitely, we would choose the one with minimum residual.

As mentioned above, SRC behaves well in human frontal face recognition. But SRC loses its classification ability even for the linearly separable task in which the data from different classes have the same direction. The main reason is that the data in the same direction would overlap each other after the normalization process. So we can not essentially distinguish them, which accounts for the bad performance of SRC in this case. To resolve this problem occurred in SRC, we will introduce the kernel trick into SRC in the next section and generate a kernel sparse representation-based classifier.

B. Kernel Trick

The kernel trick is a very well-known technique in machine learning. It has been successfully applied to many methods, e.g., SVM [22], [23] , KPCA [29], and KFDA [30]. The use of the kernel trick can easily generalize a linear algorithm to a nonlinear algorithm.

Only a kernel satisfying Mercer's condition is called a Mercer kernel which is generally used in kernel methods. In other words, a Mercer kernel is continuous, symmetric, positive semidefinite kernel function. Given a Mercer kernel $k : \mathcal{X} \times \mathcal{X} \longrightarrow \mathbb{R}$, there is a unique associated reproducing kernel Hilbert space (RKHS) $\mathcal{H}_k$. Usually, a Mercer kernel k can be expressed as

$$k(\mathbf{x}, \mathbf{x}') = \Phi(\mathbf{x})^T \Phi(\mathbf{x}') \tag{7}$$

where T denotes the transpose of a matrix or vector, $\mathbf{x}$ and $\mathbf{x}'$ are any two points in $\mathcal{X}$, and Φ is the implicit nonlinear mapping associated with the kernel function $k(\cdot, \cdot)$. The kernel function is actually Euclidian vector inner product between two images. In kernel methods, we don't need to know what Φ is and just adopt the kernel function (7). Some commonly used kernels in kernel methods are the linear kernel, polynomial kernels, Gaussian radial basis function (RBF) kernels, and wavelet kernels [23], [39], [40]. The Linear kernel has the form

$$k(\mathbf{x}, \mathbf{x}') = \mathbf{x}^T \mathbf{x}' \tag{8}$$

and RBF kernels can be expressed as

$$k(\mathbf{x}, \mathbf{x}') = \exp\left(-\gamma \|\mathbf{x} - \mathbf{x}'\|_2^2\right) \tag{9}$$

where $\gamma > 0$ is the parameter for RBF kernels.

III. Kernel Sparse Representation-Based Classifier

This section proposes a new kernel method, kernel sparse representation-based classifier (KSRC) which is a nonlinear extension of SRC.

A. Kernel Feature Space

Consider a c-class classification task. Let the training set be $\{\mathbf{x}_i, y_i\}_{i=1}^n$, where n is the total number of training samples, $\mathbf{x}_i \in \mathcal{X} \subset \mathbb{R}^m$, and $y_i \in \{1, 2, \ldots, c\}$. Given an arbitrary sample $\mathbf{x}$ in the input space $\mathcal{X}$, the goal is to assign a label to it. Let Φ be the nonlinear mapping function corresponding to a kernel $k(\cdot, \cdot)$. To make the training samples separable, we employ Φ to map the data from the input space $\mathcal{X}$ to a high-dimensional (possibly infinite dimensional) kernel feature space $\mathcal{F}$. In the finite dimensional case, we have

$$\Phi : \mathbf{x} \in \mathcal{X} \longrightarrow \Phi(\mathbf{x}) = [\phi_1(\mathbf{x}), \phi_2(\mathbf{x}), \ldots, \phi_D(\mathbf{x})]^T \in \mathcal{F} \tag{10}$$

where $\Phi(\mathbf{x}) \in \mathbb{R}^D$ is the image of $\mathbf{x}$ in $\mathcal{F}$, $D \gg m$ is the dimension of the feature space $\mathcal{F}$, and $\phi_j(\mathbf{x}) \in \mathbb{R}$. The conclusions obtained from the finite dimensional case can be applied to the case of infinite dimension. Thus, hereafter our discussion only focus on the finite dimensional case for the convenience of description. The images of the training samples $\mathbf{x}_i$ are $\Phi(\mathbf{x}_i), i = 1, \ldots, n$. In SRC, the test sample can be linearly represented by training samples in the input space $\mathcal{X}$. Similarly,

we can linearly represent the image of test sample in terms of the images of all training samples in this kernel feature space $\mathcal{F}$

$$\Phi(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i \Phi(\mathbf{x}_i) = \mathbf{\Phi}\boldsymbol{\alpha} \tag{11}$$

where $\boldsymbol{\alpha} = [\alpha_1, \alpha_2, \ldots, \alpha_n]^T$ is the coefficient vector, α_i are the coefficients corresponding to the images $\Phi(\mathbf{x}_i)$, and the sample matrix in $\mathcal{F}$ can be expressed as

$$\mathbf{\Phi} = [\Phi(\mathbf{x}_1), \Phi(\mathbf{x}_2), \ldots, \Phi(\mathbf{x}_n)] \in \mathbb{R}^{D \times n}. \tag{12}$$

If we directly replace the constraint in (4) by (11), then we can have

$$\begin{aligned} \min_{\boldsymbol{\alpha}} \quad & \|\boldsymbol{\alpha}\|_1 \\ \text{subject to} \quad & \Phi(\mathbf{x}) = \mathbf{\Phi}\boldsymbol{\alpha}. \end{aligned} \tag{13}$$

It is not practical to directly solve the optimization problem (13). The main reason is summarized as follows. There are two situations for $\mathcal{F}$: $\mathcal{F}$ is known or unknown. In the case of knowing $\mathcal{F}$, we can see that the computational complexity of (13) must be much larger than that of (4) since the dimensionality of the kernel feature space $\mathcal{F}$ is far greater than that of the input space $\mathcal{X}$, or $D \gg m$. Moreover, the solution to (13) is not sparse in the case of $D \gg n$. If $\mathcal{F}$ is unknown, we can not explicitly obtain the sample matrix $\mathbf{\Phi}$ in $\mathcal{F}$. Thus, we can not solve (13). Fortunately, we can access $\mathcal{F}$ by the kernel functions and transform (13) into a feasible optimization problem by using kernel-based dimensionality reduction methods.

B. Dimensionality Reduction in Kernel Feature Space

Since the kernel feature space $\mathcal{F}$ has a very high or possibly infinite dimensionality, it is necessary to perform dimensionality reduction in $\mathcal{F}$. In other words, the implicit images need to be projected from $\mathcal{F}$ into a low-dimensional (reduced) subspace. Now the task is to construct a transformation matrix in $\mathcal{F}$. Let $\mathbf{P} \in \mathbb{R}^{D \times d}$ be the transformation matrix. By using it, we can modify (11) as

$$\mathbf{P}^T \Phi(\mathbf{x}) = \mathbf{P}^T \mathbf{\Phi}\boldsymbol{\alpha}. \tag{14}$$

Observation on (14) indicates that $\mathbf{P}$ must be related to the images such that dot products of images can be replaced by a kernel. Now we consider the representation of the transformation matrix in kernel-based dimensionality reduction methods, KPCA [29] and KFDA [30]. In both KPCA and KFDA, the projection vector is a linear combination of images in $\mathcal{F}$. Namely

$$\mathbf{P}_j = \sum_{i=1}^{n} \beta_{j,i} \Phi(\mathbf{x}_i) = \mathbf{\Phi}\boldsymbol{\beta}_j \tag{15}$$

where $\mathbf{P}_j$ is the jth transformation vector of $\mathbf{P} = [\mathbf{P}_1, \ldots, \mathbf{P}_d]$, and $\boldsymbol{\beta}_j = [\beta_{j,1}, \ldots, \beta_{j,n}]^T$ is called the pseudo-transformation vector corresponding to the jth transformation vector. Let $\mathbf{B} = [\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_d]$ be the pseudo-transformation matrix. Then the transformation matrix can be expressed as

$$\mathbf{P} = \mathbf{\Phi}\mathbf{B}. \tag{16}$$

Substituting (16) into (14), we get

$$\mathbf{B}^T \mathbf{k}(\cdot, \mathbf{x}) = \mathbf{B}^T \mathbf{K}\boldsymbol{\alpha} \tag{17}$$

where $\mathbf{k}(\cdot, \mathbf{x}) = [k(\mathbf{x}_1, \mathbf{x}), \ldots, k(\mathbf{x}_n, \mathbf{x})]^T = \mathbf{\Phi}^T \Phi(\mathbf{x})$, $\mathbf{K} = \mathbf{\Phi}^T \mathbf{\Phi} \in \mathbb{R}^{n \times n}$ is the kernel Gram matrix which is symmetric and positive semidefinite, and $K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$. Compared to (11), (17) is definitely feasible since we can obtain $\mathbf{K}$ and $\mathbf{k}(\cdot, \mathbf{x})$ when given a kernel function $k(\cdot, \cdot)$. In (17), the pseudotransformation matrix $\mathbf{B}$ is required. So far, the problem of finding the transformation matrix in $\mathcal{F}$ is formulated into that of finding the pseudo-transformation matrix in RKHS.

Now we concern on finding the pseudo-transformation matrix $\mathbf{B}$. In the following, we not only describe the schemes for $\mathbf{B}$ in both KPCA and KFDA, but also give random and determinate schemes for $\mathbf{B}$. Generally, the dimensionality of reduced subspace is smaller than or equal to the number of training samples, or $d \leq n$.

- In KPCA, the pseudo-transformation vectors $\boldsymbol{\beta}_j \in \mathbb{R}^n$ are normalized eigenvectors corresponding to nonzero Eigenvalues which are the solution of the following Eigenvalue problem [29]

  $$n\lambda\boldsymbol{\beta} = \mathbf{K}\boldsymbol{\beta} \tag{18}$$

  $\boldsymbol{\beta}_j$ are normalized such that $\lambda_j(\boldsymbol{\beta}_j)^T\boldsymbol{\beta}_j = 1$. We take d eigenvectors corresponding to the first d largest eigenvalues $\lambda_j, j = 1, \ldots, d$, where $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_d$. $\mathbf{B} = [\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_d] \in \mathbb{R}^{n \times d}$.
- For the case of KFDA, $\mathbf{B} \in \mathbb{R}^{n \times d}$ is the solution of the following maximization problem [30]

  $$\max_{\mathbf{B}} \quad \frac{tr(\mathbf{B}^T \mathbf{S}_b^K \mathbf{B})}{tr(\mathbf{B}^T \mathbf{S}_w^K \mathbf{B})} \tag{19}$$

  where $tr(\cdot)$ denotes the trace of a matrix, $\mathbf{S}_w^K$ and $\mathbf{S}_b^K$ are quasi within-class and between-class scatter matrices, respectively. Generally, we have $d < c$.
- In $\mathcal{F}$, we can not directly use the RP method to reducing its dimensionality. But we can make $\mathbf{B}$ be a random matrix. Here, this scheme is still called random projection.
- We also design a determinate scheme. In this scheme, $\mathbf{B} \in \mathbb{R}^{n \times n}$ is defined as the identity matrix with ones on the main diagonal and zeros elsewhere. Since the reduced subspace has a dimensionality of n, which is the number of training samples, this determinate scheme does not reduce the size of $\mathbf{B}^T\mathbf{K}$. Therefore, we take it as the case of none-dimensionality reduction in KSRC.

Typically, $D \gg m$. If the number of training samples is less than the dimensionality of the sample space $\mathcal{X}$ or $n < m$ which is a typical small sample size problem (e.g., face recognition task), there is $D \gg n$. Even if $n > m$, we still could have $n < D$ in most cases, such as the case of using RBF kernels. By using any one of four dimensionality reduction methods mentioned

above, we can reduce the images in $\mathcal{F}$ into a reduced subspace. Thus, the high dimensionality of $\mathcal{F}$ does not lead to a curse.

C. KSRC

In the following, we give the classification based on KSRC in the kernel feature space. Replacing the constraint of (13) by (17), we can get a feasible optimization problem

$$\begin{aligned} \min_{\boldsymbol{\alpha}} \quad & \|\boldsymbol{\alpha}\|_1 \\ \text{subject to} \quad & \mathbf{B}^T\mathbf{k}(\cdot,\mathbf{x}) = \mathbf{B}^T\mathbf{K}\boldsymbol{\alpha}. \end{aligned} \tag{20}$$

Consider the case of noisy data, the constraint of (20) can be modified as

$$\mathbf{B}^T\mathbf{k}(\cdot,\mathbf{x}) = \mathbf{B}^T\mathbf{K}\boldsymbol{\alpha} + \boldsymbol{\xi} \tag{21}$$

where $\boldsymbol{\xi} \in \mathbb{R}^n$ is a noise and $\|\boldsymbol{\xi}\|_2 < \varepsilon$. Similar to the case of SRC, the quadratically constrained ℓ_1-minimization problem can be exploited in noisy data. Namely

$$\begin{aligned} \min_{\boldsymbol{\alpha}} \quad & \|\boldsymbol{\alpha}\|_1 \\ \text{subject to} \quad & \|\mathbf{B}^T\mathbf{k}(\cdot,\mathbf{x}) - \mathbf{B}^T\mathbf{K}\boldsymbol{\alpha}\|_2 \le \varepsilon. \end{aligned} \tag{22}$$

Usually, ε is a small positive constant, say 10^{-5}.

Solving (20) or (22), we can get the coefficient vector $\boldsymbol{\alpha}$. Now we need to classify $\mathbf{x}$ in terms of $\boldsymbol{\alpha}$. Likewise, we also use the minimum residual between $\mathbf{x}$ and its c approximations in the reduced subspace to determine the label of $\mathbf{x}$. For class i, we define a characteristic function δ_i which can pick up the coefficients corresponding to the ith class. Namely

$$\delta_i(\alpha_j) = \begin{cases} \alpha_j, & \text{if } y_j = i \\ 0, & \text{otherwise .} \end{cases} \tag{23}$$

By using which, we get only the coefficients of samples belonging to class i and denote them by a new vector

$$\boldsymbol{\delta}_i = [\delta_i(\alpha_1), \delta_i(\alpha_2), \ldots, \delta_i(\alpha_n)]^T. \tag{24}$$

Thus, the ith approximation to the test sample $\mathbf{x}$ in the reduced subspace can be expressed as $\mathbf{B}^T\mathbf{K}\boldsymbol{\delta}_i$. We get the estimated label $\hat{y}$ for $\mathbf{x}$ by minimizing residual between the $\mathbf{B}^T\mathbf{k}(\cdot,\mathbf{x})$ and its approximations. Then, we get

$$\hat{y} = \arg\min_{i=1,\ldots,c} \quad r_i(\mathbf{x}) = \|\mathbf{B}^T\mathbf{k}(\cdot,\mathbf{x}) - \mathbf{B}^T\mathbf{K}\boldsymbol{\delta}_i\|_2. \tag{25}$$

The complete classification procedure of KSRC is shown in Algorithm 1. In the Step 5 of Algorithm 1, it requires to normalize the columns of two matrixes to have unit ℓ_2-norm. The feature values for the same point are usually of different order of magnitude. Generally, the larger the feature value is, the smaller the corresponding coefficient is. The normalization step can map all data points onto a hypersphere. To represent a test point which is also located on this hypersphere, the coefficients are only affected by the location relations between training points and this test point instead of the unbalanced feature values. In addition, if the norm of a column vector is nearly 0, a small perturbation can be added into $\mathbf{B}^T\mathbf{K}$. Namely, let $\mathbf{B}^T\mathbf{K} + 10^{-8}\mathbf{1}$, where $\mathbf{1}$ is a matrix with all ones and has the same size with $\mathbf{B}^T\mathbf{K}$.

Algorithm 1 Kernel Sparse Representation-Based Classification Method

1. **Input**: A set of training samples $\{\mathbf{x}_i, y_i\}_{i=1}^n$, where $\mathbf{x} \in \mathbb{R}^m$, $y_i \in \{1, 2, \ldots, c\}$, a test sample $\mathbf{x} \in \mathbb{R}^m$, and an optional error tolerance $\varepsilon > 0$.
2. Select a Mercer kernel $k(\cdot,\cdot)$ and its parameters.
3. Compute the kernel Gram matrix $\mathbf{K}$ where $K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$, and a vector $\mathbf{k}(\cdot,\mathbf{x}) = [k(\mathbf{x}_1,\mathbf{x}), \ldots, k(\mathbf{x}_n,\mathbf{x})]^T$.
4. Select a projection method and get the corresponding pseudo-transformation matrix $\mathbf{B}$.
5. Normalize the columns of $\mathbf{B}^T\mathbf{K}$ and $\mathbf{B}^T\mathbf{k}(\cdot,\mathbf{x})$ to have unit ℓ_2-norm.
6. Solve the ℓ_1-minimization problem **(20)**, or the quadratically constrained ℓ_1-minimization problem **(22)** to get the coefficient vector $\boldsymbol{\alpha}$.
7. Compute the c residuals $r_i(\mathbf{x}) = \|\mathbf{B}^T\mathbf{k}(\cdot,\mathbf{x}) - \mathbf{B}^T\mathbf{K}\boldsymbol{\delta}_i\|_2$, $i = 1, \ldots, c$.
8. **Output**: The estimated label $\hat{y}$ for $\mathbf{x}$ according to **(24)**.

Remarks: In compressed sensing, solving sparse signal reconstruction problems can be approached via several different equivalent formulations. The sparse signal in reconstruction problems is identical to the coefficient vector $\boldsymbol{\alpha}$ here. So we can also use other optimization formulations except the quadratically constrained ℓ_1-minimization problem (22). In the following, we give the other two forms for KSRC. One form is to solve a QP problem [17]. Namely

$$\min_{\boldsymbol{\alpha}} \quad \frac{1}{2}\|\mathbf{B}^T\mathbf{k}(\cdot,\mathbf{x}) - \mathbf{B}^T\mathbf{K}\boldsymbol{\alpha}\|_2^2 + \lambda\|\boldsymbol{\alpha}\|_1 \tag{26}$$

where $\lambda > 0$ is a regularization parameter. The other form is more well-known as the LASSO [14]

$$\begin{aligned} \min_{\boldsymbol{\alpha}} \quad & \|\mathbf{B}^T\mathbf{k}(\cdot,\mathbf{x}) - \mathbf{B}^T\mathbf{K}\boldsymbol{\alpha}\|_2^2 \\ \text{subject to} \quad & \|\boldsymbol{\alpha}\|_1 \le \tau \end{aligned} \tag{27}$$

where $\tau > 0$ is a parameter. Standard optimization theory asserts that these three problems (22), (26), and (27) are of course equivalent provided that ε, λ and τ obey some special relationships [35].

D. Connection to KSR

The KSR method is proposed in [33], which is motivated by the fact that the kernel trick can capture nonlinear similarity of features. The optimization problem of KSR for face recognition can be expressed as

$$\min_{\boldsymbol{\alpha}} \quad \frac{1}{2}\|\Phi(\mathbf{x}) - \boldsymbol{\Phi}\boldsymbol{\alpha}\|_2^2 + \lambda\|\boldsymbol{\alpha}\|_1. \tag{28}$$

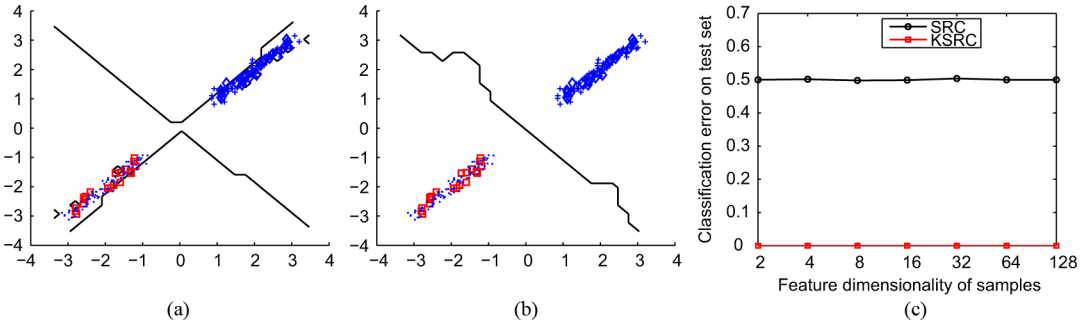


Fig. 1. Performance comparison on Type 1. In figures (a) and (b), the bolded lines are decision boundaries. Training data are denoted by "□" and ◊, respectively; Test data are denoted by "·" and "+," respectively. (a) Boundary obtained by SRC. (b) Boundary obtained by KSRC. (c) Test error versus dimensionality.

Since the nonlinear mapping Φ is unknown, (28) can not be solved by using sparse reconstruction methods as (26). So, (28) can be rewritten as

$$\min_{\boldsymbol{\alpha}} \quad \frac{1}{2}\boldsymbol{\alpha}^T\mathbf{K}\boldsymbol{\alpha} - \mathbf{k}(\cdot,\mathbf{x})^T\boldsymbol{\alpha} + \lambda\|\boldsymbol{\alpha}\|_1 \tag{29}$$

which can be solved by standard methods for QP problems, which is less efficient than the methods for sparse reconstruction. In addition, KSR does not discuss the dimensionality reduction method in kernel feature space, and simply reduces the dimensionality of input space. In order to compare KSRC and KSR, let $\mathbf{B} = \mathbf{I}$ and rewrite (26) as follows:

$$\min_{\boldsymbol{\alpha}} \quad \frac{1}{2}\boldsymbol{\alpha}^T\mathbf{K}\mathbf{K}\boldsymbol{\alpha} - \mathbf{k}(\cdot,\mathbf{x})^T\mathbf{K}\boldsymbol{\alpha} + \lambda\|\boldsymbol{\alpha}\|_1. \tag{30}$$

Obviously, (30) would yield a solution which is different with (29). However, we can find a connection of them.

At present, Zhang *et al.* construct a family of empirical mapping with kernel functions, and use them in SVM [41], KPCA [42] and KFDA [43]. The kernel empirical mapping can be explicitly computed in an empirical mapping space (feature space). Typically, the kernel empirical mapping on the data set $X = \{\mathbf{x}\}_{i=1}^n$ is $[k(\mathbf{x}_1,\mathbf{x}), k(\mathbf{x}_2,\mathbf{x}), \ldots, k(\mathbf{x}_n,\mathbf{x})]^T$. If the nonlinear mapping Φ takes a family of empirical mapping with kernel functions, then KSR is identical to KSRC. In other words, (28) is identical to (26). In this situation, for KSR we get

$$\Phi(\mathbf{x}) = [k(\mathbf{x}_1,\mathbf{x}), k(\mathbf{x}_2,\mathbf{x}), \ldots, k(\mathbf{x}_n,\mathbf{x})]^T = \mathbf{k}(\cdot,\mathbf{x}) \tag{31}$$

and

$$\boldsymbol{\Phi} = \mathbf{K}. \tag{32}$$

IV. Numerical Experiments

In this section, we give numerical experimental results of KSRC on toy and real-world benchmark data sets, and compare KSRC with other nonparametric methods, including KNN, NS, and SRC, and other kernel methods, such as SVM and KSR. In KNN, the Euclidean distance is taken as the distance measure and $K = 3$. For NS, we use the least square method to find the representation coefficients. The quadratically constrained ℓ_1-minimization problem for both KSRC and SRC is solved by exploiting ℓ_1-MAGIC software package [38]. The optimization problem of SVM can be solved by using LIBSVM software package [44]. KSR is solved by using the quadprog.m in the optimization toolbox of MATLAB. If there is no other statements, we set parameters in all methods used here according to the following description. In both SRC and KSRC, let $\varepsilon = 0.001$. In KSRC, SVM and KSR, the linear kernel or RBF kernels are used. The parameter γ of RBF kernel is set by the median value of $\frac{1}{(\|\mathbf{x}_i - \bar{\mathbf{x}}\|^2)}, i = 1,\ldots,n$, where $\bar{\mathbf{x}}$ is the mean of all training samples. For KSR, $\lambda = 0.001$. For SVM, the regularization parameter C is selected from the set $\{10^{-3}, 10^{-3}, \ldots, 10^3\}$ by using 10-fold cross validation in the experiments of benchmark data sets and 5-fold cross validation in the experiments of face data sets.

All numerical experiments are performed on the personal computer with a 1.8 GHz Pentium III and 1 G bytes of memory. This computer runs on Windows XP, with MATLAB 7.01 and VC++ 6.0 compiler installed.

A. Evaluations on toy Data Sets

The purpose of experiments is to compare KSRC with SRC on two toy data sets. Here KSRC adopts RBF kernel. We perform 100 trials on randomly sampled training and test sets and report the average test results.

1) Type 1: In this experiment, we generate m-dimensional two-class data X_1 and X_2, where each feature in X_1 and X_2 takes value from the interval $[-3,-1]$ and $[1,3]$, respectively, and m is changed from 2 to 128. All features are corrupted by Gaussian noise with zero mean and 0.01 variance. Figs. 1(a)–(b) show the case of two-dimensional data X_1 and X_2. There are 20 training and 100 test points in X_1 and X_2, respectively.

The average test results are shown in Fig. 1(c). As the feature dimensionality increases, the classification errors of two methods are almost unchanged. KSRC has a zero error, and SRC has about 50% error rate. Figs. 1(a)–(b) just shows the decision boundaries obtained by two methods in one trial. Obviously, this data set is linearly separable, but SRC could not get a good solution to it. However, KSRC with the RBF kernel can solve it well.

2) Type 2: Here, we generate m-dimensional two-class data X_1 and X_2 with different Gaussian distributions. The X_1 data are randomly sampled from the Gaussian distribution with a mean $[0,\ldots,0]^T \in \mathbb{R}^m$ and a covariance matrix $\mathbf{I}$, while the

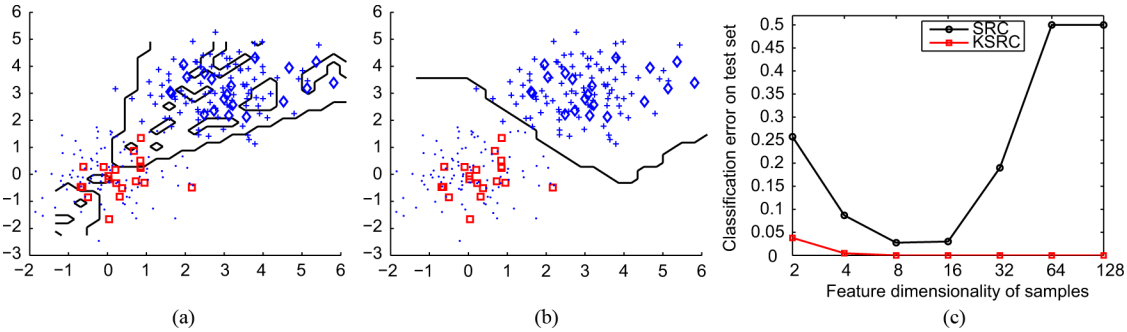


Fig. 2. Performance comparison on Type 2. In figures (a) and (b), the bolded lines are decision boundaries. Training data are denoted by "□" and "◇," respectively. Test data are denoted by "·" and "+," respectively. (a) Boundary obtained by SRC. (b) Boundary obtained by KSRC. (c) Test error versus dimensionality.

X_2 data are randomly sampled from another Gaussian distribution with a mean $[3,\ldots,3]^T \in \mathbb{R}^m$ and a covariance matrix $\mathbf{I}$, where $\mathbf{I}$ is an $m \times m$ identity matrix, and m is also changed from 2 to 128. Figs. 2(a)–(b) show the case of two-dimensional data X_1 and X_2. There still are 20 training and 100 test points in X_1 and X_2, respectively.

Figs. 2(a)–(b) just shows the decision boundaries obtained by the two methods in one trial. The classification results of KSRC with the RBF kernel are the best. The relationship of average test errors as a function of dimensionality is shown in Fig. 2(c). SRC would improve its performance as the dimensionality varied from 2 to 16. When the dimensionality is larger than or equal to 64, SRC gets its worst performance (50%). When the dimensionality is larger than or equal to 8, the test error obtained by KSRC is zero. The worst test error of KSRC is 3.8% in the case of two feature dimensions.

These results show that KSRC with the RBF kernel which has the best performance here can solve the problem that occurs on SRC.

B. Evaluation on Leukemias Data Set

Here the Leukemias data set from [44] and [45] is a typical small sample task which contains a training set and an independent test set. The goal of this task is to distinguish between two classes of leukemia (ALL and AML) which require different clinical treatment. The training set consists of 38 samples (27 ALL and 11 AML) from bone marrow specimens. The test set has 34 samples (20 ALL and 14 AML). All samples have 7129 features, corresponding to some normalized gene expression value extracted from the micro-array image.

The goal of this experiment is to show the sparsity of KSRC when using the dimensionality reduction in the kernel feature space. RBF kernel is used here. As mentioned before, we design four schemes for yielding pseudo-transformation matrix for KSRC, including KPCA, KFDA, RP and determinate scheme. Specially, the determinate scheme does not reduce the dimensionality of kernel feature space, so the size of $\mathbf{B}^T\mathbf{K}$ is still 34×34. For both KPCA and RP, we reduce the size of $\mathbf{B}^T\mathbf{K}$ to 20×34. For KFDA, the size of $\mathbf{B}^T\mathbf{K}$ is 1×34.

Now we represent the test samples. Without loss of generality, we take the first sample (whose label is ALL) in the test set as an illustration. For other samples in the test set, we have the similar results. Sparsity is related not only to the data set at hand, but also the parameter ε. In the following, we vary the parameter ε in the set $\{10^{-7}, 10^{-6}, \ldots, 10^{-1}\}$. Fig. 3 shows the curves of the sparsity and residual as the variation of ε. Here sparsity is defined by the ratio of the zero coefficient number to the training sample number. Generally, the larger the parameter ε is, the higher the sparsity and the residual, and slower the test speed. But here since the training sample is too small, the test speed is almost a constant for all ε used here. Usually, we would choose a parameter ε to balance of the sparsity and the residual. In this case, we can set $\varepsilon = 0.001$. For the first sample, we have its representation coefficient vectors given by the four schemes, as shown in Fig. 4. We can see that the coefficient vector obtained by these four schemes is sparse, which consists with Fig. 3. In the KFDA scheme, we get the highest sparsity, only one coefficient has nonzero value. Both KPCA and RP have the similar sparsity.

Finally, we compare the classification errors on the test set without using any dimensionality reduction method. The errors on the test set are 29.41% for SVM, 58.82% for SRC, 26.47% for KSR, and 26.47% for KSRC (with the determinate scheme), respectively. Both KSR and KSRC get the same classification performance, but the test speed of KSR is slower than that of KSRC. The test time for them are 4.072 s and 0.187 s, respectively.

C. Evaluations on Benchmark Data Sets

In this experiment, we check the performance of KSRC on a collection of benchmark data sets from UCI Machine Learning Repository [46]. Information on these data sets is summarized in Table I. For each data set, we run 10 trials where the training set contains $\frac{2}{3}$ of samples (randomly selected) of each class, and the test set contains the remaining $\frac{1}{3}$. The RBF kernels are adopted for SVM, KSR ,and KSRC here.

1) Experiment Without Dimensionality Reduction: First, we do not use any dimensionality reduction methods for KNN, NS, SRC, SVM, and KSR. For KSRC, we use the determinate scheme described in Section III.

The mean and standard deviation of test error rates are reported in Table II. Since KSR is time-consuming, we only compare KSR and KSRC on 12 UCI data sets. The test time comparison of KSR and KSRC is given in Table III. For each UCI data set, two-tailed t-tests with the significant level 0.05 are performed to determine whether there is a significant difference between KSRC and other methods. A win-loss-tie (W-L-T)

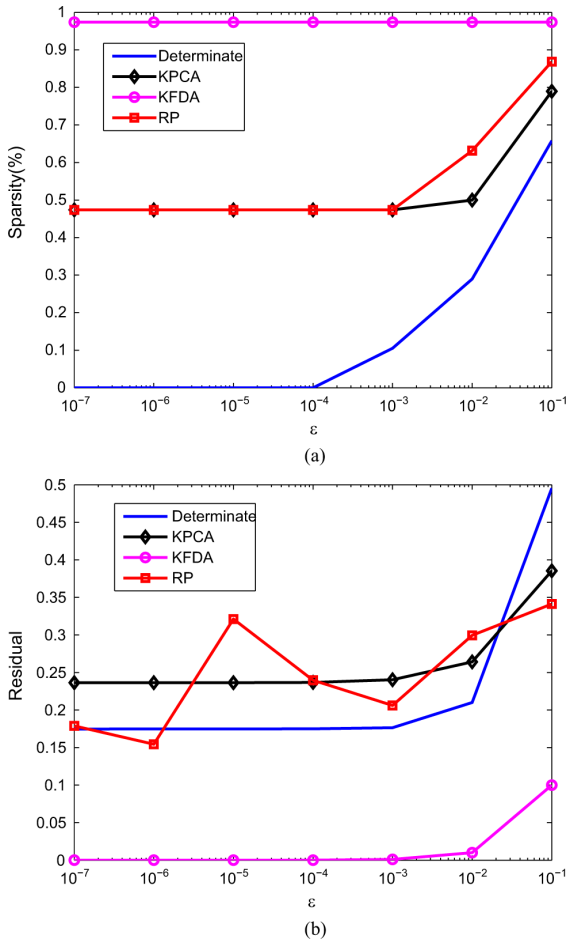


Fig. 3. Performance of four schemes versus ε. (a) Sparsity versus ε. (b) Residual versus ε.

summarization based on mean and t-test is also attached at the bottom of Table II, respectively. A win and a loss mean that the classifier being compared to KSRC is better and worse than KSRC on a data set, respectively. So, losses are good for KSRC. A tie means that both classifier have the same performance. The conclusions are summarized as follows.

- KSRC performs well when compared to other linear nonparametric learning methods, namely, KNN, NS, and SRC. For example, KSRC are significantly better than NS in 14 out of 16 data sets. KSRC is significantly better than NN and SRC in seven and eight out of sixteen data sets, respectively.
- When compared to SVM, the state-of-the-art method in machine learning, KSRC only performs better on a few data sets. There is no significant difference between KSRC and SVM in nine out of sixteen data sets.
- The performance of KSRC and KSR are similar to each other. But KSRC has a faster test speed than KSR does, which is supported by the data shown in Table III.

The second conclusion is not surprising. SVM is one of the best general algorithms in machine learning and pattern recognition. In the most case, SVM can achieve good performance by implementing the structural risk minimization rule, or minimizing both the empirical risk and the complexity of hypothesis function space [22]. Moreover, SRC, KSR, and KSRC only take a sparse representation of a sample in terms of all training samples into account. As it is well-known, the best representation does not mean the best separability. Note that the goal of this paper is to improve the classification performance of SRC, and improve test speed of KSR on the given tasks. From Table II, we can see that KSRC work better than SRC on most UCI data sets used here. According to Table III, the test speed of KSRC is 50 times as fast as KSR.

2) Experiment With Dimensionality Reduction: To see which dimensionality reduction methods in the kernel feature space is efficient for KSRC, we choose six relative high-dimensional data sets from Table I: Ionosphere (32D), Musk (166D), Sonar(60D), Soy (208D), Wdbc(30D), and Wpbc(33D), where D denotes dimensionality. We compare KSRC with KNN, SRC, SVM, KSR on the six data sets when using three dimensionality reduction methods, including PCA, FDA and RP. Except KSR and SRC, PCA and FDA mean KPCA and KFDA for other three algorithms, respectively.

We reduce the dimensionality of the original input space or the kernel feature space by half when using PCA/KPCA, and RP. In FDA/KFDA, the dimensionality of subspace is $c-1$, where c is the number of classes. We report the mean of test errors on six UCI data sets in Tables IV–VI with different dimensionality reduction methods, respectively. When PCA/KPCA is used, KSRC is better than SRC on four data sets, SVM on four data sets, and KSR on five data sets, respectively. When Adopting RP, KSRC has the similar performance as the case of KPCA. But, KSRC has a bad performance when applying KFDA. Actually, these five methods almost get a worse performance in the reduced space obtained by using FDA/KFDA than that in the input space. The first reason is that the reduced space has a low dimensionality compared to the space yielded by PCA or RP, so the useful information would be lost. The second reason is that the performance of kernel methods is closely dependent on the selection of kernel parameter. Here, we only consider a simple method for selecting kernel parameter since it is time-consuming for selecting the optimal kernel parameter.

To sum up, It is possible to improve classification accuracy when using dimensionality reduction methods, which can be observed from the comparison of Tables II and IV. PCA/KPCA is effective in SRC-like algorithms. The main reason is that the goal of PCA/KPCA is also to find a good representation of data. Therefore, for SRC-like algorithms, PCA is a good dimensionality reduction method. Alternatively, we can use RP for its low computational complexity.

D. Evaluations on Face Data Sets

Here, we compare KSRC with nonparametric learning methods (SRC, NN, and NS) and SVM on publicly available databases for face database. Three face data sets are considered here, including ORL face database [47], UMIST face database [48], and Extended Yale B database [21], [49] . The original features of each face image is obtained by stacking its columns. Then for a gray-scale face image with the size of $m_1 \times m_2$, we get $m_1 m_2$ features. In each database, we randomly select half of images in each subject as the training samples, and the remaining as the test samples. The RP method is used to perform dimensionality reduction in this experiment. [18] and [19]

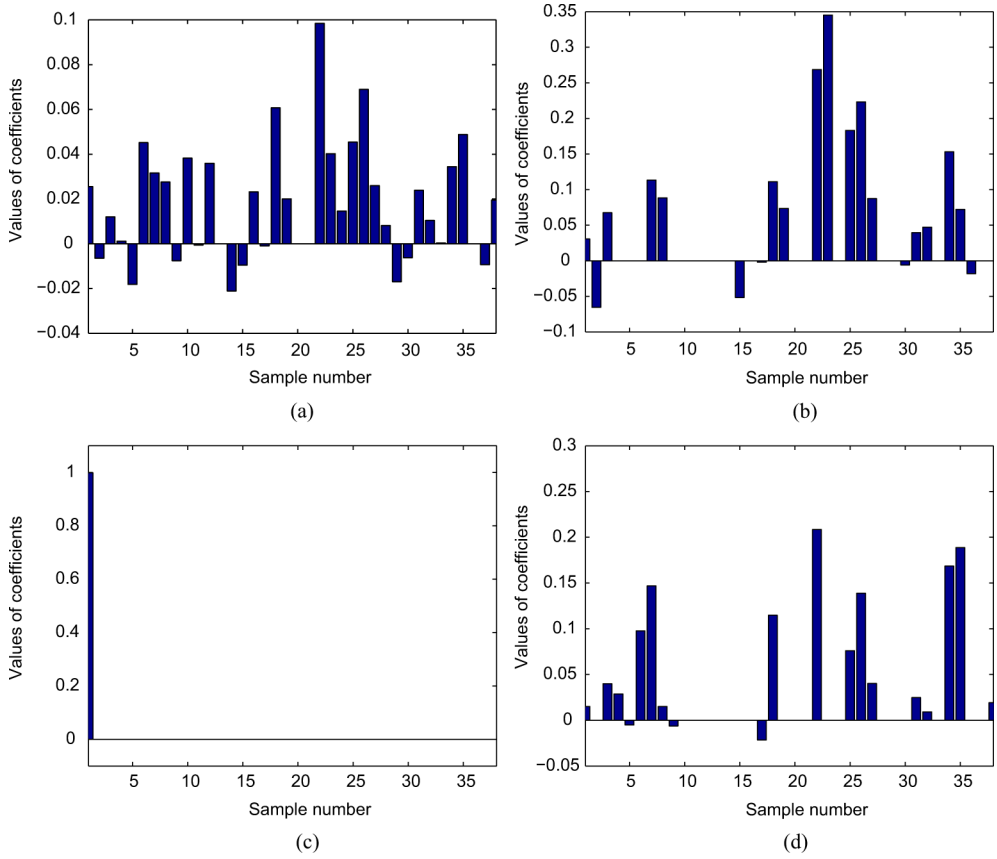


Fig. 4. Coefficient vectors obtained by (a) determinate scheme, (b) KPCA, (c) KFDA, and (d) RP. $\varepsilon = 0.001$.

TABLE I
INFORMATION ON UCI DATA SETS

Data set	Attribute	Class	Sample
Breast	9	2	699
Glass	9	6	214
Heart-Cleveland	13	2	303
Hepatitis	19	2	155
Ionosphere	32	2	351
Iris	4	3	150
Liver	6	2	345
Musk	166	2	476
Pima	8	2	768
Sonar	60	2	208
Soy	208	17	289
Vehicle	18	4	846
Vote	16	2	435
Wdbc	30	2	569
Wine	13	3	178
Wpbd	33	2	198

have showed that the RP method is an effective dimensionality reduction method in face recognition. We run 10 trials. For each trial, random projection is performed 10 times. The linear kernel is considered as in [19]. Let $\varepsilon = 0.0001$ for SRC and KSRC. For SVM, the linear kernel is adopted.

1) ORL Database: There are 10 different images for each subject in the ORL face database composed of 40 distinct subjects. All the subjects are in up-right, frontal position (with tolerance for some side movement). The size of each face image is 112×92, and the resulting standardized input vectors are of dimensionality 10 304. Fig. 5(a). shows 6 images of the same subject. The number of images for both training and test is 200.

We compute the average test error rates with the feature subspace dimensions of 10, 20, 30, 40, 50, 60, 80, 100, 120, and 140. The final results are shown in Fig. 6(a). From this figure, We observe that SRC is much worse than NN, NS, and SVM because the ORL face data contain varying pose. Fortunately, we can see that KSRC can deal with this, and is much better than SRC. The best test error rates for SRC is 25.54% and for KSRC is 5.78%. Moreover, KSRC is better than other four methods when the feature subspace has a low dimensionality, say 20. Actually, KSRC slightly improves its performance when the dimensionality is larger than about 40.

2) UMIST Database: The UMIST face database is a multi-view database which consists of 574 cropped gray-scale images of 20 subjects, each covering a wide range of poses from profile to frontal views as well as race, gender and appearance. Each image in the database is resized into 112×92. Fig. 5(b) depicts some sample images of a typical subset in the UMIST database. The total number of the training samples is 290, and that of the test samples is 284.

The average test errors with the feature subspace dimensions of 20, 40, 60, 80, 100, 120, 140, 160, 180, and 200 are reported in Fig. 6(b). Since the UMIST face database also contain varying pose, SRC also behaves badly. SRC is only better than NS when the dimensionality is smaller than 40. Clearly, KSRC also performs well on the UMIST face database.

3) Extended Yale B Database: The Extended Yale B database has been used in [18], [19]. It consists of 2414 frontal-face images of 38 subjects which are manually aligned, cropped, and

TABLE II
MEAN AND STANDARD DEVIATION OF TEST ERROR RATE (%) ON UCI DATA SETS WITHOUT DIMENSIONALITY REDUCTION

Data set	*K*NN	NS	SRC	SVM	KSR	KSRC
Breast	3.67±1.11	63.24±9.53	54.57±23.09	4.09±1.08	-	5.78±1.11
Glass	33.12±7.05	51.30±6.08	33.77±8.17	30.29±7.05	-	32.46±6.52
Heart-Cleveland	21.60±3.62	51.40±4.77	22.80±3.74	18.70±2.87	22.5±4.28	23.80±4.42
Hepatitis	40.59±5.32	43.53±6.97	45.49±5.53	34.51±5.56	40.00±5.41	38.04±5.64
Ionosphere	15.64±2.82	60.60±2.33	8.21±2.02	5.38±1.40	15.30±3.52	13.42±2.42
Iris	5.83±3.07	20.83±5.81	20.00±7.03	5.63±3.26	5.63±3.11	4.79±1.98
Liver	38.16±5.04	49.47±7.55	35.88±6.35	31.05±3.99	35.26±4.47	32.81±3.83
Musk	15.95±3.89	43.67±0.00	14.75±3.01	6.84±2.81	8.55±3.45	10.00±3.59
Pima	27.49±2.31	57.06±12.01	34.00±2.11	24.67±2.14	-	30.40±2.83
Sonar	18.70±4.86	41.45±3.94	23.48±4.36	12.90±4.50	11.88±3.26	12.46±3.94
Soy	11.65±2.90	4.47±2.06	11.65±2.96	4.35±2.78	4.59±2.11	3.41±1.70
Vehicle	29.21±2.33	33.07±3.24	18.72±1.80	18.50±2.54	-	22.96±2.45
Vote	8.00±1.50	65.45±8.80	7.11±1.69	5.59±1.57	6.28±2.53	7.04±1.68
Wdbc	2.81±1.09	17.46±4.74	6.40±1.45	2.70±1.01	2.54±0.54	3.44±0.72
Wine	3.97±1.42	35.00±5.69	2.41±1.67	0.86±1.22	2.41±1.45	1.55±1.27
Wpbd	25.087±3.91	23.08±0.00	26.46±6.91	20.31±2.03	24.15±2.90	26.00±2.85
W-L-T (mean)	5-11-0	1-15-0	3-13-0	13-3-0	6-6-0	-
W-L-T (*t*-test)	2-7-7	1-14-1	2-8-6	7-0-9	2-2-8	-

TABLE III
COMPARISON OF TEST TIME (S) ON 12 UCI DATA SETS

Data set	KSR	KSRC
Heart-Cleveland	4.51×10^3	11.29
Hepatitis	384.35	1.50
Ionosphere	1.04×10^4	28.44
Iris	421.86	0.51
Liver	1.02×10^4	30.77
Musk	7.11×10^3	37.53
Sonar	349.55	4.68
Soy	1.38×10^3	8.14
Vote	4.57×10^3	81.30
Wdbc	2.95×10^4	113.36
Wine	386.80	2.50
Wpbd	300.60	5.02

TABLE IV
COMPARISON OF FIVE METHODS WITH PCA/KPCA ON 6 UCI DATA SETS

Data set	*K*NN	SVM	KSR	SRC	KSRC
Ionosphere	5.38±1.71	**5.30**±1.84	28.63±5.79	8.12±1.90	11.37±2.02
Musk	12.98±3.44	9.49±1.81	43.48±2.91	12.78±3.32	**9.24**±3.94
Sonar	14.93±2.17	16.81±4.84	17.25±4.71	21.60±3.31	**14.20**±3.40
Soy	10.24±2.77	4.82±3.21	4.92±1.90	**4.12**±1.69	6.12±3.27
Wdbc	4.81±1.23	4.07±1.00	8.68±2.56	5.87±1.01	**2.44**±0.67
Wpbc	29.39±3.44	23.54±1.78	30.31±7.32	24.31±2.79	**22.16**±3.10

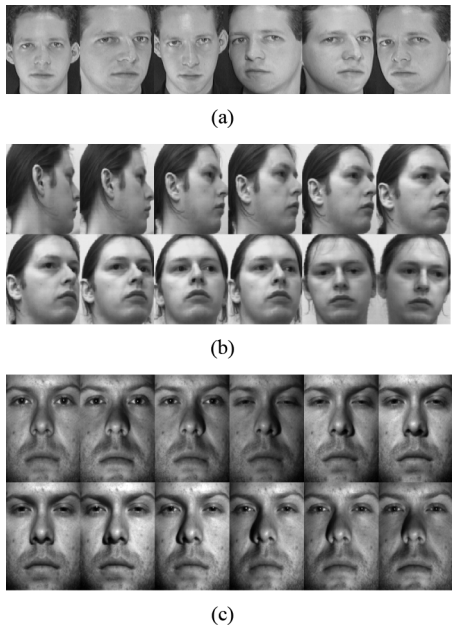
TABLE V
COMPARISON OF FIVE METHODS WITH RP ON SIX UCI DATA SETS

Data set	*K*NN	SVM	KSR	SRC	KSRC
Ionosphere	13.08±1.61	11.20±1.99	13.94±2.21	**8.72**±2.98	12.11±1.65
Musk	15.25±3.87	43.42±0.33	14.22±3.20	13.42±3.01	**10.52**±3.82
Sonar	16.96±2.65	43.48±1.53	15.57±2.47	24.49±4.34	**14.10**±3.53
Soy	9.53±1.61	74.47±5.87	10.14±2.64	5.18±2.09	**4.21**±2.30
Wdbc	5.34±1.51	21.85±2.07	3.35±12.10	9.37±2.30	**2.47**±0.70
Wpbc	31.08±2.27	**23.08**±0.00	31.58±3.76	26.77±5.24	24.22±3.37

TABLE VI
COMPARISON OF FIVE METHODS WITH FDA/KFDA ON SIX UCI DATA SETS

Data set	*K*NN	SVM	KSR	SRC	KSRC
Ionosphere	36.15±9.97	32.39±7.56	**20.09**±8.56	27.18±3.06	34.96±2.97
Musk	9.18±2.47	**7.22**±2.82	31.65±4.52	43.61±0.20	37.72±12.96
Sonar	14.35±3.77	**14.06**±3.42	29.13±4.29	40.15±7.49	27.83±11.22
Soy	4.12±2.56	**3.65**±2.69	52.82±9.63	4.24±1.49	3.88±2.48
Wdbc	12.01±3.88	6.14±1.68	**5.98**±3.20	37.04±0.00	34.24±8.87
Wpbc	25.85±3.05	22.92±3.03	**21.23**±3.68	76.92±0.00	52.46±25.25

then resized to 168x192 images [21]. These images were captured under various laboratory-controlled lighting conditions.



(a) (b) (c)

Fig. 5. Face data. (a) Images of a subject from the ORL database, (b) images of a subject from UMIST database, and (c) images of a subject from Extended Yale B database.

Fig. 5(c) shows some sample images of a typical subset in the Extended Yale B database. The total number of the training samples could be 1207, and that of the test samples is also 1207. Each sample has 32 256 features.

We report the test error rates with the feature subspace dimensions 20, 40, 60, 80, 100, 120, 140, 160, 180, and 200 in Fig. 6(c). The results is consistent with that in [19], SRC is better than NN, NS and SVM. KSRC perform better than SRC. On this face data, KSRC achieves the best performance of 1.85% in the 200-dimensional subspace. KSRC would slightly change the performance when the projected dimensionality is increased to some degree as the cases of ORL and UMIST.

Based on the results of three face databases, we have the following conclusions:

- KSRC mostly behaves better than SRC on the same feature dimensions when using the RP method, which indicates

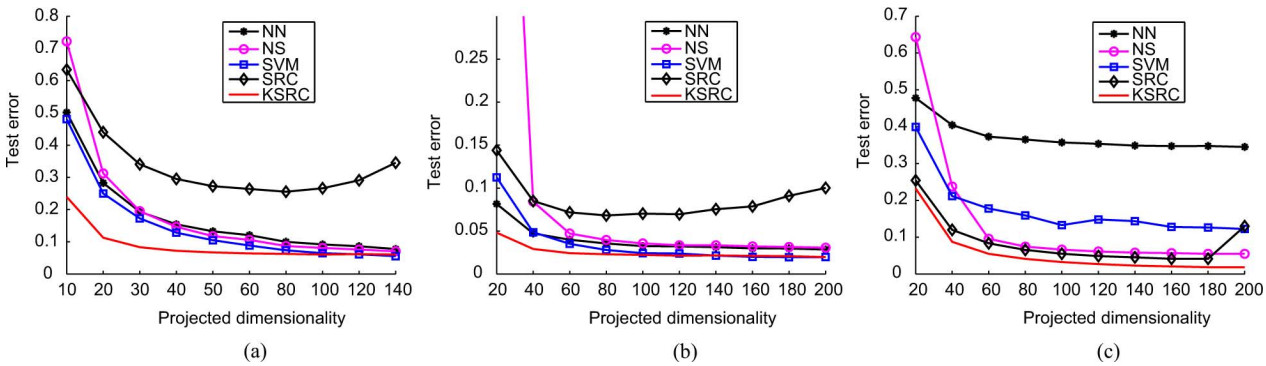


Fig. 6. Mean of test error on three face databases: (a) ORL, (b) UMIST, and (c) Extended Yale B.

that our method improves the classification performance of SRC;

- the RP method is also an effective dimension reduction method when exploiting KSRC to deal with face recognition;
- KSRC is much better than KNN and NS on these face data sets;
- compared to SVM, KSRC obtains a better performance when the reduced space has a low dimensionality on the ORL and UMIST data sets, and much better on the Extended Yale B database.

V. Conclusion

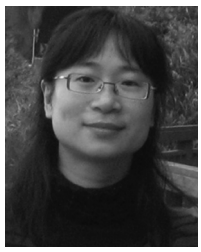
This paper proposes a kernel sparse representation-based classifier which is a nonlinear extension of SRC. On the high-dimensional data such as frontal face images, SRC could get good performance as long as the number of features is large enough. But for a general classification problem in which samples belonging to different classes have the same orientation, SRC loses its classification ability. KSRC can solve this problem by using the RBF kernel, which is supported by experiments on toy data sets. KSR is also a nonlinear extension of SRC, but it can not use the methods available for sparse signal reconstruction and has a long test time. The comparison on UCI data sets shows that KSR and KSRC have similar performance, but KSRC is much faster than KSR. We also perform extensive experiments on public face and compare KSRC with KNN, NS, SRC, and SVM. For face data sets, we reduce the dimensionality by using the RP method. On face data sets containing varying pose (ORL and UMIST) and varying illumination (Extended Yale B), KSRC achieves the best performance. Experimental results indicate that KSRC is a promising nonparametric classifier.

In KSRC, if we choose a parametric kernel, e.g., a RBF kernel, we have to determine the corresponding parameters. Usually, the cross-validation method is adopted to select kernel parameters, which is time-consuming. In our experiments, we simply set the kernel parameter by the median of distances between samples and their mean. Therefore, the performance of RBF-KSRC is worse than SRC on three UCI data sets. So we want to find a method for estimating the kernel parameter in the future.

References

[1] A. Barron, A. Cohen, W. Dahmen, and R. DeVore, "Approximation and learning by greedy algorithm," *Annal. Statist.*, vol. 36, no. 1, pp. 64–94, 2008.
[2] R. Baraniuk, M. Davenport, R. DeVore, and M. Wakin, "A simple proof of the restricted isometry property for random matrices," *Construct. Approx.*, vol. 28, no. 3, pp. 253–263, 2008.
[3] E. Candès and T. Tao, "Decoding by linear programming," *IEEE Trans. Inf. Theory*, vol. 51, no. 12, pp. 4203–4215, Dec. 2005.
[4] E. Candès, J. Romberg, and T. Tao, "Stable signal recovery from incomplete and inaccurate measurements," *Commun. Pure Appl. Math.*, vol. 59, no. 8, pp. 1207–1223, 2006.
[5] E. Candès and M. Wakin, "An introduction to compressed sampling," *IEEE Signal Process. Mag.*, vol. 25, no. 2, pp. 21–30, Mar. 2008.
[6] R. A. DeVore, "Deterministic constructions of compressed sensing matrices," *J. Complex.*, vol. 23, no. 4–6, pp. 918–925, 2007.
[7] D. Donoho, "Compressed sensing," *IEEE Trans. Inf. Theory*, vol. 52, no. 4, pp. 1289–1306, Apr. 2006.
[8] J. M. Duarte-Carvajalino and G. Sapiro, "Learning to sense sparse signals: Simultaneous sensing matrix and sparsifying dictionary optimization," *IEEE Trans. Image Process.*, vol. 18, no. 7, pp. 1395–1408, Jul. 2009.
[9] L. Zhang and W. D. Zhou, "On the sparseness of 1-norm support vector machine," *Neural Netw.*, vol. 23, no. 3, pp. 373–385, 2010.
[10] T. Graepel, R. Herbrich, and J. Shawe-Taylor, "Generalisation error bounds for sparse linear classifiers," in *Proc. 13th Annu. Conf. Comput. Learn. Theory*, Stanford, CA, 2000, pp. 298–303.
[11] S. Floyd and M. Warmuth, "Sample compression learnability, and vapnik-chervonenkis dimension," *Mach. Learn.*, vol. 21, no. 3, pp. 269–304, 1995.
[12] R. Duda, P. Hart, and D. Stork, *Pattern Classification*, 2nd ed. New York: Wiley, 2000.
[13] S. S. Chen, "Basis Pursuit," Ph.D. dissertation, Dept. Statist., Stanford Univ., Stanford, CA, 1995.
[14] R. Tibshirani, "Regression shrinkage and selection via the lasso," *J. Roy .Statist. Soc., Series B: Statist. Method.*, vol. 58, no. 1, pp. 267–288, 1996.
[15] R. Tibshirani, "The lasso method for variable selection in the Cox model," *Statist. Med.*, vol. 16, no. 4, pp. 385–395, 1997.
[16] B. Efron, T. Hastie, L. Johnstone, and R. Tibshirani, "Least angle regression," *Annal. Statist.*, vol. 32, pp. 407–499, 2004.
[17] M. A. T. Figueiredo, R. D. Nowak, and S. J. Wright, "Gradient projection for sparse reconstruction: Application to compressed sensing and other inverse problems," *IEEE J. Sel. Topics Signal Proc.: Special Issue Convex Optimiz. Method. Signal Process.*, vol. 1, no. 4, pp. 586–598, Aug. 2007.
[18] A. Y. Yang, J. Wright, Y. Ma, and S. S. Sastry, Feature Selection in Face Recognition: A Sparse Representation Perspective EECS Dept., Univ. California, Berkeley, CA, 2007, Tech. Rep. Ucb/eecs-2007-99.
[19] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma, "Robust face recognition via sparse representation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 2, pp. 210–226, Feb. 2009.
[20] S. Z. Li, "Face recognition based on nearest linear combinations, IEEE computer society," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Washington, DC, 1998, pp. 839–844.
[21] K.-C. Lee, J. Ho, and D. Kriegman, "Acquiring linear subspaces for face recognition under variable lighting," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 5, pp. 684–698, May 2005.
[22] V. N. Vapnik, *Statistical Learning Theory*. New York: Wiley-Interscience, 1998.

[23] C. J. C. Burges, "A tutorial on support vector machines for pattern recognition," *Data Mining Knowledge Discov.*, vol. 2, no. 2, pp. 121–167, 1998.
[24] A. J. Smola and B. Schölkopf, "A tutorial on support vector regression," *Statist. Comput.*, vol. 14, no. 3, pp. 199–222, 2004.
[25] N. Aronszajn, "Theory of reproducing kernels," *Trans. Amer. Math. Soc.*, vol. 68, no. 3, pp. 337–404, 1950.
[26] S. Saitoh, *Theory of Reproducing Kernels and Its Applications*. Harlow, U.K.: Longman Scientific & Technical, 1988.
[27] K.-R. Müller, S. Mika, G. Rätsch, K. Tsuda, and B. Schölkopf, "An introduction to kernel-based learning algorithms," *IEEE Trans. Neural Netw.*, vol. 12, no. 2, pp. 181–202, Apr. 2001.
[28] T. Hofmann, B. Schölkopf, and A. J. Smola, "Kernel methods in machine learning," *Annal. Statist.*, vol. 36, no. 3, pp. 1171–1220, 2008.
[29] B. Schölkopf, A. J. Smola, and K.-R. Müller, "Nonlinear component analysis as a kernel eigenvalue problem," *Neural Comput.*, vol. 10, no. 5, pp. 1299–1319, 1998.
[30] S. Mika, G. Rätsch, J. Weston, B. Schölkopf, and K.-R. Müller, "Fisher discriminant analysis with kernels," in *IEEE Int. Workshop Neural Netw. Signal Process. IX*, Madison, WI, Aug. 1999, pp. 41–48.
[31] L. Zhang and W. Zhou, "On the sparseness of 1-norm support vector machines," *Neural Netw.*, vol. 23, pp. 373–385, Apr. 2010.
[32] V. Roth, "Sparse kernel regressors," in *Proc. Int. Conf. Artif. Neural Netw., ser. Lecture Notes Comput. Sci.*, 2001, vol. 2130, pp. 339–346.
[33] S. Gao, I. W.-H. Tsang, and L.-T. Chia, "Kernel sparse representation for image classification and face recognition," in *Proc. 11th Eur. Conf. Comput. Vis.: Part IV*, 2010.
[34] E. van den Berg and M. P. Friedlander, "Probing the pareto frontier for basis pursuit solutions," *SIAM J. Sci. Comput.*, vol. 31, no. 2, pp. 890–912, 2008.
[35] S. Becker, J. Bobin, and E. Candès, NESTA: A Fast and Accurate First-Order Method for Sparse Recovery Apr. 2009 [Online]. Available: http://www.acm.caltech.edu/~emmanuel/papers/NESTA.pdf
[36] P. Belhumeur, J. Hespanha, and D. Kriegman, "Eigenfaces versus fisherfaces: Recognition using class specific linear projection," *IEEE Trans. Pattern Anal. Mach. Intell., Special Issue Face Recognit.*, vol. 19, no. 7, pp. 711–720, Jul. 1997.
[37] S. Chen, D. Donoho, and M. Saunders, "Atomic decomposition by basis pursuit," *SIAM Rev.*, vol. 43, no. 1, pp. 129–159, 2001.
[38] E. Candès and J. Romberg, ℓ_1-Magic: Recovery of Sparse Signals via Convex Programming Oct. 2005 [Online]. Available: http://www.acm.caltech.edu/l1magic
[39] L. Zhang, W. D. Zhou, and L. C. Jiao, "Wavelet support vector machine," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 34, no. 1, pp. 34–39, Feb. 2004.
[40] L. Zhang, W. D. Zhou, and L. C. Jiao, "Support vector machines based on the orthogonal projection kernel of father wavelet," *Int. J. Comput. Intell. Appl.*, vol. 5, no. 3, pp. 283–303, 2005.
[41] L. Zhang, W. D. Zhou, and L. C. Jiao, "Hidden space support vector machines," *IEEE Trans. Neural Netw.*, vol. 16, no. 6, pp. 1424–1434, Dec. 2004.
[42] W. D. Zhou, L. Zhang, and L. C. Jiao, "Hidden space principal component analysis," in *Advances in Knowledge Discovery and Data Mining, ser. Lecture Notes in Computer Science*. Berlin, Germany: Springer-Verlag, 2006, vol. 3918, pp. 801–805.
[43] L. Zhang, W. D. Zhou, and P.-C. Chang, "Generalized nonlinear discriminant analysis and its small sample size problems," *Neurocomputing*, vol. 74, no. 4, pp. 568–574, Jan. 2011.
[44] C.-C. Chang and C.-J. Lin, LIBSVM: A Library for Support Vector Machines 2001 [Online]. Available: http://www.csie.ntu.edu.tw/~cjlin/libsvm
[45] T. R. Golub, D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. P. Mesirov, H. Coller, M. L. Loh, J. R. Downing, M. A. Caligiuri, C. D. Bloomfield, and E. S. Lander, "Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring," *Science*, vol. 286, no. 5439, pp. 531–537, 1999.
[46] A. Frank and A. Asuncion, UCI Machine Learning Repository 2010 [Online]. Available: http://archive.ics.uci.edu/ml
[47] F. S. Samaria and A. C. Harter, "Parameterisation of a stochastic model for human face identification," in *Proc. 2nd IEEE Int. Workshop Appl. Comput. Vis.*, Sarasota, FL, Dec. 1994, pp. 138–142.
[48] D. B. Graham and N. M. Allinson, "Characterizing virtual eigensignatures for general purpose face recognition," *Face Recognit.: From Theory to Appl., NATO ASI Series F, Comput. Syst. Sci.*, vol. 163, pp. 446–456, 1998.
[49] A. S. Georghiades, P. N. Belhumeur, and D. J. Kriegman, "From few to many: Illumination cone models for face recognition under variable lighting and pose," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 23, no. 6, pp. 643–660, Jun. 2001.



Li Zhang (M'08) received the B.Sc. and Ph.D. degrees in electronic engineering from Xidian University, Xi'an, China, in 1997 and 2002, respectively.

From 2003 to 2005, she was a Postdoctor at the Institute of Automation of Shanghai Jiao Tong University, Shanghai, China. From 2005 to 2010, she worked as an Associate Professor at Xidian University. She is currently a Full Professor of Soochow University, Suzhou, China. Her research interests include machine learning, pattern recognition, neural networks, and intelligent information processing.

Wei-Da Zhou (M'08) received the B.Sc. and Ph.D. degrees in electronic engineering from Xidian University, Xi'an, China, in 1996 and 2003, respectively.

He was an Associate Professor at the School of Electronic Engineering at Xidian University, Xi'an, China from 2006 to 2009. He has been working at AI Speech Ltd., Jiangsu, China, since 2009. His research interests include machine learning, learning theory, and intelligent information processing.

Pei-Chann Chang received his M.Sc. and Ph.D. degrees in industrial engineering from Lehigh University, Bethlehem, PA, in 1985 and 1989, respectively.

He is currently a Professor at Yuan-Ze University, Taiwan. His research interests include production scheduling, sales forecasting, case-based reasoning, ERP, global logistics, and applications of soft computing. He has published over 80 papers in such journals as IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS—PART C: APPLICATIONS AND REVIEWS, *European Journal of Operational Research*, *International Journal of Production Economics*, etc.

Dr. Chang is a Senior Editor for the *Journal of Chinese Institute of Industrial Engineering*.

Jing Liu (M'06) received the B.Sc. degree in computer science and technology from Xidian University, Xi'an, China, in 2000, and the Ph.D. degree in circuits and systems from the Institute of Intelligent Information Processing of Xidian University, in 2004.

She is currently a Full Professor of Xidian University. Her research interests include evolutionary computation, multiagent systems, and data mining.

Zhe Yan received the B.Sc. degree in pattern recognition and intelligent systems from Xidian University, Xi'an, China, in 2009.

His research interests include pattern recognition.

Ting Wang received the B.Sc. degree in pattern recognition and intelligent systems from Xidian University, Xi'an, China, in 2010.

Her research interests include pattern recognition.

Fan-Zhang Li received the B.Sc. degree in computer science and technology from the University of Science and Technology of China, Hefei, Anhui, in 1995.

From 1996 to 2000, he worked in Yunnan University. He is currently a Full Professor of Soochow University, Suzhou, China. He has published three books on dynamic fuzzy logic and one book on Lie group machine learning. His research interests include artificial intelligence, machine learning, and dynamic fuzzy logic.