Neurocomputing 218 (2016) 172–184

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Nonlinear discriminant analysis based on vanishing component analysis

Yunxue Shao [a,*], Guanglai Gao [a], Chunheng Wang [b]

[a] College of Computer Science, Inner Mongolia University, China
[b] Institute of Automation, Chinese Academy of Sciences, China

ARTICLE INFO

Article history:
Received 16 March 2016
Received in revised form
25 June 2016
Accepted 22 August 2016
Communicated by Zhi Yong Liu
Available online 27 August 2016

Keywords:
Kernel discriminant analysis
Linear discriminant analysis
Vanishing component analysis
Support vector machine
Random forest

ABSTRACT

Most kernel-based nonlinear discriminant analysis methods need to compute the kernel distance between test samples and all of the training samples, but this approach consumes large volumes of time and memory, and it may be impractical when there is a large number of training samples. In this study, we propose a vanishing component analysis (VCA) based nonlinear discriminant analysis (VNDA) method. First, VNDA learns nonlinear mapping functions explicitly using the modified VCA method, before employing these functions to map the input feature onto a high-dimensional polynomial feature space, where the linear discriminant analysis (LDA) method is then applied. We prove that principal components analysis plus LDA is a special case of VNDA and that the set of mapping functions learned by VNDA is the best solution to the ratio trace problem in the degree bounded polynomial feature space. Unlike kernel-based methods, VNDA only stores these mapping functions instead of all the training samples in the test step. Experimental results obtained based on four simulated data sets and 15 real data sets demonstrate that the proposed method yields highly competitive test recognition results compared to the state-of-the-art methods, while consuming less memory and time resources.

1. Introduction

Extracting suitable features for classification or other tasks is fundamentally important, but the feature extraction method may differ significantly for different tasks. Discriminant analysis is often used to find the best feature or feature set to discriminate different classes, and numerous discriminant analysis methods have been proposed for pattern recognition. The best known method is linear discriminant analysis (LDA) [1], where the objective of LDA is obtained by maximizing the between-class projection distance while simultaneously minimizing the within-class projection distance, which can be represented as a ratio trace optimization problem. LDA has been shown to be successful at solving classification problems [2–4], but it still has some disadvantages and thus various extensions of LDA have been proposed. According to [5], these extensions can be categorized into three groups. The first approach is the two stage LDA [6–8], which employs an intermediate dimensionality reduction stage before applying LDA. The second approach is the regularized LDA [9–11], which adds a perturbation to the total scatter matrix to avoid the singularity problem. The third approach [5,7,12–14] applies the pseudo-inverse to solve the singularity problem. However, LDA and its variants fail to solve nonlinear problems. To address this limitation, nonlinear extensions of LDA have been proposed based on the kernel method. Kernel Fisher discriminant analysis [15] and generalized discriminant analysis [16] are two independently developed kernel-based discriminant analysis methods, but they are essentially equivalent so we refer to them as kernel discriminant analysis (KDA) in the following. The main idea of KDA is to map the input data onto another feature space by nonlinear mapping, where the inner products in the mapped feature space can be computed by a kernel function without knowing the nonlinear mapping functions explicitly. The KDA optimization problem often involves the singularity problem of the total scatter matrix. The traditional approaches for addressing this issue employ the eigen-decomposition of the kernel matrix, which is computationally expensive. Mika et al. [17] used a greedy approximation technique to speed up KDA for binary classification problems. Cai et al. [18] proposed an algorithm for KDA called spectral regression kernel discriminant analysis (SRKDA). Using spectral graph analysis, SRKDA casts discriminant analysis into a regression framework, which facilitates both efficient computation and the use of regularization techniques. In addition, various other methods based on the kernel method have been proposed for nonlinear discriminant analysis. Cawley and Talbot [19] demonstrated that

* Corresponding author.
E-mail addresses: csshyx@imu.edu.cn (Yunxue Shao), csggl@imu.edu.cn (G. Gao), chunheng.wang@ia.ac.cn (C. Wang).

http://dx.doi.org/10.1016/j.neucom.2016.08.058
0925-2312/

leave-one-out cross-validation of kernel Fisher discriminant classifiers can be implemented significantly faster than the conventional k-fold cross-validation procedure. Yang et al. [20] proposed a complete kernel Fisher discriminant framework based on KPCA plus LDA. Lei et al. [21] proposed a feature space locality constraint-based kernel nonlinear discriminant analysis method to integrate the sample geometric structure information into the traditional kernel-based methods. Zhang et al. [22] proposed generalized nonlinear discriminant analysis to deal with the small sample size problem in LDA.

However, most of these kernel-based methods need to calculate the kernel distances between test samples and all of the training samples, and thus they consume large amounts of time and memory, which may be impractical for some applications. Xu et al. [23] proposed a fast kernel-based nonlinear method (FKNM) to handle this problem. However, FKNM is designed for binary classification problems and one-against-one or one-against-the-rest strategies should be employed for multi-class classification. In the present study, instead of storing the training samples and calculating the kernel distances between test samples and the training samples in the test step, we learn the nonlinear mapping functions explicitly and we propose a new nonlinear discriminant analysis method based on vanishing component analysis (VCA) [24]. First, the proposed VCA-based nonlinear discriminant analysis (VNDA) method learns a set of polynomial generators using the modified VCA method. It can be proved that any degree bounded polynomial that can extract useful feature for classification can be represented as a linear combination of these generators. These generators are then used to map the input data onto a high-dimensional feature space, where the LDA method is then applied to learn a projection matrix. Experimental results and statistical comparisons [25] demonstrate that the proposed method achieves equivalent test recognition results compared to the state-of-the-art methods, while consuming much less memory and time resources. The main contributions of this study comprise the following.

(a) We propose an effective and efficient nonlinear discriminant analysis method called VNDA, which uses the non-vanishing polynomial features of the whole samples instead of using the vanishing ideals of each class as the intermediate feature domain. We prove that principal components analysis (PCA) plus LDA is a special case of VNDA and that the set of polynomials produced by the VNDA method is the best solution to the ratio trace problem in the degree bounded polynomial feature space. These advantages demonstrate that the proposed method has theoretical significance.
(b) We performed detailed evaluations of the proposed method and state-of-the-art methods using four well designed simulated data sets and 15 various real data sets. The experimental results obtained using simulated data sets with various distributions demonstrated that the proposed method achieved comparable cross-validation accuracy with these data sets while delivering better generalizability than the other methods. The experimental results obtained using the real data sets showed that the proposed method achieved competitive test recognition results while consuming far less memory and time resources than the kernel-based nonlinear discriminant analysis methods, especially when there was a large number of training samples.

The remainder of this paper is organized as follows. In the next section, we briefly review the VCA method. In Section 3, we describe the proposed VCA-based nonlinear discriminant analysis method. The results of experiments performed using simulated and real data sets are presented to demonstrate the effectiveness and efficiency of the proposed method in Section 4. Finally, we discuss this study and suggest future research in Section 5.

2. Brief review of VCA

According to [24], the aim is to obtain a set of polynomial functions $f_1(x), \ldots, f_n(x)$ such that $f_i(x) \approx 0$ for all i and $x \in S$, where the set of points S belongs to a particular class. These functions capture the nonlinear structure of the data and they can be used for supervised learning. The set of all polynomials $f(x)$ that reach a value of zero for a data set S is known as the vanishing ideal of S, which is denoted by $I(S)$. However, this set contains an infinite number of polynomials. The VCA method allows us to find a set of generators $V = \{f_1(x), \ldots, f_k(x)\}$ of the ideal $I(S)$ such that any polynomial $g(x) \in I(S)$ can be represented as

$$g(x) = \sum_i f_i(x) h_i(x) \tag{1}$$

for some polynomials $h_i(x)$.

In the VCA algorithm, two sets are maintained: the vanishing polynomials set V and the non-vanishing polynomials set F. V and F are initialized as an empty set $\varnothing$ and constant set $\{f(x) = 1/\sqrt{m}\}$, respectively, where m is the number of training samples in S. The initial candidate set C_1 is initialized as $\{f_1(x) = x_1, \ldots, f_n(x) = x_n\}$, $x \in \mathbf{R^n}$. At iteration t, given a candidate set $C_t = \{f_1, \ldots, f_k\}$ and a set of polynomials F, they first calculate $\tilde{f}_i$ by

$$\tilde{f}_i = f_i - \sum_{g \in F} \langle f_i(S_m), g(S_m) \rangle g, \tag{2}$$

as the remainder of the polynomials in C after projecting them onto F. They then use singular value decomposition (SVD) to find the range and null space of the matrix $A = [\tilde{f}_1(S_m), \ldots, \tilde{f}_k(S_m)]$ by calculating the right singular vectors of A. Let

$$A = LDU^T \tag{3}$$

and

$$g_i = \sum_{j=1}^{k} U_{j,i} \tilde{f}_j, \tag{4}$$

and if $D_{i,i} \le \epsilon$, then push g_i into the current vanishing set V_t; otherwise, push $g_i / \| g_i(S_m) \|$ into the current non-vanishing set F_t. Finally, update set $F = F \bigcup F_t$, $V = V \bigcup V_t$, and $C_{t+1} = \{gh: g \in F_t, h \in F_1\}$. If the proposed VCA method is run with $\epsilon = 0$, it can be proved that:

Theorem 1. *(1) V generates $I(S_m)$; and (2) any polynomial f can be written as $f = g + h$, where $g \in span(F)$ and $h \in I(S_m)$.*

To employ the VCA method for classification, VCA is applied to the training samples S^l for each class l. The output of the VCA is a set of polynomials $\{f_1^l(x), \ldots, f_{n_l}^l(x)\}$, which generates the vanishing ideal of class l, where n_l denotes the number of polynomials generated by VCA on S^l. We then use the mapping function

$$x \mapsto \left(|f_1^1(x)|, \ldots, |f_{n_z}^z(x)| \right) \tag{5}$$

to generate a new training data set from $S = \{S^1 \bigcup S^2 \bigcup \cdots \bigcup S^z\}$. Finally, LDA and regularized Euclidean distance are employed for linear classification based on the new training data set.

3. VCA-based nonlinear discriminant analysis

Assume that we have m samples from z classes. Let m_i represent the training number of the ith class and $x_j^i \in \mathbf{R}^n$ denotes

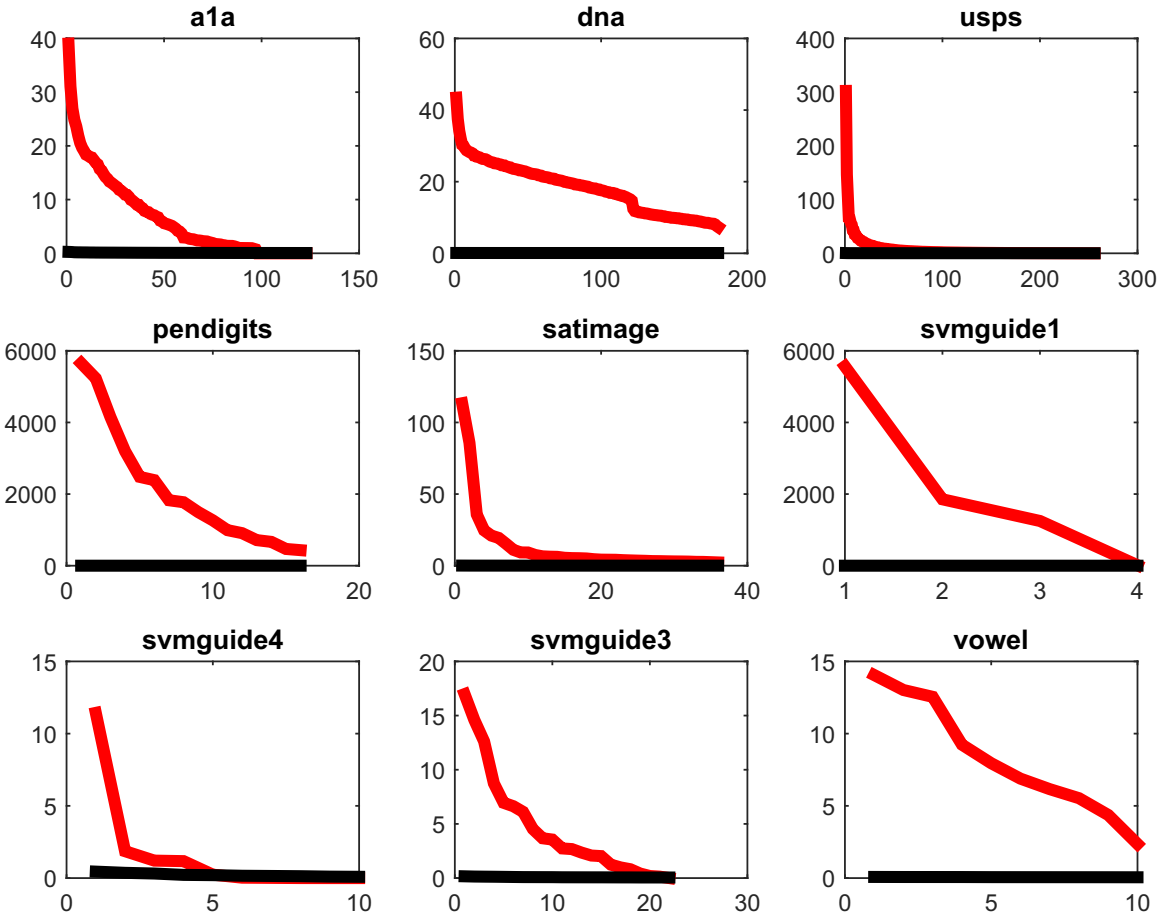

Fig. 1. Illustration of the singular values decomposed by polynomials with degree 1 (the top red line) and degree 2 (the bottom black line) using different data sets. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this paper.)

the jth sample of the ith class, $i = 1, 2, \ldots, z$, $j = 1, 2, \ldots, m_i$. Let $S^i = \{x^i_1, \ldots, x^i_{m_i}\}$ and $S = \{S^1 \bigcup S^2 \bigcup, \ldots, \bigcup S^z\}$. The target of the previously proposed method [24] is to find a set of generators V^i for each class. All of the polynomials in the set $\bigcup_{i=1}^{z} V^i$ are used as mapping functions during classification, where they find the common denominators of samples in the same class. In the present study, we consider that any polynomial $f(x)$ that does not satisfy $f(x) = 0$ for all $x \in S$ may be useful for discriminating some classes, and we propose the VNDA method. An infinite number of polynomials satisfy this constraint. Fortunately, the VCA method allows us to find a set of generators. The main idea of VNDA is that we apply the modified VCA method to the whole data set S and we obtain the vanishing component set V and the non-vanishing component set $F = \{p_1, \ldots, p_q\}$. According to Theorem 1, any polynomial f can be written as

$$f(x) = g(x) + h(x), \tag{6}$$

where $g \in span(F)$ and $h \in I(S)$. Any $h \in I(S)$ satisfies $h(x) = 0$ for all x in S, so we obtain

$$f(x) = g(x) + h(x) = \sum_{i=1}^{q} w_i p_i(x), \tag{7}$$

which implies that any useful feature extraction polynomial function $f(x)$ can be linearly represented by the polynomials in set F. This is very exciting because we can use a linear optimization method for optimal nonlinear feature extraction in the degree bounded polynomial space, where we need to learn a set of weights for the polynomials in F.

In the VCA algorithm, the candidate set C_t at iteration t is constructed from F_1 and F_{t-1}, where the polynomials in F_t are determined by the parameter ϵ. If the corresponding singular value $D_{i,i} > \epsilon$, then add this polynomial to F_t. In our experiments, we found that the singular values decomposed by polynomials with degree 1 were often much bigger than the singular values decomposed by polynomials with higher degree, where Fig. 1 illustrates this result. This figure also shows that the singular values of higher degree are very close to each other. Given this condition, if we want to explore a structure of higher degree, then we should set ϵ as a small value, which is also difficult to determine. This often means that all of the polynomials with degree 1 are included in the non-vanishing set F_1. If the original feature dimension is high (e.g., 1000), then we may obtain a very high-dimensional candidate set even at the second iteration (e.g., 1000^2). To overcome this issue, we set F_t as the corresponding right singular vectors of the k-largest singular values at iteration t, where k is determined by

$$\sum_{i=1}^{k} D_{i,i} \geq \beta \sum_{i=1}^{m} D_{i,i}. \tag{8}$$

On the one hand, if β is too small then F_t loses too much discriminant information which results in a poor test accuracy. On the other hand, if β is too big then the dimension of F_t is still too high, which increases the time and memory consumption and may result in the curse of dimensionality. The parameter β is selected by cross-validation in this paper. If $\beta = 1$, then the candidate set constructed using this method is equal to the set constructed by the VCA method with $\epsilon = 0$. In other cases, to compensate for discarding the minor singular values, we add some discriminating

components U_L to F_t at iteration t by finding the projection matrix from the space P_S to the sample label Y

$$U_L = P_S^{\ddagger} Y, \tag{9}$$

where P_S is the remainder of the candidates after projecting onto the set $F(S)$, as described formally in Algorithm 2. This projection matrix with a small number of polynomials saves most of the discrimination information that is ignored by discarding the minor singular values. We must note that if the category number is also very large, then this compensation process is impractical and this issue remains an open problem.

Algorithm 1. Framework of the VNDA method.

Require: Training samples $S = \{S^1 \bigcup S^2 \bigcup, \ldots, \bigcup S^z\}$ with label Y, degree d, proportions α and β, and the regularization coefficient δ for LDA.

$F = \{f(\cdot) = 1/\sqrt{m}\}$
$C_1 = \{f_1, \ldots, f_n\}$ where $f_i = x_i$
$(F_1, e_1) = FindBasis(F, C_1, S, Y, \alpha, \beta)$
$F = F \bigcup F_1$
for $t = 2, \ldots, d$ **do**
  $C_t = \{gh, g \in F_{t-1}(1{:}\, e_{t-1}), h \in F_1(1{:}\, e_1)\}$
  if $C_t = \varnothing$ **then**
    break
  end if
  $(F_t, e_t) = FindBasis(F, C_t, S, Y, \alpha, \beta)$
  $F = F \bigcup F_t$
end for
$\hat{S} = F(S)$
$[P, D] = LDA(\hat{S}, \delta)$
return Non-vanishing polynomial set F and the projection matrix P.

The details of the proposed VNDA method are described in Algorithms 1 and 2. In these algorithms, $F(1{:}\, e)$ denotes the vectors from the first column to the e-th column in set F and $F(S)$ is an $m \times n$ matrix, which denotes the evaluations of n polynomials in F based on m samples in S. The *FindBasis* procedure shown in Algorithm 2 is used to construct the new polynomials in F_t from the candidates in C_t. In this procedure, the selected non-vanishing polynomials are determined by α and the k principal components used to construct the higher degree candidates are determined by β. If we apply the VNDA method with $\beta = 1$, then the added U_L can be represented by F_t. In this case, the modified method is equivalent to the original proposed VCA method. Therefore, Theorem 1 is tenable. To demonstrate the superior performance of the proposed method, we provide two theorems related to the VNDA, which show that the proposed method has theoretical importance.

Algorithm 2. Framework of the *FindBasis* procedure.

Require: $F, C, S, Y, \alpha, \beta$
$S_F = F(S),\ S_C = C(S)$
$P_C = S_F^{\ddagger} S_C,\ P = \begin{bmatrix} -P_C \\ I \end{bmatrix},\ P_S = S_C - S_F P_C$
$P_S = LDU^T$ using SVD
$e = j$ where $\sum_{i=1}^{j} D(i,i) > \alpha \sum_{i=1}^{m} D(i,i)$
$g = PU(1{:}\, e)$
$k = j$ where $\sum_{i=1}^{j} D(i,i) > \beta \sum_{i=1}^{e} D(i,i)$
$U_L = P_S^{\ddagger} Y$
$g = [PU_L, g]$
$F_1 = g / \| g(S) \|$
$e_1 = k + z$
return F_1 and e_1.

Theorem 2. *PCA plus LDA is a special case of VNDA.*

Proof. In the VNDA method, F_1 is first computed by the *FindBasis* procedure, where S_C equals the original samples S and S_F equals the constant vector $(1/\sqrt{m}, \ldots, 1/\sqrt{m})^T$ when computing F_1. Therefore,

$$P_C = S_F^{\ddagger} S_C = (\sqrt{m}/m, \ldots, \sqrt{m}/m) S. \tag{10}$$

It follows that

$$\begin{aligned} P_S &= S_C - S_F P_C \\ &= S - (1/\sqrt{m}, \ldots, 1/\sqrt{m})^T (\sqrt{m}/m, \ldots, \sqrt{m}/m) S \\ &= S - mean(S), \end{aligned} \tag{11}$$

where $mean(S)$ denotes the matrix in which every row vector is the mean vector of S. This implies that the vectors in F_1 are equivalent to the eigenvectors produced by applying the PCA method to S. If we bound the degree of VNDA to 1, then the non-vanishing polynomials in set F are equivalent to the linear projection vectors produced by PCA. This proves Theorem 2.□

Theorem 3. *The set of polynomials produced by the VNDA method is the best combination of polynomials in the degree bounded polynomial feature space $\mathcal{P}^d$ that optimizes*

$$\underset{w}{\operatorname{argmax}}\ tr\{(w^T S_w w)^{-1} (w^T S_b w)\}, \tag{12}$$

where S_w and S_b are the within-class covariance and between-class covariance in $\mathcal{P}^d$, respectively.

Proof. Suppose that $\mathcal{P}^d$ is constructed by polynomials $\Phi\{\varphi_1, \varphi_2, \ldots, \varphi_\eta, \varphi_{\eta+1}, \ldots, \varphi_\zeta\}$, where φ_1 to φ_η are the non-vanishing polynomials and $\varphi_{1+\eta}$ to φ_ζ are the vanishing polynomials. $\forall\, x$ in $\mathbf{R}^n$, x can be mapped onto the space $\mathcal{P}^d$ by

$$\begin{aligned} \Phi(x) &= (\varphi_1(x), \ldots, \varphi_\eta(x), \varphi_{\eta+1}(x), \ldots, \varphi_\zeta(x))^T \\ &= (\varphi_1(x), \ldots, \varphi_\eta(x), 0, \ldots, 0)^T. \end{aligned} \tag{13}$$

By ignoring the zero features and rewriting $\Phi(x)$, then according to Theorem 1 we obtain

$$\Phi(x) = \left(\sum_{j=1}^{q} w_j^1 p_j(x), \ldots, \sum_{j=1}^{q} w_j^\eta p_j(x) \right)^T = WF(x), \tag{14}$$

where $p_1, \ldots, p_q$ comprises the non-vanishing set F. The within-class covariance S_w in $\mathcal{P}^d$ is computed by

$$\begin{aligned} S_w &= \sum_i \sum_j (\Phi(x_{i,j}) - \sum_k \Phi(x_{i,k})) \left(\Phi(x_{i,j}) - \sum_k \Phi(x_{i,k}) \right)^T \\ &= \sum_i \sum_j (WF(x_{i,j}) - \sum_k WF(x_{i,k})) \left(WF(x_{i,j}) - \sum_k WF(x_{i,k}) \right)^T \\ &= \sum_i \sum_j W(F(x_{i,j}) - \sum_k F(x_{i,k})) \left(F(x_{i,j}) - \sum_k F(x_{i,k}) \right)^T W^T \\ &= W S'_w W^T, \end{aligned} \tag{15}$$

where S'_w is the within-class covariance in the space spanned by F. In a similar manner, the between-class covariance S_b can be represented by the between-class covariance S'_b in the space spanned by F. Based on these results, Eq. (12) can be rewritten as

$$\begin{aligned}&\underset{w}{\operatorname{argmax}}\, tr\{(w^T S_w w)^{-1}(w^T S_b w)\}\\&= \underset{w}{\operatorname{argmax}}\, tr\{(w^T W S'_w W^T w)^{-1}(w^T W S'_b W^T w)\}\\&= \underset{w'}{\operatorname{argmax}}\, tr\{(w'^T S'_w w')^{-1}(w'^T S'_b w')\}.\end{aligned} \tag{16}$$

Therefore, Theorem 3 is proved. □

4. Experiments

4.1. Data sets and setup

To prove the generalizability of the proposed method for data sets with various distributions, we generated four simulated data sets called *XOR*, *RDM*, *TRI*, and *CIR*. Without loss of generality, we generated two two-dimensional classes and studied each type of simulated data.

For the *XOR* data set, a set of 200 points in the first class was generated according to $X \sim N(0, \nu)$, $Y \sim N(1, \nu)$ and another set of 200 points in the first class was generated according to $X \sim N(1, \nu)$, $Y \sim N(0, \nu)$. The first 100 points in each set were used for the learning step and the remaining points were used for the test step. The points in the second class were generated in a similar manner except that these two sets were generated according to $X \sim N(0, \nu)$, $Y \sim N(0, \nu)$ and $X \sim N(1, \nu)$, $Y \sim N(1, \nu)$. The parameter ν controls the complexity of the problem and it was selected from {0.1, 0.2, 0.3}.

In the *RDM* data set, the points in each class comprised points generated from τ normal distributions $X \sim N(\mu_x^i, \nu_x^i)$, $Y \sim N(\mu_y^i, \nu_y^i)$. In particular, we first generated 2τ centers $(\mu_x^1, \mu_y^1), \ldots, (\mu_x^{2\tau}, \mu_y^{2\tau})$ from $U(0, 1)$ and we then found the nearest neighbor (μ_x^j, μ_y^j) of each center (μ_x^i, μ_y^i), where we set the variance of the i-th normal distribution by $(\nu_x^i, \nu_y^i) = (|\mu_x^i - \mu_x^j|/3, |\mu_y^i - \mu_y^j|/3)$. A set of 200 points was generated for each normal distribution, where the first 100 points were used for the learning step and the remaining points were used for the test step. The points generated from the first τ normal distributions were appended to the set for the first class. The remaining points were appended to the set for the second class. The parameter τ controls the complexity of the problem and it was selected from {3, 4, 5}. The *RDM* problem can be considered a general case of the *XOR* problem.

The training data set for *TRI* was generated as follows. The first feature of each class was generated randomly from the uniform distribution $U(-\iota, \iota)$. The second feature values in the first and second classes were the $\sin(\cdot)$ and $\cos(\cdot)$ values for the corresponding first feature value, respectively. Normal noise with a variance of 0.1 was added to the second feature value in each class. A set of 200 points was generated for each class. The parameter ι controls the complexity of the problem and it was selected from $\{\pi, 2\pi\}$. The test data set for *TRI* was generated in the same manner except that the first feature of each class was generated randomly from the uniform distribution $U(-\iota_t - \iota, -\iota)$ and $U(\iota, \iota + \iota_t)$, where ι_t controls the variation of the test points and it was selected from $\{\pi/4, \pi/2, 3\pi/4\}$.

The training data set for *CIR* was generated as follows. We assumed that T_1 was generated randomly from the uniform distribution $U(0, a)$ and T_2 was generated randomly from the uniform distribution $U(2\pi - a, 2\pi)$, where a controls the angle of the arcs and it was selected from $\{7\pi/6, 5\pi/3\}$. The first and the second features in the first class were computed by $r_* \sin(T_1) + 1$ and $r_* \cos(T_1) + 1$, respectively. The first and the second features in the second class were computed by $r_* \sin(T_2)$ and $r_* \cos(T_2)$, respectively. Normal noise with a variance of 0.1 was added to the two feature values in each class. A set of 200 points was generated for each class. The parameter r is the radius of the circles and it was selected from {2, 3, 4}. The test data set for *CIR* was generated in the same manner, except that T_1 was generated randomly from the uniform distribution $U(11\pi/6, 2\pi)$ and $U(a, a + \pi/6)$, T_2 was generated randomly from the uniform distribution $U(11\pi/6 - a, 2\pi - a)$, and $U(0, \pi/6)$.

The training points and test points in the four data sets are illustrated in Fig. 2.

The experimental results were then confirmed based on 15 real data sets obtained from various data sources. The structures of the real data sets are summarized in Table 1. All of our experiments were implemented using Matlab 2014 and they were conducted on a server with an Intel Xeon 2.10 GHz and 16 GB of memory. To facilitate reproducibility, we have provided all of the source codes and data sets used in our experiments.[1] The training and test data sets can also be downloaded from the LibSVM website.[2]

We first compared the proposed VNDA method with the state-of-the-art discriminant analysis methods LDA, orthogonal LDA (OLDA) [12], KDA, SRKDA, and the original VCA method. To ensure a fair comparison between VNDA and VCA, we also applied LDA to the new training data set generated by Eq. (5). This comparison tested the novelty of VNDA compared with VCA. LDA and OLDA provide a baseline for the performance of linear methods. KDA and SRKDA are state-of-the-art nonlinear discriminant analysis methods. In our experiments, the radial basis function kernel was employed in KDA and SRKDA. The kernel width parameter γ and the regularization parameter θ used in KDA and SRKDA were selected from {10000, 1000, 100, 10, 1, 0.1, 0.01, 0.001} and {100, 10, 1, 0.1, 0.01, 0.001, 0.0001}, respectively, based on five-fold cross-validations of the training set. The parameter ϵ and the regularized value used in VCA were selected from {1000, 500, 100, 50, 10, 5, 1, 0.5, 0.1, 0.05, 0.01, 0.005, 0.001, 0.0005, 0.0001, 0.00005, 0.00001} and {100, 10, 1, 0.1, 0.01, 0.001, 0.0001} , respectively. The parameters α, β, and δ, which are used in the proposed VNDA method, were selected from {0.1: 0.1: 1}, {0.1: 0.1: 1} and {100, 10, 1, 0.1, 0.01, 0.001, 0.0001}, respectively, by five-fold cross-validation. The regularized Euclidean distance obtained by Eq. (17) was used to evaluate the classification performance.

$$f_{ed} = \sum_i \frac{(x_i - m_i)^2}{\delta_i}. \tag{17}$$

We then employed SVM [26,27] and Random Forest (RF) [28–30] as classifiers and evaluated the performance of SVM and RF with the original feature space and the transformed feature space using KDA, SRKDA and VNDA. The tradeoff parameter and the kernel width parameter used in SVM (LibSVM [31] was adopted) were selected from {1, 10, 100, 1000} and {1, 0.1, 0.01, 0.001, 0.0001, 0.00001, 0.000001, 0.0000001}, respectively, based on a five-fold cross validation. The number of decision trees used in RF (Matlab function *TreeBagger* was adopted) was selected from {1, 10, 50, 100, 500, 1000} based on a five-fold cross validation.

4.2. Results obtained using the simulated data sets

This experiment demonstrated that the proposed method exhibited comparable generalizability to the state-of-the-art methods using data sets with various distributions. In the next experiment, we evaluated the performance of these methods in terms of their resource consumption. In this experiment, the

[1] http://202.207.12.136:81/vnda/vnda.html
[2] http://ntucsu.csie.ntu.edu.tw/cjlin/libsvmtools/datasets/

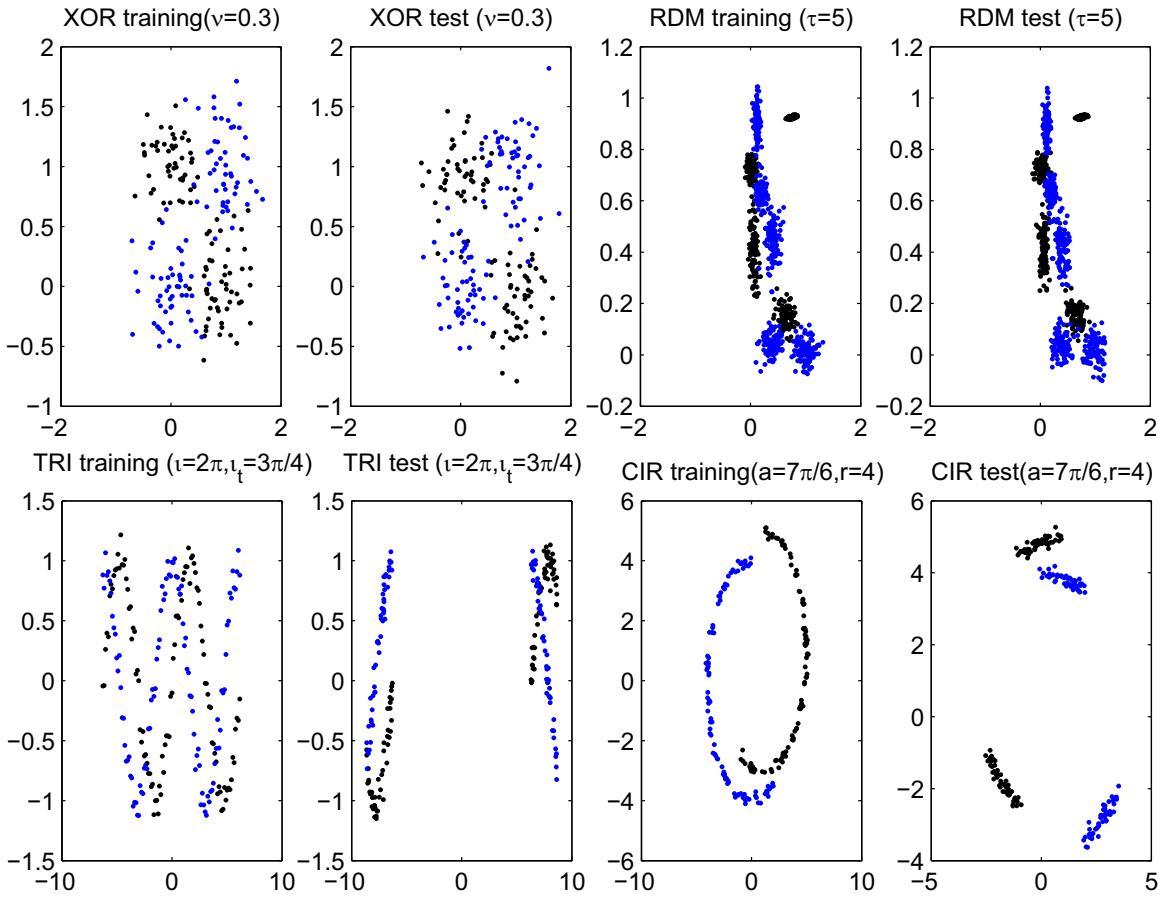

Fig. 2. Illustration of the training points and test points in the four data sets.

Table 1
Summary of the contents of the real data sets.

Data set	# features	# training	# test	# classes
acoustic	50	78,823	19,705	3
a1a	123	1605	30,956	3
a2a	123	2265	30,296	3
a3a	123	3185	29,376	3
a4a	123	4781	27,780	3
dna	180	2000	1186	3
letter	16	15,000	5000	26
madelon	500	2000	600	2
mnist	780	60,000	10,000	10
pendigits	16	7494	3498	10
protein	357	17766	6621	3
satimage	36	4435	2000	6
seismic	50	78,823	19,705	3
svmguide1	4	3089	4000	2
svmguide3	21	1243	41	2
svmguide4	10	300	312	6
usps	256	7291	2007	10
vowel	10	528	462	11

Table 2
Results obtained using the *XOR* data set.

ν	Method	CV (%)	Test (%)	Time (s)	Memory (double)
0.1	LDA	–	51	0.0000	2
	KDA	100	**100**	0.0010	1200
	SRKDA	100	**100**	0.0000	1200
	VNDA	100	**100**	0.0010	40
	VCA	100	**100**	0.0010	20
0.2	LDA	–	51	0.0000	2
	KDA	100	98	0.0000	1200
	SRKDA	100	98	0.0010	1200
	VNDA	100	**98.5**	0.0000	273
	VCA	100	98	0.0000	70
0.3	LDA	–	52.5	0.0010	2
	KDA	93.5	**91.5**	0.0010	1200
	SRKDA	93.5	91	0.0000	1200
	VNDA	93.5	**91.5**	0.0000	40
	VCA	94	91	0.0000	70

The bold values indicate the best results obtained under the current parameter settings.

maximum degree used in VNDA and VCA was selected from {2, 3, …, 20} based on a five-fold cross validation. The performance of OLDA in this experiment is not shown because it was similar to the performance of LDA.

First, the results obtained using the *XOR* data set are shown in Table 2. In this and the following tables, "CV (%)" denotes the cross-validation accuracy for different methods based on the training data set, "Test (%)" is the test accuracy using different methods, "Time (s)" is the CPU time (seconds) consumed by different methods, and "Memory (double)" is the memory (number of double type) consumed by different methods. This table shows that LDA failed to solve this problem, whereas KDA, SRKDA, VNDA, and VCA achieved comparable cross-validation accuracy and test recognition accuracy using this data set. In particular, all of the four nonlinear methods achieved a recognition rate of 100% when $\nu = 0.1$. This problem became nonseparable as ν increased. The proposed method and the other nonlinear methods obtained comparable test recognition results, which were much better than the results obtained using linear methods. Thus, the results of the experiments performed using this data set demonstrate that the

Table 3
Results obtained using the *RDM* data set.

τ	Method	CV (%)	Test (%)	Time (s)	Memory (double)
3	LDA	–	64.50	0.0000	2
	KDA	99.33	98.66	0.0080	1800
	SRKDA	99.33	**98.83**	0.0080	1800
	VNDA	99.33	97.83	0.0020	2143
	VCA	96.83	96.66	0.0020	1794
4	LDA	–	75.62	0.0000	2
	KDA	98.62	97.50	0.0120	2400
	SRKDA	98.62	96.87	0.0140	2400
	VNDA	99.12	**97.87**	0.0020	3319
	VCA	95.25	92.87	0.0020	1794
5	LDA	–	60.20	0.0000	2
	KDA	98.90	**98.10**	0.0210	3000
	SRKDA	98.70	97.90	0.0230	3000
	VNDA	98.60	97.80	0.0110	22699
	VCA	96.40	84.80	0.0020	96

The bold values indicate the best results obtained under the current parameter settings.

Table 4
Results obtained using the *TRI* data set.

ι	ι_t	Method	CV (%)	Test (%)	Time (s)	Memory (double)
π	$\pi/4$	LDA	–	49.50	0.0000	2
		KDA	97.00	57.50	0.0030	600
		SRKDA	97.00	59.50	0.0020	600
		VNDA	97.50	62.50	0.0040	1967
		VCA	91.50	**71.50**	0.0030	2970
	$\pi/2$	LDA	–	49.50	0.0010	2
		KDA	97.00	49.00	0.0040	600
		SRKDA	97.00	43.00	0.0030	600
		VNDA	97.50	**72.50**	0.0030	1967
		VCA	91.50	71.50	0.0020	2970
	$3\pi/4$	LDA	–	50.00	0.0010	2
		KDA	97.00	43.00	0.0040	600
		SRKDA	97.00	39.00	0.0020	600
		VNDA	97.50	**69.50**	0.0020	1967
		VCA	91.50	68.00	0.0030	2970
2π	$\pi/4$	LDA	–	52.00	0.0010	2
		KDA	96.50	56.00	0.0010	600
		SRKDA	96.50	**75.00**	0.0020	600
		VNDA	95.00	73.00	0.0010	1586
		VCA	81.50	72.50	0.0030	6902
	$\pi/2$	LDA	–	46.50	0.0010	2
		KDA	96.50	53.50	0.0020	600
		SRKDA	96.50	48.50	0.0030	600
		VNDA	95.00	62.00	0.0030	1586
		VCA	81.50	**63.00**	0.0030	6902
	$3\pi/4$	LDA	–	44.50	0.0000	2
		KDA	96.50	52.00	0.0020	600
		SRKDA	96.50	40.50	0.0010	600
		VNDA	95.00	**61.00**	0.0030	1586
		VCA	81.50	56.50	0.0030	6902

The bold values indicate the best results obtained under the current parameter settings.

proposed method successfully solved the classical *XOR* problem.

After evaluating the proposed method based on the classical *XOR* problem, we showed that the proposed method could achieve comparable recognition performance compared to the state-of-the-art methods with the complex *RDM* data set, which can be considered an extension of the *XOR* problem. The experimental results obtained using the *RDM* data set are shown in Table 3, which demonstrates that VNDA, KDA, and SRKDA achieved

Table 5
Results obtained using the *CIR* data set.

a	r	Method	CV (%)	Test (%)	Time (s)	Memory (double)
$7\pi/6$	2	LDA	-	24.50	0.0010	2
		KDA	100.00	62.00	0.0010	600
		SRKDA	100.00	47.50	0.0000	600
		VNDA	100.00	**76.50**	0.0000	88
		VCA	100.00	57.50	0.0010	94
	3	LDA	–	16.50	0.0000	2
		KDA	100.00	42.50	0.0010	600
		SRKDA	100.00	66.00	0.0000	600
		VNDA	100.00	**74.50**	0.0000	88
		VCA	100.00	49.50	0.0000	94
	4	LDA	–	11.00	0.0000	2
		KDA	100.00	27.50	0.0000	600
		SRKDA	100.00	43.00	0.0010	600
		VNDA	100.00	**72.00**	0.0010	88
		VCA	100.00	38.00	0.0020	94
$5\pi/3$	2	LDA	–	50.00	0.0010	2
		KDA	97.00	**89.50**	0.0000	600
		SRKDA	96.50	66.50	0.0010	600
		VNDA	97.50	68.50	0.0000	1750
		VCA	95.50	80.50	0.0010	454
	3	LDA	–	42.00	0.0000	2
		KDA	96.50	37.00	0.0010	600
		SRKDA	96.50	45.00	0.0000	600
		VNDA	97.00	**78.50**	0.0020	1240
		VCA	96.50	78.00	0.0000	590
	4	LDA	–	37.50	0.0000	2
		KDA	96.00	21.00	0.0010	600
		SRKDA	96.50	37.00	0.0010	600
		VNDA	97.00	68.00	0.0000	463
		VCA	96.50	**69.50**	0.0020	742

The bold values indicate the best results obtained under the current parameter settings.

comparable performance to that with the *RDM* data set. In particular, VNDA, KDA, and SRKDA achieved the highest values in terms of the test accuracy. In all cases, these three nonlinear methods obtained competitive test accuracy results, which were much better than the performance achieved by LDA. The performance of the VCA method tended to decrease as τ increased. The results of the experiments performed using this data set showed that the proposed method could obtain competitive test accuracy with the *RDM* data set.

Finally, we used the *TRI* and *CIR* data sets to evaluate the generalizability of these methods. The experimental results obtained using the *TRI* data set are shown in Table 4, which demonstrates that VNDA, KDA, and SRKDA had comparable cross-validation accuracies, but the test accuracy achieved by the VNDA method was higher than that using the KDA and SRKDA methods. The cross-validation accuracy of the VCA method was poor but the test accuracy was comparable to that with the VNDA method. This result was the same with various values of ι and ι_t. For each value of ι, the test accuracy declined in most cases as ι_t increased using all of the methods. Only the proposed method and the VCA method always performed better than a random guess. The experimental results obtained using the *CIR* data set are shown in Table 5, which demonstrates that the cross-validation accuracies of VNDA, KDA, SRKDA, and VCA were comparable, but the test accuracy achieved by VNDA was mostly the highest compared with the other methods. This result was the same with various values of a and r. Using all of the methods, the test accuracy decreased as r increased, but only the proposed method always performed better than a random guess. The performance of the VCA method was unstable with various values of a. The results of

Table 6
Comparison of the cross-validation accuracy and test accuracy using the real data sets.

Data set	CV (%)				Test (%)					
	KDA	SRKDA	VNDA	VCA	LDA	OLDA	KDA	SRKDA	VNDA	VCA
dna	96.19	96.19	**96.29**	95.44	94.26	94.51	**96.12**	**96.12**	95.19	95.62
letter	**96.26**	95.47	95.61	83.59	67.90	63.66	**96.80**	96.22	96.06	86.86
madelon	62.25	62.20	**65.65**	54.55	57.66	57.66	65.33	65.00	**66.83**	57.16
pendigits	99.59	99.62	**99.67**	99.61	80.24	78.87	**97.97**	97.91	97.71	97.48
satimage	**88.22**	86.87	87.46	86.11	81.10	82.15	**88.05**	86.90	87.55	85.35
svmguide1	96.63	**96.89**	96.53	95.27	89.42	89.42	96.65	**96.85**	96.72	95.25
svmguide3	79.49	80.05	**80.13**	79.41	90.24	90.24	**1.000**	97.56	90.24	97.56
svmguide4	56.05	60.36	**78.27**	67.17	62.17	65.06	51.60	58.97	**84.93**	69.23
usps	98.09	**98.14**	97.76	97.91	88.24	87.09	**95.61**	95.56	94.86	95.11
vowel	**99.63**	97.89	99.27	99.25	37.66	43.93	**58.00**	35.49	51.08	50.86
a1a	80.43	80.74	**81.24**	79.81	79.59	79.59	80.93	**81.14**	80.93	80.11
a2a	79.56	79.64	**80.66**	79.51	78.98	78.98	80.41	80.21	**80.60**	79.11
a3a	81.44	81.47	**81.85**	81.03	79.54	79.54	80.74	81.09	**81.61**	80.25
a4a	82.03	81.92	**82.51**	81.26	80.44	80.44	81.77	81.46	**81.86**	81.16
$acoustic_5$	71.96	**72.08**	70.89	70.08	64.89	63.64	71.30	**71.43**	69.71	68.96
$acoustic_{10}$	72.45	**72.49**	71.55	69.74	65.40	65.44	72.59	**72.78**	71.12	69.36
$acoustic_{15}$	72.82	**73.08**	71.76	70.31	65.48	65.02	72.83	**72.97**	71.57	69.96
$acoustic_{20}$	**73.66**	73.64	72.58	70.73	65.58	65.56	73.19	**73.39**	72.26	70.22
$mnist_5$	**94.46**	94.16	94.43	94.00	81.55	75.56	**95.25**	94.75	94.76	94.55
$mnist_{10}$	**95.48**	95.26	95.18	95.10	84.53	80.49	**96.45**	96.24	95.95	95.90
$mnist_{15}$	96.50	**96.57**	96.05	87.05	85.63	81.69	**96.90**	96.95	96.36	87.66
$mnist_{20}$	**97.04**	96.93	96.45	87.24	86.01	79.00	**97.36**	97.34	96.61	87.80
$protein_{10}$	**63.31**	62.74	62.91	62.52	59.55	50.73	60.68	61.59	61.75	**61.77**
$protein_{20}$	63.13	62.97	62.77	**63.19**	59.71	56.33	62.16	62.45	**63.10**	62.13
$protein_{30}$	63.30	63.56	**65.10**	63.15	61.19	61.36	62.83	**65.38**	64.35	62.12
$protein_{40}$	64.95	66.38	**66.98**	65.21	63.05	63.37	63.47	**66.86**	64.94	65.77
$seismic_5$	69.20	69.55	**70.24**	69.40	67.84	66.10	69.33	69.49	**70.31**	70.04
$seismic_{10}$	69.93	70.61	**71.11**	69.44	67.75	67.48	70.13	70.24	**70.83**	69.61
$seismic_{15}$	70.20	70.83	**71.16**	69.21	67.93	68.10	70.56	70.61	**71.42**	68.63
$seismic_{20}$	70.78	71.16	**71.52**	69.10	67.77	66.37	70.67	70.74	**71.45**	68.45

The bold values indicate the best results obtained under the current parameter settings.

Table 7
Comparisons of the memory and CPU time consumption using the real data sets.

Data set	Memory (double)				Time (s)			
	KDA	SRKDA	VNDA	VCA	KDA	SRKDA	VNDA	VCA
dna	364,000	364,000	**39,213**	18,486,485	0.1100	0.1290	**0.0160**	1.8800
letter	**615,000**	615,000	9,011,588	1,473,779	2.9210	1.7900	**1.1450**	4.8680
madelon	1,002,000	1,002,000	**11,044**	36,683,002	0.0930	0.1350	**0.0110**	1.8560
pendigits	**187,350**	187,350	682,652	855,835	0.5550	0.5960	**0.1030**	1.1030
satimage	**181,835**	181,835	211,274	5,919,853	0.1990	0.1910	**0.0350**	1.5690
svmguide1	15,445	15,445	2380	**754**	0.2490	0.2600	**0.0060**	0.0070
svmguide3	27,346	27,346	**954**	200,869	0.0020	**0.0010**	**0.0010**	0.0030
svmguide4	4500	4500	**2401**	7969	0.0020	0.0020	**0.0010**	0.0040
usps	**1,932,115**	1,932,115	1,956,117	15,637,367	0.8050	**0.7130**	1.5780	3.7620
vowel	**10,560**	10,560	16,061	214,533	0.0060	0.0060	**0.0030**	0.0430
a1a	199,020	199,020	**5796**	589,730	0.9850	1.0470	**0.1090**	2.3980
a2a	280,860	280,860	**1025**	30,506	1.3040	1.4230	**0.0960**	0.2680
a3a	394,940	394,940	**2259**	9,153,349	1.8300	1.9000	**0.1120**	15.930
a4a	592,844	592,844	**1158**	4,899,951	2.7310	2.8040	**0.0760**	8.8330
$acoustic_5$	**204,932**	204,932	336,084	3,014,465	2.2290	2.2490	**0.9670**	5.5420
$acoustic_{10}$	409,864	409,864	**71307**	6,934,602	9.2480	5.5230	**0.9650**	10.330
$acoustic_{15}$	614,796	614,796	**130,308**	8,990,917	12.481	14.469	**0.5850**	11.790
$acoustic_{20}$	819,780	819,780	**282,290**	11,550,598	18.683	10.890	**1.2450**	16.660
$mnist_5$	2,367,000	2,367,000	**1,806,762**	4,019,135	2.6000	2.7580	**1.9840**	6.2850
$mnist_{10}$	4,734,000	4,734,000	**3,674,672**	13,457,414	4.3960	4.5310	**3.0010**	14.702
$mnist_{15}$	7,101,000	7,101,000	**3,441,395**	6,091,810	6.1870	6.6230	**2.8520**	6.2170
$mnist_{20}$	9,468,000	9,468,000	**4,181,542**	6,091,810	8.0540	8.8090	**3.2800**	6.2460
$protein_{10}$	637,584	637,584	**113,863**	383,421	0.6490	0.6310	**0.1540**	0.3570
$protein_{20}$	1,275,527	1,275,527	**277,974**	472,289	1.1980	1.1470	**0.3740**	4.7220
$protein_{30}$	1,913,111	1,913,111	**149,729**	5,076,133	1.5220	1.6620	**0.2640**	2.4390
$protein_{40}$	2,551,054	2,551,054	**150,325**	25,454,617	2.0840	2.2540	**0.2390**	9.0270
$seismic_5$	204,932	204,932	**8057**	5,601,046	2.5670	2.5220	**0.8470**	8.4540
$seismic_{10}$	409,864	409,864	**24,353**	5,902,668	8.2160	9.9630	**1.1920**	7.8260
$seismic_{15}$	614,796	614,796	**133,146**	5,432,061	6.3720	6.7970	**0.5460**	7.5290
$seismic_{20}$	819,780	819,780	**116,181**	5,434,551	9.2630	8.6370	**0.5880**	7.3770

The bold values indicate the best results obtained under the current parameter settings.

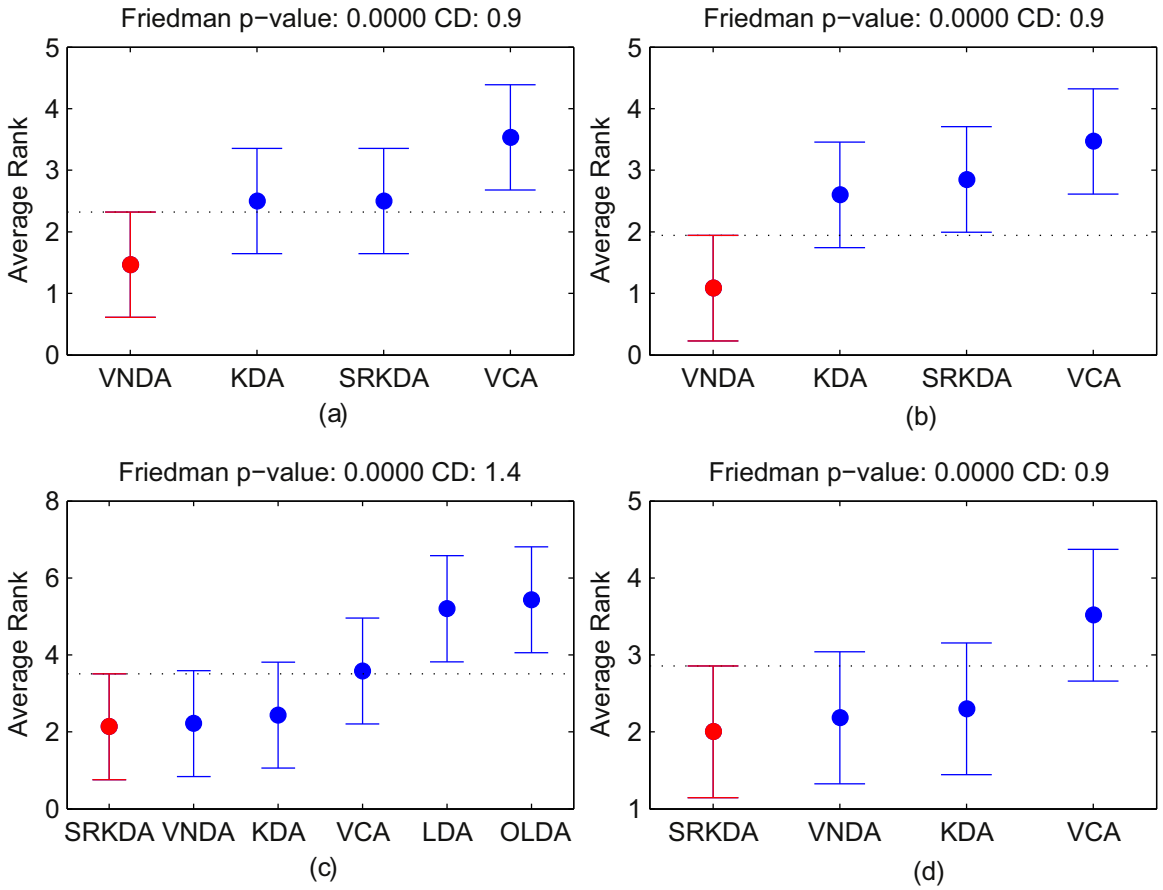

Fig. 3. Results of the Friedman tests and Nemenyi tests for: (a) memory consumption by KDA, SRKDA, VNDA, and VCA; (b) time consumption by KDA, SRKDA, VNDA, and VCA; (c) test accuracy using LDA, OLDA, KDA, SRKDA, VNDA, and VCA; and (d) test accuracy using KDA, SRKDA, VNDA, and VCA. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this paper.)

this experiment demonstrate that the proposed method had better generalizability for the *TRI* and *CIR* data sets than the state-of-the-art methods.

The experimental results obtained using the four simulated data sets demonstrated that the cross-validation accuracy and test accuracy obtained for data sets with various distributions were comparable using the proposed method to those with other methods, but the generalizability of the proposed method with the *TRI* and *CIR* data sets was better than that of the state-of-the-art methods. In the next experiment, we demonstrated that the proposed method could achieve competitive recognition rates with low time and memory consumption based on the real data sets.

4.3. Results obtained using the real data sets

The experimental results obtained using the real data sets are shown in Tables 6 and 7. In this experiment, the maximum degrees used in VNDA and VCA were selected from {2, 3, …, 4} based on five-fold cross validations. In these tables, the data set designated as xxx_i indicates that we only used the first i percent of the training samples in *xxx* during the learning step. All of the test samples were used in the test step. The cross-validation accuracy and test accuracy were comparable using VNDA, KDA, and SRKDA, but the memory and time consumption were higher with KDA and SRKDA than VNDA. In particular, KDA had eight and 11 top values, and SRKDA had six and nine top values in terms of the cross-validation accuracy and test accuracy, respectively, while VNDA had 15 and 10 top values, respectively. VCA only had one and one top value in terms of the cross-validation accuracy and test accuracy, respectively. The evaluation of the memory consumption performance showed that VCA had one top value, KDA and SRKDA had six top values, and VNDA had 23 top values. The evaluation of the time consumption performance showed that VNDA has most of the top values and SRKDA only had one top value with the *usps* data set.

Finally, we used the Friedman test [25] to test whether KDA, SRKDA, VNDA, and VCA were equivalent in terms of their test accuracy and resource consumption. If the Friedman test shows that the methods are not equivalent (p-value $p < 0.05$) in one respect, then we can use the Nemenyi test as a post hoc test to compare these methods in detail. According to the Nemenyi test, the performance of two classifiers is significantly different if the corresponding average ranks differ by at least the critical difference (CD). Fig. 3 shows the results of the Friedman tests and Nemenyi tests. Each subfigure shows the p-value obtained by the Friedman test and the CD for the Nemenyi test. The red or blue point at the center of each bar is the average rank for the corresponding method. The length of each bar is the doubled CD value. The top left and top right subfigures show the results of the statistical tests performed to determine whether the KDA, SRKDA, VNDA, and VCA methods were equivalent in terms of their memory and time consumption, respectively. According to the results of these two tests, we can see that the Friedman p-value was much smaller than 0.05, so we rejected the hypothesis that these methods were equivalent in terms of their memory and time consumption. The Nemenyi test showed that the CD value was 0.9, thereby demonstrating that the VNDA method was not in the same group as KDA, SRKDA, and VCA. Thus, VNDA performed better than KDA, SRKDA, and VCA in terms of the memory and time consumption. To demonstrate that VNDA could achieve

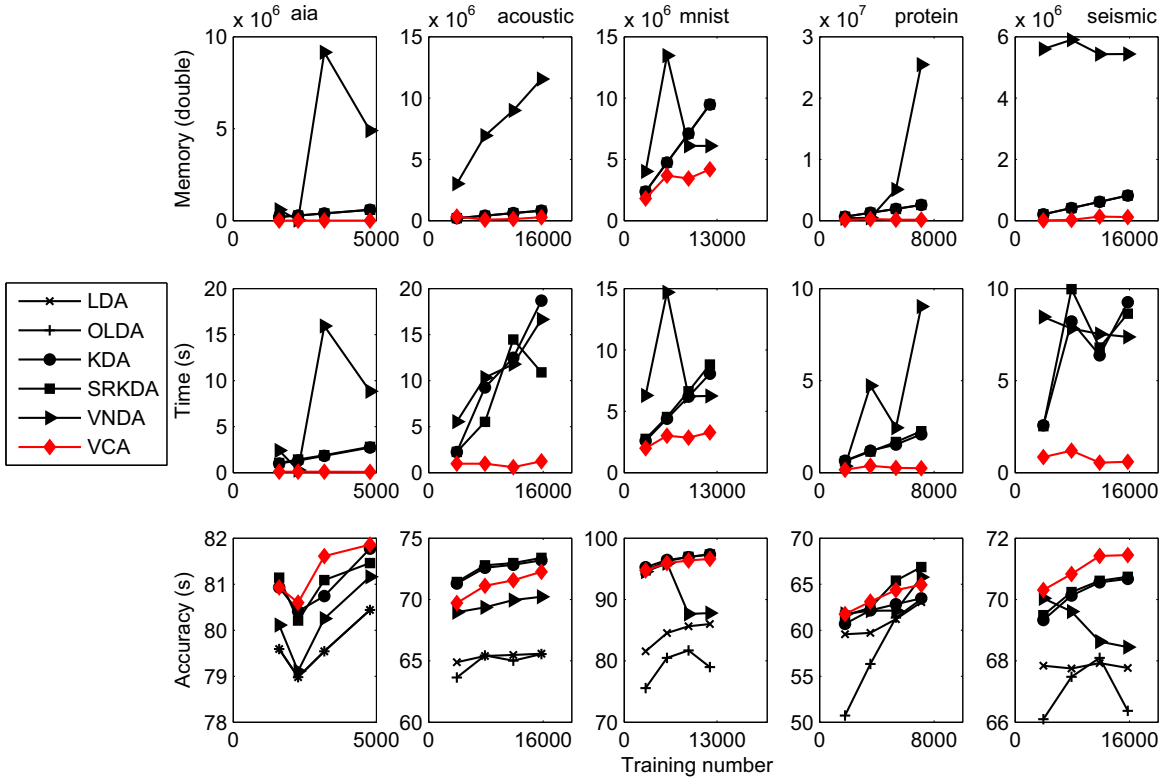

Fig. 4. Trends in the of memory consumption, time consumption, and test accuracy using different methods with increasing numbers of training samples. The first, second, and third rows show the memory consumption, time consumption, and test accuracy using different methods, respectively, for the *aia*, *acoustic*, *mnist*, *protein*, and *seismic* data sets. The horizontal axis in each subfigure shows the number of training samples.

Table 8
Comparison of the test accuracy using the real data sets.

Data set	SVM				Random Forest			
	Original	KDA	SRKDA	VNDA	Original	KDA	SRKDA	VNDA
dna	94.85	**96.37**	96.12	94.94	94.68	91.98	95.45	95.19
letter	97.62	**97.92**	97.26	96.76	96.50	97.62	97.36	96.26
madelon	68.33	66.67	65.83	67.83	**74.83**	65.66	65.33	59.66
pendigits	**98.39**	98.23	98.28	97.82	96.51	97.94	98.23	97.34
satimage	**91.60**	91.20	91.35	89.90	91.05	90.25	91.05	90.20
svmguide1	96.80	96.63	96.88	96.87	**97.05**	95.35	94.95	94.82
svmguide3	75.60	85.37	75.61	63.41	**1.00**	**1.00**	**1.00**	92.68
svmguide4	56.08	55.13	36.22	**83.01**	82.05	61.54	66.99	82.05
usps	95.41	**96.16**	95.96	95.26	93.87	96.01	95.62	95.01
vowel	**64.93**	57.58	50.22	36.79	57.35	36.36	36.36	39.61
a1a	**84.29**	84.07	84.04	83.82	83.54	80.31	81.32	79.93
a2a	84.18	84.18	**84.67**	83.68	83.18	79.66	78.88	78.86
a3a	84.41	84.58	84.35	**84.59**	83.50	79.21	80.01	78.88
a4a	84.55	**84.65**	76.05	84.34	84.06	79.52	79.70	79.77
$acoustic_5$	71.33	71.24	71.61	69.95	**73.61**	68.85	69.16	66.49
$acoustic_{10}$	72.27	72.59	72.74	71.55	**74.39**	69.99	70.49	67.98
$acoustic_{15}$	72.68	72.68	72.90	71.56	**75.35**	70.30	70.78	68.69
$acoustic_{20}$	73.14	73.17	73.32	72.55	**75.99**	70.44	71.21	68.65
$mnist_5$	94.73	**95.71**	95.52	94.65	93.76	92.50	88.92	93.76
$mnist_{10}$	95.89	**96.81**	96.67	95.87	94.85	96.14	92.77	95.65
$mnist_{15}$	96.50	**97.28**	97.09	96.73	95.28	96.74	94.00	96.52
$mnist_{20}$	96.89	**97.63**	97.48	96.85	95.62	96.91	94.47	96.54
$protein_{10}$	63.81	61.86	**63.84**	62.37	55.73	57.29	60.66	61.60
$protein_{20}$	**65.65**	65.19	64.79	64.50	61.50	59.85	60.26	62.43
$protein_{30}$	66.60	65.68	**66.77**	66.19	63.60	60.29	64.87	63.37
$protein_{40}$	67.45	66.58	**67.81**	67.19	65.05	61.58	65.88	63.67
$seismic_5$	69.31	69.74	69.72	70.43	**71.92**	66.41	65.65	66.35
$seismic_{10}$	69.93	70.47	70.52	71.36	**72.48**	67.06	67.71	67.36
$seismic_{15}$	70.40	71.03	70.97	71.59	**73.08**	67.77	67.76	67.63
$seismic_{20}$	70.53	71.14	71.13	71.43	**73.37**	67.51	67.70	67.87

The bold values indicate the best results obtained under the current parameter settings.

Table 9
Comparison of the CPU time consumption using the real data sets.

Data set	SVM				Random Forest			
	Original	KDA	SRKDA	VNDA	Original	KDA	SRKDA	VNDA
dna	1.3300	0.1310	0.1590	**0.0300**	3.1040	0.1950	0.5570	0.4280
letter	3.7980	3.8140	3.5460	**1.7560**	15.1620	10.2830	26.7050	24.9150
madelon	1.5820	0.1280	0.1870	**0.0520**	6.2600	0.1180	0.7400	0.6050
pendigits	**0.3230**	0.6000	1.5170	0.9210	8.6520	1.3180	5.9360	5.3400
satimage	0.3830	0.2470	0.2900	**0.0990**	0.6810	5.8560	0.8720	0.6810
svmguide1	**0.0370**	0.2800	0.3220	0.0620	0.6930	0.4340	0.4540	0.1940
svmguide3	0.0030	0.0030	0.0020	**0.0010**	2.1300	0.2070	0.0460	0.0450
svmguide4	0.0060	0.0050	0.0040	**0.0020**	0.2240	0.2270	0.0560	0.0540
usps	2.1830	**0.8310**	2.3470	1.6340	9.5850	3.6610	8.5010	7.7880
vowel	**0.0110**	0.0220	0.0240	0.0180	5.1750	0.0670	0.0710	0.0650
a1a	9.4830	1.4070	1.8570	**0.8100**	27.9100	2.0620	2.8500	1.8030
a2a	12.5410	2.0290	2.4880	**1.0650**	3.8640	2.6620	2.7130	1.2900
a3a	16.8190	2.7560	3.1540	**1.2540**	52.3900	3.0150	3.4190	1.5190
a4a	25.0980	4.0150	4.5510	1.7470	26.3580	4.6090	4.0010	**1.1970**
$acoustic_5$	7.3540	3.2000	4.3720	**2.1230**	14.5780	29.4380	27.5330	25.2840
$acoustic_{10}$	14.9380	11.3110	9.0540	**3.5310**	4.0590	39.1960	38.9400	33.4170
$acoustic_{15}$	24.0090	15.5740	19.0290	**4.5600**	40.3420	36.4920	31.8300	17.3610
$acoustic_{20}$	30.0230	22.7880	17.2110	6.3210	55.6140	34.5100	14.6990	**3.8090**
$mnist_5$	46.8550	3.0180	4.9290	**2.1710**	59.0060	3.0450	7.0810	4.3230
$mnist_{10}$	76.8980	5.1860	7.6630	**3.1320**	66.3930	10.6190	27.4410	22.9100
$mnist_{15}$	118.8910	6.4820	10.0690	**3.4460**	39.7670	20.5200	19.3440	12.7210
$mnist_{20}$	131.2320	8.1910	12.7550	**3.9460**	72.9510	20.5620	31.9120	23.1030
$protein_{10}$	17.8970	0.8680	0.9480	**0.3170**	24.8440	5.0070	6.4060	5.7750
$protein_{20}$	37.0350	1.7100	1.8790	**0.7320**	13.0240	2.4020	7.6760	6.5290
$protein_{30}$	55.6750	2.2290	2.4660	**0.8040**	29.6170	12.8280	12.3700	10.7080
$protein_{40}$	67.9500	3.0650	3.3310	**1.0770**	18.8640	8.4800	15.3870	13.1330
$seismic_5$	7.6250	3.7960	4.8590	**2.3370**	28.9440	12.5110	18.3730	15.8510
$seismic_{10}$	18.4490	10.5280	14.1670	**4.2040**	17.7810	10.8570	28.7900	18.8270
$seismic_{15}$	24.8920	9.9740	11.3050	4.5080	44.5470	19.2600	9.9110	**3.1140**
$seismic_{20}$	35.2050	14.1280	14.5010	**5.8640**	58.1020	51.9970	31.1390	22.5020

The bold values indicate the best results obtained under the current parameter settings.

Table 10
Comparison of the memory consumption using the real data sets.

Data set	SVM				Random Forest			
	Original	KDA	SRKDA	VNDA	Original	KDA	SRKDA	VNDA
dna	187,578	366,713	404,816	**40,816**	117,974,589	370,953	466,738	102,738
letter	**283,615**	687,847	9,674,904	9,059,904	92,544,564	13,938,961	52,258,761	51,643,761
madelon	1,004,001	1,008,001	1,017,212	**15,212**	696,207,372	1,006,054	2,539,549	1,537,549
pendigits	**23,445**	191,632	936,224	748,874	28,194,250	413,547	1,989,975	1,802,625
satimage	**67,467**	187,790	401,264	219,429	6,352,695	8,230,930	1,531,272	1,349,437
svmguide1	**1537**	16,208	19,242	3797	361,189	51,366	51,570	36,125
svmguide3	10,801	28,211	29,645	**2299**	8,206,521	198,431	88,081	60,735
svmguide4	3999	7144	7895	**3395**	177,960	157,179	22,404	17,904
usps	**406493**	1,934,364	3,897,587	1,965,472	525,671,919	2,869,450	7,845,587	5,913,472
vowel	**7867**	21,703	36,693	26,133	5,559,697	25,069	42,558	31,998
a1a	77,751	200,413	206,611	**7591**	70,471,413	207,244	262,409	63,389
a2a	111,751	283,327	284,292	**3432**	9,783,243	370,013	377,534	96,674
a3a	147,376	398,211	400,536	**5596**	245,477,353	526,093	416679	21,739
a4a	221,751	597,621	598,857	**6013**	171,656,837	1,566,717	802,875	210,031
$acoustic_5$	**117,345**	212,760	550,204	345,272	67,250,196	18,651,436	23,537,596	23,332,664
$acoustic_{10}$	238,185	426,637	501,954	**92,090**	26,414,210	39,572,442	51,478,101	51,068,237
$acoustic_{15}$	348,584	639,434	775,497	**160,701**	382,712,081	60,589,529	38,110,881	37,496,085
$acoustic_{20}$	461,103	853,003	1,143,313	**323,533**	503,023,016	40,799,532	6,182,738	5,362,958
$mnist_5$	**1,225,335**	2,381,067	4,179,640	1,812,640	2,060,649,776	2,407,207	4,518,027	2,151,027
$mnist_{10}$	**2,026,395**	4,760,664	8,412,954	3,678,954	3,148,640,388	5,551,737	13,939,549	9,205,549
$mnist_{15}$	**2,596,775**	7,111,001	10,555,303	3,454,303	2,054,447,528	9,275,573	14,840,180	7,739,180
$mnist_{20}$	**3,134,765**	9,472,548	13,665,072	4,197,072	5,024,264,636	12,214,461	22,806,691	13,338,691
$protein_{10}$	588,963	642,807	753,935	**116,351**	659,361,424	6,984,924	3,765,583	3,127,999
$protein_{20}$	1,147,683	1,287,920	1,560,799	**285,272**	575,820,658	399,2002	10,484,816	9,209,289
$protein_{30}$	1,643,043	1,930,169	2,075,878	**162,767**	1,632,469,718	42,970,342	34,129,539	32,216,428
$protein_{40}$	2,109,963	2,574,697	2,721,527	**170,473**	1,054,254,050	29,509,873	52,319,806	49,768,752
$seismic_5$	126,567	214,845	224,032	**19,100**	130,809,455	12,412,187	13,742,236	13,537,304
$seismic_{10}$	286,203	428,602	458,620	**48,756**	127,404,699	5,112,879	26,543,856	2,6133,992
$seismic_{15}$	423,314	643,504	779,855	**165,059**	376,378,013	36,022,389	4,663,311	4,048,515
$seismic_{20}$	564,718	858,128	978,824	**159,044**	498,404,420	95,376,704	53,673,213	52,853,433

The bold values indicate the best results obtained under the current parameter settings.

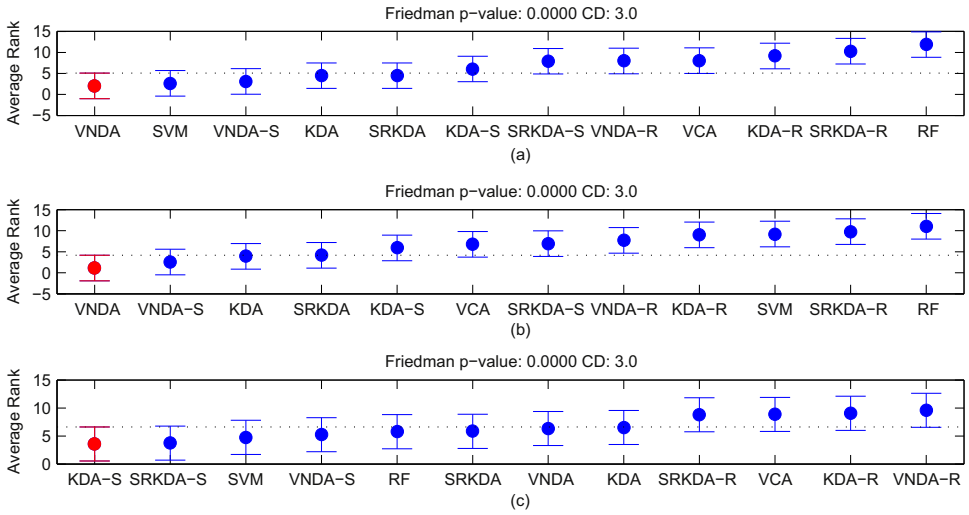

Fig. 5. Results of the Friedman tests and Nemenyi tests for: (a) memory consumption by these methods; (b) time consumption by these methods; and (c) test accuracy using these methods. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this paper.)

equivalent or better test recognition accuracy than KDA, SRKDA, and VCA, we performed these statistical tests two times. The bottom left subfigure shows the results of the statistical tests performed to determine whether the LDA, OLDA, KDA, SRKDA, VNDA, and VCA methods were equivalent in terms of the test recognition accuracy. The bottom right subfigure shows the results of the statistical tests conducted to determine whether the KDA, SRKDA, VNDA and VCA methods were equivalent in terms of the test recognition accuracy. According to these two subfigures, we can see that the test recognition accuracy using VNDA was equivalent to that with KDA and SRKDA.

Fig. 4 shows the performance of KDA, SRKDA, VNDA, and VCA with the *aia*, *acoustic*, *mnist*, *protein*, and *seismic* data sets using various numbers of training samples. All of these subfigures share the same legend. The first, second, and third rows show the memory consumption, time consumption, and test accuracy for these methods, respectively. As the number of training samples increased, the memory and time consumption with KDA, SRKDA, and VNDA increased in an almost linear manner, whereas the memory and time consumption with VNDA increased very little. In particular, we determined the memory and time consumption ratios for VNDA relative to KDA. The memory consumption ratios with the four training data sets, i.e., *aia*, *acoustic*, *mnist*, *protein*, and *seismic*, were 0.2%, 34.4%, 44.1%, 5.8%, and 14.1%, respectively, and the time consumption ratios were 2.8%, 6.7%, 40.7%, 11.5%, and 6.3%. As the number of training samples increased, these ratios continued to decrease. In addition, the test accuracies of the nonlinear methods were all grouped together, which showed that they performed much better than the linear methods. Based on these experiments, we can state that the proposed method achieves comparable recognition performance with low time and memory consumption in the test step.

4.4. Results of these discriminant analysis methods based on different classifiers

In this experiment, we evaluated the performance of SVM [26,27] and Random Forest (RF) [28–30] with the original feature space and the transformed feature space using VNDA, KDA and SRKDA. SVM and RF with the transformed space using VNDA, KDA and SRKDA were denoted as VNDA-S, KDA-S, SRKDA-S and VNDA-R, KDA-R, SRKDA-R, respectively. Parameters used in VNDA, KDA and SRKDA were initialized as the selected values in Section 4.3. The experimental results obtained using the real data sets are shown in Tables 8–10.

The test accuracy was comparable using SVM with different feature spaces and RF with original feature space. But the memory and time consumption of SVM and RF with original feature space were higher than that of VNDA-S. We also used the Friedman test to test whether SVM, RF, KDA, SRKDA, VNDA, VCA, KDA-S, KDA-R, SRKDA-S, SRKDA-R, VNDA-S and VNDA-R were equivalent in terms of their test accuracy and resource consumption. Fig. 5 shows the results of the Friedman tests and Nemenyi tests. Each subfigure shows the *p*-value obtained by the Friedman test and the CD for the Nemenyi test. The red or blue point at the center of each bar is the average rank for the corresponding method. The length of each bar is the doubled CD value. According to the results of these two tests, we can see that the Friedman *p*-value was much smaller than 0.05, so we rejected the hypothesis that these methods were equivalent in terms of their test accuracy and memory and time consumption. The Nemenyi test showed that the CD value was 3.0, thereby demonstrating that the VNDA, SVM, VNDA-S, KDA and SRKDA were not in the same group as KDA-S, SRKDA-S, VNDA-R, VCA, KDA-R, SRKDA-R and RF in terms of their memory consumption, and the VNDA, VNDA-S, KDA and SRKDA were not in the same group as KDA-S, VCA, SRKDA-S, VNDA-R, KDA-R, SVM, SRKDA-R and RF in terms of their time consumption, and the KDA-S, SRKDA-S, SVM, VNDA-S, RF, SRKDA, VNDA and KDA were not in the same group as SRKDA-R, VCA, KDA-R and VNDA-R in terms of their test accuracy. According to this figure, we can see that the test recognition accuracies using VNDA and VNDA-S were equivalent to that with the state-of-the-art methods. The time and memory consumptions of VNDA and VNDA-S were lower than those of the state-of-the-art methods except SVM but the time consumptions of VNDA and VNDA-S were much lower than that of SVM.

5. Conclusion and future work

In this study, we proposed the VNDA method, which does not need to store the training samples during the test step unlike most kernel-based nonlinear discriminant analysis methods. First, the VNDA method learns nonlinear mapping functions explicitly with the modified VCA method, before using these functions to map the input data onto a high-dimensional polynomial feature space, where the LDA method can be applied. Only the mapping functions are stored and computed in the test step. In addition, the VNDA method has some

useful properties. The first is that PCA plus LDA is a special case of VNDA when we bound the degree of the learned mapping functions to 1. The second is that the set of polynomials produced by the VNDA method is the best combination of polynomials in the degree bounded polynomial feature space $\mathcal{P}^d$ that optimizes the ratio trace problem in $\mathcal{P}^d$. To demonstrate the effectiveness and efficiency of the proposed method, we compared the VNDA method with the state-of-the-art methods using four simulated data sets and 15 real data sets. VNDA achieved comparable test accuracy with the *XOR* and *RDM* simulated data sets, and its generalizability was much better with the *TRI* and *CIR* data sets. The results of the experiments conducted using the real data sets showed that VNDA achieved competitive recognition performance with low time and memory consumption in the test step, especially with a large number of training samples. These results were confirmed by the Friedman and Nemenyi tests. We should note that VNDA consumes more memory in the training step if the original feature dimension or category number is very large. The proposed method tries to reduce the number of polynomials in the candidate set, but balancing the recognition performance and memory consumption is still an open problem. Moreover, it is still possible to develop further extensions because LDA is an intensive research field and many ideas that have been developed previously in the input space could be applied to the VNDA method.

Acknowledgments

We would like to express our sincere appreciation to the editors and anonymous reviewers for their insightful comments, which have greatly aided us in improving the quality of the paper. This study was supported by the program of higher-level talents of Inner Mongolia University (135137) and the National Natural Science Foundation of China (NSFC) under Grant nos. 61563039 and 61531019.

References

[1] R.O. Duda, P.E. Hart, D.G. Stork, Pattern Classification, John Wiley & Sons, New York, 2001.
[2] K. Fukunaga, Introduction to Statistical Pattern Recognition, Academic Press, New York, 1972.
[3] P.N. Belhumeur, J.P. Hespanha, D.J. Kriegman, Eigenfaces vs. fisherfaces: recognition using class specific linear projection, IEEE Trans. Pattern Anal. Mach. Intell. 19 (7) (1997) 711–720.
[4] A.K. Jain, R.P. Duin, J. Mao, Statistical pattern recognition: a review, IEEE Trans. Pattern Anal. Mach. Intell. 22 (1) (2000) 4–37.
[5] J. Ye, R. Janardan, C.H. Park, H. Park, An optimization criterion for generalized discriminant analysis on undersampled problems, IEEE Trans. Pattern Anal. Mach. Intell. 26 (8) (2004) 982–994.
[6] D.L. Swets, J.J. Weng, Using discriminant eigenfeatures for image retrieval, IEEE Trans. Pattern Anal. Mach. Intell. 8 (1996) 831–836.
[7] P. Howland, H. Park, Generalizing discriminant analysis using the generalized singular value decomposition, IEEE Trans. Pattern Anal. Mach. Intell. 26 (8) (2004) 995–1006.
[8] J. Ye, Q. Li, A two-stage linear discriminant analysis via qr-decomposition, IEEE Trans. Pattern Anal. Mach. Intell. 27 (6) (2005) 929–941.
[9] D.-Q. Dai, P.C. Yuen, Regularized discriminant analysis and its application to face recognition, Pattern Recognit. 36 (3) (2003) 845–847.
[10] J. Ye, T. Wang, Regularized discriminant analysis for high dimensional, low sample size data, in: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, Philadelphia, USA, 2006, pp. 454–463.
[11] W.-K. Ching, D. Chu, L.-Z. Liao, X. Wang, Regularized orthogonal linear discriminant analysis, Pattern Recognit. 45 (7) (2012) 2719–2732.
[12] J. Duchene, S. Leclercq, An optimal transformation for discriminant and principal component analysis, IEEE Trans. Pattern Anal. Mach. Intell. 10 (6) (1988) 978–983.
[13] J. Ye, Characterization of a family of algorithms for generalized discriminant analysis on undersampled problems, J. Mach. Learn. Res. (2005) 483–502.
[14] D. Chu, G.S. Thye, A new and fast implementation for null space based linear discriminant analysis, Pattern Recognit. 43 (4) (2010) 1373–1379.
[15] B. Scholkopft, K.-R. Mullert, Fisher discriminant analysis with kernels, Neural Netw. Signal Process. IX (1) (1999) 1.
[16] G. Baudat, F. Anouar, Generalized discriminant analysis using a kernel approach, Neural Comput. 12 (10) (2000) 2385–2404.
[17] S. Mika, A. Smola, B. Schölkopf, An improved training algorithm for kernel fisher discriminants, in: Proceedings AISTATS, vol. 2001, Morgan Kaufmann, Florida, USA, 2001, pp. 98–104.
[18] D. Cai, X. He, J. Han, Speed up kernel discriminant analysis, VLDB J. 20 (1) (2011) 21–33.
[19] G.C. Cawley, N.L. Talbot, Efficient leave-one-out cross-validation of kernel fisher discriminant classifiers, Pattern Recognit. 36 (11) (2003) 2585–2592.
[20] J. Yang, A.F. Frangi, J.-y. Yang, D. Zhang, Z. Jin, Kpca plus lda: a complete kernel fisher discriminant framework for feature extraction and recognition, IEEE Trans. Pattern Anal. Mach. Intell. 27 (2) (2005) 230–244.
[21] Z. Lei, Z. Zhang, S.Z. Li, Feature space locality constraint for kernel based nonlinear discriminant analysis, Pattern Recognit. 45 (7) (2012) 2733–2742.
[22] L. Zhang, W. Da Zhou, P.-C. Chang, Generalized nonlinear discriminant analysis and its small sample size problems, Neurocomputing 74 (4) (2011) 568–574.
[23] Y. Xu, D. Zhang, Z. Jin, M. Li, J.-Y. Yang, A fast kernel-based nonlinear discriminant analysis for multi-class problems, Pattern Recognit. 39 (6) (2006) 1026–1033.
[24] R. Livni, D. Lehavi, S. Schein, H. Nachliely, S. Shalev-Shwartz, A. Globerson, Vanishing component analysis, in: Proceedings of the 30th International Conference on Machine Learning, 2013, pp. 597–605.
[25] J. Demšar, Statistical comparisons of classifiers over multiple data sets, J. Mach. Learn. Res. 7 (2006) 1–30.
[26] C. Cortes, V. Vapnik, Support vector network, Mach. Learn. 20 (3) (1995) 273–297.
[27] V.N. Vapnik, Statistical Learning Theory, Wiley, New York, NY, 1998.
[28] T.K. Ho, The random subspace method for constructing decision forests, IEEE Trans. Pattern Anal. Mach. Intell. 20 (8) (1998) 832–844.
[29] L. Breiman, Random forests, Mach. Learn. 45 (1) (2001) 5–32.
[30] T.G. Dietterich, An experimental comparison of three methods for constructing ensembles of decision trees: bagging, boosting, and randomization, Mach. Learn. 40 (2) (2000) 139–157.
[31] C.-C. Chang, C.-J. Lin, Libsvm: a library for support vector machines, ACM Trans. Intell. Syst. Technol. 2 (3) (2011) 27.

Yunxue Shao received the B.S. degree in Electronic Engineering from Hohai University, Nanjing, China and the Ph.D. degree in Pattern Recognition and Artificial Intelligence from the Institute of Automation, Chinese Academy of Sciences, Beijing, China, in 2008 and 2013, respectively. From 2013, he has been a Assistant Professor at the College of Computer Science, Inner Mongolia University, Hohhot, China. His research interests include pattern recognition and machine learning.

Guanglai Gao received the B.S. degree in Electronic Engineering from Inner Mongolia University, Hohhot , China and the M.E. degree in Electronic Engineering from the College of Computer ,National University of Defense Technology, Changsha, China, in 1985 and 1987, respectively. From 1999, he has been a Professor at the College of Computer Science, Inner Mongolia University, Hohhot, China. His research interests include artificial intelligence and multimedia information processing.

Chunheng Wang received the B.S. degree and the M.E. degree in Electronic Engineering from Dalian University of Technology, Dalian, China, and the Ph.D. degree in Pattern Recognition and Intelligent control from the Institute of Automation, Chinese Academy of Sciences, Beijing, China, in 1993, 1996 and 1999, respectively. From 2004, he has been a Professor at the State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing, China. His research interests include pattern recognition, image processing, neural networks and machine learning.