Pattern Anal Applic (2013) 16:609–622
DOI 10.1007/s10044-012-0308-4

SHORT PAPER

Classification by nearness in complementary subspaces

Menglong Yang · Yiguang Liu · Baojiang Zhong · Zheng Li

Received: 24 March 2011 / Accepted: 16 October 2012 / Published online: 20 November 2012

Abstract This study introduces a classifier founded on *k*-nearest neighbours in the complementary subspaces (NCS). The global space, spanned by all training samples, can be decomposed into the direct sum of two subspaces in terms of one class: the projection vectors of this class into one subspace are nonzero, and that into another subspace are zero. A query sample is projected into the two subspaces for each class, respectively. In each subspace, the distance from the projection vector to the mean of its *k*-nearest neighbours can be calculated, and the final classification rules are designed in terms of the two distances calculated in the two complementary subspaces, respectively. Allowing for the geometric meaning of Gram determinant and kernel trick, the classifier is naturally implemented in the kernel space. The experimental results on 1 synthetic, 13 IDA binary class, and five UCI multi-class data sets show that NCS compares favourably to the comparing classifiers, which is founded on the *k*-nearest neighbours or the nearest subspace, on almost all the data sets. The classifier can straightforwardly solve multi-classification problems, and the performance is promising.

Keywords Complementary subspaces · Gram determinant · Projection vector · Kernel function

M. Yang · Y. Liu (✉) · Z. Li
Vision and Image Processing Laboratory, School of Computer, Sichuan University, Chengdu 610064, Sichuan, People's Republic of China
e-mail: lygpapers@yahoo.com.cn

B. Zhong
School of Computer Science and Technology, Soochow University, Suzhou 215006, Jiangsu, China

1 Introduction

The nearest neighbour (NN) approach is an old yet appealing classification method, and is among the most successful and robust methods for many classification problems. In this naive approach, a query sample is assigned to the class which contains the nearest sample [1]. The NN classifier is very flexible but easily overfits training data set. Thus, instead of the NN rule, *k*-nearest neighbouring data samples (*k*NN) are generally considered [2]. It is theoretically shown that the *k*NN rule exhibits good performance, and the error committed by *k*NN is at most twice the Bayes error [1]. Additionally, it is empirically observed that the *k*NN classifier with a well-chosen distance metric outperforms many more sophisticated classifiers in many situations [3]. Thus, *k*NN has been widely used. For instance, in computer vision, *k*NN has been used in face recognition, articulated pose estimation as well as character recognition [4]. Despite its advantages, the *k*NN algorithm suffers from several disadvantages [5, 6]: with a limited number of samples, *k*NN tends to yield a poor generalization ability and becomes less effective in high-dimensional spaces [3]; when the samples having different class labels are comparable in the neighbourhood of a query, the *k*NN probably fails to assign the query; and for large data sets, the computational and storage demands leave *k*NN prohibitive in classifying query samples [7].

To overcome the pitfall of *k*NN in high-dimensional space, various methods have been proposed in the literature. The nearest subspace method classifies a query sample into the class, whose subspace has the maximal projection of the query sample [3]. CLAss Featuring Information Compression (CLAFIC) is the first classifier using the linear subspaces, and its extension into the nonlinear subspace is the Kernel CLAFIC (KCLAFIC), which

is built by incorporating the kernel-based-eigenvalue trick into CLAFIC [8]. Motivated in part by the ability of softness to reduce the variance and (or) locality to reduce the bias, a soft and local adaptation of CLAFIC is presented in [9], and the oriented soft regional subspace classifier is introduced in [10]. The nearest convex hull classification method is proposed with the distance, from a query sample to the convex hull of a class, serving as the classification measure [11]. Actually, the nearest convex hull classification method can be seen as a type of instance-based large-margin classifier [12, 13]. In the K-local Hyperplane distance Nearest Neighbour (HKNN) method [14], each class is considered as a low-dimensional smooth manifold embedded in a high-dimensional space. The manifold is considered to be locally linear, and a query sample is classified based on the distances from the query sample to the local linear manifolds. In many cases, hyper-disks are preferable to affine and convex hulls as well as hyper-spheres because hyper-disks bound the classes more tightly than affine hulls or hyper-spheres while avoiding much of the sample overfitting and computational complexity [15]. All the aforementioned classifiers share the common property that the extension from binary classification to multi-class classification can be carried out in a straightforward way and face the issue of k-nearest neighbour finding [16].

In this study, a sample-based classifier is introduced. In terms of the samples of each class, the global space, spanned by all training samples, can be decomposed into the direct sum of two complementary subspaces. In each subspace, the distance from the projection vector of a query sample to the mean of the k-nearest neighbours is obtained, and the classification rule is built on the fusion of the distances. Through tuning the neighbourhood size k, the proposed classifier offers an altered compromise between the nearest mean classifier and the nearest neighbour classifier in the subspaces. With the help of the geometric denotation of Gram determinant as well as the kernel trick, the classifier is naturally extended into the nonlinear feature space. The advantages of the classifier versus the conventional kNN, KCLAFIC, HKNN, NHD as well as the excellent classifier, Support Vector Machine (SVM) [17], are the following: (1) the proposed classifier is built on the fusion of the closeness from the query sample and the class in each subspace; (2) the classifier is implemented in kernel space creatively, and a simple classification discriminant is given without employing the linear or quadratic optimization programming or an iterative process; and (3) the classification performance of the proposed classifier is much competitive to the classifiers founded on the k-nearest neighbours [1] or the near subspaces [3, 8, 11, 15].

The remainder of this study is organized as follows. In Sect. 2, some preliminaries (the geometric meaning of Gram determinant, the projection of a vector into a subspace, etc.) are introduced. In Sect. 3, the classifier is designed based on the preliminaries, and some properties are investigated. The experiments along with some explanations are presented in Sect. 4. Finally, some concluding remarks are given in Sect. 5.

2 Preliminaries

Though some classical geometric fundamentals, which will be used in designing the classifier, have been introduced in [18, 19], we still revisit the fundamentals here to make this study self-contained. Some nomenclatures used are listed in Table 1.

2.1 Parallelotope volume

If the column vectors of $\mathcal{X}_{1,m} \in R^{n\times m}$ are linearly independent, the volume of the parallelotope with these vectors as its sides is [20, 21]:

$$V(\mathcal{X}_{1,m}) = \sqrt{|G(\mathcal{X}_{1,m})|} \tag{1}$$

where

$$G(\mathcal{X}_{1,m}) \triangleq [\mathcal{X}_{1,m}]^T[\mathcal{X}_{1,m}] = \begin{bmatrix} x_1^T x_1 & \dots & x_1^T x_m \\ \vdots & \vdots & \vdots \\ x_m^T x_1 & \dots & x_m^T x_m \end{bmatrix} \tag{2}$$

is called Gram matrix, whose entry $x_i^T x_j$ involves the length values of x_i and x_j, as well as the angle between them for $1 \le i, j \le m$. When $m > n$, or $m \le n$ and two columns of $\mathcal{X}_{1,m}$ are collinear , it does not hold that the column vectors

Table 1 Nomenclatures

n	Tthe dimension of the original input samples
$\|.\|$	The determinant of a matrix, or the cardinality of a set
$\|\|x\|\|$	The 2-norm of $x \in R^n$
$\mathcal{X}_{i,j}$	The matrix $[x_i, \dots, x_j] \in R^{n\times(j-i+1)}$
Ω	The space spanned by nonlinear mapped $x_1, \dots, x_m$
Γ_i	The parallelotope with the nonlinear mapped $x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_m$ as its sides
Γ_i^s	The minimal linear subspace containing Γ_i in Ω
h	The total number of classes
m_i	The number of the training samples of the ith class
$z_{i,j}$	The jth training sample of the ith class, $1 < i \le h$ and $1 \le j \le m_i$
$\mathcal{Z}_i$	The matrix $[z_{i,1}, \dots, z_{i,m_i}]$
$\mathcal{L}_i$	The subspace associated with the ith class
$\mathcal{L}_i'$	The complementary subspace of $\mathcal{L}_i$
l	The neighbourhood size in $\mathcal{L}_1, \dots, \mathcal{L}_h$
s_q	A query sample

of $\mathcal{X}_{1,m}$ are independent of each other. In such circumstances, $G(\mathcal{X}_{1,m})$ is rank deficient, and the volume of the parallelotope is zero under the m-dimensional volume metric.

To improve the rank of $G(\mathcal{X}_{1,m})$ when $G(\mathcal{X}_{1,m})$ is rank deficient, a natural way is to map the column vectors of $\mathcal{X}_{1,m}$ into a high-dimensional space, wherein the mapped vectors span a subspace with the dimension larger than the dimension of the subspace straightforwardly spanned by $\mathcal{X}_{1,m}$. Let $\psi(.)$ denote the mapping, and

$$\psi(\mathcal{X}_{1,m}) \triangleq [\psi(x_1), \ldots, \psi(x_m)]$$

In the mapped space, $G(\mathcal{X}_{1,m})$ becomes

$$G(\mathcal{X}_{1,m}) = \psi^T(\mathcal{X}_{1,m})\psi(\mathcal{X}_{1,m}) = \begin{bmatrix} \psi^T(x_1)\psi(x_1) & \ldots & \psi^T(x_1)\psi(x_m) \\ \vdots & \vdots & \vdots \\ \psi^T(x_m)\psi(x_1) & \ldots & \psi^T(x_m)\psi(x_m) \end{bmatrix} \quad (3)$$

From [17], we know that for any kernel function, $\kappa(x, y)$, satisfying the Mercer's condition, there exists a feature space where $\kappa(x, y)$ denotes the inner product of the nonlinearly mapped x and y, $\psi(x)$ and $\psi(y)$, where $\psi(.)$ is implicitly determined by $\kappa(x, y)$. Replacing $\psi^T(x_i)\ \psi(x_j)$ in (3) by $\kappa(x_i, x_j)$ yields

$$G_\kappa(\mathcal{X}_{1,m}) \triangleq \kappa(\mathcal{X}_{1,m}, \mathcal{X}_{1,m}) = \begin{bmatrix} \kappa(x_1, x_1) & \ldots & \kappa(x_1, x_m) \\ \vdots & \vdots & \vdots \\ \kappa(x_m, x_1) & \ldots & \kappa(x_m, x_m) \end{bmatrix} \quad (4)$$

The value of $\sqrt{|G_\kappa(\mathcal{X}_{1,m})|}$ is the volume of the parallelotope with the column vectors of $\psi(\mathcal{X}_{1,m})$ as its sides.

2.2 Distance from a sample to a linear space

When $G_\kappa(\mathcal{X}_{1,m})$ is full rank, the column vectors of $\psi(\mathcal{X}_{1,m})$ span an m-dimensional space, denoted by Ω, without relations to the dimension of x_i for $1 \le i \le m$. In Ω, use Λ to denote the m-dimensional parallelotope with the column vectors of $\psi(\mathcal{X}_{1,m})$ as its sides. For Λ, any of its faces is an $(m-1)$-dimensional parallelotope with $m-1$ column vectors of $\psi(\mathcal{X}_{1,m})$ as its sides. Let Γ_i denote the $(m-1)$-dimensional parallelotope with the column vectors of $[\psi(\mathcal{X}_{1,i-1}), \psi(\mathcal{X}_{i+1,m})]$ as its sides. For instance, in Fig. 1, any two of $\psi(x_1)$, $\psi(x_2)$ as well as $\psi(x_3)$ determine a face of the parallelepiped *oabcdefg*, and each face is a parallelogram.

In view of (1) and (4), the volume of Γ_i is

$$V([\mathcal{X}_{1,i-1}, \mathcal{X}_{i+1,m}]) = \sqrt{\left|G_\kappa([\mathcal{X}_{1,i-1}, \mathcal{X}_{i+1,m}])\right|} \quad (5)$$

In space Ω, let d_i stand for the distance from $\psi(x_i)$ to Γ_i. In view of the geometric relation between the parallelotope Γ_i and the parallelotope Λ, we have

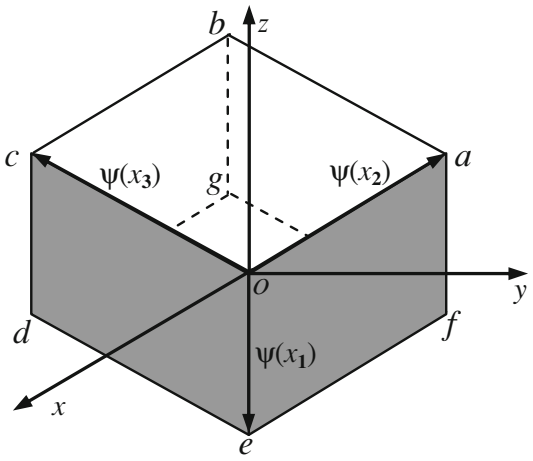


Fig. 1 Parallelepiped *oabcdefg* is determined by three samples $\psi(x_1)$, $\psi(x_2)$ and $\psi(x_3)$

$$V(\mathcal{X}_{1,m}) = d_i V([\mathcal{X}_{1,i-1}, \mathcal{X}_{i+1,m}])$$

In addition to (1) and (5), we have

$$d_i^2 = \frac{|G_\kappa(\mathcal{X}_{1,m})|}{|G_\kappa([\mathcal{X}_{1,i-1}, \mathcal{X}_{i+1,m}])|} \quad (6)$$

For instance, in Fig. 1, $d_1^2 = \frac{|G_\kappa(\mathcal{X}_{1,3})|}{|G_\kappa(\mathcal{X}_{2,3})|}$, $d_2^2 = \frac{|G_\kappa(\mathcal{X}_{1,3})|}{|G_\kappa([\mathcal{X}_{1,1}, \mathcal{X}_{3,3}])|}$ and $d_3^2 = \frac{|G_\kappa(\mathcal{X}_{1,3})|}{|G_\kappa(\mathcal{X}_{1,2})|}$. Based on [22], in Appendix 1, from (6) it follows that

$$d_i^2 = G_\kappa(x_i, x_i) - \kappa(x_i, [\mathcal{X}_{1,i-1}, \mathcal{X}_{i+1,m}]) \times G_\kappa^{-1}([\mathcal{X}_{1,i-1}, \mathcal{X}_{i+1,m}])\kappa([\mathcal{X}_{1,i-1}, \mathcal{X}_{i+1,m}], x_i) \quad (7)$$

2.3 The projection of a sample into a linear space

Let Γ_i^s denote the minimal linear subspace containing the parallelotope Γ_i, $\psi^{(i)}(x_i)$ denote the projection vector of $\psi(x_i)$ into Γ_i^s. That is to say, $\psi^{(i)}(x_i)$ belongs to Γ_i^s, and $\psi(x_i) - \psi^{(i)}(x_i)$ is perpendicular to any vector in Γ_i^s. Thus, amongst $\psi(x_i)$, $\psi^{(i)}(x_i)$ as well as Γ_i^s, there exists

$$\psi^{(i)}(x_i) \in \Gamma_i^s, *\psi(\xi_\iota) - \psi^{(\iota)}(\xi_\iota) \perp \psi^{(\iota)}(\xi_\iota) \quad (8)$$

so, in view of Pythagoras's theorem we have

$$\|\psi^{(i)}(x_i)\|^2 = \|\psi(x_i)\|^2 - \|\psi(x_i) - \psi^{(i)}(x_i)\|^2 \quad (9)$$

From the geometrical relation amongst $\psi(x_i)$, $\psi^{(i)}(x_i)$ and d_i, we know that

$$d_i^2 = \|\psi(x_i) - \psi^{(i)}(x_i)\|^2 \quad (10)$$

Noting that $|\psi(x_i)|^2 = \kappa(x_i, x_i) = G_\kappa(x_i, x_i)$, from (7), (9) and (10), we have

$$\|\psi^{(i)}(x_i)\|^2 = \kappa(x_i, [\mathcal{X}_{1,i-1}, \mathcal{X}_{i+1,m}])G_\kappa^{-1}([\mathcal{X}_{1,i-1}, \mathcal{X}_{i+1,m}]) \times \kappa([\mathcal{X}_{1,i-1}, \mathcal{X}_{i+1,m}], x_i) \quad (11)$$

Because $\psi^{(i)}(x_i) \in \Gamma_i^s$ and Γ_i^s is spanned by the column vectors of $\psi([\mathcal{X}_{1,i-1}), \mathcal{X}_{i+1,m}])$, $\psi^{(i)}(x_i)$ can be represented by the linear combination of the column vectors of $\psi([\mathcal{X}_{1,i-1}), \mathcal{X}_{i+1,m}])$. That is to say, there exists $\alpha \in R^{m-1}$ satisfying

$$\psi^{(i)}(x_i) = \psi([\mathcal{X}_{1,i-1}, \mathcal{X}_{i+1,m}])\alpha \tag{12}$$

From (11) and (12), in Appendix 2, it follows that

$$\alpha = G_k^{-1}([\mathcal{X}_{1,i-1}, \mathcal{X}_{i+1,m}])\kappa([\mathcal{X}_{1,i-1}, \mathcal{X}_{i+1,m}], x_i) \tag{13}$$

Substituting (13) into (12), we obtain the projection vector of $\psi(x_i)$ into the subspace Γ_i^s as follows

$$\begin{aligned}\psi^{(i)}(x_i) = {} & \psi([\mathcal{X}_{1,i-1}, \mathcal{X}_{i+1,m}])G_k^{-1}([\mathcal{X}_{1,i-1}, \mathcal{X}_{i+1,m}]) \\ & \times \kappa([\mathcal{X}_{1,i-1}, \mathcal{X}_{i+1,m}], x_i)\end{aligned} \tag{14}$$

3 The classifier and its properties

3.1 Classification algorithm

All nonlinearly mapped training samples (i.e. the $m_1 + \cdots + m_h$ column vectors of $\psi([\mathcal{Z}_1, \ldots, \mathcal{Z}_h])$ span a space. And the space can be decomposed into the direct sum of two subspaces—$\mathcal{L}_i$ and $\bar{\mathcal{L}}_i$, in terms of the nonlinearly mapped m_i training samples of the ith class. The subspace $\mathcal{L}_i$ contains all column vectors of $\psi(\mathcal{Z}_i)$, and $\bar{\mathcal{L}}_i$ is the complementary subspace of $\mathcal{L}_i$. Because $\mathcal{L}_i$ and $\bar{\mathcal{L}}_i$ contained in the space spanned the column vectors of $\psi([\mathcal{Z}_1, \ldots, \mathcal{Z}_h])$, the basis of $\mathcal{L}_i$ or $\bar{\mathcal{L}}_i$ can be represented by a set of the linear combinations of the column vectors of $\psi([\mathcal{Z}_1, \ldots, \mathcal{Z}_h])$.

Assume the jth basis vector of $\mathcal{L}_i$ is represented as $\psi([\mathcal{Z}_1,\ldots,\mathcal{Z}_h])\alpha_{i,j}$ with $\alpha_{i,j} \in R^{m_1+\ldots+m_h}$. Then $\psi([\mathcal{Z}_1,\ldots,\mathcal{Z}_h])\alpha_{i,j}$ for $1 \le j \le m_i$ need to satisfy

$$\psi^T(\mathcal{Z}_i)\psi([\mathcal{Z}_1, \ldots, \mathcal{Z}_h])[\alpha_{i,1}, \alpha_{i,2}, \ldots] \neq 0 \tag{15}$$

That is to say, the projection of any column vector of $\psi(\mathcal{Z}_i)$ into subspace $\mathcal{L}_i$ (spanned by the column vectors of $\psi([\mathcal{Z}_1, \ldots, \mathcal{Z}_h])[\alpha_{i,1}, \alpha_{i,2}, \ldots]$) is a nonzero vector. To solve $[\alpha_{i,1}, \alpha_{i,2}, \ldots]$, (15) is changed into

$$\kappa(\mathcal{Z}_i, [\mathcal{Z}_1, \ldots, \mathcal{Z}_h])[\alpha_{i,1}, \alpha_{i,2}, \ldots] \neq 0 \tag{16}$$

where $\kappa(\mathcal{Z}_i, [\mathcal{Z}_1, \ldots, \mathcal{Z}_h])$ is an $m_i \times (m_1 + \cdots + m_h)$ matrix. Equation (16) indicates that the subspace spanned by $\alpha_{i,1}, \alpha_{i,2}, \ldots$ is the same as the space spanned by the row vectors of $\kappa(\mathcal{Z}_i, [\mathcal{Z}_1, \ldots, \mathcal{Z}_h])$. So, $[\alpha_{i,1}, \alpha_{i,2}, \ldots]$ can be solved by the Gram–Schmidt orthogonalization of the row vectors of $\kappa(\mathcal{Z}_i, [\mathcal{Z}_1, \ldots, \mathcal{Z}_h])$. In the platform Matlab, the process for solving $[\alpha_{i,1}, \alpha_{i,2}, \ldots]$ is ready-to-wear, and using

$$[\alpha_{i,1}, \alpha_{i,2}, \ldots] = \texttt{orth}\,(\kappa(\mathcal{Z}_\texttt{i}, [\mathcal{Z}_1, \ldots, \mathcal{Z}_\texttt{h}])) \tag{17}$$

we are able to straightforwardly work out $[\alpha_{i,1}, \alpha_{i,2}, \ldots]$.

Let $\psi([\mathcal{Z}_1, \ldots, \mathcal{Z}_h])\beta_{i,j}$ denote the jth basis vector of $\bar{\mathcal{L}}_i$. Because $\bar{\mathcal{L}}_i$ is the complementary subspace of $\mathcal{L}_i$, any column vector of $\psi([\mathcal{Z}_1, \ldots, \mathcal{Z}_h])$ has zero projection onto any basis vector of $\bar{\mathcal{L}}_i$. Hence, there exists

$$\psi^T(\mathcal{Z}_i)\psi([\mathcal{Z}_1, \ldots, \mathcal{Z}_h])[\beta_{i,1}, \beta_{i,2}, \ldots] = 0 \tag{18}$$

i.e.

$$\kappa(\mathcal{Z}_i, [\mathcal{Z}_1, \ldots, \mathcal{Z}_h])[\beta_{i,1}, \beta_{i,2}, \ldots] = 0 \tag{19}$$

Equation (19) indicates that $\beta_{i,1}, \beta_{i,2}, \ldots$ are the basis vectors of the null space of the matrix $\kappa(\mathcal{Z}_i, [\mathcal{Z}_1, \ldots, \mathcal{Z}_h])$. Therefore, we can use the following Matlab command to calculate $[\beta_{i,1}, \beta_{i,2}, \ldots]$

$$[\beta_{i,1}, \beta_{i,2}, \ldots] = \texttt{null}\,(\kappa(\mathcal{Z}_\texttt{i}, [\mathcal{Z}_1, \ldots, \mathcal{Z}_\texttt{h}])) \tag{20}$$

By (17) and (20), the basis vectors of $\mathcal{L}_i$ and $\bar{\mathcal{L}}_i$ are obtained, $\psi([\mathcal{Z}_1, \ldots, \mathcal{Z}_h])[\alpha_{i,1}, \alpha_{i,2}, \ldots]$ and $\psi([\mathcal{Z}_1, \ldots, \mathcal{Z}_h])[\beta_{i,1}, \beta_{i,2}, \ldots]$, respectively. In terms of (14), the projections of the nonlinear mapped training samples of the ith class [i.e. the column vectors of $\psi(\mathcal{Z}_i)$] into $\mathcal{L}_i$ are the column vectors of

$$\psi_{(\mathcal{L}_i)}(\mathcal{Z}_i) = \psi([\mathcal{Z}_1, \ldots, \mathcal{Z}_h])\xi_i\kappa([\mathcal{Z}_1, \ldots, \mathcal{Z}_h], \mathcal{Z}_i) \tag{21}$$

where

$$\begin{aligned}\xi_i \triangleq {} & [\alpha_{i,1}, \alpha_{i,2}, \ldots]\left[[\alpha_{i,1}, \alpha_{i,2}, \ldots]^T\right. \\ & \left. \times G_\kappa([\mathcal{Z}_1, \ldots, \mathcal{Z}_h])[\alpha_{i,1}, \alpha_{i,2}, \ldots]\right]^{-1}[\alpha_{i,1}, \alpha_{i,2}, \ldots]^T\end{aligned} \tag{22}$$

In terms of the definition of $\bar{\mathcal{L}}_i$, the projection of $\psi(\mathcal{Z}_i)$ into $\bar{\mathcal{L}}_i$ is

$$\psi_{(\bar{\mathcal{L}}_i)}(\mathcal{Z}_i) = 0 \tag{23}$$

Similarly, the projection vectors of any nonlinear mapped query sample $\psi(s_\text{q})$ into $\mathcal{L}_i$ and $\bar{\mathcal{L}}_i$ are

$$\psi_{(\mathcal{L}_i)}(s_q) = \psi([\mathcal{Z}_1, \ldots, \mathcal{Z}_h])\xi_i\kappa([\mathcal{Z}_1, \ldots, \mathcal{Z}_h], s_q) \tag{24}$$

and

$$\psi_{(\bar{\mathcal{L}}_i)}(s_q) = \psi([\mathcal{Z}_1, \ldots, \mathcal{Z}_h])\bar{\xi}_i\kappa([\mathcal{Z}_1, \ldots, \mathcal{Z}_h], s_q) \tag{25}$$

where

$$\begin{aligned}\bar{\xi}_i \triangleq {} & [\beta_{i,1}, \beta_{i,2}, \ldots]\left[[\beta_{i,1}, \beta_{i,2}, \ldots]^T\right. \\ & \left. \times G_\kappa([\mathcal{Z}_1, \ldots, \mathcal{Z}_h])[\beta_{i,1}, \beta_{i,2}, \ldots]\right]^{-1}[\beta_{i,1}, \beta_{i,2}, \ldots]^T\end{aligned} \tag{26}$$

In $\mathcal{L}_i$, with respect to $\psi_{(\mathcal{L}_i)}(s_q)$, the l training samples, whose associated column vectors of $\psi_{(\mathcal{L}_i)}(\mathcal{Z}_i)$ are closer than that of the remaining samples, are contained in the following set

$$\begin{aligned}\mathcal{N}_i \triangleq \{z_{ij} \Big| & \|\psi_{(\mathcal{L}_i)}(z_{i,j}) - \psi_{(\mathcal{L}_i)}(s_q)\|_{z_{i,j} \in \mathcal{N}_i} \le \|\psi_{(\mathcal{L}_i)}(z_{i,k}) \\ & - \psi_{(\mathcal{L}_i)}(s_q)\|_{z_{i,k} \notin \mathcal{N}_i}\}, |\mathcal{N}_i| = l\end{aligned} \tag{27}$$

Table 2 The pseudocode of the proposed approach

Given the samples $\mathcal{Z}_1, \mathcal{Z}_2, \ldots, \mathcal{Z}_h$, the kernel function $\kappa(x, y)$ as well as the neighbourhood size l, the pseudocode implementing NCS is as follows

1. As used in [23], we can implement orth() in (17) which denotes the Gram–Schmidt orthogonalization. In light of (17), $[\alpha_{i,1}, \alpha_{i,2}, \ldots]$ is calculated, and further ξ_i can be worked out in terms of (22) for $i = 1, \ldots, h$
2. To implement `null (A)` in (20), we can take the eigen-vectors associated to zero eigen-values of A^TA as the basis vectors of the null space of A. In light of (20), $[\beta_{i,1}, \beta_{i,2}, \ldots]$ is calculated, and further $\bar{\xi}_i$ is calculated according to (26)
3. For a query $s_q, d^2_{(\mathcal{L}_i)}$ and $d^2_{(\bar{\mathcal{L}}_i)}$ are calculated according to (29) and (30), respectively
4. In terms of $g(d_{(\mathcal{L}_j)}, d_{(\bar{\mathcal{L}}_j)}) = \min_{1 \le i \le h} g(d_{(\mathcal{L}_i)}, d_{(\bar{\mathcal{L}}_i)})$ proposed in (31), s_q is classified into a class

To work out $\mathcal{N}_i$, the nonlinear mapping, $\psi(.)$, in (27) should be removed. To the end, (27) is equivalently transformed into

$$\begin{aligned}\mathcal{N}_i &= \{z_{i,j} \Big| \|\psi_{(\mathcal{L}_i)}(z_{i,j}) - \psi_{(\mathcal{L}_i)}(s_q)\|^2_{z_{i,j} \in \mathcal{N}_i} \le \|\psi_{(\mathcal{L}_i)}(z_{i,k}) \\ &\quad -\psi_{(\mathcal{L}_i)}(s_q)\|^2_{z_{i,k} \notin \mathcal{N}_i}\} \\ &= \{z_{i,j} | f^T([\mathcal{Z}_1, \ldots, \mathcal{Z}_h], s_q, z_{i,j})\xi_i \\ &\quad \times f([\mathcal{Z}_1, \ldots, \mathcal{Z}_h], s_q, z_{i,j})_{z_{i,j} \in \mathcal{N}_i} \\ &\le f^T([\mathcal{Z}_1, \ldots, \mathcal{Z}_h], s_q, z_{i,k})\xi_i f([\mathcal{Z}_1, \ldots, \mathcal{Z}_h], s_q, z_{i,j})_{z_{i,k} \notin \mathcal{N}_i}\}\end{aligned} \tag{28}$$

where $f([\mathcal{Z}_1, \ldots, \mathcal{Z}_h], x, y) \triangleq \kappa([\mathcal{Z}_1, \ldots, \mathcal{Z}_h], x) - \kappa([\mathcal{Z}_1, \ldots, \mathcal{Z}_h], y)$. The condition, $|\mathcal{N}_i| = l$ for $1 \le i \le h$, means that the parameter l takes the same value for all classes.

In this study, the distance from $\psi_{(\mathcal{L}_i)}(s_q)$ to the mean of the column vectors, $\psi_{(\mathcal{L}_i)}(z_{i,j})$ with $z_{i,j} \in \mathcal{N}_i$, is used to calibrate the relation between s_q and class i in space $\mathcal{L}_i$. The squared value of the distance is calculated as follows:

$$\begin{aligned}d^2_{(\mathcal{L}_i)} &= \|\psi_{(\mathcal{L}_i)}(s_q) - \frac{1}{l}\sum_{z_{i,j} \in \mathcal{N}_i} \psi_{(\mathcal{L}_i)}(z_{i,j})\|^2 \\ &= \kappa^T([\mathcal{Z}_1, \ldots, \mathcal{Z}_h], s_q)\xi_i\kappa([\mathcal{Z}_1, \ldots, \mathcal{Z}_h], s_q) \\ &\quad - \frac{2}{l}\sum_{z_{i,j} \in \mathcal{N}_i} \kappa^T([\mathcal{Z}_1, \ldots, \mathcal{Z}_h], s_q)\xi_i\kappa([\mathcal{Z}_1, \ldots, \mathcal{Z}_h], z_{i,j}) \\ &\quad + \frac{1}{l^2}\sum_{z_{i,j}, z_{i,h} \in \mathcal{N}_i} \kappa^T([\mathcal{Z}_1, \ldots, \mathcal{Z}_h], z_{i,j})\xi_i \\ &\quad \times \kappa([\mathcal{Z}_1, \ldots, \mathcal{Z}_h], z_{i,h})\end{aligned} \tag{29}$$

Due to $\psi_{(\bar{\mathcal{L}}_i)}(\mathcal{Z}_i) = 0$ given in (23), $\|\psi_{(\bar{\mathcal{L}}_i)}(s_q)\|$ reflects the closeness between s_q and class i in space $\bar{\mathcal{L}}_i$. Similar to $d^2_{(\mathcal{L}_i)}$, we use

$$d^2_{(\bar{\mathcal{L}}_i)} = \kappa^T([\mathcal{Z}_1, \ldots, \mathcal{Z}_h], s_q)\bar{\xi}_i\kappa([\mathcal{Z}_1, \ldots, \mathcal{Z}_h], s_q) \tag{30}$$

to calibrate the closeness.

The two metrics, $d_{(\mathcal{L}_i)}$ and $d_{(\bar{\mathcal{L}}_i)}$, uncover the proximities from the query sample s_q to the ith class in two complementary subspaces, respectively. The smaller $d_{(\mathcal{L}_i)}$ or $d_{(\bar{\mathcal{L}}_i)}$ is, the larger the possibility, that s_q belongs to the ith class, is. Thus, the final classification rule should consider the two metrics as a whole. Based on $d^2_{(\mathcal{L}_i)}$ and $d^2_{(\bar{\mathcal{L}}_i)}$, the following two classification rules can be used: (1) if

$$\begin{aligned}&g(d_{(\mathcal{L}_j)}, d_{(\bar{\mathcal{L}}_j)}) = \min_{1 \le i \le h} g(d_{(\mathcal{L}_i)}, d_{(\bar{\mathcal{L}}_i)}) \\ &\quad \text{with } g(x, y) = x^2 + y^2\end{aligned} \tag{31}$$

then s_q is assigned into class j; or (2) replace $g(x, y)$ with

$$g(x, y) = x^2y^2 \tag{32}$$

and the classification rule is the same as (31). The function $g(., .)$ serves as the integrator to fuse the two metrics. Actually, any function $g(x, y)$, which increases with respect to x and y, is an alternative to be used in designing the classification rule. However, different $g(x, y)$ will lead to different classification decision surfaces. For instance, the condition, that one of $d^2_{(\mathcal{L}_j)}$ and $d^2_{(\bar{\mathcal{L}}_j)}$ is zero, will bring little influence on $d^2_{(\mathcal{L}_j)} + d^2_{(\bar{\mathcal{L}}_j)}$ in rule (31), and yet make $d^2_{(\mathcal{L}_j)}d^2_{(\bar{\mathcal{L}}_j)}$ zero without relations to the nonzero one of $d^2_{(\mathcal{L}_j)}$ and $d^2_{(\bar{\mathcal{L}}_j)}$ in rule (32).

In the implementation of the proposed approach, only (17), (20), (22), (26), and (31) are involved. When training samples, kernel functions, etc. are given, NCS is realized in Table 2.

3.2 Classifier properties

Some properties of the proposed classifier, NCS, are investigated with the aim of understanding its fundamentals.

Property 1 If the subspaces respectively spanned by the column vectors of $\psi(\mathcal{Z}_1), \ldots, \psi(\mathcal{Z}_h)$ are the same, then $d^2_{(\bar{\mathcal{L}}_i)}$ for $1 \le i \le h$ are also the same.

Proof When the subspaces, spanned by the column vectors of $\psi(\mathcal{Z}_1), \ldots, \psi(\mathcal{Z}_h)$, respectively, are the same, (15) can be reduced to

$$\psi^T(\mathcal{Z}_i)\psi(\mathcal{Z}_i)[\alpha_{i,1}, \alpha_{i,2}, \ldots] \ne 0 \tag{33}$$

with $\alpha_{i,j} \in R^{m_i}$ because the subspace spanned by $\psi(\mathcal{Z}_i)$ is the same as the space spanned by $\psi([\mathcal{Z}_1, \ldots, \mathcal{Z}_h])$. Surely,

$[\alpha_{i,1}, \alpha_{i,2}, \ldots]$ with $\alpha_{i,j} \in R^{m_1+\cdots+m_h}$ solved by (15) is different from $[\alpha_{i,1}, \alpha_{i,2}, \ldots]$ with $\alpha_{i,j} \in R^{m_i}$ solved by (33), but the space spanned by the column vectors of $\psi(\mathcal{Z}_i)[\alpha_{i,1}, \alpha_{i,2}, \ldots]$ with $[\alpha_{i,1}, \alpha_{i,2}, \ldots]$ solved by (33) is the same as the space spanned by the column vectors of $\psi([\mathcal{Z}_1, \ldots, \mathcal{Z}_h])[\alpha_{i,1}, \alpha_{i,2}, \ldots]$ with $[\alpha_{i,1}, \alpha_{i,2}, \ldots]$ solved by (15). Equation (33) indicates that the space, $\mathcal{L}_i$, spanned by the column vectors of $\psi(\mathcal{Z}_i)[\alpha_{i,1}, \alpha_{i,2}, \ldots]$ is the same as the space spanned by $\psi(\mathcal{Z}_i)$. So, $\mathcal{L}_i$ for $1 \leq i \leq h$ are the same, so are $\bar{\mathcal{L}}_i$ for $1 \leq i \leq h$. Thus, the projection of any $\psi(s_q)$ into $\bar{\mathcal{L}}_i$ is the same for $1 \leq i \leq h$, which means $d^2_{(\bar{\mathcal{L}}_i)}$ for $1 \leq i \leq h$ are the same. □

If $d^2_{(\bar{\mathcal{L}}_i)}$ for $1 \leq i \leq h$ are the same, the contribution of $d^2_{(\bar{\mathcal{L}}_i)}$ toward classifying s_q becomes trivial. So, based on Property 1, we ought to make the subspaces respectively spanned by $\psi(\mathcal{Z}_i)$ for $1 \leq i \leq h$ different as much as possible. In the following experiments, a partial aim of tuning the free parameters of NCS is to make $\mathcal{L}_i$ for $1 \leq i \leq h$ different.

Property 2 When $\psi(s_q) \in \mathcal{L}_1 \bigcap \ldots \bigcap \mathcal{L}_h, d^2_{(\bar{\mathcal{L}}_i)} = 0$ for $1 \leq i \leq h$.

Proof Because $\mathcal{L}_i$ and $\bar{\mathcal{L}}_i$ are complementary with each other, the only intersection of the both subspaces is the origin point. Equations (15) and (18) demonstrate that each vector in $\mathcal{L}_i$ has zero projection into $\bar{\mathcal{L}}_i$. So $d^2_{(\bar{\mathcal{L}}_i)} = 0$ when $\psi(s_q) \in \mathcal{L}_1 \bigcap \ldots \bigcap \mathcal{L}_h$. □

The Property 2 indicates that, to make the contribution of $d^2_{(\bar{\mathcal{L}}_i)}$ with $1 \leq i \leq h$ more prominent for classification aims, it is necessary to minimize the intersection spaces amongst $\mathcal{L}_1, \ldots, \mathcal{L}_h$. Tuning the free parameters of NCS can reduce, even remove the intersection.

Property 3 The regularization trick is necessary for stably calculating ξ_i as well as $\bar{\xi}_i$.

Proof Based on (22) and (26), whether the regularization trick is necessary for calculating ξ_i or $\bar{\xi}_i$, is dependent on, whether $[\alpha_{i,1}, \alpha_{i,2}, \ldots]^T G_\kappa([\mathcal{Z}_1, \ldots, \mathcal{Z}_h])[\alpha_{i,1}, \alpha_{i,2}, \ldots]$ or $[\beta_{i,1}, \beta_{i,2}, \ldots]^T G_\kappa([\mathcal{Z}_1, \ldots, \mathcal{Z}_h])[\beta_{i,1}, \beta_{i,2}, \ldots]$ is invertible. Because

$$[\alpha_{i,1}, \alpha_{i,2}, \ldots]^T G_\kappa([\mathcal{Z}_1, \ldots, \mathcal{Z}_h])[\alpha_{i,1}, \alpha_{i,2}, \ldots] = \begin{bmatrix} \alpha_{i,1}^T \\ \alpha_{i,2}^T \\ \vdots \end{bmatrix} \begin{bmatrix} \kappa([\mathcal{Z}_1, \ldots, \mathcal{Z}_{i-1}], [\mathcal{Z}_1, \ldots, \mathcal{Z}_h])[\alpha_{i,1}, \alpha_{i,2}, \ldots] \\ \kappa(\mathcal{Z}_i, [\mathcal{Z}_1, \ldots, \mathcal{Z}_h])[\alpha_{i,1}, \alpha_{i,2}, \ldots] \\ \kappa([\mathcal{Z}_{i+1}, \ldots, \mathcal{Z}_h], [\mathcal{Z}_1, \ldots, \mathcal{Z}_h])[\alpha_{i,1}, \alpha_{i,2}, \ldots] \end{bmatrix}$$

the only restriction $\kappa(\mathcal{Z}_i, [\mathcal{Z}_1, \ldots, \mathcal{Z}_h])[\alpha_{i,1}, \alpha_{i,2}, \ldots] \neq 0$ in (16) cannot ensure $[\alpha_{i,1}, \alpha_{i,2}, \ldots]^T G_\kappa([\mathcal{Z}_1, \ldots, \mathcal{Z}_h])[\alpha_{i,1}, \alpha_{i,2}, \ldots]$ to be full rank [i.e. to be positive definite as $[\alpha_{i,1}, \alpha_{i,2}, \ldots]^T G_\kappa([\mathcal{Z}_1, \ldots, \mathcal{Z}_h])[\alpha_{i,1}, \alpha_{i,2}, \ldots]$ is positive semi-definite by itself]. Similarly, the only restriction (19) on $[\beta_{i,1}, \beta_{i,2}, \ldots]$ cannot ensure $[\beta_{i,1}, \beta_{i,2}, \ldots]^T G_\kappa([\mathcal{Z}_1, \ldots, \mathcal{Z}_h])[\beta_{i,1}, \beta_{i,2}, \ldots]$ to be full rank. □

Based on Property 3, in calculating ξ_i and $\bar{\xi}_i$, the inverse of $[\alpha_{i,1}, \alpha_{i,2}, \ldots]^T G_\kappa([\mathcal{Z}_1, \ldots, \mathcal{Z}_h])[\alpha_{i,1}, \alpha_{i,2}, \ldots]$ or $[\beta_{i,1}, \beta_{i,2}, \ldots]^T G_\kappa([\mathcal{Z}_1, \ldots, \mathcal{Z}_h])[\beta_{i,1}, \beta_{i,2}, \ldots]$ is likely to not exist. To guarantee the two matrix to be invertible, some regularization tricks are necessary. For instance, Tikhonov regularization and the well-known method of truncated singular function expansions can be employed [24].

Property 4 With the increase of l, the proposed classifier, NSC, offers an altered compromise between the standard nearest neighbour classification rule and the nearest mean classification rule.

Proof When $l = 1$, $d^2_{\mathcal{L}_i}$ defined in (29) denotes the squared distance from $\psi_{(\mathcal{L}_i)}(s_q)$ to its nearest vector in $\psi_{(\mathcal{L}_i)}(\mathcal{Z}_i)$, and $d^2_{\bar{\mathcal{L}}_i}$ denotes the squared distance from $\psi_{(\bar{\mathcal{L}}_i)}(s_q)$ to its nearest sample in $\bar{\mathcal{L}}_i$. When $l = m_i$, $d^2_{\mathcal{L}_i}$ denotes the squared distance from $\psi_{(\mathcal{L}_i)}(s_q)$ to the mean of $\psi_{(\mathcal{L}_i)}(\mathcal{Z}_i)$, and $d^2_{\bar{\mathcal{L}}_i}$ denotes the squared distance from $\psi_{(\bar{\mathcal{L}}_i)}(s_q)$ to the mean of the projection vectors of class i. For a classification function $g(.,.)$ with $d^2_{\mathcal{L}_i}$ and $d^2_{\bar{\mathcal{L}}_i}$ as input variables, such that defined in (31) or (32), the classification performance of NCS will be close to the nearest neighbour classifier for $l = 1$, and be close to the nearest mean classifier when $l = m_i$ for class i. When $1 \leq l \leq m_i$, NCS offers an altered compromise between the nearest neighbour classifier and the nearest mean classifier for class i. It is worthwhile to be noted that NCS with $1 \leq l \leq m_i$ only offers an altered, not intact, compromise between the two classifiers, the classification decision surface is also related to the special form of $g(.,.)$. □

4 Experimental results and discussions

In the section, to demonstrate the effectiveness of the proposed approach, we presented a number of experimental results on several test benchmarks while comparing it to some nearest neighbour-related classifiers and SVM. First, an example was carried out on a toy benchmark [25] to show the classification performance of NCS in a so-called severe case. Then, IDA repository (available at http://theoval.cmp.uea.ac.uk/%7Egcc/matlab/default.html#benchmarks) was used to evaluate the binary classification

performance. Finally, the multi-classification capability was assessed on five multi-class UCI data sets [26]. In the experiment, each data set is split into two: one is for training and the other is for test. Since the proposed classifier is founded on k-nearest neighbours, the classification algorithms, founded on k-nearest neighbours or the nearest subspace, are chosen in the comparing experiment, including kNN, NHD [15], KCLAFIC [8], HKNN [14]. Also, the prevalent classifier, SVM [17], is chosen to imply what level the classification performance of NCS can achieve. Gaussian kernel is normally preferable to the other kernel functions due to its many perfect properties [27], so $\kappa(x, y)$ takes Gaussian kernels in the experiment.

The neighbourhood size l, the band parameter σ of the Gaussian kernel $\kappa(x, y)$, the Tikhonov regularization μ, and the parameter C for SVM need to be predefined. Because each of these parameters can take any real positive value, exhaustive search for the optimal value for each parameter is not practical. Thus, we predefine a data set for each parameter as follows: $l \in \{1, 3, 6, \ldots, 27, 31, \ldots, 43, 70, 90, 110, 125\}$, $\sigma \in \{10^{-1.5}, 10^{-1}, \ldots, 10^{1.5}\}$, $\mu \in \{10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}\}$ and $C \in \{1, 10, 100, 1000\}$. Specially, the weight parameter λ of HKNN (see [14] for detail) and the parameter D of KCLAFIC (see [8] for detail) are restricted as l. Each of the methods joining the experiments has more than one parameter, and with each parameter combination, we can get a training error value by fivefold cross validation over the training data. Within all combinations of the free parameters, the lowest error rate is called training error (e_{trn}), and the associated parameter combination is chosen in the test over the test data. The test error (e_{tst}) is obtained by classifying the test data using the trained classifier.

4.1 Synthetic problems

This experiment is devoted to assessing the classification capability of NCS in a so-called severe case. The spiral data set [25] is synthetic and contains two classes. The samples of the two classes are intertwined as two spirals in a 2-dimensional space. The data set is difficult to classify, and often employed as a severe test benchmark. Two-thirds of the data are randomly selected as the training data, and the remaining samples serve as the test data.

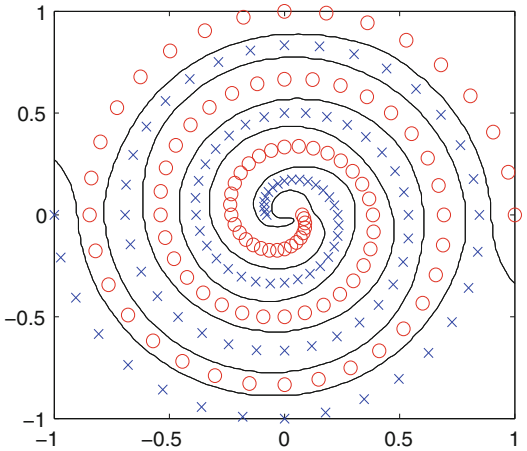
For NCS and the comparing classifiers, the test error rates as well as the corresponding partitions of the original input sample space are shown in Fig. 2. From Fig. 2, we can see that, on the spiral data set, the test error rate of NCS is obviously lower than that of the comparing classifiers. The partition produced by NCS in the original input sample space is more in accordance with the true sample distribution than that produced by other comparing approaches. Within the comparing classifiers, SVM, kNN as well as HKNN have almost the same error rate, and KCLAFIC attains the highest error rate.

4.2 IDA two-class problems

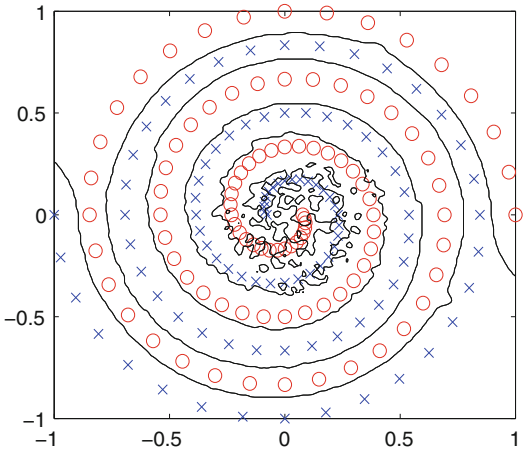
The goal of this experiment is to investigate the binary classification performance of NCS. The IDA repository data set contains 13 data sets which have been used in [28]. The dimensionality of the original input sample space, the number of training and test samples, and the number of realizations for each data set are summarized in Table 3.

On each data set, the test is repeated for 20 times. In each time, the training samples and test samples are prescribed in the IDA repository. On each data set, the experimental result of each classifier consists of two parts: one is about the training and the other is about the test. Each part consists of the average value and the associated variance of the outcomes of the 20 times. The experimental results are shown in Tables 4 and 5, with the best average e_{trn} and e_{tst} values of each data set in bold. The first noticeable result in Tables 4 and 5 is that the proposed classifier, NCS, has outperformed the conventional kNN method, and the other three classification methods, KCLAFIC, NHD as well as HKNN in most of the data sets. It is also clear that NCS is close to SVM in training and test error rates.

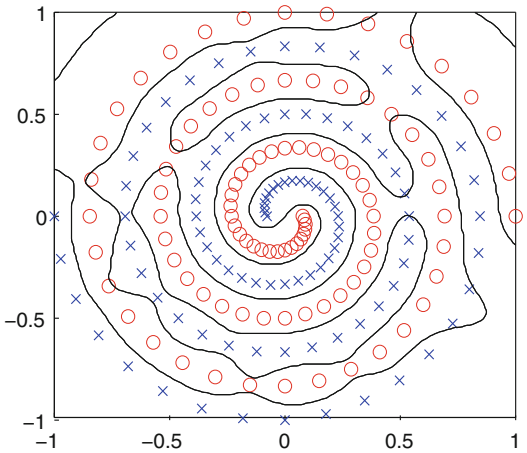
To perform a statistical comparison of the classifiers, the paired tests [29] comparing the twenty repeat results of NCS to that of kNN, KCLAFIC, NHD, HKNN as well as SVM are performed, respectively. If the p value is below the threshold chosen for statistical significance level (usually 5.00 %), the compared two distributions are thought of being significantly different. Thus, on each data set, if the p value is less than 5.00 %, the comparing two methods are thought of having different classification performance. In Table 6, the p values, produced by the t test on the training error distributions (p_{trn}) and the test error distributions (p_{tst}), respectively, are shown with the values, less than 5.00 %, followed by '↓' indicating the compared classifier is inferior to NCS. From Table 6, we can see that the p values of SVM are all larger than 5.00 % on both training and test error distributions. This means that the classification performance of NCS is close to that of SVM. Comparatively, on many IDA data sets, the p values corresponding to kNN, KCLAFIC, NHD as well as HKNN are less than 5.00 % and are followed by '↓', showing that NCS is almost favourable to the four classifiers. Meanwhile, the results in Tables 4, 5 as well as 6 demonstrate that the performance of NCS is robust over the IDA data sets, and on each IDA data set NCS achieves the first-rate classification performance.



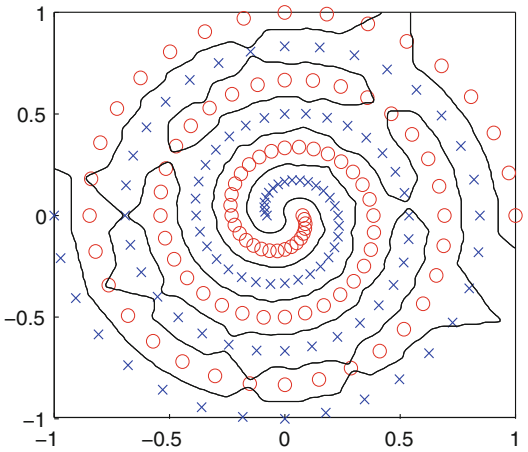
(a) NCS, $e_{tst} = 0.00\%$, $l = 1$, $\sigma = 10^{0.5}$ and $\mu = 10^{-5}$



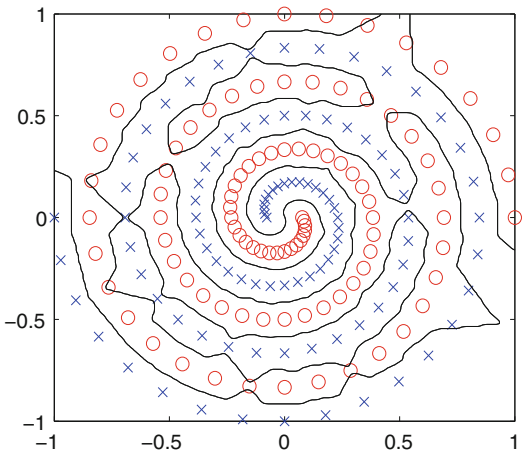
(b) NHD, $e_{tst} = 12.82\%$, $l = 15$ and $\sigma = 10^{0.5}$



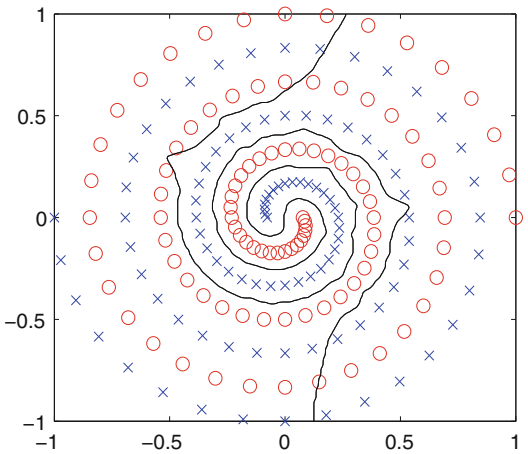
(c) SVM, $e_{tst} = 15.38\%$, $C = 10$ and $\sigma = 0.1$



(d) kNN, $e_{tst} = 15.38\%$, $l = 1$



(e) HKNN, $e_{tst} = 15.38\%$, $l = 1$ and $\lambda = 1$



(f) KCLAFIC,$e_{tst} = 25.64\%$, $\sigma = 10^{-1.5}$ and $D = 39$

Fig. 2 On the spiral data set [25], NCS achieves the lowest error rate

Table 3 IDA repository data set summary

Name	Dimension	Train	Test	Realizations
Banana	2	400	4900	100
Breast cancer	9	200	77	100
Diabetes	8	468	300	100
German	20	700	300	100
Heart	13	170	100	100
Image	18	1,300	1,010	20
Ringnorm	20	400	7,000	100
Solar flare	9	666	400	100
Splice	60	1,000	2,175	20
Thyroid	5	140	75	100
Titanic	3	150	2,051	100
Twonorm	20	400	700	100
Waveform	21	400	4,600	100

4.3 UCI multi-class data sets

To demonstrate the performance of NCS on solving multi-classification issues, we compare NCS to *k*NN, NHD, KCLAFIC, HKNN as well as one-against-one SVM [30] on five multi-class UCI data sets [26]. Though many SVM models have been developed that formulate the multi-classification problem as a single optimization problem [17, 31, 32], here one-against-one SVM is employed to serve as a benchmark multi-classification classifier. A one-against-one SVM is an ensemble of $0.5\ n\ (n-1)$ SVMs to perform n-classifications. Between each pair of two classes, there exists a SVM. Each SVM will give a vote for a query sample, and the final decision is made for the class with majority votes. The experimental data sets consist of iris plants database (IRIS), Wine database (Wine), Glass Identification Database (Glass), Balance Scale Weight & Distance Database (BSWD) and Car Evaluation Database (CED). IRIS contains 3 classes (setosa, versicolor and virginica) of 50 instances each, and each instance includes 4 numeric attributes. The class of setosa is linearly separable from the other two classes which are not linearly separable. The dimensionality of the original input sample space, the class number, the sample number and sample distribution of each data set are summarized in Table 7.

We randomly select 50.00 % of the data as training and remaining 50.00 % as testing. Similar to the experiment on IDA data sets, the test on each UCI data set is repeated for twenty times, and in each time the training samples are randomly selected with the remaining as the test samples. Through twenty rounds, the training and test error rate distributions are obtained. The average and variance values on each data set are shown in Tables 8 and 9, and the associated p values are listed in Table 10. From the three tables, we can see that on IRIS, Wine and Glass, the multi-class classification performance of NCS is close to that of the comparing classifiers apart from KCLAFIC, which has obviously higher error rates. The p values of KCLAFIC are followed by '↓', confirming that KCLAFIC is inferior to NCS in training and test. On BSWD, NCS is inferior to one-against-one SVM in training as p_{trn} value of one-against-one SVM is only 0.05 %. On CED, one-against-one SVM is obviously superior to the other classifiers including NCS.

4.4 Computation times of the classifier implementations

All experiments were done on a desktop computer with 2 GB RAM, a CPU with Intel Pentium dual CPU T2390 @

Table 4 IDA repository experimental result (I)

Data	NCS		*k*NN		KCLAFIC	
	$e_{\text{trn}}(\%)$	$e_{\text{tst}}(\%)$	$e_{\text{trn}}(\%)$	$e_{\text{tst}}(\%)$	$e_{\text{trn}}(\%)$	$e_{\text{tst}}(\%)$
Banana	**9.35** ± 0.90	**11.26** ± 0.58	9.95 ± 0.56	11.82 ± 0.61	15.50 ± 2.75	18.14 ± 0.29
Breast cancer	24.70 ± 1.50	**28.35** ± 5.84	24.70 ± 1.03	29.61 ± 4.82	25.90 ± 2.80	30.39 ± 6.39
Diabetes	**20.97** ± 1.13	**22.67** ± 1.01	24.11 ± 1.19	25.47 ± 0.62	29.70 ± 1.33	28.87 ± 1.67
Solar flare	**31.38** ± 0.56	33.75 ± 1.80	34.62 ± 0.77	34.10 ± 1.90	37.50 ± 2.57	36.60 ± 1.97
German	**24.69** ± 0.74	22.67 ± 2.43	26.20 ± 1.40	25.13 ± 1.75	29.74 ± 0.75	30.20 ± 1.47
Heart	**11.24** ± 3.40	**16.40** ± 3.93	15.65 ± 3.38	18.80 ± 5.19	20.00 ± 4.40	22.40 ± 6.50
Image	**3.38** ± 0.21	2.76 ± 1.10	5.51 ± 0.29	4.36 ± 0.62	15.43 ± 2.04	14.08 ± 2.65
Ringnorm	2.15 ± 0.87	2.25 ± 0.61	45.00 ± 0.57	42.30 ± 1.12	40.45 ± 1.62	40.16 ± 2.02
Splice	2.00 ± 0.79	2.08 ± 0.48	45.45 ± 0.94	42.30 ± 1.12	41.10 ± 1.72	39.63 ± 1.94
Thyroid	**2.14** ± 0.73	**4.87** ± 2.00	8.00 ± 0.95	9.33 ± 1.89	5.57 ± 1.05	8.53 ± 1.36
Titanic	21.20 ± 1.36	**22.42** ± 0.40	20.53 ± 1.71	23.73 ± 1.78	22.80 ± 4.07	24.99 ± 3.31
Twonorm	2.40 ± 0.51	**2.55** ± 0.12	2.55 ± 0.60	2.63 ± 0.23	5.25 ± 0.82	5.65 ± 1.49
Waveform	9.45 ± 1.72	10.67 ± 0.83	9.70 ± 1.55	11.58 ± 0.74	16.50 ± 1.97	17.57 ± 2.31

For each data set, best average e_{trn} and e_{tst} values are set in bold

Table 5 IDA repository experimental result (II)

Data	NHD		HKNN		SVM	
	$e_{trn}(\%)$	$e_{tst}(\%)$	$e_{trn}(\%)$	$e_{tst}(\%)$	$e_{trn}(\%)$	$e_{tst}(\%)$
Banana	9.63 ± 1.12	12.22 ± 0.35	10.75 ± 1.50	12.67 ± 0.29	9.63 ± 0.38	11.64 ± 0.16
Breast cancer	28.20 ± 1.44	31.95 ± 3.35	25.40 ± 1.32	28.57 ± 6.57	**24.20** ± 0.93	28.83 ± 5.59
Diabetes	26.24 ± 0.75	26.87 ± 1.53	22.39 ± 1.21	24.40 ± 1.20	21.58 ± 0.82	24.60 ± 1.06
Solar flare	35.74 ± 1.05	38.20 ± 1.37	34.29 ± 0.88	36.15 ± 2.12	32.79 ± 0.88	**31.90** ± 1.43
German	26.86 ± 1.43	25.27 ± 1.27	25.09 ± 1.00	23.53 ± 0.86	24.74 ± 0.94	**22.60** ± 1.88
Heart	19.29 ± 3.67	21.40 ± 5.99	17.41 ± 3.26	19.80 ± 3.97	13.29 ± 2.37	17.60 ± 3.72
Image	11.85 ± 0.58	13.01 ± 1.06	3.91 ± 0.16	3.62 ± 0.68	**3.38** ± 0.28	**2.73** ± 0.59
Ringnorm	5.25 ± 1.52	5.94 ± 0.51	12.10 ± 2.21	12.31 ± 1.10	**1.80** ± 0.29	**1.87** ± 0.24
Splice	5.40 ± 0.98	5.53 ± 0.33	11.75 ± 1.98	12.48 ± 1.16	**1.65** ± 0.41	**1.85** ± 0.22
Thyroid	3.71 ± 1.05	5.87 ± 2.00	4.57 ± 1.32	5.87 ± 2.17	2.29 ± 0.83	5.07 ± 3.71
Titanic	24.53 ± 2.81	29.66 ± 2.59	**20.40** ± 1.55	22.54 ± 0.69	21.73 ± 1.44	22.47 ± 0.59
Twonorm	3.15 ± 0.64	3.56 ± 0.05	2.85 ± 0.64	3.45 ± 0.48	**2.25** ± 0.52	2.69 ± 0.20
Waveform	12.00 ± 0.69	12.36 ± 0.31	10.95 ± 1.91	11.53 ± 0.61	**9.35** ± 1.71	**10.33** ± 0.24

For each data set, best average e_{trn} and e_{tst} values are set in bold

Table 6 The p values produced by t test on the IDA test result

Data	kNN		KCLAFIC		NHD		HKNN		SVM	
	$p_{trn}(\%)$	$p_{tst}(\%)$	$p_{trn}(\%)$	$p_{tst}(\%)$	$p_{trn}(\%)$	$p_{tst}(\%)$	$p_{trn}(\%)$	$p_{tst}(\%)$	$p_{trn}(\%)$	$p_{tst}(\%)$
Banana	35.03	9.23	19.00	4.33↓	38.40	10.51	25.78	4.54↓	38.40	12.57
Breast cancer	100.00	89.79	43.28	44.49	3.84↓	16.47	49.80	96.04	80.31	92.21
Diabetes	0.83↓	16.98	0.00↓	5.32	0.11↓	13.36	12.17	89.79	77.89	75.10
Solar flare	19.06	11.09	2.85↓	4.32↓	6.06	1.78↓	21.10	4.69↓	29.26	7.69
German	2.97↓	1.49↓	0.06↓	0.01↓	2.67↓	1.02↓	71.74	15.14	23.47	21.79
Heart	11.80	68.85	0.00↓	6.87	0.01↓	21.40	0.09↓	47.98	56.35	86.62
Image	0.11↓	19.75	0.03↓	5.07	0.09↓	7.04	9.49	77.80	87.23	92.35
Ringnorm	0.00↓	0.00↓	0.00↓	0.00↓	5.12	5.29	3.29↓	4.02↓	9.65	7.84
Splice	0.00↓	0.00↓	0.00↓	0.00↓	5.68	4.87↓	3.05↓	2.01↓	67.05	57.08
Thyroid	0.01↓	0.29↓	4.18↓	7.05	79.90	40.40	18.90	40.40	83.09	61.35
Titanic	35.13	16.49	19.03	24.99	25.47	1.97	56.14	19.37	65.43	31.11
Twonorm	30.46	51.05	0.04↓	2.42↓	0.54↓	4.33↓	8.76	7.88	30.46	37.58
Waveform	41.30	6.46	2.23↓	2.33↓	3.05↓	4.15↓	6.84	8.11	74.89	8.73

The p values less than 5.00 % are followed by '↓', indicating the performance of the compared classifier is inferior to NCS

1.86 GHz 1.87 GHz, and a MATLAB Version 2008a platform. In most applications, a classifier will be trained only once, and used for testing individual samples for many times. So the test efficiency is important in reality. When the neighbourhood size of kNN and NCS, as well as the dimension of the nearest subspace for KCLAFIC, NHD and HKNN, take 25 on the data set, heart, while 100 on the data sets, diabetes, twonorm and splice, the computation test time values of twenty repeats for all the classifier implementations are shown in Fig. 3. It is worthwhile to note

Table 7 The summary of several UCI multi-class data sets

Name	Dimension	Classes	Sample numbers	Sample distributions
IRIS	4	3	150	50: 50: 50
Wine	13	3	178	59: 71: 48
Glass	10	6	214	70: 17: 76: 13: 9: 29
BSWD	4	3	625	49: 288: 288
CED	6	4	1728	1210: 384: 69: 65

Table 8 The test result on some UCI multi-class data sets (I)

Data	NCS		*k*NN		KCLAFIC	
	$e_{trn}(\%)$	$e_{tst}(\%)$	$e_{trn}(\%)$	$e_{tst}(\%)$	$e_{trn}(\%)$	$e_{tst}(\%)$
IRIS	2.93 ± 2.59	**5.33** ± 1.46	3.73 ± 2.29	6.67 ± 1.69	14.67 ± 2.39	14.67 ± 4.62
Wine	1.79 ± 0.53	**1.57** ± 0.55	4.03 ± 1.96	3.60 ± 1.80	6.50 ± 1.47	8.31 ± 2.31
Glass	29.58 ± 2.82	**33.27** ± 4.29	30.91 ± 3.49	33.83 ± 2.54	51.43 ± 2.39	47.29 ± 3.77
BSWD	8.39 ± 1.32	5.96 ± 2.01	10.98 ± 0.96	10.38 ± 1.79	32.80 ± 2.31	30.29 ± 1.10
CED	12.44 ± 0.79	13.31 ± 1.07	17.50 ± 0.80	15.35 ± 0.70	29.51 ± 1.40	30.44 ± 1.40

For each data set, best average e_{trn} and e_{tst} values are set in bold

Table 9 The test result on some UCI multi-class data sets (II)

Data	NHD		HKNN		One-against-one SVM	
	$e_{trn}(\%)$	$e_{tst}(\%)$	$e_{trn}(\%)$	$e_{tst}(\%)$	$e_{trn}(\%)$	$e_{tst}(\%)$
IRIS	8.80 ± 6.46	12.80 ± 6.40	2.93 ± 1.77	5.60 ± 1.55	**2.13** ± 1.36	5.87 ± 1.81
Wine	2.01 ± 1.30	4.49 ± 2.46	**1.35** ± 0.85	2.92 ± 1.15	1.79 ± 1.14	1.80 ± 1.35
Glass	33.72 ± 3.22	35.14 ± 4.45	**28.07** ± 2.12	34.02 ± 1.73	29.94 ± 1.95	33.34 ± 2.18
BSWD	8.63 ± 0.92	9.52 ± 1.59	7.91 ± 0.54	8.08 ± 1.50	**3.55** ± 0.81	**3.37** ± 1.01
CED	14.42 ± 1.10	13.01 ± 0.68	14.40 ± 0.62	13.70 ± 0.93	**7.57** ± 0.75	**6.85** ± 0.68

For each data set, best average e_{trn} and e_{tst} values are set in bold

Table 10 The p values produced by t test on the UCI test result

Data	*k*NN		KCLAFIC		NHD		HKNN		One-against-one SVM	
	$p_{trn}(\%)$	$p_{tst}(\%)$	$p_{trn}(\%)$	$p_{tst}(\%)$	$p_{trn}(\%)$	$p_{tst}(\%)$	$p_{trn}(\%)$	$p_{tst}(\%)$	$p_{trn}(\%)$	$p_{tst}(\%)$
IRIS	42.63	14.19	0.05↓	1.65↓	11.12	9.23	100.00	62.13	42.63	47.66
Wine	6.27	7.05	0.31↓	0.20↓	70.40	7.32	37.39	10.87	100.00	74.89
Glass	13.36	77.80	0.06↓	0.45↓	3.46↓	32.62	5.77	64.21	75.32	53.71
BSWD	2.26↓	4.72↓	0.01↓	0.00↓	59.68	0.85↓	44.78	7.14	0.59↑	9.14
CED	0.09↓	0.41↓	0.00↓	0.00↓	2.36↓	4.69↑	3.99↓	3.54↓	0.05↑	0.05↑

The p values less than 5.00 % are followed by '↓' indicating that the performance of the compared classifier is significantly lower than NCS.

that the experimental results are closely related to, but are not completely dependent on the classifiers themselves, and the particular implementations of the classifiers influence the results.

From Fig. 3, we can see that the implementation of NCS is a little more time consuming than the implementations of the comparing classifiers apart from NHD, which will require more time due to the quadratic programming issue for each test sample. *k*NN and HKNN need to find the k nearest samples of a test sample, while NCS needs to find k nearest samples in a subspace and calculate two metrics in both subspaces for each query sample, so NCS is a little more time consuming. When SVM is trained well, only a metric needs to be calculated for each test sample, so the test efficiency of SVM is also better.

5 Discussions

Subspace classifiers classify a pattern based on its distance from different vector subspaces. Many models of subspace classification are based on the assumption that individual classes lie in unique subspaces [9], such as CLAFIC, NHD, HKNN, etc. The drawback of the classical subspace method, CLAFIC, is that the subspaces corresponding to each class are determined without taking into account the features of the other class(es) [9]. NHD approximates each class with the smallest bounding hyper-disk of its k nearest training samples [15], while HKNN replaces the hyper-disk with a hyperplane. By introducing the concepts of locality and softness, the assumption (that a class lie in only a subspace) is extended, and a query is assumed to be

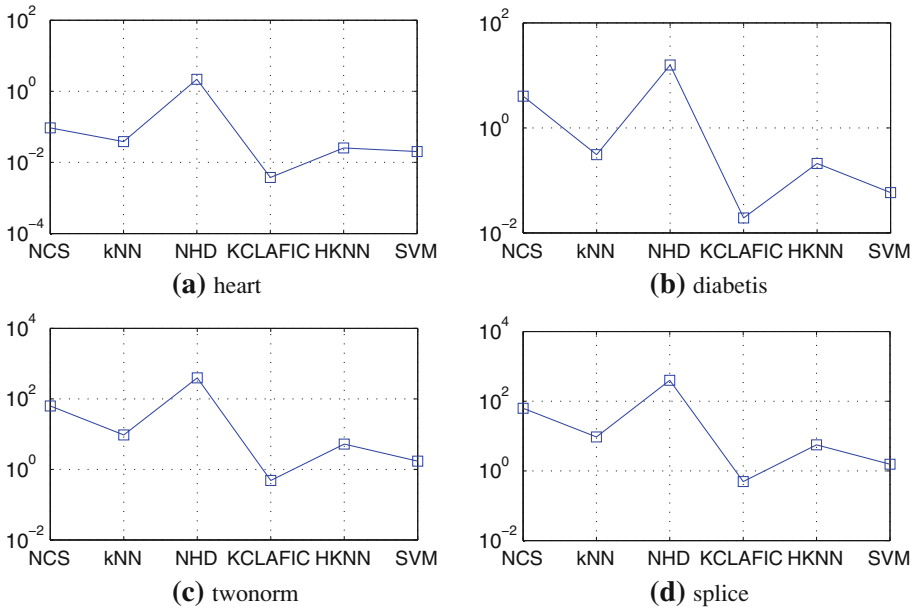


Fig. 3 The test time of NCS and the comparing classifiers on the four data sets, heart, diabetes, twonorm as well as splice. The unit of y-axis is second

associated with more than one subspaces [9, 10]. The soft regional subspace classifiers (introduced in [9, 10]) are built on the extension. NCS proposed here is different from CLAFIC, NHD and HKNN, because NCS associates each class with, not one, but two, subspaces which are complementary to each other. The terms in (29) [such as $\kappa^T([\mathcal{Z}_1,\ldots,\mathcal{Z}_h],s_q)\xi_i\kappa([\mathcal{Z}_1,\ldots,\mathcal{Z}_h],s_q)$] and the right hand side of (30) look like Mahalanobis distance. However, this is not true, and any factor in the terms is not related to covariance.

The kNN classifier is sensitive to the hole artifacts [3] in high-dimensional space, and the accuracy of kNN will decrease greatly in classifying the samples within or nearby the hole artifacts. In the proposed approach, each class actually involves two subspaces whose direct sum forms the global space, and thus the hole artifacts are removed. Usually, training samples of each class cannot disclose all classifying features of the class. To make the best of the training samples of a class for classifications, we use all training samples of a class to define the two complementary subspaces. A query is projected into the two complementary subspaces, and the nearness metrics are computed in the two subspaces, respectively. The final classification metric is obtained by integrating the two metrics. The rationale behind the technique seems to be using the projection to beat the curse of dimensionality in the techniques based on nearest neighbours, and the extension to kernel space further beat the shortage. So, the error rates of the proposed technique are lower than that of kNN in most of the tests.

6 Conclusion

In this study, a classifier has been introduced based on the distances calculated in two complementary subspaces, respectively. By training samples of each class, the space spanned by all training samples can be decomposed into the direct sum of two subspaces: one is completely related to the class and the other is completely unrelated (i.e. all the samples of the class have zero projections in this subspace). In classifications, a query sample is projected into the two subspaces. In the related subspace, use the distance from the projection vector to the mean of the k-nearest neighbours as a metric, locally indicating the closeness between the query sample and the class. While in the unrelated subspace, the norm of the projection vector is employed to serve as the closeness metric because all training samples of the class have zero projections in the subspace. Based on the two metrics, the classification rule is designed, along with the requirements for a function with the two metrics as the input variables.

The classifier is tested on 1 synthetic [25], 13 IDA binary-class (available at http://theoval.cmp.uea.ac.uk/%7Egcc/matlab/default.html#benchmarks) as well as five UCI multi-class [26] data sets when compared with kNN, Kernel CLAFIC (KCLAFIC) [8], Nearest Hyper-disk classifier (NHD) [15], K-local Hyperplane Distance Nearest Neighbour classifier (HKNN) [14] (which are all founded on the k-nearest neighbours or the nearest subspace) as well as SVM. The experimental results indicate that the proposed classifier is favourable to the classification techniques founded on the k-nearest neighbours or the

nearest subspace on almost all the data sets. In addition, the proposed classifier can straightforwardly perform multi-classifications, and its performance is promising. As revealed by the experimental results, the implementation of the proposed classifier is a little more time consuming than that of the comparing classifiers other than NHD in the test period.

Appendix 1

Proof of (7)

Based on [22], there exists

$$\begin{vmatrix} A & B \\ C & D \end{vmatrix} = |A||D - CA^{-1}B|$$

when A^{-1} exists. In addition to

$$G_\kappa(\mathcal{X}_{1,m}) = \begin{bmatrix} G_\kappa([\mathcal{X}_{1,i-1}, \mathcal{X}_{i+1,m}]) & \kappa([\mathcal{X}_{1,i-1}, \mathcal{X}_{i+1,m}], x_i) \\ \kappa(x_i, [\mathcal{X}_{1,i-1}, \mathcal{X}_{i+1,m}]) & G_\kappa(x_i, x_i) \end{bmatrix}$$

we have

$$\begin{aligned} |G_\kappa(\mathcal{X}_{1,m})| =& |G_\kappa([\mathcal{X}_{1,i-1}, \mathcal{X}_{i+1,m}])|[G_\kappa(x_i, x_i) \\ & - \kappa(x_i, [\mathcal{X}_{1,i-1}, \mathcal{X}_{i+1,m}]) \\ & G_\kappa^{-1}([\mathcal{X}_{1,i-1}, \mathcal{X}_{i+1,m}])\kappa([\mathcal{X}_{1,i-1}, \mathcal{X}_{i+1,m}], x_i)] \end{aligned} \tag{34}$$

Substituting (34) into (6) yields (7).

Appendix 2

Proof of (13)

From (11) and (12), we have

$$\begin{aligned} & \alpha^T \psi^T([\mathcal{X}_{1,i-1}, \mathcal{X}_{i+1,m}])\psi([\mathcal{X}_{1,i-1}, \mathcal{X}_{i+1,m}])\alpha \\ =& \alpha^T G_\kappa([\mathcal{X}_{1,i-1}, \mathcal{X}_{i+1,m}])\alpha \\ =& \|\psi^{(i)}(x_i)\|^2 \\ =& \kappa(x_i, [\mathcal{X}_{1,i-1}, \mathcal{X}_{i+1,m}])G_\kappa^{-1}([\mathcal{X}_{1,i-1}, \mathcal{X}_{i+1,m}]) \\ & \times \kappa([\mathcal{X}_{1,i-1}, \mathcal{X}_{i+1,m}], x_i) \\ =& \left[G_\kappa^{-1}([\mathcal{X}_{1,i-1}, \mathcal{X}_{i+1,m}])\kappa([\mathcal{X}_{1,i-1}, \mathcal{X}_{i+1,m}], x_i)\right]^T \\ & \times G_\kappa([\mathcal{X}_{1,i-1}, \mathcal{X}_{i+1,m}]) \\ & G_\kappa^{-1}([\mathcal{X}_{1,i-1}, \mathcal{X}_{i+1,m}])\kappa([\mathcal{X}_{1,i-1}, \mathcal{X}_{i+1,m}], x_i) \end{aligned}$$

From (35), we have

$$\begin{aligned} 0 =& \alpha^T G_\kappa([\mathcal{X}_{1,i-1}, \mathcal{X}_{i+1,m}])\alpha - \left[G_\kappa^{-1}([\mathcal{X}_{1,i-1}, \mathcal{X}_{i+1,m}])\right. \\ & \left. \times \kappa([\mathcal{X}_{1,i-1}, \mathcal{X}_{i+1,m}], x_i)\right]^T \\ & G_\kappa([\mathcal{X}_{1,i-1}, \mathcal{X}_{i+1,m}])G_\kappa^{-1}([\mathcal{X}_{1,i-1}, \mathcal{X}_{i+1,m}]) \\ & \times \kappa([\mathcal{X}_{1,i-1}, \mathcal{X}_{i+1,m}], x_i) \end{aligned}$$

i.e.

$$\begin{aligned} 0 =& [\alpha + G_\kappa^{-1}([\mathcal{X}_{1,i-1}, \mathcal{X}_{i+1,m}])\kappa([\mathcal{X}_{1,i-1}, \mathcal{X}_{i+1,m}], x_i)]^T \\ & \times G_\kappa([\mathcal{X}_{1,i-1}, \mathcal{X}_{i+1,m}])[\alpha - G_\kappa^{-1}([\mathcal{X}_{1,i-1}, \mathcal{X}_{i+1,m}]) \\ & \times \kappa([\mathcal{X}_{1,i-1}, \mathcal{X}_{i+1,m}], x_i)] \end{aligned}$$

so α takes

$$\alpha = G_\kappa^{-1}([\mathcal{X}_{1,i-1}, \mathcal{X}_{i+1,m}])\kappa([\mathcal{X}_{1,i-1}, \mathcal{X}_{i+1,m}], x_i)$$

and (13) is proved.

References

1. Cover TM, Hart PE (1967) Nearest neighbor pattern classification. IEEE Trans Inf Theory 13:21–27
2. Veenman CJ, Reinders MJ (2005) The nearest subclass classifier: a compromise between the nearest mean and nearest neighbor classifier. IEEE Trans Pattern Anal Mach Intell 27: 1417–1429
3. Cevikalp H, Larlus D, Douze M, Jurie F (2007) Local subspace classifiers: linear and nonlinear approaches. In: The 2007 IEEE Machine Learning for Signal Processing Workshop, Thessaloniki, Greece
4. García-Pedrajas N (2009) Constructing ensembles of classifiers by means of weighted instance selection. IEEE Trans Neural Netw 20: 258–277
5. Li B, Chen YW, Chen YQ (2008) The nearest neighbor algorithm of local probability centers. IEEE Trans Syst Man Cybern B 38: 141–154
6. Jiang L, Cai Z, Wang D, Jiang S (2007) Survey of improving k-nearest-neighbor for classification. In: Fourth international conference on fuzzy systems and knowledge discovery, 2007 (FSKD 2007) 1: 679–683
7. Fayed HA, Atiya AF (2009) A novel template reduction approach for the *k*-nearest neighbor method. IEEE Trans Neural Netw 20: 890–896
8. Balachander T, Kothari R (1999) Kernel based subspace pattern classification. In: Proceedings of the international joint conference on neural networks, vol 5, pp 3119–3122
9. Balachander T, Kothari R (1999) Introducing locality and softness in subspace classification. Pattern Anal Appl 2(1): 53–58
10. Balachander T, Kothari R (1999) Oriented soft localized subspace classification. In: Proceedings of the 1999 IEEE international conference on acoustics, speech, and signal processing, 1999, vol 2, pp 1017–1020
11. Nalbantov GI, Groenen PJF, Bioch JC (2007) Nearest convex hull classification. Tech. Rep. EI 2006-50, Econometric Institute
12. Weinberger KQ, Blitzer J, Saul LK (2006) Distance metric learning for large margin nearest neighbor classification. In: NIPS. MIT Press
13. Kumar MP, Torr P, Zisserman A (2007) An invariant large margin nearest neighbour classifier. In: IEEE 11th international conference on computer vision, 2007 (ICCV 2007), vol 2, pp 1–8

14. Vincent P, Bengio Y (2001) K-local hyperplane and convex distance nearest neighbor algorithms. In: NIPS
15. Cevikalp H, Triggs B, Polikar R (2008) Nearest hyperdisk methods for high-dimensional classification. In: Cohen WW, McCallum A, Roweis ST (eds) CICML ACM international conference proceeding series, vol 307, Helsinki, Finland. ACM, pp 120–127
16. Sam H (2008) K-nearest neighbor finding using maxnearestdist. IEEE Trans Pattern Anal Mach Intell 30: 243–252
17. Vapnik VN (1998) Statistical learning theory. Wiley-Interscience, New York
18. Liu Y, Ge SS, Li C, You Z (2011) k-NS: a classifier by the distance to the nearest subspace. IEEE Trans Neural Netw 22: 1017–1020
19. Liu Y, Cao X, Liu JG (2011) Classification using distances from samples to linear manifolds. Pattern Anal Appl. http://www.springerlink.com/content/k261001001288257/
20. Barth N (1999) The gramian and *k*-volume in *n*-space: some classical results in linear algebra. J Young Investig 2. Accessed 19 July 2011
21. Meyer CD (2001) Matrix analysis and applied linear algebra. SIAM, Philadelphia
22. Horn RA, Johnson CR (1986) Matrix analysis. Cambridge University Press, Cambridge
23. Laaksonen J (1997) Subspace classifiers in recognition of handwritten digits. PhD thesis, Helsinki University of Technology, Finland
24. Bertero M (1986) Regularization methods for linear inverse problems. In: Lecture notes in mathematics. Springer, Berlin, pp 52–112
25. Wieland A (1995) Twin spiral dataset. http://www.cs.cmu.edu/afs/cs/project/ai-repository/ai/areas/neural/bench/cmu/bench.tgz. Accessed 19 July 2011
26. Asuncion A, Newman D (2007) UCI machine learning repository. Accessed 19 July 2011
27. Keerthi SS, Lin C-J (2003) Asymptotic behaviors of support vector machines with gaussian kernel. Neural Comput 15: 1667–1689
28. Räetsch G, Onoda T, Müller K-R (2001) Soft margins for adaboost. Mach Learn 43: 287–320
29. Mangasarian OL, Wild EW (2006) Multisurface proximal support vector machine classification via generalized eigenvalues. IEEE Trans Pattern Anal Mach Intell 28: 69–74
30. Mu T, Nandi AK (2009) Multiclass classification based on extended support vector data description. IEEE Trans Syst Man Cybern B 39: 1206–1216
31. Schölkopf B, Smola AJ (2001) Learning with Kernels: support vector machines, regularization, optimization, and beyond. MIT Press, Cambridge
32. Liu Y, You Z, Cao L (2006) A novel and quick SVM-based multi-class classifier. Pattern Recognit 39: 2258–2264