Pattern Recognition 43 (2010) 3448–3457

Contents lists available at ScienceDirect

Pattern Recognition

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

An asymmetric classifier based on partial least squares

Hai-Ni Qu, Guo-Zheng Li*, Wei-Sheng Xu

The Key Laboratory of Embedded System and Service Computing, Ministry of Education, Department of Control Science and Engineering, Tongji University, Shanghai 201804, China

ARTICLE INFO

Article history:
Received 31 December 2009
Received in revised form
7 April 2010
Accepted 4 May 2010

Keywords:
Partial least squares
Dimension reduction
Classification
Unbalanced data

ABSTRACT

This paper investigates the effect of partial least squares (PLS) in unbalanced pattern classification. Beyond dimension reduction, PLS is proved to be superior to generate favorable features for classification. The PLS classifier (PLSC) is illustrated to give extremely better prediction accuracy to the class with the smaller data number. In this paper, an asymmetric PLS classifier (APLSC) is proposed to boost the poor performance of PLSC to the class with the larger data number. PLSC and APLSC are compared with five state-of-arts algorithms, support vector machines (SVMs), unbalanced SVMs, asymmetric principal component and discriminant analysis (APCDA), SMOTE and Adaboost. Experimental results on six UCI data sets show that APLSC improves PLSC in promoting overall classification accuracy, at the same time, APLSC and PLSC perform better than other five algorithms even under seriously unbalanced distribution.

1. Introduction

In many pattern recognition tasks, the data sets are highly unbalanced. For example, disease cases are much rarer than normal cases in the medical field; scammers appear rarer than the defaults in business. The rare objects generally arouse more interests. An example is that detecting disease cases gains so much significance that a patient will be cured as early as possible. In this paper, the rare cases (or the class of interest) are named by minority class or positive class, and the normal cases overwhelming in number are the majority class or negative class. For this kind of asymmetric class distribution, regular classifiers even like support vector machines (SVMs), may treat the minority class data as noise and make the class-boundary extremely benefit for the majority class, resulting in a drop of precision in classifying the minority class [1].

The unbalanced problem is illustrated in Fig. 1. The majority class data is expressed in green color, where the training data is with 'cross' shape and test with 'diamond'. The left part of the plot in red color is the minority class, where the training data is in 'circle' shape and test in 'square'. The solid line separated the data of two classes represents the boundary learned by linear SVMs as shown in Fig. 1. The line is apparently biased to the minority class, giving more margins to the majority class. However, it risks of under-learning the minority class. As in Fig. 1, one of the minority class test examples is mislabeled as majority according to the SVMs boundary. Therefore, an ideal boundary line denoted by dashed line should be more favorable to the minority class and the overall accuracy on the whole data set.

Lots of works have been taken for unbalanced classification [2]. Cost-sensitive learning, over-sampling and under-sampling constitute the major methods. Empirical studies have been taken to compare the effect of over-sampling, under-sampling and threshold-moving by using several base classifiers [3–9]. Ensemble methods, such as Bagging and boosting, are applied frequently in this field and good results have been reported [10–13]. Adaboost is a typical boosting method, which attempts to reduce the bias generated by the regular classifiers to the majority class data by focusing intensively on misclassified examples [13]. Synthetic minority oversampling technique (SMOTE) enhances classification performance primarily from the diversity caused by generating pseudominority class data [14].

There are some recent works of studying asymmetric classifiers [15], especially promoting the classification performance of SVMs in asymmetric data classification. Chew proposed an unbalanced support vector machines (it is simplified by UnSVMs in this paper) to adjust error penalties of each class, so as to counter the effects of the uneven class sizes [16]. Granular SVMs repetitive under sampling was presented in [17]. Its effectiveness has proved that information loss is minimized and positive effect of data cleaning is maximized in the under sampling process.

Besides these efforts, some researches put forward to improve dimension reduction methods for the asymmetric data [18–20], including principal component analysis (PCA), and linear discriminant analysis (LDA). The core of these methods is an eigen-decomposition problem, which concerns extremely with the

* Corresponding author at: Room 605, Dianxin College Building, Cao'an Road 4800, China. Tel.: +86 21 6958 3706; fax: +86 21 6958 9241.
E-mail addresses: gzli@tongji.edu.cn, drgzli@gmail.com (G.-Z. Li).

0031-3203/$ - see front matter © 2010 Elsevier Ltd. All rights reserved.
doi:10.1016/j.patcog.2010.05.002

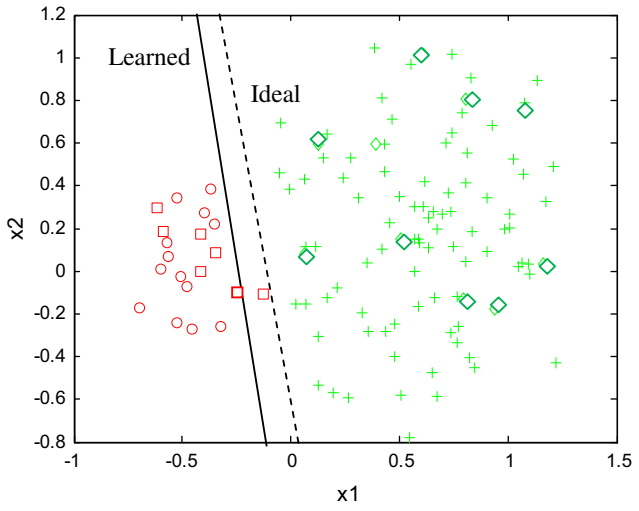

Fig. 1. Plot of one unbalanced data set and its classification boundary in the original feature place by using SVMs. ('circle' demonstrates the minority class training examples, 'cross' means the majority class training examples, 'Square' denotes the minority class test examples while 'diamond' is majority class test examples. Solid line is the boundary line learned by linear SVMs, while the dashed one is the optimal boundary line.)

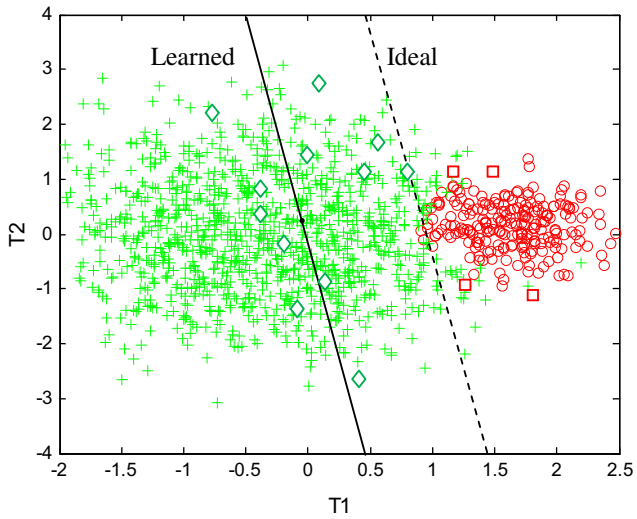

Fig. 2. Plot of one unbalanced data set and its classification boundary in the score vector place by using PLS classifiers. ('circle' demonstrates the minority class training samples, 'cross' means the majority class training samples, 'Square' denotes the minority class test samples, while 'diamond' is majority class test samples. Solid line is the boundary line learned by regular PLS classifier, while the dashed one is generated by APLSC.)

data structure. Including the between-class distance and the within-class distance, the data structure is dramatically related to class distribution, which is asymmetric if the data is skewed. To retrieve the harmful effect of unbalanced data to PCA, an asymmetric principal component and discriminant analysis (APCDA) method is proposed in [18].

In this paper, we address the effect of unbalanced data on another common dimension reduction method, partial least squares (PLS), and propose an asymmetric classifier based on PLS to tackle the unbalanced classification problem, especially promoting the prediction accuracy of the minority class. We limit our discussion to binary classification to focus on the skew class distribution.

In our work, PLS, for the first time, is proved to possess the essential immunity to class distribution to some extent. Based on the fundamental knowledge of PLS [21,22], we unify PLS, PCA and LDA in an eigen-decomposition framework. Barker pointed out that PLS concerns eigen-decomposition of between-class scatter matrix solely [23]. Solving this problem only involves calculation of mean vectors of different classes. The number of data in each class, however, is not concerned. In other words, class distribution has little influence in PLS. As a contrast, PCA aims at distinguishing gross scatter matrix, which is composed by within-class and between-class scatter matrix. The class with more training examples results in heavier influence to dimension reduction of PCA. Consequently, the class with the less number is prone to be misclassified. PLS is illustrated to overcome this deficiency. We will prove that PLS outperforms PCA and APCDA in dimension reduction for the unbalanced problem in theoretical and empirical analyses in this paper.

Furthermore, we show that the PLS classifier places the classification plane far away from the minority class, making the classifier have a strong estimation bias toward the minority class. However, the bias results in a large number of false positives as in Fig. 2. Stimulated from this characteristic of PLS classifiers, we take advantage of its benefit and further make a bias to alleviate the majority class loss. The new classifier based on PLS, generating the new classification hyper plane (shown as the dashed line in Fig. 2), is named as Asymmetric PLS Classifier (APLSC).

Compared with the previous works, APLSC has two advantages. Firstly, previous works emphasize on over-learning the minority class data or under-learning the majority class. We take no additional measure to train both of them, so that the computational complexity is lower. Secondly, present studies on the effect of skewed class distribution to dimension reduction methods have revealed that PCA and LDA are inevitable to bias to the majority class. However, our work uncovers the priority of PLS over PCA and APCDA, that it is less sensitive to class distribution and more capable to generate favorable features for classification.

Six UCI data sets are taken to validate the performance of APLSC. Several state-of-the-art methods are compared with APLSC in this paper. Our experimental results show the effectiveness of APLSC in classifying unbalanced data. The rest of this paper is organized as follows. Section 2 introduces a unified eigen decomposition framework for PLS and related methods. Theoretical study on the effect of PLS for asymmetric classification and an asymmetric PLS classifier is presented in Section 3. Experimental observations and discussions are reported in Section 4. Section 5 makes conclusions.

2. PLS and related works

2.1. PLS

Consider a set of M-dimensional features and its class label vector denoted as $\mathbf{X} \in R^{N \times M}$ and $\mathbf{y} \in R^N$, where N is the number of examples. Only binary classification problems are discussed in this paper, so $\mathbf{y}$ is one dimensional. Partial least squares (PLS) is commonly used to model the relations between these two blocks by means of score vectors. Its classical form is based on the nonlinear iterative partial least squares (NIPALS) algorithm [24]. NIPALS interprets PLS as finding appropriate weight vectors $\mathbf{w}$,$\mathbf{c}$ such that

$$\max[\mathrm{cov}(\mathbf{t},\mathbf{u})]^2 = \max[\mathrm{cov}(\mathbf{Xw},\mathbf{yc})]^2 = \max_{|\mathbf{a}|=|\mathbf{b}|=1}[\mathrm{cov}(\mathbf{Xa},\mathbf{yb})]^2 \tag{1}$$

where $\mathbf{t}$ and $\mathbf{u}$ are named as score vectors of $\mathbf{X}$ and $\mathbf{y}$, respectively. $\mathrm{cov}(\mathbf{t},\mathbf{u}) = \mathbf{t}^T\mathbf{u}/N$ denotes the sample covariance between score

vectors. PLS is an iterative method. The initial value of $\mathbf{u}$ is randomly chosen, then $\mathbf{t}$, $\mathbf{w}$ and $\mathbf{c}$ are obtained according to NIPALS. The final step in one iteration is deflation of $\mathbf{X}$ and $\mathbf{y}$. Different forms of deflation correspond to different forms of PLS. In this paper, we adopt PLS mode A [21]. NIPALS may be iterated for k times, where k is the number of components. The iteration process of PLS is summarized as Algorithm 1.

Algorithm 1. Partial least squares (PLS)

Input: Feature set $\mathbf{X}$
Label $\mathbf{y}$
Number of components k
Output: Projection matrix $\mathbf{w}$, score vectors $\mathbf{T}$ and $\mathbf{U}$, parameter $\mathbf{V}$ and loading $\mathbf{Q}$
$\mathbf{E}_0=\mathbf{X}$, $\mathbf{F}_0=\mathbf{y}$, $\mathbf{w}=[]$; $\mathbf{T}=[]$; $\mathbf{U}=[]$; $\mathbf{V}=[]$; $\mathbf{Q}=[]$; $\mathbf{u}_0=\mathbf{y}$
for $i=1$ to k **do**
  $\mathbf{w}_i=\mathbf{E}_{i-1}^T\mathbf{u}_{i-1}/(\mathbf{u}_{i-1}^T\mathbf{u}_{i-1})$;
  $\mathbf{w}_i=\mathbf{w}_i/\|\mathbf{w}_i\|$;
  $\mathbf{t}_i=\mathbf{E}_{i-1}\mathbf{w}_i$;
  $\mathbf{c}_i=\mathbf{F}_{i-1}^T\mathbf{t}_i/(\mathbf{t}_i^T\mathbf{t}_i)$;
  $\mathbf{c}_i=\mathbf{c}_i/\|\mathbf{c}_i\|$;
  $\mathbf{u}_i=\mathbf{F}_{i-1}\mathbf{c}_i$;
  $v_i=\mathbf{u}_i^T\mathbf{t}_i/(\mathbf{t}_i^T\mathbf{t}_i)$;
  $q_i=\mathbf{F}_{i-1}^T\mathbf{u}_i/(\mathbf{u}_i^T\mathbf{u}_i)$;
  $\mathbf{E}_i=\mathbf{E}_{i-1}-\mathbf{t}_i\mathbf{t}_i^T\mathbf{E}_{i-1}/(\mathbf{t}_i^T\mathbf{t}_i)$;
  $\mathbf{F}_i=\mathbf{F}_{i-1}-v_i\mathbf{t}_iq_i$;
  $\mathbf{w}=[\mathbf{w},\mathbf{w}_i]$; $\mathbf{T}=[\mathbf{T},\mathbf{t}_i]$; $\mathbf{U}=[\mathbf{U},\mathbf{u}_i]$; $\mathbf{V}=[\mathbf{V},v_i]$; $\mathbf{Q}=[\mathbf{Q},q_i]$;
end for

After extracting score vectors, the PLS regression model are written in the matrix form as

$$\mathbf{y}=\mathbf{VTQ}^T+\mathbf{F} \tag{2}$$

where $\mathbf{T}=[\mathbf{t}_1,\mathbf{t}_2,\ldots,\mathbf{t}_k]$, $\mathbf{v}=[v_1,v_2,\ldots,v_k]$ and $\mathbf{Q}=[q_1,q_2,\ldots,q_k]$, $\mathbf{F}$ is an matrix of residuals. To make prediction on new sample data, we rewrite (2) into (3)

$$\hat{\mathbf{y}}=\sum_{i=1}^{k}v_i\hat{\mathbf{t}}_iq_i \tag{3}$$

where $\hat{\mathbf{t}}_i=\hat{\mathbf{E}}_{i-1}\mathbf{w}_i$ and $\hat{\mathbf{E}}_i=\hat{\mathbf{E}}_{i-1}-\hat{\mathbf{t}}_i\mathbf{t}_i^T\mathbf{E}_{i-1}/(\mathbf{t}_i^T\mathbf{t}_i)$, $\hat{\mathbf{E}}_0=\mathbf{X}_t$.

2.2. Related methods

Barker pointed out that there is close connection among PLS, PCA, canonical correlation analysis (CCA) and LDA [23]. Before elaborating the connection in detail, some brief introductions of these feature extraction methods are presented from the data structure point of view.

Let $(\mathbf{\Sigma_x})_{M\times M}$ denotes the covariance matrix of $\mathbf{X}\in R^{N\times M}$, and $\mathbf{\Sigma}_y$ for the covariance matrix of $\mathbf{y}$. Covariance of $\mathbf{X}$ and $\mathbf{y}$ is shown by $(\mathbf{\Sigma_{xy}})_{M\times 1}$. Define the membership matrix $\mathbf{y}$ to be a dummy matrix $\mathbf{y}=\begin{pmatrix}1_{n_1}\\0_{n_2}\end{pmatrix}$, where $\{n_i\}_{i=1}^2$ denotes the number of examples in each class. 0_{n_i} or 1_{n_i} is a $(n_i\times 1)$ vector of all zeros or ones, respectively. After normalization of $\mathbf{X}$, let

$$\mathbf{S_X}=\frac{1}{N-1}\mathbf{X}^T\mathbf{X},\quad \mathbf{S}_y=\frac{1}{N-1}\mathbf{y}^T\mathbf{y},\quad \mathbf{S_{Xy}}=\frac{1}{N-1}\mathbf{X}^T\mathbf{y}$$

be the estimates of the covariance matrix $\mathbf{\Sigma_x}$, $\mathbf{\Sigma}_y$ and the cross-product covariance matrix $\mathbf{\Sigma_{xy}}$.

Suppose $\mathbf{H}=\sum_{i=1}^2 n_i(\overline{\mathbf{x}}_i-\overline{\mathbf{x}})(\overline{\mathbf{x}}_i-\overline{\mathbf{x}})^T$ denotes the between-class scatter matrix and $\mathbf{E}=\sum_{i=1}^2\sum_{j=1}^{n_i}(\mathbf{x}_{ij}-\overline{\mathbf{x}}_i)(\mathbf{x}_{ij}-\overline{\mathbf{x}}_i)^T$ the within-class scatter matrix, where $\overline{\mathbf{x}}_i$ is class-conditional mean vector, $\overline{\mathbf{x}}$ is mean vector of total observations and $\mathbf{x}_{ij}$ is the jth observation in the ith group. Using the relationship $\mathbf{E}+\mathbf{H}=(N-1)\mathbf{S_x}$, LDA turns out to be an optimization problem $\mathrm{argmax}_{a\in R^M}\{\mathbf{a}^T\mathbf{Ha}/\mathbf{a}^T\mathbf{Ea}\}$. The solution is equivalent to that of the eigen-decomposition problem $\mathbf{E}^{-1}\mathbf{Ha}=\lambda\mathbf{a}$.

Bartlett has pointed out that CCA discrimination directions are the same with directions given by LDA under the dummy group membership y and feature set $\mathbf{X}$ [25]. According to the CCA definition, it finds the direction of maximal correlation between $\mathbf{Xa}$ and $\mathbf{yb}$, as shown below

$$\max_{|\mathbf{a}|=|\mathbf{b}|=1}[\mathrm{corr}(\mathbf{Xa},\mathbf{yb})]^2 \tag{4}$$

A solution is obtained by solving the eigen-decomposition problem $\mathbf{S_x}^{-1}\mathbf{S_{xy}}\mathbf{S}_y^{-1}\mathbf{S_{yx}}\mathbf{a}=\lambda\mathbf{a}$. According to the relationship $\mathbf{S_{xy}}\mathbf{S}_y^{-1}\mathbf{S_{yx}}=\mathbf{H}/(N-1)$ and $\mathbf{E}+\mathbf{H}=(N-1)\mathbf{S_x}$, the solution of CCA is transformed into solving an equivalently eigen-decomposition problem $\mathbf{E}^{-1}\mathbf{Ha}=\mu\mathbf{a}$, where $\mu=\lambda/(1-\lambda)$.

PLS direction is determined by the below optimization problem:

$$\begin{aligned}&\underset{\mathbf{a}\in R^M}{\operatorname{argmax}}\left\{\frac{[\mathrm{cov}(\mathbf{Xa},\mathbf{yb})]^2}{(\mathbf{a}^T\mathbf{a})(\mathbf{b}^T\mathbf{b})}\right\}=\{\mathbf{a}_{k+1},\mathbf{b}_{k+1}\}\\&\mathbf{b}\in R\\&\mathbf{a}^TA=0\end{aligned} \tag{5}$$

where $\mathbf{A}=[\mathbf{a}_1,\mathbf{a}_2,\ldots,\mathbf{a}_k]_{M\times k}$, $\mathbf{a}_{k+1}$ is the eigenvector of $\mathbf{\Sigma_{xy}}\mathbf{\Sigma_{yx}}$ corresponding to the $(k+1)$th largest eigenvalue, and $\mathbf{b}_{k+1}=\mathbf{\Sigma_{yx}}\mathbf{a}_{k+1}$. Since $[\mathrm{cov}(\mathbf{Xa},\mathbf{yb})]^2=\mathrm{var}(\mathbf{Xa})[\mathrm{corr}(\mathbf{Xa},\mathbf{yb})]^2\mathrm{var}(\mathbf{yb})$, the objective function of PLS is considered as a form of CCA balanced with variance in both $\mathbf{X}$ and $\mathbf{y}$ space. Coded with dummy variables, the $\mathbf{y}$ space is not considered meaningful. Removing $\mathbf{y}$ space from the CCA objective function, it turns out

$$\frac{[\mathrm{cov}(\mathbf{Xa},\mathbf{yb})]^2}{\mathrm{var}(\mathbf{yb})}=\mathrm{var}(\mathbf{Xa})[\mathrm{corr}(\mathbf{Xa},\mathbf{yb})]^2 \tag{6}$$

This gives a new form of PLS with orthogonal constraints. Its objective function is shown below

$$\begin{aligned}&\max_{\mathbf{a}\in R^M}\left\{\frac{[\mathrm{cov}(\mathbf{Xa},\mathbf{yb})]^2}{(\mathbf{a}^T\mathbf{a})\mathrm{var}(\mathbf{yb})}\right\}=\max_{\mathbf{a}\in R^M}\left\{\frac{\mathbf{a}^T\mathbf{S_{xy}}\mathbf{S}_y^{-1}\mathbf{S_{yx}}\mathbf{a}}{\mathbf{a}^T\mathbf{a}}\right\}\\&\mathbf{b}\in R\qquad\qquad\qquad\qquad\mathbf{a}^TA=0\\&\mathbf{a}^TA=0\end{aligned} \tag{7}$$

Applying the correlation between $\mathbf{H}$ and $\mathbf{S_{xy}}$, $\mathbf{S_{xy}}\mathbf{S}_y^{-1}\mathbf{S_{yx}}=\mathbf{H}/(N-1)$, (7) is considered as an optimization problem of $\mathbf{H}$, that is $\mathrm{argmax}_{\mathbf{a}\in R^M}\{\mathbf{a}^T\mathbf{Ha}/\mathbf{a}^T\mathbf{a}\}$. Furthermore, its solution is achieved by solving the eigen-decomposition problem $\mathbf{Ha}=\lambda\mathbf{a}$. Eigenvalue λ_k corresponds to eigenvector $\mathbf{a}_k$ on which the between-class scatter matrix is projected.

Similarly, PCA, as another common data analysis method, is treated as an eigen-decomposition problem $(\mathbf{E}+\mathbf{H})\mathbf{a}=\lambda\mathbf{a}$, and its original optimization objective function is $\max_{|a|=1}[\mathrm{var}(\mathbf{Xa})]$, in which $\mathrm{var}(\mathbf{t})=\mathbf{t}^T\mathbf{t}/N$.

2.3. A unified eigen-decomposition framework of PLS, PCA, CCA and LDA

From the above discussions, it is clear that PLS, CCA, LDA and PCA have some interesting information in common. Though in different optimization problems and using implementation techniques, they come to the same type of eigen-decomposition problems $\Sigma\mathbf{a}=\lambda\mathbf{a}$, under the condition that dummy group

Table 1
Eigen-decomposition of CCA, LDA, PCA and PLS.

Method	Objective function	Eigen-decomposition of Σ
CCA (y is dummy matrix)	$\max_{\|\mathbf{a}\|=\|\mathbf{b}\|=1}[\mathrm{corr}(\mathbf{Xa},\mathbf{yb})]^2$	$\Sigma=\mathbf{E}^{-1}\mathbf{H}$
LDA	$\max_{\|\mathbf{a}\|=1}\{\mathbf{a}^T H\mathbf{a}/\mathbf{a}^T E\mathbf{a}\}$	$\Sigma=\mathbf{E}^{-1}\mathbf{H}$
PCA	$\max_{\|\mathbf{a}\|=1}[\mathrm{var}(\mathbf{Xa})]$	$\Sigma=\mathbf{E}+\mathbf{H}$
PLS (y is dummy matrix)	$\max_{\|\mathbf{a}\|=\|\mathbf{b}\|=1}[\mathrm{cov}(\mathbf{Xa},\mathbf{yb})]^2$	$\Sigma=\mathbf{H}$

membership matrix $\mathbf{y}$ is adopted. Σ is a square matrix, and different methods apply different Σs. Table 1 illustrates this point distinctly.

According to the definition, $\mathbf{E}$ is represented by $\mathbf{E}=p_o\sum_o+p_c\sum_c$, where p_o denotes the number of positive samples, and p_c is the negative, Σ_o represents the positive covariance matrices and Σ_c is the negative. Obviously, class distribution has a great impact on $\mathbf{E}$. Consequently, dimension reduction by PCA and LDA is inevitably affected by asymmetric distribution. Therefore, we want to know the impact of asymmetric data on $\mathbf{H}$, specifically on PLS.

3. PLS based asymmetric pattern classification

3.1. Illustration: PLS, PCA, APCDA in asymmetric pattern classification

Between-class scatter matrix $\mathbf{H}$ is quite important in discrimination. An ideal class distribution for classification is one in which the distance between class centers is large and data within each class assembles tight, so as to maximize the margin between classes. Fisher's LDA applies this concept, which minimizes the ratio between $\mathbf{H}$ and $\mathbf{E}$. As for PLS, which ignores class structure of $\mathbf{E}$, the goal is only to maximize $\mathbf{H}$. According to the definition of $\mathbf{H}$, it equals to $(N/\mathrm{rate})(\bar{\mathbf{x}}_1-\bar{\mathbf{x}})(\bar{\mathbf{x}}_1-\bar{\mathbf{x}})^T$ in binary classification, where $\bar{\mathbf{x}}_1$ denotes the mean vector of the minority class, and "rate" is the proportion of the two classes in number. It is apparent that the eigenstructure problem to maximize $\mathbf{H}$ is equivalent to eigen decomposition of $(\bar{\mathbf{x}}_1-\bar{\mathbf{x}})(\bar{\mathbf{x}}_1-\bar{\mathbf{x}})^T$ [24]. The distribution rate is not concerned in solving this eigenstructure problem. In other words, PLS is immune to asymmetric class distribution to some extent in binary classification.

As a comparison, PCA only distinguishes gross variability, which is sum of $\mathbf{H}$ and $\mathbf{E}$. Surely, PCA suffers from the asymmetric class distribution owing to asymmetric class proportion in $\mathbf{E}$. Jiang [18] improved PCA by giving heavier weight in $\mathbf{E}$ to less reliable covariance matrix caused by less training samples, so that more dimensions characterized by the small variances of this class are removed. This is the first step in APCDA, named by asymmetric PCA (APCA). The corresponding eigen-decomposition problem is $(p_c\sum_o+p_o\sum_c+\mathbf{H})=\Phi\Lambda\Phi^T$, where Φ is m eigenvectors corresponding to the m largest eigenvalues. After that, LDA and covariance discriminant analysis (CDA) are integrated with some adjustable parameters to help extract meaningful features in the class which occupies larger subspace. This is the second step of APCDA. It is realized by working out the eigenvectors to $(\hat{\sum}_o+\beta\hat{\sum}_c)^{-1}(\hat{\sum}_o+\gamma\hat{\mathbf{H}})=\hat{\Phi}\Lambda\hat{\Phi}^T$, where $\hat{\sum}_o=\Phi^T\sum_o\Phi$, $\hat{\sum}_c=\Phi^T\sum_c\Phi$, $\hat{\mathbf{H}}=\Phi^T\mathbf{H}\Phi$. β is a parameter about asymmetry of two classes , and γ is the tradeoff between LDA and CDA.

In order to compare performance of three dimension reduction methods, i.e. PLS, PCA and APCDA, on unbalanced data, Fisher's ratio $R_i=H_i/E_i$ is adopted, where i means the ith dimension of the score vectors. It is reasonable to consider that the value of R_i represents the contribution of the ith dimension to classification. Since the first dimension of a latent variable contains majority of information, we merely compare the contribution of the first dimension of score vectors generated by the three methods. Therefore, $R=H_1/E_1$ is selected as an index, where $H_1=\boldsymbol{\alpha}^T\mathbf{H}\boldsymbol{\alpha}$ and $E_1=\boldsymbol{\alpha}^T\mathbf{E}\boldsymbol{\alpha}$, $\boldsymbol{\alpha}$ is the eigenvector corresponding to the first largest eigenvalue. We use a toy problem to make the comparison. The data set $\mathbf{X}\in R^4$ has 1000 examples with four features, and the asymmetric rate, ratio of the number of the negative class examples to that of the positive examples, varies from 1:1 to 30:1. The data is created by using the following formula:

$$Y=\mathrm{sign}(\sin(X_1+2X_2)-\cos(X_3)+X_4) \tag{8}$$

where $\mathbf{X}=[X_1,X_2,X_3,X_4]$ is randomly distributed with mean 0 and stand deviation 1 in each dimension. The 'sign' is a sign function. The number of principal components extracted is set to 2 for both PLS and PCA. APCDA extracts 3 in the first step, and then selects 2 from the 3 components. β is set to be 1 and γ equals 1000.

We calculate the value of R as the asymmetric rates increase from 1:1 to 30:1. The variance of R is illustrated in Fig. 3, where x-axis is the asymmetric rate and y-axis is R. From the figure, we can see that the value of R declines as the rate ascends. Examining the definition of H_1, $H_1=(N/rate)(\bar{x}_1-\bar{x})(\bar{x}_1-\bar{x})^T$, it is obvious to see that the value of H_1 is significantly influenced by the asymmetric rate. As the increase of rate, the weak growth of $(\bar{x}_1-\bar{x})(\bar{x}_1-\bar{x})^T$ is killed by sharp reduction of $N/rate$. As a result, H_1 gets lower and lower. Meanwhile, E_1 benefits from the part $p_c\sum_c$ since the negative class occupies more proportion, which makes E_1 get larger. Consequently, the ratio of H_1 to E_1 descends. Though H_1 is closely related to asymmetric rate, the eigenstructure problem to maximize $\mathbf{H}$ is unrelated to that. Consequently, the score vectors created by PLS are immune to class distribution. An important thing that deserves one's notice is that under the same asymmetric rate, PLS always provides the largest R. It reveals the great capability of PLS in handling asymmetric data analysis. APCDA improves PCA quite a lot, but is still not as effective as PLS. The main reason is that in the APCA step, the goal of maximizing gross variability $\mathbf{H}+\mathbf{E}$, disadvantages extracting favorable features to asymmetric classification. From Fig. 3, we also find that the R values provided by APCDA are not as stable as those by PCA and PLS. One main reason is that the two steps designed in APCDA disturb the dominating impact of H_1 or E_1 in calculating R.

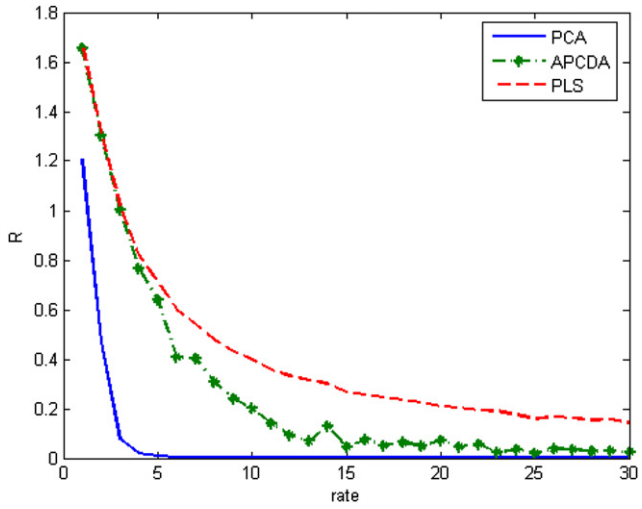

Fig. 3. Ratio of between-class scatter matrix to within-class scatter matrix projected on the first dimension of score vectors generated by PCA, APCDA and PLS, as the increase of asymmetric rates of simulated data set.

3.2. An asymmetric PLS classifier

For one dimensional output, the PLS model for classification is represented as

$$\tilde{Y} = \text{sign}\left(\sum_{i=1}^{k} m_i \mathbf{t}_i\right) = \text{sign}(\mathbf{t}\cdot\mathbf{m}) \tag{9}$$

where $\mathbf{t}_i$ is the ith score vector of $\mathbf{X}$, and $m_i = v_i q_i$. Take a two dimensional binary toy dataset for example, the dominated class is signed by 'cross', while the minority class by 'circle'. Class distribution is in accord with Gaussian distribution. After the PLS operation on feature set $\mathbf{X}$, the distribution of score vectors (SV distribution) is shown in Fig. 4, in which x-axis is T_1 meaning the first dimension of score vector $\mathbf{t}$, while y-axis is T_2 meaning the second dimension of $\mathbf{t}$. We can get the conclusion from Fig. 4 that, the larger number of examples in majority class; the closer between point O (0, 0) and the center of that class. This phenomenon may be explained by the requirement of zero mean of $\mathbf{X}$ during normalization. PLS procedure does not change the characteristic.

Dashed line $y = \mathbf{t}\cdot\mathbf{m}$ is denoted by L. It goes through the point O, apart from the center of the minority class. According to the discrimination function of PLS model, $\tilde{Y} = \text{sign}(\mathbf{t}\cdot\mathbf{m})$, points located on the left side of L are expected to be classified as 'cross' class, while those located on the right side are considered as 'circle'. Since L passes across the majority class, away from the minority class, the PLS Classifier (PLSC) provides strong bias in detecting the minority class, resulting in loss of true negatives. In other words, PLSC risks misclassifying some majority class data to label correctly most of the minority class data. In some real world applications, e.g. medical diagnosis, misclassification of the minority class will cause larger cost than the majority class. From this point of view, PLSC has a prominent advantage in treating asymmetric data analysis.

In practice, we hope to keep classification sensitivity of the minority class while enhance specificity of the majority class. Since score vectors $\mathbf{t}$ put most of the information on the first dimension, we only take the first dimension into consideration. Label the center of two classes at the first dimension as A and B shown in Fig. 2. Make circles with points A and B as the centers, and standard deviation of two classes as radius, respectively, shown in dashed line. Thus 68% of the data in each class is included inside the circles, supposing the data distribution is in Gaussian. Described in a mathematic way, the relationship between two circles in position is expressed in the formula:

$$M_o - M_c = c(r_o + r_c) \tag{10}$$

where M_o and M_c are centers of score vectors belonging to the positive and negative classes on the first dimension, r_o and r_c indicates radius of two circles, and c is a parameter. The larger value of c, the less overlapping between both circles.

In the case that both circles are tangent ($c=1$), if we move line L along the first dimension to the tangent point, the new line is signed as L^*, as shown in Fig. 4. L^* is located closer to the positive class than L, which may risk to lower sensitivity of the positive class. However, L^* restores intensively the specificity of the negative class damaged by L. The novel PLS classifier with L^* as the classification plane is named as Asymmetric PLS Classifier (APLSC) in this paper. It is represented by $\tilde{Y} = \text{sign}(\mathbf{t}\cdot\mathbf{m} - b)$, where

$$b = m_1(M_o - r_o c). \tag{11}$$

When both circles are intersected or separated, L^* is determined by using shrink or expansion of two circles to the same extent till they get touched (circles shown in solid line in Fig. 4), and then move the line L to the touched point. Calculation of b is the same with (11).

Suppose $\hat{\mathbf{t}}$ expresses the score vectors of test examples, modeling of APLSC is summarized as in Algorithm 2.

Algorithm 2. Asymmetric PLS classifier (APLSC)

Input: Training Feature set $\mathbf{X}$, Test Feature set $\mathbf{X}_t$
Target Label $\mathbf{y}$ of $\mathbf{X}$
Number of components k

Output: Output: Predicted Label $\tilde{Y}$ of $\mathbf{X}_t$

1: Use Algorithm 1 to obtain Projection matrix $\mathbf{w}$, parameter $\mathbf{V}$ and loading $\mathbf{Q}$;
2: Calculate $\mathbf{m}$, M_o, M_c, r_o and r_c according to their definitions in Eqs. (9) and (10);
3: Calculate b according to $b = m_1(M_o - r_o(M_o - M_c)/(r_c + r_o))$;
4: $\hat{\mathbf{E}}_0 = \mathbf{X}_t$;
5: **for** $i = 1$ to k **do**
6: $\hat{\mathbf{t}}_i = \hat{\mathbf{E}}_{i-1}\mathbf{w}_i$;
7: $\hat{\mathbf{E}}_i = \hat{\mathbf{E}}_{i-1} - \hat{\mathbf{t}}_i\hat{\mathbf{t}}_i^T\hat{\mathbf{E}}_{i-1}/(\hat{\mathbf{t}}_i^T\hat{\mathbf{t}}_i)$;
8: end for
9: $\tilde{Y} = \text{sign}\left(\sum_{i=1}^{k} m_i\hat{\mathbf{t}}_i - b\right)$

APLSC originates from the need to balance sensitivity and specificity, which inherits the advantage of PLSC which is prone to protect the minority class from being misclassified and boosts the performance on the majority class.

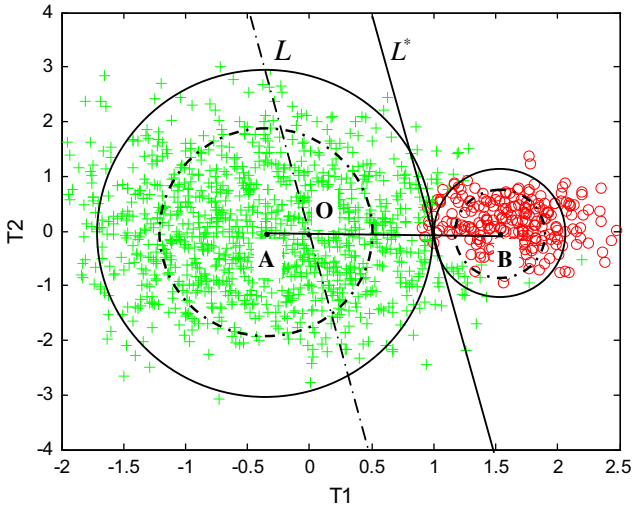

Fig. 4. Two-dimensional distribution and the optimal classification line L^* of score vectors produced by APLSC on one artificial dataset (the majority class data is denoted by 'cross', while the minority class by 'circle').

Table 2
Information of data sets from UCI repository.

Data sets	Features	Size	Class distribution
Abalone	8	4177	36/4141
Breast cancer (diagnostic)	30	569	212/357
Breast cancer (original)	9	699	241/458
Hepatitis	19	155	32/123
Hypothyroid	25	3163	151/3012
SPECT	22	267	55/212

Table 3
Statistical TPR values of seven classifiers on six UCI data sets.

Classifiers	SVMs	UnSVMs	APCDA	PLSC	APLSC	SMOTE	Adaboost
Abalone	0.00(0.00)	0.00(0.00)	0.00 (0.00)	**0.99(0.04)**	0.93(0.12)	0.17(0.16)	0.00(0.00)
Breast Cancer (Diagnostic)	0.95(0.02)	0.96(0.02)	0.92(0.03)	**0.97(0.01)**	0.96(0.01)	0.96(0.03)	0.94(0.03)
Breast Cancer (Original)	0.95 (0.02)	0.96(0.02)	0.98(0.01)	0.98 (0.01)	**0.99(0.01)**	0.97(0.02)	0.93(0.03)
Hepatitis	0.45(0.17)	0.56(0.18)	0.82(0.13)	**0.85(0.05)**	0.76(0.13)	0.51(0.18)	0.60(0.16)
Hypothyroid	0.73(0.07)	0.76(0.05)	0.78(0.04)	**0.99(0.01)**	0.98(0.02)	0.75(0.11)	0.73(0.07)
SPECT	0.45(0.12)	0.45(0.16)	0.62(0.07)	**0.87(0.06)**	0.78(0.09)	0.53(0.13)	0.56(0.01)
Average	0.59(0.06)	0.62(0.07)	0.68(0.07)	**0.94(0.06)**	0.90(0.09)	0.65(0.11)	0.63(0.06)

Table 4
Statistical AUC values of seven classifiers on six UCI data sets.

Classifiers	SVMs	UnSVMs	APCDA	PLSC	APLSC	SMOTE	Adaboost
Abalone	0.50(0.00)	0.50 (0.00)	0.50(0.00)	0.77(0.04)	**0.79(0.05)**	0.57(0.08)	0.50(0.00)
Breast Cancer (Diagnostic)	0.96(0.02)	0.96(0.02)	0.92(0.02)	**0.97(0.02)**	0.96 (0.02)	**0.97(0.02)**	0.96(0.02)
Breast Cancer (Original)	0.96(0.01)	0.96(0.01)	0.97(0.01)	**0.98(0.007)**	**0.98 (0.01)**	0.96(0.02)	0.95(0.02)
Hepatitis	0.69(0.12)	0.74(0.09)	0.78(0.08)	0.78(0.07)	**0.79(0.07)**	0.72(0.12)	0.76(0.09)
Hypothyroid	0.86(0.03)	0.88(0.03)	0.86(0.03)	0.84(0.01)	**0.95(0.03)**	0.87(0.05)	0.86(0.04)
SPECT	0.69(0.09)	0.68(0.08)	0.74(0.04)	0.73(0.05)	**0.75(0.08)**	0.72(0.08)	0.73(0.06)
Average	0.77(0.04)	0.78(0.04)	0.80(0.03)	0.84(0.04)	**0.87(0.04)**	0.80(0.06)	0.79(0.04)

4. Experiments and discussions

In order to validate the analysis in the above section and examine the effect of APLSC, we carry out an empirical study on UCI benchmark data sets. APCDA is a significant improvement in dimension reduction for asymmetric data classification. The minimum Mahalanobis distance classifier is employed with APCDA as a base classifier. UnSVMs are especially designed for handling the unbalanced problem based on general SVMs. SMOTE and ensemble methods have been widely applied to address the unbalanced classification problems. We pick out Adaboost, one of the classical ensemble methods, accompanied with SMOTE, UnSVMs and APCDA in comparison with APLSC on six UCI data sets with different asymmetric levels. The regular PLSC and SVMs are also considered in comparison. SVMs and UnSVMs are applied directly to data sets without any dimension reduction procedure.

4.1. Data sets and experimental setting

Six data sets are collected from UCI repository [26], where missing values on continuous features are set to the average value and those on nominal features are set to the majority value. The information of data sets is concluded in Table 2. All the data sets except for Abalone are binary classification. The Abalone data set is about the age information of abalone from physical measurements. We classify those whose age is over 20 as the positive class and those younger or equal to 20-year-old as the negative class. The processed data sets in MATLAB format are published.[1]

For the SVMs and UnSVMs methods, 'rbf' kernel function is applied with the form $K(x,y)=\exp(-|x-y|^2/(2\sigma^2))$, where soft margin of SVMs C, kernel parameter σ and balanced-ridge designed in UnSVMs for adjusting weights between the majority and the minority classes, are selected from $[10^{-1}, 1, 10, 10^2]$, $[2^{-1}, 1, 2, 2^2, 2^3]$ and $[10^{-1}, 1, 10, 10^2]$, respectively, by using 3-folds cross validation on the training set. The base learner of SMOTE is SVMs. The minority class is oversampled at 1000 percent for the Abalone and Hypothyroid data sets, and 200 percent for the rest data sets. We select 5 SVMs as the base learners of Adaboost. The hyper parameters of base classifiers are obtained for each run using 3-folds cross validation. The number of principle components for PCA varies with various applications. For APCDA, we extract half of the dimension number of feature set for dimension reduction in the first APCA step, and in the second step the same number of score vectors is adopted as that in APLSC. $\beta=1$ and $\gamma=1000$ are set for APCDA according to [18].

In experiments, two-thirds examples in each data set are selected randomly for training and the rest one-third for test. Both the training and test sets have the same class distribution. Experiments on each data set are repeated 20 times with different training-test partitions. Average results and their standard deviation are calculated for comparative analysis.

Different criteria have been adopted to measure the classification performance [16]. We make TP, TN, FP, FN stand for the number of true positive, true negative, false positive, false negative examples, respectively. Hence, TPR is defined as $TP/(TP+FN)$, also known as sensitivity. TNR is defined as $TN/(TN+FP)$, also called as specificity. TPR is an important criterion in the unbalanced problem, because 'minority class' is always more attractive to us. Another great concern in this paper is to promote the overall prediction accuracy while maintaining high values of TPR. The area under a receiver operating characteristic curve (AUC) actually evaluates classifier ability between sensitivity and specificity when varying a classification threshold [27]. It is a commonly used metric to evaluate the overall prediction performance. In this paper, AUC, together with TPR, is selected as indications of the model performance, while solely one of them merely introduces one-side view.

4.2. Experimental results

We implemented the algorithms on the MATLAB platform, and the code of APLSC is published here.[2] Experiments are performed on a PC with 2.66 G CPU and 4 G RAM. Tables 3 and 4 show the

[1] http://levis.tongji.edu.cn/gzli/data/aplsc_data.zip.

[2] http://levis.tongji.edu.cn/gzli/code/aplsc_code.zip

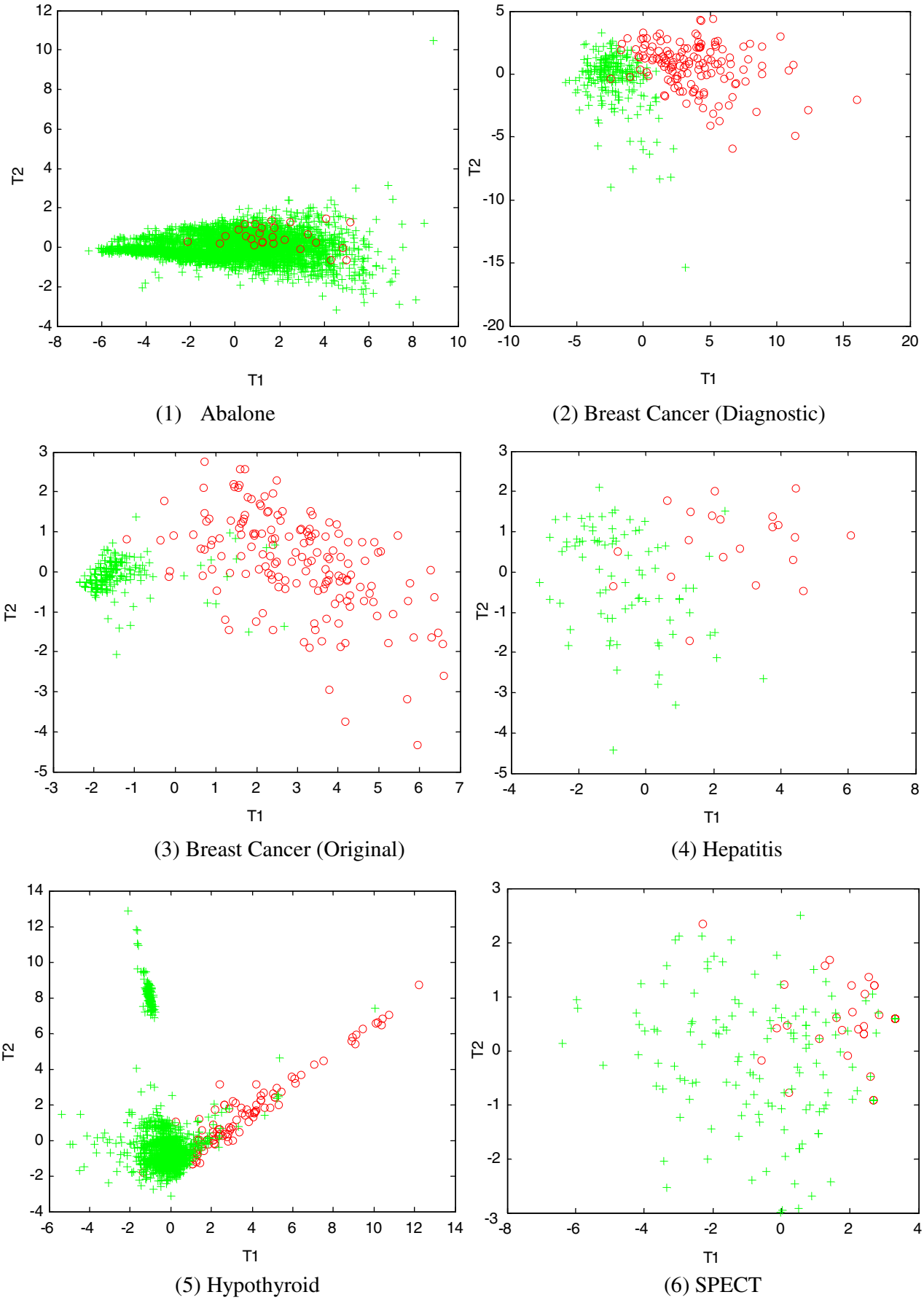

Fig. 5. Two-dimensional distribution of score vectors of six UCI data sets (T1 represents the first dimension of score vectors, and T2 means the second dimension. the majority class data is denoted by 'cross', while the minority class by 'circle').

statistical results of seven classifiers working on six UCI data sets, where the table entries are the mean values of TPR and AUC, and the values in the bracket are their standard deviations. The best results on each data set are shown in bold.

From Tables 3 and 4, it is apparent that: (1) PLSC outperforms the other classifiers in correctly classifying the minority class. APLSC provides almost as good results as PLSC does in TPR, except on the Hepatitis and SPECT data sets. But it is still better than other five classifiers. It verifies the fact that on seriously overlapping data sets (for example SPECT), the bias designed by APLSC to retrieve the true negative loss produces a few false positive. However, APLSC provides larger values of AUC than PLSC does on five out of six data sets, which confirms the capability of APLSC to promote the overall prediction accuracy. Under seriously asymmetric class distribution, e.g. on Abalone, both of them give high prediction accuracy to the minority class, while other methods corrupt. (2) UnSVMs, as the unbalanced version of SVMs, help to emphasize the positive class in kernel matrix, and present slightly better results than SVMs. (3) APCDA provides similar results with PLSC. However, it fails in predicting the minority class on Abalone data set. It appears less stable than PLSC and APLSC. (4) SMOTE and Adaboost enhance the classification performance of SVMs on unbalanced data. However, they appear weaker than APLSC when facing highly skewed data.

In order to present analysis in more details, the score vector distributions of the six data sets are shown in Fig. 5. Since the first two dimensions of score vectors generally contain most of the information of feature sets, two-dimension graph is plotted. The x-axis of each figure, $T1$, represents the first dimension of score vectors, and the y-axis, $T2$, means the second dimension.

From Fig. 5, we can see that: (1) On Abalone, data points from the majority class nearly cover those from the minority class. (2) On the next two data sets, Breast Cancer (Diagnostic) and Breast Cancer (Original), the two classes locate separately. Especially for Breast Cancer (original), the distance between two classes is larger than that in Breast Cancer (diagnostic). (3) On SPECT and Hepatitis, data points scatter loosely. The majority class and the minority overlap to each other to some extent. (4) On Hypothyroid, data points from both classes tightly cluster together.

Observing the performance of classifiers in Tables 3 and 4 and the SV distribution in Fig. 5, we can see that (1) APLSC is affected by SV distribution significantly. Under seriously overlapping, such as on Abalone and SPECT data sets, APLSC provides relatively lower prediction accuracy than it does on other data sets. (2) SVMs and UnSVMs are more stable to data overlapping than the other methods. The nonlinear mapping of feature set to high dimension kernel space overcomes the overlapping problem. However, serious unbalance of class proportion makes both classifiers fail to distinguish the minority class, e.g. on Abalone data set. One reason is that their loss function aims at minimizing the number of classification errors and/or the structure risk, not concerning information of the class label distribution. Moreover, they suffer from overfitting to some extent on SPECT and Hepatitis, resulting less effective than APLSC. (3) The performance of APCDA is also sensitive to data overlapping. Results on SPECT confirm this point of view.

4.3. Discussions

APLSC outperforms other methods on five data sets out of six with a less number of parameters. The main factors that influence performance of APLSC are overlapping of both classes and the number of components in score vectors, which are discussed below.

Table 5
Position information of class centers on six UCI data sets (M denotes center location on the first dimension of score vector, r is standard deviation of class, and c means position relationship between both classes).

Data sets	M_o	M_c	r_o	r_c	*rate*	c
Abalone	2.790	−0.024	1.861	2.590	**115.000**	0.633
Breast Cancer (Diagnostic)	3.851	−2.280	2.920	1.450	7.141	1.406
Breast Cancer (Original)	2.990	−1.580	1.581	0.636	4.550	**2.063**
Hepatitis	2.543	−0.651	1.859	1.363	3.980	1.001
Hypothyroid	3.710	−0.187	2.710	0.815	**20.000**	1.106
SPECT	2.110	−0.553	1.344	2.132	3.860	0.767

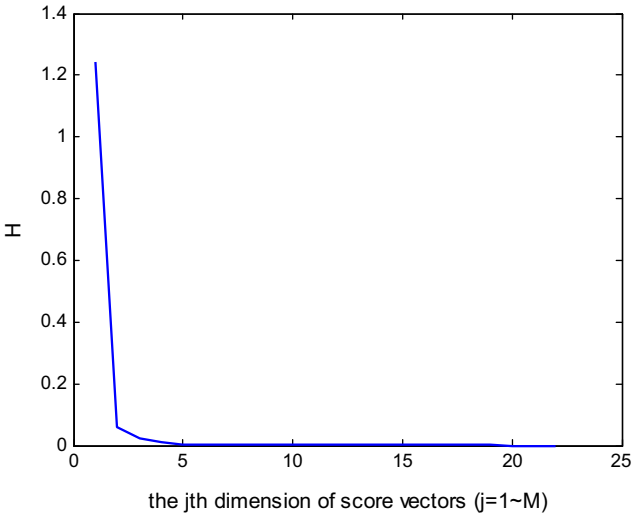

Fig. 6. The value of between-class scatter matrix projected on the eigenvectors of the SPECT data set.

4.3.1. Position relationship between the majority class and the minority class

In order to study the overlapping level of both classes, position information of class centers is collected in Table 5. Definition of c, M_o, M_c, r_o and r_c may be referred to Eq. (10). c indicates the position relationship between both classes. The bigger value of c, the larger margin between both classes, in other words, the easier for classification. Rate is defined as unbalanced ratio, denoting the rate between the majority class and the minority in number.

The column of "rate" in Table 5 shows that Abalone and Hypothyroid are the most seriously unbalanced data sets. Observing c, the largest value of c is provided by Breast Cancer (Original), whose SV distribution in Fig. 5 shows it is nearly linear separable.

Considering Table 5 and experimental results of seven classifiers, we obtain several observations: (1) PLSC and APLSC have the characteristic of protecting the minority class from misclassified no matter how "minor" it is, and dimension reduction is not affected by class distribution. Other methods, however, do not avoid the influence of unbalance. (2) Highly overlapping makes small values of c, which harms specificity of APLSC to the majority class, thus causes poor overall accuracy. The main reason lies in that APLSC is a linear classifier, which does not separate the majority class from the minority appropriately when both classes score vectors are intensively overlapping. Nonlinear techniques, such as the kernel trick, may be applied to solve this problem.

4.3.2. The number of components

The number of principle components k, in other words, dimension of score vectors, is the only but critical parameter in

Table 6
Absolute and relative computational time of seven classifiers in one training-test partition on six UCI data sets. (Measurement unit is second. The absolute values are listed on the left side of "/", and the relative on the right.)

Classifiers	SVMs	UnSVMs	APCDA	PLSC	APLSC	SMOTE	Adaboost
Abalone	1.19/74.20	2.42/151.40	0.047/2.93	0.016/1.00	0.016/1.00	2.75/181.87	9.71/606.87
Breast Cancer (Diagnostic)	0.078/4.90	0.11/6.80	0.016/1.02	0.016/1.00	0.016/1.00	0.14/8.75	0.47/29.31
Breast Cancer (Original)	0.079/4.90	0.11/6.80	0.016/1.10	0.015/1.00	0.015/1.00	0.14/8.75	0.47/29.31
Hepatitis	0.032/2.13	0.047/3.13	0.03/2.11	0.015/1.00	0.015/1.00	0.08/5.20	0.20/15.53
Hypothyroid	0.97/20.60	1.94/41.20	0.05/1.08	0.047/1.00	0.047/1.00	3.18/67.80	7.95/169.21
SPECT	0.062/3.87	0.078/4.87	0.016/1.00	0.016/1.00	0.016/1.00	0.09/5.69	0.29/18.50
Average	0.40/19.10	0.78/37.00	0.03/1.38	0.02/1.00	0.02/1.00	1.06/46.34	3.18/144.80

PLS. The cross validation method with non-error-rate (NER) is a classical option, though the computation expense is costly [28]. Nguyen and Rocke [29] simply fixed the number at three and suggested the classification accuracy is insensitive to the parameter when it goes beyond five. In this paper, we determine the dimension of score vectors by using a new approach related to data structure. This paper shows that eigen-decomposition of $\mathbf{H}$ is unrelated to the way in which PLS is decomposed. Eigenvalues correspond to their eigenvectors on which between-class scatter matrix is projected. The interesting thing is that these values decrease with the increasing of dimensions of the sample data. Fig. 6 illustrates this situation on the SPECT data set, where x-axis represents the jth dimension of score vectors($j = 1,2,\ldots,M$), while y-axis is the value of $\mathbf{H}$ projected on eigenvector $\mathbf{a}_j$, denoted by H. It is shown that the value of H decreases sharply in the first four dimensions, approaching to zero afterwards. A smaller value of H is considered as less additional contribution of that dimension to classification. This motivates us to define a threshold to obtain the optimal dimension of score vectors k. When the value is less than the threshold, the iteration calculation in the PLS decomposition is halted. Thus k is equal to the iteration number. Our experience shows that the threshold 0.01 is reasonable and helps to extract features favorable for classification.

4.3.3. Computational complexity of APLSC

PLS has high computational efficiency with computational complexity of $O(NMk)$, where N is the number of observations, M is the number of features, k is the number of components extracted. APLSC does not add additional computation burden to PLS. SVMs, however, involve calculating a high-dimensional kernel matrix, $K \in R^{N\times N}$, resulting a tremendous complexity in computation, $O(N^3)$. In most cases $M < N$ and $k < N$, so APLSC costs less time in computation than SVMs and UnSVMs do. This makes APLSC more appealing in dealing with large-scale data sets. Table 6 lists the absolute and relative running time of each classifier in one training-test partition on every data set, where the computing platform is a PC with 2.66G CPU and 4G RAM. The absolute values are listed on the left side of "/", and the relative on the right side. Measurement unit is second. Parameters C, σ and balanced-ridge for SVMs and UnSVMs are selected as 10, 4 and 10.

Table 6 demonstrated the superiority of APLSC in computational complexity. Besides, APCDA and PLSC are shown to be efficient in implementation, compared to the other four methods. In average, APLSC takes about 1/20 time of SVMs and 1/40 time of UnSVMs and SMOTE, nearly 1/150 time of Adaboost. UnSVMs take longer running time than SVMs, using the same values of C and σ, which indicates that UnSVMs carry heavier computation load than ordinary SVMs. Over sampling the minority class data in SMOTE does not cost much additional computational expense. For Adaboost, the more base classifiers are taken the more intensive computational complexity is produced.

5. Conclusion

We have analyzed the role of PLS in asymmetric classification, which is far beyond a naive technique for dimension reduction. The close connection between PLS and CCA, even LDA, reveals that dimension reduction by PLS is guided by maximizing the between-class scatter matrix, which helps to extract favorable features for unbalanced classification. This point of view is demonstrated by comparing the performance of PLS with those of PCA and APCDA. In a numerical experiment, PLS is proved to outperform the latter two methods. Furthermore, APLSC, an asymmetric classifier based on PLS, is proposed to enhance the overall prediction accuracy for the asymmetric binary classification problem. Experiments illustrate that APLSC outperforms the state-of-the-art methods on most of the six data sets collected from UCI repository.

It has been proved that PLS inherits the LDA structure for discrimination, which makes PLS orient beneficially for classification, while PCA not. This point has been confirmed by many theoretical works, but the main contribution herein is proposal of the Fishers'-like criterion to numerically measure the devotion of these methods to classification. In spite of the success achieved by APLSC over other methods, its superiority is in conditional restriction as a bilinear classifier. Future effort is expected to set up a model adapted to more general situations, e.g. a kernel form of APLSC.

Acknowledgements

This work was supported in part by the Nature Science Foundation of China under Grant nos. 60873129 and 30901897, the STCSM "Innovation Action Plan" Project of China under Grant no. 07DZ19726, the Shanghai Rising-Star Program under Grant no. 08QA1403200 and the Open Projects Program of National Laboratory of Pattern Recognition, China. We thank the LAMDA group for providing us with the SMOTE code, and we appreciate the encouraging suggestions from anonymous reviewers.

References

[1] Edward Y. Gang Wu, K.B.A. Chang, Kernel boundary alignment considering unbalanced data distribution, IEEE Trans. Knowl. Data Eng. 17 (6) (2006) 786–796.
[2] H. He, E.A. Garcia, Learning from unbalanced data, IEEE Trans. Knowl. Data Eng. 21 (9) (2009) 1263–1284.
[3] Z.H. Zhou, X.Y. Liu, Training cost-sensitive neural networks with methods addressing the class imbalance problem, IEEE Trans. Knowl. Data Eng. 18 (1) (2006) 63–77.
[4] H. He, Y. Bai, E.A. Garcia, S. Li, ADASYN: adaptice synthetic sampling approach for unbalanced learning, in: IEEE Joint Conference on Neural Networks, 2008, pp. 1322–1328.
[5] J. Laurikkala, Improving identification of difficult small classes by balancing class distribution, Conference of AI in Medicine in Europe: Artificial intelligence Medicine, 2001, pp. 63–66.

[6] A. Estabrooks, T. Jo, N. Japkowicz, A multiple resampling method for learning from unbalanced data sets, Comput. Intell. 20 (2004) 18–36.
[7] D. Mease, A.J. Wyner, A. Buja, Boosted classification trees and class probability/quartile estimation, J. Mach. Learn. Res. 8 (2007) 409–439.
[8] C. Drummond, R.C. Holte, C4.5, class imbalance and cost sensitivity: why under sampling beats over-sampling, in: International Conference of Machine Learning, Workshop on Learning from Unbalanced Data Sets II, 2003.
[9] X.Y. Liu, J. Wu, Z.H. Zhou, Exploratory under sampling for class unbalanced learning, in: International Conference of Data Mining, 2006, pp. 965–969.
[10] G.-Z. Li, H.-H. Meng, W.-C. Lu, J.Y. Yang,and, M.Q. Yang, Asymmetric bagging and feature selection for activities prediction of drug molecules, BMC Bioinformatics 9 (S6) (2008) S7.
[11] J.G. Xie, Z.D. Qiu, Z.J. Miao, Bootstrap FDA for counting positives accurately in imprecise environments, Pattern Recognition 40 (11) (2007) 3292–3298.
[12] L. Breiman, Bagging predictors, Mach. Learn. 24 (2) (1996) 123–140.
[13] Y.M. Sun, M.S. Kamel, A.K.C. Wong, Cost-sensitive boosting for classification of unbalanced data, Pattern Recognition 40 (12) (2007) 3358–3378.
[14] N.V. Chawla, K.W. Bowyer, L.O. Hall, W.P. Kegelmeyer, SMOTE: Synthetic Minority Over-sampling Technique, J. Artif. Intell. Res. 16 (2002) 321–357.
[15] A.F. Atiya, A. Al-Ani, A penalized likelihood based pattern classification algorithm, Pattern Recognition 42 (11) (2009) 2684–2694.
[16] H.G. Chew, R.E. Bogner, C.C..Lim, Dual support vector machine with error rate and training size biasing, in: Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, 2001, pp. 1269–1272.
[17] Y. Tang, Y.Q. Zhang, N.V. Chawla, S. Krasser, SVMs modeling for highly unbalanced classification, IEEE Trans. Syst. Man Cybern. 39 (1) (2009) 281–289.
[18] X.D. Jiang, Asymmetric principle component and discriminant analyses for pattern recognition, IEEE Trans. Pattern Anal. Mach. Intell. 31 (5) (2009) 931–937.
[19] J.H. Xue, D.M. Titterington, Do unbalanced data have a negative effect on LDA? Pattern Recognition 41 (5) (2008) 1558–1571.
[20] J. Xie, Z.D. Qiu, The effect of unbalanced data sets on LDA: a theoretical and empirical analysis, Pattern Recognition 40 (2) (2007) 557–562.
[21] R. Rosipal, N. Kramer, Overview and recent advances in partial least squares, in: Subspace, Latent Structure and Feature Selection, Statistical and Optimization, Perspectives Workshop, 2005, pp. 34–51.
[22] X.D. Jiang, B. Mandal, A. Kot, Eigen feature regularization and extraction in face recognition, IEEE Trans. Pattern Anal. Mach. Intell. 30 (3) (2008) 383–394.
[23] M. Barker, W. Rayens, Partial least squares for discrimination, J. Chemometr. 17 (2003) 166–173.
[24] H. Wold, Path models with latent variables: the NIPALS approach, in: H.M. Blalock et al.(Ed.), Quantitative Sociology: International Perspectives on Mathematical and Statistical Model Building, Academic Press, New York, 1975, pp. 307–357.
[25] M.S. Bartlett, Further aspects of the theory of multiple regressions, Proc. Cambridge Philos. Soc. 34 (1938) 33–40.
[26] C. Blake, E. Keogh, C.J. Merz, UCI repository of machine learning databases, 〈http://www.ics.uci.edu/~mlearn/MLRepository.html〉, Department of Information and Computer Science, University of California, Irvine, CA.
[27] T. Fawcett, An introduction to ROC analysis, Pattern Recognition Lett. 27 (2006) 861–874.
[28] D. Ballabio, T. Skov, R. Leardi, R. Bro, Classification of GC–MS measurements of wines by combining data dimension reduction and variable selection techniques, J. Chemometr. 22 (2008) 457–463.
[29] D.V. Nguyen, D.M. Rocke, Tumor classification by partial least squares using microarray gene expression data, Bioinformatics 18 (2002) 39–50.

Hai-Ni Qu received her B.Sc. degree and M.Sc. degree from Suzhou University in 2003 and 2006, respectively. She is currently a Ph.D. student in Department of Control Science & Engineering, Tongji University, China. Her research interests include machine learning and pattern recognition.

Guo-Zheng Li received his Ph.D. degree from Shanghai JiaoTong University in 2004. He is an Associate Professor in the Department of Control Science & Engineering, Tongji University, China. He is currently serving on the Committees at CCF Artificial Intelligence and Pattern Recognition Society, CAAI Machine Learning Society, International Society of Intelligent Biological Medicine and IEEE Computer Society. His research interests include feature selection, classifier design and machine learning in bioinformatics, traditional Chinese medicine and other intelligent applications.

Wei-Sheng Xu received his Ph.D. degree from Tongji University in 1996. He is currently a Professor in the Department of Control Science & Engineering, Tongji University. His research interests include control theory and engineering, integrated circuits design.