Regularized Label Relaxation Linear Regression

Xiaozhao Fang, *Student Member, IEEE*, Yong Xu, *Senior Member, IEEE*, Xuelong Li, *Fellow, IEEE*, Zhihui Lai, Wai Keung Wong, and Bingwu Fang

***Abstract*— Linear regression (LR) and some of its variants have been widely used for classification problems. Most of these methods assume that during the learning phase, the training samples can be exactly transformed into a strict binary label matrix, which has too little freedom to fit the labels adequately. To address this problem, in this paper, we propose a novel regularized label relaxation LR method, which has the following notable characteristics. First, the proposed method relaxes the strict binary label matrix into a slack variable matrix by introducing a nonnegative label relaxation matrix into LR, which provides more freedom to fit the labels and simultaneously enlarges the margins between different classes as much as possible. Second, the proposed method constructs the class compactness graph based on manifold learning and uses it as the regularization item to avoid the problem of overfitting. The class compactness graph is used to ensure that the samples sharing the same labels can be kept close after they are transformed. Two different algorithms, which are, respectively, based on ℓ_2-norm and $\ell_{2,1}$-norm loss functions are devised. These two algorithms have compact closed-form solutions in each iteration so that they are easily implemented. Extensive experiments show that these two algorithms outperform the state-of-the-art algorithms in terms of the classification accuracy and running time.**

***Index Terms*— Class compactness graph, computer vision, label relaxation, linear regression (LR), manifold learning.**

I. Introduction

LEAST squares regression (LSR) is a widely used regression technique in the fields of computer vision and pattern recognition. LSR is usually mathematically tractable and computationally efficient. Many variants, such as weight LSR [1], partial LSR [2], and nonnegative least squares (NNLS) [3], have been proposed for classification. LSR has also been used for feature selection and semisupervised learning. For example, Xiang *et al.* [4] proposed a discriminant LSR (DLSR) framework for multiclass classification and feature selection. The core idea is to enlarge the distance between different classes by using the ε-draggings technique. Nie *et al.* [5] proposed an adaptive loss minimization for semisupervised elastic embedding (SEE). The purpose of SEE is to solve the semisupervised leaning problem by using LSR and the graph embedding [6]. Belkin *et al.* [7] proposed a general framework for semisupervised learning where the regularized least squares can be regarded as a special case of the proposed framework. Linear regression (LR) is also a simple and very effective regression analysis method. For a collection of n training samples represented as a matrix $X = [x_1, \ldots, x_n]^T \subset \Re^{n\times m}$, the objective function of LR is as follows:

$$\min_A \|XA - Y\|_F^2 + \lambda\|A\|_F^2 \tag{1}$$

where $Y = [y_1, \ldots, y_n]^T \subset \Re^{n\times c}$ ($c \geq 2$ is the number of classes) is the corresponding binary label matrix and $A \subset \Re^{m\times c}$ is the transformation matrix. Y is defined as follows: for each training sample x_i ($i = 1, \ldots, n$), $y_i \in \Re^c$ is its label vector. If x_i is from the kth class ($k = 1, \ldots, c$), then only the kth entry of y_i is one and all the other entries are zero.

For classification problem, margins between different classes are expected to be as large as possible after they are transformed into their label space. This criterion is also used for distance learning [8] and discriminant analysis [9], [10]. In order to enlarge the margins between different classes, Leski [11] proposed an LSR model via the squares approximations of the misclassification errors [4]. Most of the above methods assume that the training samples should be exactly transformed into a linear model or a strict binary label matrix, such as Y in (1). In practice, however, as shown in some literature, this assumption is too rigid to learn a discriminative transformation matrix. To this end, some methods are proposed to solve this problem. For example, the essential of using the ε-draggings technique in [4] is to relax this rigid assumption. In [5], the elastic embedding constraint is used to capture the manifold structure and simultaneously relaxes the constraint that the predicted label matrix should be exactly equal to a linear model. Despite their great success based on the this relaxation, these methods have to encounter the problem of being easy to excessively fit the labels (overfitting). For example, in order to pursue large margin, the characteristics of some training samples may be ignored in the process

Manuscript received January 5, 2016; revised April 22, 2016 and August 18, 2016; accepted December 31, 2016. This work was supported in part by the National Basic Research Program of China (973 Program) the Hong Kong Polytechnic University (Project Code: 1-YW1C) under Grant 2012CB316400, and in part by the National Natural Science Foundation of China under Grant 61370163, Grant 61332011, Grant 61573248, and Grant 61375012. *(Corresponding author: Yong Xu.)*

X. Fang is with the School of Computer Science and Technology, Guangdong University of Technology, Guangzhou 518055, China (e-mail: xzhfang168@126.com).

Y. Xu is with the Bio-Computing Research Center, Shenzhen Graduate School, Harbin Institute of Technology, Shenzhen 518055, China, and also with the Key Laboratory of Network Oriented Intelligent Computation, Shenzhen 518055, China (e-mail: yongxu@ymail.com).

X. Li is with the State Key Laboratory of Transient Optics and Photonics, Center for OPTical IMagery Analysis and Learning, Xi'an Institute of Optics and Precision Mechanics, Chinese Academy of Sciences, Xi'an 710119, China (e-mail: xuelongli@opt.ac.cn).

Z. Lai is with the College of Computer Science and Software Engineering, Shenzhen University, Shenzhen 518055, China (e-mail: lai_zhi_hui@163.com).

W. K. Wong is with the Institute of Textiles and Clothing, The Hong Kong Polytechnic University, Hong Kong, and also with The Hong Kong Polytechnic University Shenzhen Research Institute, Shenzhen, China (e-mail: calvin.wong@polyu.edu.hk).

B. Fang is with the School of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, China (e-mail: bingwufang@163.com).

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

Digital Object Identifier 10.1109/TNNLS.2017.2648880

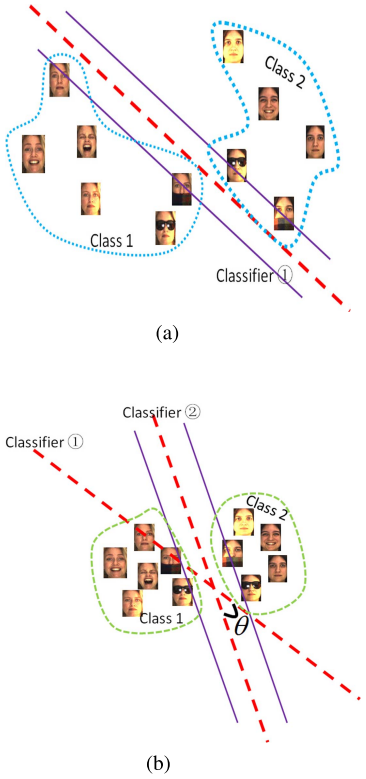


Fig. 1. Visualization of an overfitting classifier and a better classifier (the red dotted line denotes the Classifier). (a) Training data and an overfitting Classifier ①. (b) Training data and a better classifier ②. Classifier ② is obtained by rotating classifier ① with angle θ.

of training. As shown in Fig. 1(a), since the noises (faces with sunglass and scarf occlusion) greatly deviate from the normal samples, classifier ① overly fits these noises in order to classify class 1 and class 2 as soon as possible.

The above observation motivates us to develop a method which not only relaxes the above rigid assumption, but also solves the problem of overfitting. A feasible solution to avoid overfitting is to ensure that the samples sharing the same labels can be kept close together after they are transformed. Since manifold learning can capture some underlying data structures by using the graph embedding, we consider using the graph embedding to address the problem of overfitting. A brief review of some representative manifold learning methods is as follows.

Manifold learning methods aim to find a new space in which the transformed samples well preserve the intrinsic geometry structure of the samples [12]. For example, locality preserving projection [13], neighborhood preserving embedding [14], and isometric projection [15]. However, these methods do not exploit label information to improve the discriminant ability of algorithms. Therefore, Yan *et al.* [16] proposed the margin Fisher analysis method to simultaneously preserve both the intrinsic geometry structure and the discriminant structure of the samples by using the label information. Some similar methods, such as locality Fisher discriminant analysis [10], [17], locality sensitive discriminant analysis [18], and local discriminant embedding [19], are also proposed. In some semisupervised learning methods, the adjacency graph is usually used to capture the manifold structure for label propagation [5], [7]. The underlying idea of introducing the adjacency graph is to ensure that similar samples have nearly the same labels.

The adjacency graph only captures the local structure of the samples. In order to ensure that the samples from the same class can be kept close together in the transformed space, the distribution of samples from the same class should be captured by using the sample affinity. To this end, the concept of the class compactness graph is introduced by using the label information. In the class compactness graph, two different samples from the same class are linked by an undirected edge. By using the class compactness graph, the samples from the same class can be kept close together in the transformed space so that the problem of overfitting can be avoided to a great extent. As shown in Fig. 1(b), by using the class compactness graph, the samples, even the samples corrupted by the noise, sharing the same class labels keep close together when they are transformed into their label space. A better Classifier ② can be obtained in this case.

Inspired by DLSR and the manifold learning methods, in this paper, a regularized label relaxation (RLR) LR method is proposed, which not only relaxes the strict binary label matrix into a slack variable matrix by introducing a nonnegative label relaxation matrix, but also avoids the problem of overfitting by constructing the class compactness graph. The advantages of the proposed method are twofold: 1) it can enlarge the margins between different classes as much as possible and 2) it has more freedom to fit the labels better. To avoid the problem of overfitting, the method uses the class compactness graph to guarantee that the samples from the same class can be kept close together in the transformed space. In this proposed method, two different loss functions are used. The first is based on ℓ_2-norm loss function and we refer to it as RLR. The second is based on $\ell_{2,1}$-norm [20] loss function and we refer to it as robust RLR (RRLR), because it can handle noise well. Two optimization algorithms are devised to solve RLR and RRLR. RLR and RRLR have closed-form solutions in each iteration so that they are implemented easily. Extensive experiments demonstrate the superiority of RLR and RRLR. In summary, the contributions of this paper include the following.

1) It overcomes the problem of overfitting by using the class compactness graph as the regularization item. By solving the regularized minimization problem, it can obtain the near-optimal margins.
2) Two algorithms are devised for the proposed methods and both have closed-form solutions in each iteration, which makes them implement easily. The validity of these two algorithms are tested on low-dimensional, high-dimensional data sets, and the corrupted image sets and are illustrated by high classification accuracy and fast running time.
3) Theoretical and experimental analyses of convergence behaviors and computation complexities of these two algorithms are given.

The rest of this paper is organized as follows. In Section II, RLR and RRLR and their solutions are presented. In Section III, we give the algorithmic analyses.

TABLE I

INTRODUCTION OF MANY VARIABLES IN WHICH n IS THE NUMBER OF TRAINING SAMPLES, m IS THE DIMENSIONALITY OF TRAINING SAMPLES, AND c IS THE NUMBER OF CLASSES

Notation	Size	Description
X	$n \times m$	The data matrix
A	$m \times c$	The transformation matrix
Y	$n \times c$	The binary label matrix
Y'	$n \times c$	The slack variable matrix
B	$n \times c$	The luxury matrix
M	$n \times c$	The non-negative label relaxation matrix
Z	$n \times n$	The diagonal matrix
L	$n \times n$	The graph Laplacian matrix

Experiments and corresponding analyses are reported in Section IV. Section V gives the kernel extensions of RLR and RRLR. This paper is concluded in Section VI.

II. PROPOSED METHOD

A. Notation

The definitions of the training samples matrix X, label matrix Y, and transformation matrix A are the same as those in (1). $\|A\|_F^2 = \mathrm{tr}(A^T A) = \mathrm{tr}(AA^T)$ stands for the squares Frobenius norm of matrix A, where $\mathrm{tr}(\cdot)$ is the trace operator of a matrix. $\|A\|_{2,1} = \sum_{i=1}^{m} \left(\sum_{j=1}^{c} a_{ij}^2\right)^{1/2} = \sum_{k=1}^{m} \|a^k\|_2$ is $\ell_{2,1}$-norm of A, where a^k denotes the kth row of A. The $\ell_{2,1}$-norm is first presented in [21] as a rotational invariant ℓ_1-norm [22]–[25]. Table I gives the introduction of many variables in this paper.

B. RLR

Our method inherits some ideas from DLSR [4]. First, four samples are taken as an example to show how to relax the strict binary label matrix into a slack variable matrix. Let

$$Y = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} \in \Re^{4\times 3}$$

be the corresponding label matrix of the four samples. It is obvious that these four samples belong to the second, first, third, and second class, respectively. It is expected that the strict binary constraints in Y can be relaxed into the soft constraints so that it has more freedom to fit the labels. To this end, a slack variable matrix is used to substitute for the original label matrix Y. The slack variable matrix is defined as follows:

$$Y' = \begin{bmatrix} -m_{11} & 1+m_{12} & -m_{13} \\ 1+m_{21} & -m_{22} & -m_{23} \\ -m_{31} & -m_{32} & 1+m_{33} \\ -m_{41} & 1+m_{42} & -m_{43} \end{bmatrix}, \quad \text{s.t. } m_{ij} \geq 0.$$

It can be seen that this setting $m_{ij} \geq 0$ can help to enlarge the margins between different classes so that a discriminative transformation matrix can be obtained. Therefore, a more discriminant version of LR can be formulated as follows:

$$\min_A \|XA - Y'\|_F^2 + \lambda \|A\|_F^2. \tag{2}$$

A luxury matrix is devised and then combined with the original label matrix Y to develop the slack variable matrix Y'. Let B be the luxury matrix and defined as

$$B_{ij} = \begin{cases} +1 & \text{if } Y_{ij} = 1 \\ -1 & \text{if } Y_{ij} = 0 \end{cases}$$

then $Y' = Y + B \odot M$, where $\odot$ is a Hadamard product operator of matrices. The nonnegative label relaxation matrix M is defined as

$$M = \begin{bmatrix} m_{11} & \cdots & m_{1c} \\ \vdots & m_{i,j} & \vdots \\ m_{n1} & \cdots & m_{nc} \end{bmatrix} (i = 1, \ldots, n;\ j = 1, \ldots, c;\ m_{ij} \geq 0).$$

Thus, (2) can be rewritten as follows:

$$\begin{aligned} &\min_{A,M} \|XA - (Y + B \odot M)\|_F^2 + \lambda \|A\|_F^2 \\ &\text{s.t. } M \geq 0 \end{aligned} \tag{3}$$

where $\lambda \geq 0$ is a positive regularization parameter.

Although the transformation matrix A in (3) is discriminative, the problem of overfitting may occur due to this label relaxation. Based on the idea from manifold learning, the class compactness graph is proposed to address this problem. The core idea is that the samples sharing the same labels should be kept close together in the transformed space. In the class compactness graph, two nodes corresponding to two different samples from the same class are linked by an undirected edge. Therefore, the weight of the class compactness graph is defined as follows:

$$W_{ij} = \begin{cases} e^{-\frac{\|x_i - x_j\|^2}{\sigma}} & \text{if } x_i \text{ and } x_j \text{ have the same labels} \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

where σ is the heat kernel parameter. A natural assumption here could be that if W can capture the relationship that the samples sharing the same labels should be kept close together, then any two transformed samples f_i and f_j with the same labels have a bigger weight W_{ij}. A reasonable criterion for choosing a "good" mapping is to minimize the following objective function:

$$\min_f \sum_{ij} \|f_i - f_j\|^2 W_{i,j} \tag{5}$$

where $f_i = x_i A$ denotes the transformation result of sample x_i. Minimizing it is to further ensure that f_i and f_j are close in the transformed space. Based on the above insights, we propose the following objective function:

$$\min_A \sum_{ij} \|f_i - f_j\|^2 W_{i,j} = \min_A \mathrm{tr}(A^T X^T L X A) \tag{6}$$

where L is the graph Laplacian. L is defined as $L = Z - W$, where Z is a diagonal matrix and its diagonal entries are defined as $Z_{ij} = \sum_j W_{ij}$.

By integrating (6) with (3), we propose the following objective function for RLR:

$$\begin{aligned} &\min_{A,M} \|XA - (Y + B \odot M)\|_F^2 + \lambda \mathrm{tr}(A^T X^T L X A) \\ &\text{s.t. } M \geq 0. \end{aligned} \tag{7}$$

Note that in (7), since all training samples are used to learn the transformation matrix A, the regularization item $\|A\|_F^2$ is not added as in (3). This leads to the merit that the burden of parameter tuning is reduced. Let us revisit Fig. 1(b) which explicitly shows that the better classifier ② is obtained by rotating the overfitting classifier ① with an angle θ. In other words, the regularization item $\text{tr}(A^T X^T L X A)$ imposes a perturbation on the transformation matrix A in order to rotate the overfitting classifier ① with the angle θ so that the problem of overfitting in Fig. 1(a) can be addressed to some extent.

Once the optimal solution A is obtained, a test sample x_{te} is classified by using the following way. Let $y_{\text{te}} = x_{\text{te}} A$ be the transformation result of this test sample. If $h = \arg\max_i y_{\text{te}}^i$ $(i = 1, \ldots, c)$, then test sample x_{te} is assigned to the hth class, where y_{te}^i denotes the ith element of vector y_{te}.

According to [4], a direct optimization is impossible to solve problem (7). Therefore, an iterative update rule is devised to solve it and guarantee that it has a closed-form solution in each iteration. The first step of the algorithm is to solve A by fixing M. The following theorem is used for solving A.

Theorem 1: Given M, and let $Y + B \odot M = D \in \Re^{n \times c}$, then the optimal A in (6) can be calculated as

$$A = (X^T X + \lambda X^T L X)^{-1} X^T D. \tag{8}$$

Proof: Given an arbitrary M, (6) can be rewritten as

$$J(A) = \arg\min_A \|XA - D\|_F^2 + \lambda \text{tr}(A^T X^T L X A) \tag{9}$$

where $D = Y + B \odot M$. The optimal A can be obtained by making the derivation of (8) with respect to A and set it to zero. That is

$$\begin{aligned} &\frac{\partial J}{\partial A} = 2X^T XA - 2X^T D + 2\lambda X^T L X A = 0 \\ &\Rightarrow A = (X^T X + \lambda X^T L X)^{-1} X^T D. \end{aligned} \tag{10}$$

Thus, the proof is finished. □

Let $P = (X^T X + \lambda X^T L X)^{-1} X^T$, we find that P is independent of D, and thus, it can be precalculated before going to the loop. Parameter λ plays an important role in RLR, which is generally selected from a large set of candidates via cross validation. However, for image classification, the number of features is usually much larger than that of the samples ($n \ll m$). Thus, the size of $X^T X$ and $X^T L X$ is large and the cross validation is computationally expensive. To this end, first we decompose $X = U \Sigma V^T$ by the singular value decomposition, where $U \in \Re^{n \times n}$, $\Sigma \in \Re^{n \times m}$ and $V \in \Re^{m \times m}$. Denote $t = rank(X)$. Let $U_t \in \Re^{n \times t}$ and $V_t \in \Re^{m \times t}$ consist of the first t columns of U and V, respectively. Let the square matrix $\Sigma_t \in \Re^{t \times t}$ consist of the first t rows and the first t columns of Σ, then $X = U \Sigma V^T = U_t \Sigma_t V_t^T$. Then, this decomposition is substituted into P, and thus, P can be expressed as

$$\begin{aligned} P &= (X^T (I + \lambda L) X)^{-1} X^T \\ &= V \begin{pmatrix} \left(\Sigma_t U_t^T (I + \lambda L) U_t \Sigma_t\right)^{-1} & 0 \\ 0 & 0 \end{pmatrix} V^T V_t \Sigma_t U_t^T \\ &= V_t (\Sigma_t U_t^T (I + \lambda L) U_t \Sigma_t)^{-1} V_t^T V_t \Sigma_t U_t^T \\ &= V_t (\Sigma_t U_t^T (I + \lambda L) U_t \Sigma_t)^{-1} \Sigma_t U_t^T \end{aligned} \tag{11}$$

Algorithm 1 RLR

Input: Training samples matrix X; Label matrix Y;
The luxury matrix B; Laplacian matrix L;
Parameter λ;
Output: The transformation matrix A.
Initialization: $M = \mathbf{1}_{n \times c}$;
Set $t = 0$;
repeat
1. Update $A^{t+1} = (X^T X + \lambda X^T L X)^{-1} X^T D^t$, where $D^t = Y + B \odot M^t$.
2. Update $M^{t+1} = max(B \odot R^{t+1}, 0)$, where $R^{t+1} = XA^{t+1} - Y$.
3. Update $t = t + 1$.

until *Convergence*

where the size of $(\Sigma_t U_t^T (I + \lambda L) U_t \Sigma_t) \in \Re^{t \times t}$ is much smaller than that of $(X^T (I + \lambda L) X) \in \Re^{m \times m}$ in the original formulation. In this way, we finally need to calculate an inverse $(t \times t)$ matrix in advance, and thus, this dramatically reduces the computation cost when the number of features is much larger than that of training samples.

The second step of the algorithm is to solve M by fixing A. Let $XA - Y = R$. Since $A^T X^T L X A$ is uncorrelated with M, (7) can be rewritten as

$$\begin{aligned} &\min_M \|R - B \odot M\|_F^2 \\ &\text{s.t. } M \geq 0. \end{aligned} \tag{12}$$

According to [4], the optimal solution of M is finally obtained by

$$M = \max(B \odot R, 0). \tag{13}$$

The algorithm of RLR is described in Algorithm 1.

For (7), it is found that LR is equivalent to a special case of RLR, and thus, we have Proposition 1.

Proposition 1: LR is equivalent to the special case of RLR when $\lambda = 0$ and $M = 0$.

For (7), when $\lambda = 0$ and $M = 0$, it degrades into $\min_{A^*} \|XA^* - Y\|_F^2$. It is easy to find that RLR and LR are exactly the same in essence besides there is no regularization item in RLR in this case. Moreover, their solutions are represented as the following unified form: $A = (X^T X + \gamma I)^{-1} X^T Y$, where $\gamma \geq 0$ is used to avoid singular of the matrix $X^T X$.

C. RRLR

$\ell_{2,1}$-norm is usually used to perform feature selection due to the row-sparsity property [26]–[30]. Let $\delta = ||A^T x_i - (y_i + B_i \odot M_i)||_2$, then δ is not squared, and thus, outliers have less important than the squared residual $||A^T x_i - (y_i + B_i \odot M_i)||^2$. This loss function has a rotational invariant, while the ℓ_1-norm loss function does not has such property [26]. Thus, we use the $\ell_{2,1}$-norm-based loss function to replace the ℓ_2-norm-based loss function in (7) and refer to it as RRLR. The objective

function of RRLR is defined as follows:

$$\min_{A,M} \|XA-(Y+B\odot M)\|_{2,1}+\lambda \mathrm{tr}(A^TX^TLXA) \quad \text{s.t. } M\geq 0. \tag{14}$$

Some algorithms have been proposed to solve the $\ell_{2,1}$-norm minimization problem, such as the iterative update rule shown in [4], [20], [21], [26], and [27]. However, in these algorithms, the problems have no closed-form solutions. Here, we propose an efficient algorithm to tackle (14) via the alternating direction method of multipliers (ADMM) [31], [32] and guarantee that (14) has a closed-form solution in each iteration. According to ADMM, we introduce a slack variable E to replace $XA-(Y+B\odot M)$ so that objective function (14) is separable.

Equation (14) can be rewritten as

$$\min_{A,E,M,C} \|E\|_{2,1}+\lambda \mathrm{tr}(A^TX^TLXA) \quad \text{s.t. } E=XA-(Y+B\odot M), M\geq 0. \tag{15}$$

ADMM solves the following problem:

$$\min_{A,E,M,C} \|E\|_{2,1}+\mathrm{tr}(C^T(E-XA+Y+B\odot M))+\frac{\mu}{2} \times \|E-XA+Y+B\odot M\|_F^2+\lambda \mathrm{tr}(A^TX^TLXA) \quad \text{s.t. } E=XA-(Y+B\odot M), M\geq 0 \tag{16}$$

where C is Lagrange multipliers and $\mu \geq 0$ is the penalty parameter. We iteratively update A, M, E, C, and μ.

The objective of (16) can be further transformed into the following minimization problem:

$$\min_{A,E,M,C} \|E\|_{2,1}+\frac{\mu}{2}\|E-XA+Y+B\odot M+\frac{C}{\mu}\|_F^2 + \lambda \mathrm{tr}(A^TX^TLXA) \quad \text{s.t. } E=XA-(Y+B\odot M), M\geq 0. \tag{17}$$

1) Updating A and M: To obtain the solutions of A and M, the problem is translated into the problem: Solving A and M by an alternative update rule while fixing E. For this problem of solving A and M while fixing E, the following theorem is presented.

Theorem 2: Given E, the solutions of A and M can be calculated by Algorithm 1.

Proof: Given E, we have the following optimization problem:

$$\min_{A,M} \frac{\mu}{2}\|E-XA+Y+B\odot M+\frac{C}{\mu}\|_F^2+\lambda \mathrm{tr}(A^TX^TLXA) \quad \text{s.t. } M\geq 0. \tag{18}$$

Equation (18) is rewritten as

$$\min_{A,M} \frac{\mu}{2}\|XA-\left(Y+E+B\odot M+\frac{C}{\mu}\right)\|_F^2 + \lambda \mathrm{tr}(A^TX^TLXA) \quad \text{s.t. } M\geq 0. \tag{19}$$

When M is fixed and let $U=Y+E+B\odot M+(C/\mu)$, (19) can be reformulated as

$$\min_{A} \frac{\mu}{2}\|XA-U\|_F^2+\lambda \mathrm{tr}(A^TX^TLXA) \tag{20}$$

which is the form of (8) and can be solved by Theorem 1.

Next, M is solved while fixing A. Let $K=XA-E-Y-(C/\mu)$, we have the following optimization problem:

$$\min_{M} \|K-B\odot M\|_F^2 \quad \text{s.t. } M\geq 0 \tag{21}$$

which is the form of (12) and can be solved by (13), $M=\max(B\odot K,0)$.

Therefore, we have proven our claim. □

2) Updating E: In order to obtain the solution of E while fixing A and M, (17) can be converted into the following problem:

$$\min_{E} \|E\|_{2,1}+\frac{\mu}{2}\|E-H\|_F^2 \tag{22}$$

where $H=XA-Y-B\odot M-(C/\mu)$. Let e^i and h^i be the ith rows of matrix E and H, respectively. Equation (22) can be decomposed into n independent problems

$$\min_{e^i} \sum_{i=1}^{n}\|e^i\|_2^2+\frac{\mu}{2}\|e^i-h^i\|_2^2 \tag{23}$$

which can be solved by the proximal operator in [31] and [33] and the solution is

$$e^i=\begin{cases}\left(1-\dfrac{1}{\mu\|h^i\|_2}\right)h^i & \|h^i\|_2>\dfrac{1}{\mu}\\ 0 & \|h^i\|_2\leq\dfrac{1}{\mu}.\end{cases} \tag{24}$$

3) Updating Parameters C and μ: In each iteration, C and μ are updated as

$$\begin{cases}C\leftarrow C+\mu(E-XA+Y+B\odot M)\\ \mu\leftarrow \rho\mu\end{cases} \tag{25}$$

where $\rho\geq 0$ is the incremental step size parameter.

We iteratively update A, M, E, C, and μ by solving the above problems. The algorithm of RRLR is presented in Algorithm 2.

III. Algorithm Analysis

In this section, we first discuss the computation complexity and convergence of Algorithms 1 and 2. Then, we present the differences between our method and some similar methods.

A. Computation Complexity

In order to analyze the computation complexity, we assume that $m\geq n$ (i.e., the dimensionality of the feature is larger than the number of training samples).

1) RLR: The analysis of the computation complexity is as follows.

The graph weight matrix W is computed with the cost of $\mathcal{O}(n^2m)$; the complexity to compute $(X^TX+\lambda X^TLX)^{-1}X^T$ is about $\mathcal{O}(m^3+m^2n)$; the complexity of computing $B\odot R$ is $\mathcal{O}(nc)$, where c is the number of classes; the complexity to obtain XA is $\mathcal{O}(nmc)$. The main computation complexity of RLR is $\mathcal{O}((m^3+m^2n+n^2m)+(t(nc+nmc)))$, where t is the iteration number.

Algorithm 2 Solving (14) by ADMM

Input: Training samples matrix X; Label matrix Y;
The luxury matrix B; Laplacian matrix L;
Parameter λ;
Initialization: $M = \mathbf{1}_{n\times c}$; $E = \mathbf{0}_{n\times c}$; $C = \mathbf{0}_{n\times c}$;
$\mu = 0.1$; $\rho = 1.01$;
while not converged **do**
1. Alternatively update A and M
while not converged **do**
a. Update A by solving (20).
b. Update M by solving (21).
end while
2. Update E by solving (22).
3. Update C and μ according to (25).
end while
Output: Transformation matrix A

2) RRLR: The analysis of the computation complexity is as follows.

The complexity of computing E is $\mathcal{O}(nc)$; the complexity to obtain C is $\mathcal{O}(nmc + nc)$. The other analyses of computation complexity are the same as that in RLR. The main computation complexity of RRLR is $\mathcal{O}((m^3+m^2n+n^2m)+(t_1t_2+t_1)(nmc+nc))$, where t_1 is the number of outside loop and t_2 is the number of inner loop.

Although many loops are required for RLR and RRLR for optimization, it is still computationally more efficient than some conventional representation-based methods, since the iteration numbers are usually small. In the test phase, the computation complexities of our methods are negligible, since they only perform a linear computation. In addition, some computations such as $(X^TX + \lambda X^TLX)^{-1}X^T$ can be precalculated before going to the loop, and thus, the overall computation cost is acceptable. Next, we briefly analyze the computation complexities of many representation-based methods, i.e., sparse representation classification (SRC), NNLS, and metasample-based SRC (MSRC). Please note that these methods have no training time. For SRC, the computation complexity is about $\mathcal{O}(t(Rm^2 + Rnm))$, where t is the number of iterations and R is the number of test samples. For NNLS, the computation complexity is about $\mathcal{O}(t(mnR))$, where t and R are also the number of iterations and test samples, respectively. For MSRC, the computation complexity is at least higher than that of SRC, since it uses SRC to perform the final classification. We also note that the number of iterations of these representation-based methods is much greater than that of our methods, and thus, their computation cost are somewhat high.

B. Convergence

In order to analyze the convergence behavior of RLR and RRLR, we first give Theorem 3 and then discuss the convergence of RLR and RRLR.

Theorem 3: For any positive λ, (7) is a convex problem and has at least one minimum. In the case of covariance matrix X^TX being nonsingular, problem (7) has a unique minimum.

Proof: Since both X^TX and X^TLX are positive semidefinite, (7) is a convex function with respect to variable A. On the other hand, there is no zero entry in the matrix B, and as a result, (7) is a strictly convex function with respect to variable M. Thus, problem (7) is a convex problem. According to [34, Proposition 2.1.2], the minimum of (7) is nonempty. Moreover, when X^TX is nonsingular, (7) is strictly convex function with respect to both A and M, and hence, problem (7) has a unique minimum [34, Proposition 2.1.2]. □

Theorem 4: Suppose $\{A^k, M^k\}$ is a sequence generated by Algorithm 1 and X^TX is nonsingular, then $\{A^k, M^k\}$ converges to the unique optimal solution to problem (7).

Proof: Algorithm 1 is a fundamental block ordinate descend method, and both A and M can be uniquely obtained in the case of that X^TX is nonsingular. According to [35, Proposition 2.7.1], the sequence $\{A^k, M^k\}$ converges to the unique solution of problem (6). □

The convergence of ADMM was proved for two blocks [36]. Next, we will prove that there are two blocks in Algorithm 2, and thus, Algorithm 2 converges to an optimal solution.

Theorem 5: Let μ be any positive number and $\rho \geq 1$. Suppose $\{A^k, M^k, E^k\}$ is a sequence generated by Algorithm 2, then $\{A^k, M^k, E^k\}$ converges to an optimal solution to problem (15).

Algorithm 2 is a standard ADMM for problem (15), and the convergence of which can be directly deduced from the established results in [36]. Hence, we omit the proof here. One may doubt that Algorithm 2 is an extended ADMM for three blocks, because there are three primal variables A, M, and E, and hence, its convergence cannot be fully guaranteed. However, we point out that: although three primal variables are involved in Algorithm 2, variables A and M are jointly updated via an inner loop. As a result, variables A and M can be viewed as one variable in one block, and thus, Algorithm 2 is strictly a classical ADMM for two blocks. The experiments demonstrated in Section IV show that RRLR has a good convergence behavior.

C. Difference Between Our Method and Some Similar Methods

Many previous regression methods [1], [2], [4], [11] only aimed at enlarging the margins between different classes. However, the optimal margins may be not obtained, since they do not consider the problem of overfitting. However, our method can obtain the optimal margins by solving (6) and (14). The main reason is that besides seeking large margins, our method addresses the overfitting problem by introducing the class compactness graph, which can guarantee that the samples, even the samples corrupted by the noise, with the same labels can be kept close together in the transformed space. Therefore, the optimal margins are easily achieved by solving the regularized minimization problem. In previous works [5], [7], the graph embedding idea was also introduced into the LSR framework to capture the desired prosperities for the semisupervised learning. In [5] and [7], the sample and its neighbors (may from different classes) were linked by a undirected edge in order to ensure that these neighbor

samples have similar labels, whereas the samples with the same labels are linked by a undirected edge in our method. Thus, the motivations and the purposes of introducing the graph embedding in our method and in [5] and [7] are distinctly different.

IV. Experiments and Analysis

To evaluate RLR and RRLR, they are compared with LR, sparse representation-based classification (SRC (l1_ls)) [37], support vector machine (SVM) [38], DLSR [4], locality-constrained linear coding for image classification (LLC) [39], NNLS [3], LR-based classification (LRC) [40], K-nearest neighbors (KNN), MSRC [41], extreme learning machine (ELM) [42]–[44], random forest (RF) [42], logistic regression (Log_R, one-against-all) and naive Bayesian decision tree on ORL [45], Georgia Tech (GT) [46], [47], CMU PIE [48], and AR [49] face data sets, Caltech 101 image data set [50], and 7 UCI data sets [51]. The platform is MATLAB 2011b under Windows 7 on a PC equipped with a 2.67-GHz CPU and 4-GB memory. The MATLAB code of the proposed method is publicly available at http://www.yongxu.org/lunwen.html.

A. Data Sets and Experimental Setup

1) ORL: This data set contains 400 face images taken from 40 persons, with each person providing ten face images. For some persons, the images were taken at different times, with varying lighting, facial expressions (open/closed eyes, smiling/not smiling), and facial details (glasses/no glasses). The first 3, 4, 5, and 6 images per person are selected for training and the remaining for testing.

2) GT: This data set consists of 50 persons with 15 images per person. The images were taken with several variations such as pose, expression, cluttered background, and illumination. The first 4, 5, 6, 7, and 8 images per person are selected for training and the remaining for testing.

3) CMU PIE: This data set contains 68 person with 41 368 face images as a whole. The face images were captured under varying pose, illumination, and expression. In the experiments, this subset that contains five near frontal poses (C05, C07, C09, C27, and C29) and all the images under illuminations and expressions are selected. Therefore, there are about 170 images for each person. The first 30, 40, 50, 60, 70, and 80 images per person are selected for training and the remaining for testing.

4) AR: This data set contains over 4000 color face images of 126 peoples, including frontal views of faces with different facial expressions, lighting conditions, and occlusion. In the experiments, 3120 gray images from 120 subjects are used with each subject providing 26 images. The first 5, 6, 7, and 8 images per person are selected for training and the remaining for testing.

5) Caltech 101: This data set contains 9144 images from 102 classes (i.e., 101 object classes and a "background" class), including animals, vehicles, and flowers. The samples from each category have significant shape variability. The number of images in each category varies from 31 to 800. The first 10, 15, 20, 25, and 30 images per class are selected for training and the remaining for testing.

TABLE II
Summary of Characteristics of These Seven UCI Data Sets

Data set	Dimension	Size	Class
Ionosphere	34	351	2
Breast	30	569	2
Wine	13	178	3
Semeion	256	1593	10
Splice	60	3175	3
Waveform	40	5000	3
Krvskp	36	3196	2

TABLE III
Classification Accuracies (%) of Different Algorithms on the ORL Data Set

Alg.	3 train	4 train	5 train	6 train
SVM	84.28	88.33	90.00	95.62
LR	87.50	90.00	92.00	92.50
DLSR	88.93	91.67	94.50	95.63
NNLS	88.21	92.50	93.00	95.00
LLC	87.86	93.33	94.50	94.37
KNN	83.21	87.50	89.00	90.00
LRC	82.14	85.42	89.50	95.63
SRC	87.86	91.25	92.00	93.75
MSRC	88.21	92.50	93.00	95.00
RF	83.21	85.00	89.50	96.25
Log_R	80.00	85.00	86.00	86.87
NBtree	75.71	83.33	88.50	95.00
ELM	84.64	89.17	90.50	93.75
RLR	87.86	92.50	93.00	96.25
RRLR	**90.36**	**94.58**	**95.00**	**96.88**

6) UCI Data Sets: A summary of characteristics of these seven UCI data sets is presented in Table I. For each data set, the first 5, 15, 25, and 35 samples per class are selected for training and the remaining for testing.

For LLC, we use all the training samples as the codebook. For SVM, the regularization parameter C is selected from the set $\{0.001, 0.01, 0.1, 1.0, 10, 100, 1000\}$ by using the cross validation method, which is also used to select the optimal parameters for LR, DLSR, LLC, and NNLS, respectively. For RLR and RRLR, the heart parameter σ is set to 1 and the regularization parameter λ is selected from the range from 10^{-4} to 10^{0}. The best results of all methods are selected as the final results. In all experiments, all face images are manually cropped and resized to 32×32 pixels. For Caltech 101 image data set [52], we select the spatial pyramid features of image, which is available at "http://www.umiacs.umd.edu/∼zhuolin/projectlcksvd.html", for the experiment.

B. Experimental Results and Its Analysis

Tables III–V show the classification accuracies on the ORL, GT, and CMU PIE data sets, respectively. Figs. 2 and 3 show the classification accuracies on the Caltech 101 and AR data sets, respectively.

As can be seen from Tables III–V, RLR and RRLR perform well when compared with all the other algorithms on high-dimensional data sets (face data sets and image data sets). Particularly, when using the $\ell_{2,1}$-norm based loss function, RRLR is robust to the facial expression variations. For example, in Table III, the performance of RLR is slightly inferior

TABLE IV

CLASSIFICATION ACCURACIES (%) OF DIFFERENT ALGORITHMS ON THE GT DATA SET

Alg.	4 train	5 train	6 train	7 train	8 train
SVM	56.72	61.00	69.33	72.25	73.71
LR	55.27	57.60	63.78	67.00	70.28
DLSR	59.27	61.60	68.22	71.00	73.42
NNLS	59.09	63.40	70.22	72.75	75.71
LLC	60.13	62.00	70.89	73.00	75.14
KNN	50.91	53.00	62.44	66.00	68.00
LRC	56.36	59.60	68.22	72.00	74.57
SRC	57.64	61.60	69.11	73.25	75.43
MSRC	59.09	63.20	70.89	73.25	75.71
RF	61.63	64.20	71.11	73.25	76.00
ELM	53.45	54.80	65.11	69.75	71.43
Log_R	43.45	44.20	46.67	51.25	54.00
NBtree	55.27	55.80	62.44	67.00	70.29
RLR	61.63	62.44	71.11	**73.25**	**76.00**
RRLR	**61.81**	**64.40**	**72.00**	**73.25**	75.71

TABLE V

CLASSIFICATION ACCURACIES (%) OF DIFFERENT ALGORITHMS ON THE CMU PIE DATA SET

Alg.	30 train	40 train	50 train	60 train	70 train	80 train
SVM	73.36	80.31	83.95	87.61	90.95	90.20
LR	76.46	79.83	82.93	85.08	88.15	87.68
DLSR	79.36	83.05	86.68	88.07	91.72	91.13
LRC	70.72	79.61	82.86	87.57	89.62	88.86
SRC	79.43	84.27	87.68	89.75	92.20	91.97
MSRC	79.84	84.27	88.15	89.75	92.20	91.13
RF	55.92	57.68	73.76	77.33	84.28	84.56
ELM	70.93	77.52	81.33	85.07	86.96	85.82
Log_R	56.20	61.93	67.26	70.87	77.13	77.72
NBtree	57.91	59.97	67.00	69.97	76.00	81.01
RLR	**81.50**	84.27	**88.15**	89.40	92.55	92.18
RRLR	**81.50**	**84.71**	**88.15**	**89.75**	**92.59**	**92.47**

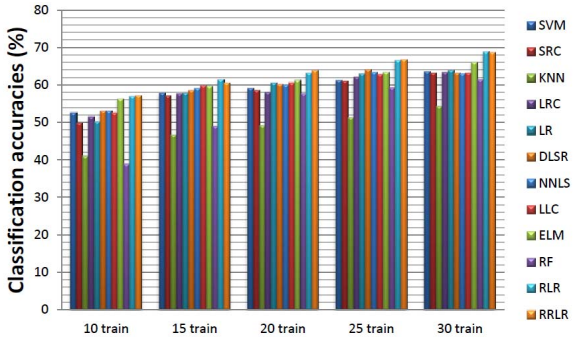


Fig. 2. Classification accuracies (%) of different algorithms on the Caltech 101 data set.

to that of DLSR when the first three and five are selected for training. However, RRLR still demonstrates its power in the two cases.

For the Caltech 101 and AR data sets, there are more complex variations in the images than the other data sets. Some classic classification algorithms may fail to classify the test samples correctly. For example, SVM, LRC, KNN, LR, and DLSR all obtain worse classification results. However, RLR and RRLR still achieve good performance on these two data sets. This is particularly evident on the AR data set.

Table VI lists the classification accuracies of different methods on seven UCI data sets. From Table VI, it can be see that RLR and RRLR do not achieve the consistently best

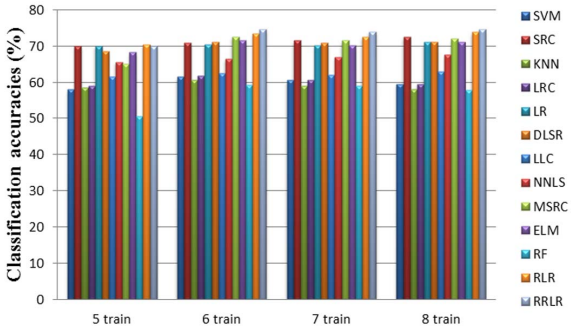


Fig. 3. Classification accuracies (%) of different algorithms on the AR data set.

TABLE VI

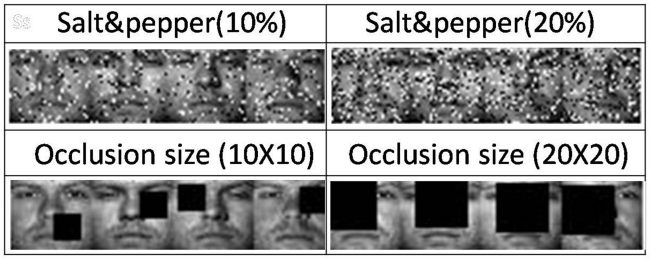
CLASSIFICATION ACCURACIES (%) OF DIFFERENT ALGORITHMS ON THE SEVEN UCI DATA SETS

Data set	KNN	LR	DLSR	SVM	RLR	RRLR
Ionosphere (5)	65.69	73.60	75.36	**76.54**	74.49	75.07
Ionosphere (15)	71.34	72.89	76.94	75.08	**77.26**	76.94
Ionosphere (25)	71.10	80.07	82.06	78.40	**82.06**	**82.06**
Ionosphere (35)	79.72	85.76	84.69	83.98	**86.12**	85.60
Breast (5)	**84.44**	82.82	80.32	83.72	82.83	81.39
Breast (15)	81.82	82.56	**83.30**	81.26	82.56	82.74
Breast (25)	82.47	84.97	86.51	84.39	86.70	**87.47**
Breast (35)	85.97	88.38	90.18	86.78	91.78	**91.98**
Wine (5)	86.50	87.73	87.73	87.73	88.96	**89.57**
Wine (15)	90.23	93.23	93.23	89.47	**93.23**	**93.23**
Wine (25)	92.23	92.23	93.20	92.23	**94.17**	**94.17**
Wine (35)	90.41	93.15	94.52	94.52	94.52	**95.89**
Semeion (5)	61.63	70.12	72.19	73.36	73.29	**73.88**
Semeion (15)	72.83	77.89	81.42	**83.29**	81.35	81.63
Semeion (25)	79.90	80.49	84.51	85.10	85.10	**85.25**
Semeion (35)	82.23	85.35	86.40	90.98	87.61	**87.77**
Splice (5)	42.25	51.04	**52.22**	51.20	51.55	52.18
Splice (15)	42.04	57.73	**60.51**	56.83	58.85	59.90
Splice (25)	43.71	62.42	64.10	59.93	61.58	**64.10**
Splice (35)	43.78	63.74	67.58	62.08	64.39	**64.62**
Waveform (5)	68.00	73.07	74.42	72.05	73.86	**74.76**
Waveform (15)	73.14	79.11	**80.16**	75.86	76.83	79.11
Waveform (25)	71.33	76.38	**76.49**	74.86	72.95	76.47
Waveform (35)	72.09	75.32	76.58	76.58	72.38	**76.58**
Krvskp (5)	63.59	75.99	76.02	75.98	76.20	**76.77**
Krvskp (15)	65.54	71.25	71.76	68.64	73.27	**73.34**
Krvskp (25)	67.55	71.55	75.17	71.03	75.77	**76.54**
Krvskp (35)	71.56	83.46	83.01	77.48	83.74	**84.32**

classification results on these data sets except on the Wine and Krvskp data sets. For example, on the Breast data set, RLR and RRLR obtain the best classification accuracies on two cases (25 and 35), but they are only the third and second best on the other two cases (5 and 15), respectively. The similar results can also be found on the other data sets.

In order to test the robustness of RLR and RRLR, experiments are also conducted on the corrupted Extended Yale B data set. The introduction of this database is shown in [52]. We randomly select 15 persons from the Extended Yale B data set to test the robustness of our method to different types of corruptions. We, respectively, simulate various levels of contiguous occlusions and random pixel corruption which are as follows.

1) *Contiguous Occlusions:* The block occlusions are randomly added to different locations in each image with the block size of 10×10 and 20×20, respectively.



(a) AR

Fig. 4. Some of the corrupted face images from the Extended Yale B data set: AR.

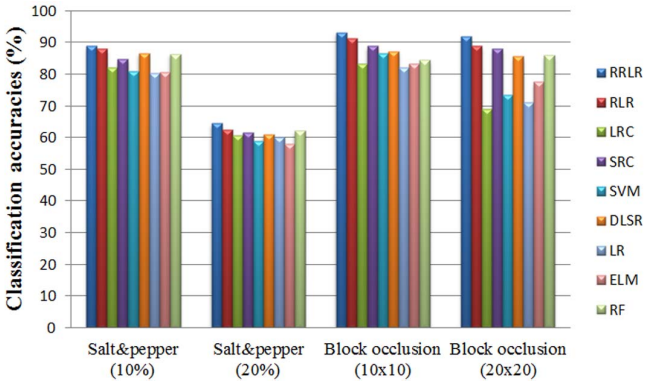


Fig. 5. Experimental results of different methods on the Extended Yale B data set.

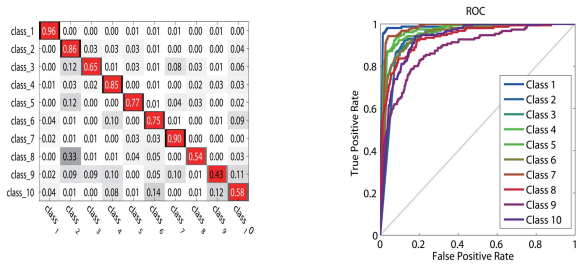
2) *Random Pixel Corruptions:* We randomly choose pixels from each image and corrupt them by salt & pepper noise. The rates of corrupted pixels are 10% and 20%, respectively.

Fig. 4 shows some of the corrupted face images from the Extended Yale B data set. We randomly select 30 samples per subject as the training set and use the remaining as the testing set. Fig. 5 shows the experimental results of different methods. It can again be seen that RLR and RRLR outperform all the other methods, including SRC, an outstanding method for the corrupted images classification [53].

Based on the experimental results on these image and UCI data sets, the following conclusions are reached.

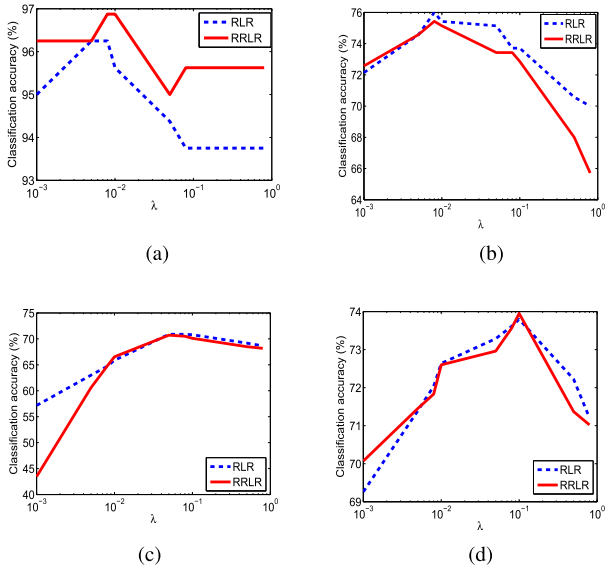
1) RLR and RRLR are significantly superior to some representation-based algorithms, such as SRC, MSRC, LCC, and LRC. The reason may be that these representation-based algorithms mainly focus on the ability of sample reconstruction. However, the best sample reconstruction does not mean the best separability [54]. By relaxing the strict binary label matrix into a slack variable matrix and using class compactness graph to avoid the problem of overfitting, RLR and RRLR can obtain the near-optimal margins. Thus, the classification results of RLR and RRLR are best.
2) When the number of training samples is very small, some classification methods (LR, SVM, DLSR, and NNLS) cannot find the true decision function. The potential reason may be that these methods are more likely to overfit with a few training samples. However, RLR and RRLR still obtain the best classification results. The main reason is that the class compactness graph plays a vital role in avoiding the problem of overfitting. For example, although DLSR obtains the large margins, DLSR does not consider the problem of overfitting. As a result, its classification performance is inferior to RLR and RRLR in most cases, especially on the GT and CMU PIE data sets. This indicates that the class compactness graph can indeed be used to address the problem of overfitting.
3) RLR and RRLR cannot obtain the best classification results on some low-dimensional data sets (see Table VI). The possible reason is that when the training samples are transformed into the label space, the dimensionality is so low that it may not support to preserve the manifold structure of the data. In other words, the samples sharing the same class may not keep close together under such low dimensionality, and thus, the problem of overfitting may be occurred in this case. As a result, the optimal margin may be not obtained on some low-dimensional data sets.
4) RRLR outperforms RLR on these corrupted face image data sets due to the use of the $\ell_{2,1}$-norm-based loss function. Particularly, when RRLR classify the faces with glasses or scarf occlusion in the AR data set, it still has advantages on classifying them correctly. However, SRC, DLSR, SVM, and LR cannot achieve satisfying classification results. We also note that LRC achieves the worst classification results in this case. We analyze the reason as follows: when the test samples are corrupted, the clean training samples may well represent some test samples that from different classes. As a result, a test sample is classified into a false class. It should be noted that on the ORL data set, RLR is slightly inferior to DLSR in two cases. However, in GT and AR data sets, two more complex data sets, RLR is superior to DLSR in the classification performance, which benefits from the use of the class compactness graph.
5) As shown in Table VI, RLR and RRLR fail to earn the consistent winner on some low-dimensional data sets. In other words, RLR and RRLR may not be capable of classifying some low-dimensional data. DLSR, SVM, RLR, and RRLR have their merits in classifying low-dimensional data. Therefore, we may select the most suitable method for a specific application. However, RLR and RRLR obtain the best classification results on high-dimensional data, and thus, they are particularly favorable for classifying high-dimensional data. Moreover, the computing of the transformation matrix A can be done offline. Thus, RLR and RRLR can be competent for practical high-dimensional data classification.

In order to better evaluate the performance of RLR, confusion matrix and receiver operating characteristic (ROC) curve is studied on the test samples of the Semeion data set. Some elements in Y' may be large which may lead to the problem that some large elements submerge some small elements when they are compared. Thus, each row elements in Y' are



(a) (b)

Fig. 6. Confusion matrix and ROC curves on the Semeion data set, in which the first 5 samples per class were selected for training and remaining for testing. (a) Confusion matrix on the Semeion data set. (b) ROC curves on the Semeion data set.



(a) (b) (c) (d)

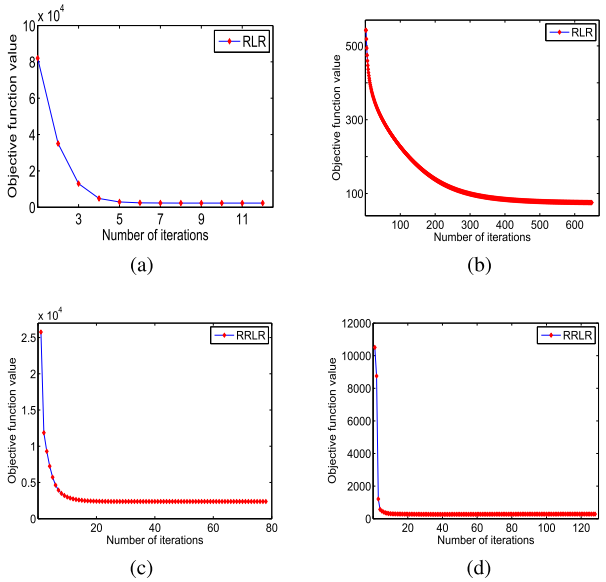
Fig. 7. Classification accuracies (%) versus the value of λ on these four data sets. (a) ORL (# 6). (b) GT (# 8). (c) Caltech 101 (# 25). (d) AR (# 8).

normalized to $[0, 1]$ for the sake of the analysis of ROC curve. The confusion matrix and the ROC curve are shown in Fig. 6. From Fig. 6, it can be seen that the performance of classifier associated with the class 1 is the best and the classifier associated with the class 9 is the worst. However, when the value of false positive rate is greater than 0.8, all classifiers achieve almost equivalent performance. Moreover, the value of area under ROC curve of the classifier associated with the class 9 (i.e., the worst classifier) is still greater than 0.80, which indicates that the proposed RLR achieves excellent classification performance on the Semeion data set.

C. Parameter Sensitiveness and Convergence Study

In order to further investigate the properties of RLR and RRLR, the classification accuracies versus the value of λ are shown in Fig. 7, in which (#) represents that for each data set the first (#) training samples per class are selected for training and the remaining for testing. This setting is also suitable for subsection E.

It can be see that RLR and RRLR are not robust to λ. This is because, if λ is very small, the class compactness graph may not effectively address the problem of overfitting; if λ is very large, it cannot achieve the optimal margins between different classes. Moreover, it is observed that when the images in the data set have strong variations, the corresponding value of λ is large, such as on the Caltech 101 and AR data sets. This also indicates that the problem of overfitting easily happens on these data sets with strong variations. The compactness graph in RLR and RRLR plays a vital role in suppressing the problem of overfitting on these data sets. How to select the optimal the value of λ is data set dependent and still an open problem. In practical, the cross validation method is used to select the optimal value of λ from the given range.



(a) (b) (c) (d)

Fig. 8. Convergence curves of RLR and RRLR on these two data sets. (a) Caltech 101 (# 25). (b) AR (# 8). (c) Caltech 101 (# 25). (d) AR (# 8).

From the convergence curves in Fig. 8, we can see that RLR and RRLR converge very fast, especially within 20 iterations for RRLR on these two data sets.

D. Comparison of Training and Test Time

In this section, we compare the running time of our methods (RLR and RRLR) with those of SRC, DLSR, NNLS, MSRC, LRC, LLC, ELM, and RF. ELM, RF, DLSR, and our methods need complete training and test phases. For example, DLSR and our methods need learn a projection in the training phase and then use a linear classifier for classification in the test phase. However, SRC, NNLS, MSRC, LRC, and LLC have no training time and only have test time, since they only need to represent input test samples as a linear combination of dictionary items, and then use the representation coefficients for classification. Therefore, we, respectively, give the training time and test time of different methods.

Table VII shows the training and test time of different methods on two representative data sets GT (the dimensionality of the feature is larger than the number of training samples) and AR (the number of training samples is larger than the dimensionality of the feature). We can see that the running speed of our methods is significantly faster than the representation-based methods in training phase, since representation-based methods require a lot of time to solve

TABLE VII

TRAINING TIME + TEST TIME ON THE GT AND AR DATA SETS (THE FIRST EIGHT IMAGES PER PERSON WERE SELECTED FOR TRAINING AND THE REMAINING FOR TESTING ON THESE TWO DATA SETS)

Alg.	GT		AR	
	Training time	Average test time	Training time	Average test time
SRC	none	1.5516	none	4.1916
DLSR	2.8891	6.5857e-4	38.5562	0.0024
NNLS	none	0.1311	none	24.2212
MSRC	none	2.2531	none	6.7449
LRC	none	0.0303	none	0.07313
LLC	none	0.0020	none	0.0034
ELM	4.9920	0.0027	20.2334	0.0027
RF	59.4848	0.0077	943.3848	0.0182
RLR	2.3756	2.0857e-4	12.9836	9.2185e-4
RRLR	18.6647	5.5256e-4	27.4672	9.4300e-4

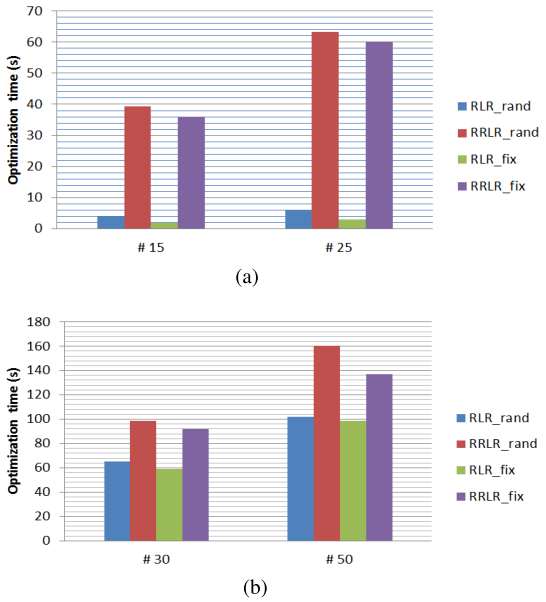


Fig. 9. Optimization time (s) on two largest data sets. (a) Caltech 101. (b) CMU PIE.

an optimization problem. In contrast, our methods only need to solve a group of linear equations in each iteration, which has linear time complexity. Therefore, RLR and RRLR are faster than these representation-based methods. In test phase, the running speed of our methods is also faster than the other methods, since our methods use a linear classifier to classify test samples. Please note that the average test time is time to classify a test sample. We also note that ELM spends less training time than RRLR. However, classification accuracies of ELM on the GT and AR data sets are significantly lower than our methods (see Table IV and Fig. 3).

E. Average Optimization Time

In this section, we evaluate the average optimization time of our methods with different settings.

In our methods, variable M needs to be set in advance. With different settings of M, we have different optimization time. To evaluate the average optimization time, for each data set, we randomly set different values for M in advance and then run our algorithms. This process is repeated 20 times, and then, the average optimization time is reported. This experiment helps evaluate how much time the convergence is guaranteed and which case the convergence is guaranteed in a reasonable amount of time. Please note that the convergence criterion used in Algorithms 1 and 2 are that the number of iterations are more than 200 and 50, respectively, or $|\Theta_{k-1} - \Theta_k|/\Theta_k \leq 0.001$, where Θ_k is the value of the objective function in the kth iteration. In this experiment, we select two largest data sets Caltech 101 and CMU PIE to evaluate the average optimization time and the results are shown in Fig. 9, in which RLR/RRLR_rand represents that M is randomly set to different values (evaluate average optimization time) and RLR/RRLR_fix represents that M is the matrix with all elements as 1. It can be seen that in the Caltech 101 data set (# 25), the average optimization time of RLR_rand and RRLR_rand is less than 5 and 65 (s), respectively. In CMU PIE data set (# 50), the average optimization time of RLR_rand and RRLR_rand is less than 100 and 160 (s), respectively. Although different data sets have different optimization time, we find that if we initialize M as a matrix with all elements as 1, the optimization time is less than that of setting M as a random value. Particularly, in CMU PIE data set (# 50), the average optimization time of RRLR_rand is 160 (s) but RRLR_fix is faster than RRLR_rand by a margin of 23 (s). Therefore, in our experimental setting, we always set M as a matrix with all elements as 1, which can guarantee that algorithms converge to a reasonable amount of time.

V. KERNEL EXTENSIONS OF RLR AND RRLR

In some real-world applications, many data are not linear, which may limit the application of RLR and RRLR. Therefore, in this section, we derive the kernelized versions of RLR and RRLR to tackle the nonlinear problems. We only give the formulations of RLR and RRLR and their solutions and experiments are not done due to space limitation.

Let ϕ: $\Re^n \rightarrow \Gamma$ be a nonlinear function which maps data from the input space $\Re^m$ to a high-dimensional feature space $\Gamma(X \rightarrow \phi(X))$. In this case, the number of dimension can be much high, or even infinite. Let $K \in \Re^{n\times n}$ be a positive semidefinite kernel Gram matrix whose elements are computed using

$$[\mathcal{K}(X, X^T)]_{ij} = [< \phi(X), \phi(X)^T >_\Gamma]_{ij} = k\left(x_i^T, x_j^T\right) \quad (26)$$

where k: $\Re^m \times \Re^m \rightarrow \Re$ is the kernel function and $\phi(X) = [\phi(x_1)^T, \phi(x_2)^T, \ldots, \phi(x_n)^T]$. Some commonly used kernel

functions include polynomial kernel $k(x, y) = (< x, y > +a)^b$, Gaussian kernel $k(x, y) = \exp(-\sigma||x - y||^2)$, and hyperbolic tangle kernel $k(x, y) = \tanh(sx^T y + o)$, where a, b, σ s, and o are the corresponding kernel parameters, respectively.

The transformation matrix A can be represented as a linear combination of all the training samples in the high-dimensional feature space $A^* = \phi(X)^T\varphi$, where $\psi = [\alpha_1^T, \alpha_2^T, \ldots, \alpha_n^T]$ is the representation coefficient matrix. With the above definitions, the nonlinear RLR can be written as follows:

$$\min_{\psi,M} ||\phi(X)\phi(X)^T\psi - (Y + B \odot M)||_F^2 + \lambda \text{tr}(\psi^T\phi(X)\phi(X)^T\Pi\phi(X)\phi(X)^T\psi). \quad (27)$$

Equation (27) can be further transformed into the following minimization problem:

$$\min_{\psi,M} ||\mathcal{K}\psi - (Y + B \odot M)||_F^2 + \lambda \text{tr}(\psi^T\mathcal{K}\Pi\mathcal{K}\psi) \quad (28)$$

where $\Pi = \Xi - \Sigma$. Σ is the weight matrix and defined as $\Sigma_{ij} = e^{-((||\phi(x_i)^T - \phi(x_j)^T||^2)/\sigma)}$. Ξ is the diagonal matrix $\Xi = \sum_j \Sigma_{ij}$.

With the same definitions, the nonlinear RRLR can be written as

$$\min_{\psi,M} ||\mathcal{K}\psi - (Y + B \odot M)||_{2,1} + \lambda \text{tr}(\psi^T\mathcal{K}\Pi\mathcal{K}\psi). \quad (29)$$

Once we obtain the representation coefficient matrix ψ, we can calculate the transformation result of test samples by $\phi(X_{\text{te}})\phi(X)^T\psi$, where $\phi(X_{\text{te}})$ is the test samples matrix.

VI. Conclusion

By relaxing the strict binary label matrix into a slack variable matrix, the proposed methods have more freedom to fit the labels and can enlarge the margins between different classes as much as possible. The class compactness graph is used to address the problem of overfitting. Two optimization algorithms are derived, which all solve a sequence of convex optimization problems, and hence, they are both tractable and scalable. We also extend RLR and RRLR into their kernel versions. Experiments indicate that these two algorithms outperform the state-of-the-art algorithms in terms of classification accuracy and running time. The limitation of two algorithms is that they may not work well on some low-dimensional data (as shown in Table VI). In the future, we plan to improve them so that they can classify low-dimensional data well.

References

[1] T. Strutz, *Data Fitting and Uncertainty: A Practical Introduction to Weighted Least squares and Beyond*. Wiesbaden, Germany: Vieweg, 2010.

[2] S. Wold, H. Ruhe, H. Wold, and W. Dunn, "The collinearity problem in linear regression. the partial least squares (PLS) approach to generalized inverses," *J. Sci. Stat. Comput.*, vol. 5, no. 3, pp. 735–743, Jan. 1984.

[3] Y. Li and A. Ngom, "Nonnegative least-squares methods for the classification of high-dimensional biological data," *IEEE/ACM Trans. Comput. Biol. Bioinf.*, vol. 10, no. 2, pp. 447–456, Mar. 2013.

[4] S. M. Xiang, F. P. Nie, G. F. Meng, C. H. Pan, and C. S. Zhang, "Discriminative least squares regressions for multiclass classification and feature selection," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 23, no. 11, pp. 1738–1754, Nov. 2012.

[5] F. P. Nie, H. Wang, H. Huang, and C. Ding, "Adaptive loss minimization for semi-supervised elastic embedding," in *Proc. 23rd Int. Joint Conf. Artif. Intell.*, Beijing, China, 2013, pp. 1565–1571.

[6] Y. Pang, Z. Ji, P. Jing, and X. Li, "Ranking graph embedding for learning to rerank," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 24, no. 8, pp. 1292–1303, Aug. 2013.

[7] M. Belkin, P. Niyogi, and V. Sindhwani, "Manifold regularization: A geometric framework for learning from labled and unlabeled examples," *J. Mach. Learn. Res.*, vol. 12, pp. 2399–2434, May 2006.

[8] K. Q. Weinberger, J. Blitzer, and L. Saul, "Distance metric learning for large margin nearest neighbor classification," *J. Mach. Learn. Res.*, vol. 10, pp. 207–244, Jul. 2009.

[9] L. Shao, L. Liu, and X. Li, "Feature learning for image classification via multiobjective genetic programming," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 7, pp. 1359–1371, Jul. 2014.

[10] Z. Fan, Y. Xu, and D. Zhang, "Local linear discriminant analysis framework using sample neighbors," *IEEE Trans. Neural Netw.*, vol. 22, no. 7, pp. 1119–1132, Jul. 2011.

[11] J. Leski, "Ho-Kashyap classifier with generalization control," *Pattern Recognit. Lett.*, vol. 24, no. 14, pp. 2281–2290, Oct. 2003.

[12] X. Li, S. Lin, S. Yan, and D. Xu, "Discriminant locally linear embedding with high-order tensor data," *IEEE Trans. Syst. Man, Cybern. B, Cybern.*, vol. 38, no. 2, pp. 342–352, Apr. 2008.

[13] X. He, S. Yan, Y. Hu, P. Niyogi, and H.-J. Zhang, "Face recognition using Laplacianfaces," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 3, pp. 328–340, Mar. 2005.

[14] X. He, D. Cai, S. Yan, and H.-J. Zhang, "Neighborhood preserving embedding," in *Proc. 10th IEEE Int. Conf. Comput. Vis.*, Beijing, China, Oct. 2005, pp. 1208–1213.

[15] D. Cai, X. He, and J. Han, "Isometric projection," in *Proc. Nat. Conf. Artif. Intell.*, Cambridge, MA, USA, Jul. 2007, pp. 528–533.

[16] S. Yan, D. Xu, B. Zhang, H.-J. Zhang, Q. Yang, and S. Lin, "Graph embedding and extensions: A general framework for dimensionality reduction," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 1, pp. 40–51, Jan. 2007.

[17] M. Sugiyama, "Dimensionality reduction of multimodal labeled data by local Fisher discriminant analysis," *J. Mach. Learn. Res.*, vol. 8, pp. 1027–1061, May 2007.

[18] D. Cai, X. He, K. Zhou, J. Han, and H. Bao, "Locality sensitive discriminant analysis," in *Proc. 20th Int. Joint Conf. Artif. Intell.*, Jan. 2007, pp. 708–713.

[19] H.-T. Chen, H.-W. Chang, and T.-L. Liu, "Local discriminant embedding and its variants," in *Proc. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2005, pp. 846–853.

[20] C. X. Ren, D. Q. Dai, and H. Yan, "Robust classification using $\ell_{2,1}$-norm based regression model," *Pattern Recognit.*, vol. 45, no. 7, pp. 2708–2718, Jul. 2012.

[21] C. Ding, D. Zhou, X. He, and H. Zha, "R1-PCA: Rotational invariant ℓ_1-norm principal component analysis for robust subspace factorization," in *Proc. 23rd Int. Conf. Mach. Learn.*, New York, NY, USA, Jun. 2006, pp. 281–288.

[22] J. Wright, Y. G. Peng, and Y. Ma, "Robust principal component analysis: Exact recovery of corrupted low-rank matrices by convex optimization," *Adv. Neural Inf. Process. Syst.*, Vancouver, BC, Canada, Dec. 2009, pp. 2080–2088.

[23] J. Yang, J. Wright, T. S. Huang, and Y. Ma, "Image super-resolution via sparse representation," *IEEE Trans. Image Process.*, vol. 19, no. 11, pp. 2861–2873, Nov. 2010.

[24] Z. Zhang, Y. Xu, J. Yang, X. Li, and D. Zhang, "A survey of sparse representation: Algorithms and applications," *IEEE Access*, vol. 3, no. 1, pp. 490–530, May 2015.

[25] X.-T. Yuan, X. Liu, and S. Yan, "Visual classification with multitask joint sparse representation," *IEEE Trans. Image Process.*, vol. 21, no. 10, pp. 4349–4360, Oct. 2012.

[26] C. P. Hou, F. P. Nie, D. Y. Yi, and Y. Wu, "Feature selection via joint embedding learning and sparse regression," in *Proc. 22th Int. Joint Conf. Artif. Intell.*, Barcelona, Spain, Jul. 2011, pp. 1324–1329.

[27] Q. Q. Gu, Z. H. Li, and J. W. Han, "Joint feature selection and subspace learning," in *Proc. 2nd Int. Joint Conf. Artif. Intell.*, Barcelona, Spain, Jul. 2011, pp. 1294–1299.

[28] Y. Yang, H. T. Shen, Z. G. Ma, and X. F. Zhou, "$\ell_{2,1}$-norm Regularized discriminative feature selection for unsupervised learning," in *Proc. 22nd Int. Joint Conf. Artif. Intell.*, Barcelona, Spain, Jul. 2011, pp. 1–6.

[29] Z. C. Li, Y. Yang, J. Liu, X. F. Zhou, and H. Q. Lu, "Unsupervised feature selection using nonnegative spectral analysis," in *Proc. 26th AAAI Conf. Artif. Intell.*, Toronto, ON, Canada, Jul. 2012, 1–7.

[30] J. Liu, S. Ji, and J. Ye, "Multi-task feature learning via efficient $\ell_{2,1}$-norm minimization," in *Proc. 21th Conf. Uncertainty Artif. Intell.*, Montreal, QC, Canada, Jun. 2009, pp. 339–348.

[31] T. Goldstein and S. Osher, "The split Bregman method for L1-regularized problems," *SIAM J. Imag. Sci.*, vol. 2, no. 2, pp. 323–343, 2009.

[32] V. Estellers, D. Zosso, R. Lai, S. Osher, J. P. Thiran, and X. Bresson, "Efficient algorithm for level set method preserving distance function," *IEEE Trans. Image Process.*, vol. 21, no. 12, pp. 4722–4734, Dec. 2012.

[33] D. Luo, C. Ding, and H. Huang, "Towards structural sparsity: An explicit ℓ_2/ℓ_0 approach," *Knowl. Inf. Syst.*, vol. 36, no. 2, pp. 411–438, Aug. 2010.

[34] D. P. Bertsekas, A. Nedi, and A. E. Ozdaglar, *Convex Analysis and Optimization*. Belmont, MA, USA: Athena Scientific, 2003.

[35] D. P. Bertsekas, *Nonlinear Programming*. Belmont, MA, USA: Athena Scientific, 1999.

[36] G. Liu, Z. Lin, S. Yan, J. Sun, Y. Yu, and Y. Ma, "Robust recovery of subspace structures by low-rank representation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 1, pp. 171–184, Jan. 2013.

[37] A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma, "Fast l1-minimization algorithms and an application in robust face recognition," EECS Dept. Univ. California, Berkeley, Berkeley, CA, USA, Tech. Rep. UCB/EECS-2010-13, Feb. 2010.

[38] Y. Tang, Y.-Q. Zhang, and Z. Huang, "Development of two-stage SVM-RFE gene selection strategy for microarray expression data analysis," *IEEE/ACM Trans. Comput. Biol. Bioinfor.*, vol. 4, no. 3, pp. 365–381, Jul. 2007.

[39] J. J. Wang, J. C. Yang, K. Yu, F. J. Lv, and T. Huang, "Locality-constrained linear coding for image classification," in *IEEE Conf. Comput. Vis. Pattern Recognit.*, San Francisco, USA, Sep. 2010, pp. 3360–3367.

[40] I. Naseem, R. Togneri, and M. Bennamoun, "Linear regression for face recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 11, pp. 2106–2112, Nov. 2010.

[41] C. Zheng, L. Zhang, T. Ng, and C. Shiu, "Metasample based sparse representation for tumor classification," *IEEE/ACM Trans. Comput. Biol. Bioinf.*, vol. 8, no. 5, pp. 1273–1282, Oct. 2011.

[42] M. Fernandez-Delgado, E. Cernadas, and S. Barro, "Do we need hundreds of classifiers to solve real world classification problem?" *J. Mach. Learn. Res.*, vol. 15, pp. 3133–3181, Oct. 2014.

[43] J. Cao and Z. Lin, "Extreme learning machines on high dimensional and large data applications: A survey," *Math. Problems Eng.*, vol. 2015, Mar. 2015, doi: 10.1155/2015/103796.

[44] J. Cao *et al.*, "Landmark recognition with sparse representation classification and extreme learning machine," *J. Franklin Inst.*, vol. 352, no. 10, pp. 4528–4545, 2015.

[45] Y. Xu *et al.*, "Data uncertainty in face recognition," *IEEE Trans. Cybern.*, vol. 44, no. 10, pp. 1950–1961, Oct. 2014.

[46] Y. Xu, X. L. Li, J. Yang, Z. H. Lai, and D. Zhang, "Integrating conventional and inverse representation for face recognition," *IEEE Trans. Cybern.*, vol. 44, no. 10, pp. 1738–1746, Oct. 2014.

[47] X.Z. Fang, Y. Xu, X. L. Li, Z. H. Lai, and W. K. Wong, "Learning a nonnegative sparse graph for linear regression," *IEEE Trans. Image Process.*, vol. 24, no. 9, pp. 2760–2771, Sep. 2015.

[48] T. Sim, S. Baker, and M. Bsat, "The CMU pose, illumination, and expression database," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 12, pp. 1615–1618, Dec. 2003.

[49] A. M. Martinez and R. Benavente, "The AR face database," Dept. CVC, Univ. Autonoma Barcelona, Barcelona, Spain, Tech. Rep. #24, Jun. 1998.

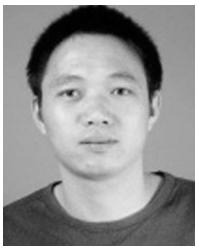
[50] Z. H. Lai, W. K. Wong, Y. Xu, J. Yang, and D. Zhang, "Approximate orthogonal sparse embedding for dimensionality reduction," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 4, pp. 723–735, Apr. 2016.

[51] M. Lichman, "UCI machine learning repository," School Inf. Comput. Sci., Univ. California, Irvine, CA, USA, Tech. Rep., 2013. [Online]. Available: http://archive.ics.uci.edu/ml

[52] Z. L. Jiang, Z. Lin, and L. S. Davis, "Label consistent K-SVD: Learning a discriminative dictionary for recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 11, pp. 2651–2664, Nov. 2013.

[53] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma, "Robust face recognition via sparse representation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 2, pp. 210–227, Feb. 2009.

[54] L. Zhang *et al.*, "Kernel sparse representation-based classifier," *IEEE Trans. Signal Process.*, vol. 60, no. 4, pp. 1684–1695, Apr. 2012.



Xiaozhao Fang (S'15) received the M.S. degree in computer science from the Guangdong University of Technology, Guangzhou, China, in 2008. He is currently pursuing the Ph.D. degree in computer science and technology with the Shenzhen Graduate School, Harbin Institute of Technology, Shenzhen, China.

He has authored over 15 journal papers. His current research interests include pattern recognition and machine learning.

Yong Xu (M'06–SM'15) was born in Sichuan, China, in 1972. He received the B.S. and M.S. degrees in 1994 and 1997, respectively, and the Ph.D. degree in pattern recognition and intelligence system with the Nanjing University of Science and Technology, Nanjing, China, in 2005.

He is currently with the Shenzhen Graduate School, Harbin Institute of Technology, Shenzhen, China. His current interests include pattern recognition, biometrics, machine learning, and video analysis.

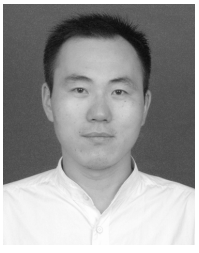
Xuelong Li (M'02–SM'07–F'12) is currently a Full Professor with the Center for OPTical IMagery Analysis and Learning, State Key Laboratory of Transient Optics and Photonics, Xi'an Institute of Optics and Precision Mechanics, Chinese Academy of Sciences, Xi'an, China.

Zhihui Lai received the B.S. degree in mathematics from South China Normal University, Guangdong, China, the M.S. degree from Jinan University, Guangdong, and the Ph.D. degree in pattern recognition and intelligence system from the Nanjing University of Science and Technology, Nanjing, China, in 2002, 2007, and 2011, respectively.

He was a Research Associate and a Post-Doctoral Fellow with The Hong Kong Polytechnic University, Hong Kong, from 2010 to 2013. He is currently a Post-Doctoral Fellow with the Bio-Computing Research Center, Shenzhen Graduate School, Harbin Institute of Technology, Harbin, China. He has authored over 30 scientific papers in pattern recognition and computer vision. His current research interests include face recognition, image processing and content-based image retrieval, pattern recognition, and compressive sense.

Wai Keung Wong received the Ph.D. degree from The Hong Kong Polytechnic University, Hong Kong.

He is currently with the Institute of Textiles & Clothing, The Hong Kong Polytechnic University, Hong Kong, and the Shenzhen Research Institute, The Hong Kong Polytechnic University. He has authored over fifty scientific articles in refereed journals, including IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, *Pattern Recognition*, *International Journal of Production Economics*, *European Journal of Operational Research*, *International Journal of Production Research*, *Computers in Industry*, and the IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS, among others. His current research interests include artificial intelligence, pattern recognition, and optimization of manufacturing scheduling, planning and control.



Bingwu Fang received the B.E. and M.S. degrees from the University of Science and Technology of China, Hefei, China, in 2001 and 2005, respectively. He is currently pursuing the Ph.D. degree with the School of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, China.

His current research interests include software engineering and machine learning.