Applied Intelligence
https://doi.org/10.1007/s10489-018-01407-y

Non-convex approximation based l_0-norm multiple indefinite kernel feature selection

Hui Xue[1,2] · **Yu Song**[1,2]

Abstract

Multiple kernel learning (MKL) for feature selection utilizes kernels to explore complex properties of features, which has been shown to be among the most effective for feature selection. To perform feature selection, a natural way is to use the l_0-norm to get sparse solutions. However, the optimization problem involving l_0-norm is NP-hard. Therefore, previous MKL methods typically utilize a l_1-norm to get sparse kernel combinations. However, the l_1-norm, as a convex approximation of l_0-norm, sometimes cannot attain the desired solution of the l_0-norm regularizer problem and may lead to prediction accuracy loss. In contrast, various non-convex approximations of l_0-norm have been proposed and perform better in many linear feature selection methods. In this paper, we propose a novel l_0-norm based MKL method (l_0-MKL) for feature selection with non-convex approximations constraint on kernel combination coefficients to select features automatically. Considering the better empirical performance of indefinite kernels than positive kernels, our l_0-MKL is built on the primal form of multiple indefinite kernel learning for feature selection. The non-convex optimization problem of l_0-MKL is further refumated as a difference of convex functions (DC) programming and solved by DC algorithm (DCA). Experiments on real-world datasets demonstrate that l_0-MKL is superior to some related state-of-the-art methods in both feature selection and classification performance.

Keywords l_0-norm · Feature selection · Multiple kernel learning · DC programming

1 Introduction

Feature selection is one of fundamental problems in machine learning. The goals of feature selection are to remove the irrelevant and redundant features, reduce store space and execution time, and avoid the course of dimensionality while preserving or improving the prediction performance [13]. In general, feature selection methods can be divided into three categories: "*filter*" which ranks the features according to some discrimination measures independent of learning algorithms, "*wrapper*" which evaluates the features by learning algorithms, and "*embedded*" which embeds feature selection into learning process. Generally, the wrapper approach is considered to produce better feature subsets but run much more slowly than a filter for a specific learning algorithm. Embedded methods do not separate the learning from the feature selection part and tend to be between filters and wrappers in terms of computational complexity.

This paper focuses on an embedded method for feature selection. For the feature selection purpose, a natural way is to use zero norm (denoted l_0 or $|\cdot|_0$) to deal with sparsity. The zero norm of a vector is defined as the number of its non-zero components. However, the optimization problem involving l_0-norm is NP-hard and hence is not practical for large scale problems. During the last two decades, works in feature selection with the l_0-norm can be divided into three categories according to the way to treat the zero norm: *convex approximation*, *non-convex approximation* and *direct solutions*.

"*Convex approximation*" creates a smoothed convex approximation such as l_1-norm regularization and has been studied extensively. Least absolute shrinkage and selection

✉ Hui Xue
hxue@seu.edu.cn

Yu Song
song_yu@seu.edu.cn

[1] School of Computer Science and Engineering, Southeast University, Nanjing, 210096, China

[2] MOE Key Laboratory of Computer Network and Information Integration, Southeast University, Nanjing, China

Published online: 16 July 2019

operator (LASSO) method penalized the regression coefficients in linear regression with the l_1 penalty, shrinking many of them to zero [21]. l_1-SVM was then presented for feature selection in the context of SVM [1]. Elastic net regularization, which is a combination of l_1-norm and l_2-norm, can form a more structured regularization and obtain better prediction accuracy [26]. To perform nonlinear feature selection, the feature selection problem of gene expression data was transformed into a multiple parameter learning problem based on multiple kernel support vector machine [3]. Varma and Babu proposed a more generalized MKL scheme for feature selection where the combination of base kernels can be generalized to be nonlinear [22]. Instead of using positive definite kernels, Xue et al. utilized indefinite kernels to select features based on the primal framework of indefinite kernel support vector machine [25]. Among these l_1-norm based methods, MKL techniques have been shown to be amongst the most effective for nonlinear feature selection [22, 25]. Gribonval and Nielsen have proven that, under suitable assumptions, a solution of the l_0-norm regularizer problem over a polyhedral set can be obtained by solving the l_1-norm regularizer problem [8]. However, these assumptions may not be satisfied in many cases. As a result, l_1-norm may fail to live up to the desired feature selection property, leading to prediction accuracy loss by shrinking both relevant and irrelevant features to zero.

Instead of using l_1-norm, "*non-convex approximation*" attempts to approximate l_0-norm by a non-convex continuous function. Various non-convex regularizations have been developed in several works in different contexts, most of them are for feature selection in SVM or regression. For example, the concave exponential approximation [1, 12, 14, 16, 19], the smoothly clipped absolute deviation [6, 13], the log penalty method [2], and the capped l_1-norm [17, 18]. However, these non-convex regularizations for feature selection are mainly based on linear methods, such as linear SVM. Unlike MKL methods, they may not be able to uncover complicated properties of the features and can be greatly limited.

In the third category of "*direct solutions*", l_0-norm is directly solved via auxiliary functions. For example, the intractable l_2/l_0 optimization was solved by simple iterative algorithms with a Lipschitz auxiliary function [15]. And the l_0-norm regularized problem was reformulated as a continuous nonconvex program with an exact penalty technique [11]. However, the additional parameters introduced by auxiliary functions may negatively affect the generalization abililty and increase the time complexity of the algorithms.

In this paper, we propose a novel non-convex approximation based l_0-norm MKL method for feature selection. Considering the superiority of indefinite kernels, our l_0-MKL is built on the primal form of multiple indefinite kernel learning for feature selection. Concretely, l_0-MKL uses an indefinite base kernel to represent each feature respectively and a l_0-norm constraint is then enforced on kernel combination coefficients in order to select features automatically. The challenging problem of l_0-norm is firstly approximated by the optimization of non-convex approximations and then reformulated as a difference of convex functions (DC) programming. An iteratively two step algorithm is further proposed to solve the non-convex optimization problem. Experimental results on real-world datasets have shown that l_0-MKL outperforms some related methods in terms of feature selection and classification.

The remainder of this paper is organized as follows. Section 2 briefly reviews the related works on sparse regularization based feature selection. In Section 3, we present the proposed l_0-MKL method. Section 4 describes the optimization algorithm for solving our model. Section 5 provides the experimental results. Section 6 concludes this paper.

2 Related work

In this section we will briefly describe the l_1-norm MKL methods for feature selection and some non-convex approximation based methods in linear feature selection. Given a training set $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$, where $\boldsymbol{x}_i \in \mathbb{R}^M$ is a training sample and $y_i \in \{-1, +1\}$ is the corresponding class label.

2.1 l_1-norm based MKL

MKL methods firstly apply a base kernel k_m on each feature of the samples and then combines these kernels into a kernel combination [4]:

$$k\left(\boldsymbol{x_i}, \boldsymbol{x_j}\right) = \sum_{m=1}^{M} d_m k_m \left(x_{i,m}, x_{j,m}\right), d_m \geq 0 \tag{1}$$

where $x_{i,m}$ denotes the mth feature of $\boldsymbol{x}_i$ and d_m represents the coefficient of the kernel k_m.

MKL methods aim to learn a sparse combination of the kernels so that the feature can be selected naturally and l_1-norm regularizer is used on $\boldsymbol{d}$ to obtain a sparse solution [3, 4, 22].

$$\begin{aligned} \min_{\boldsymbol{\alpha}, \boldsymbol{d}} \quad & \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j k(\boldsymbol{x_i}, \boldsymbol{x_j}) - \sum_{i=1}^{n} \alpha_i + \lambda \|\boldsymbol{d}\|_1 \\ s.t \quad & d_m \geq 0, m = 1, \cdots, M \\ & 0 \leq \alpha_i \leq C \\ & \sum_{i=1}^{n} y_i \alpha_i = 0, i = 1, \cdots, n \end{aligned} \tag{2}$$

where $\boldsymbol{\alpha}$ is the Lagrange multiplier, $C > 0$ is the penalty parameter of SVM and $\lambda > 0$ is the regularization parameter controlling sparsity of $\boldsymbol{d}$.

2.2 Non-convex approximation based methods

The l_0-norm is defined as:

$$\|\boldsymbol{w}\|_0 = \sum_{i=1}^{n} sign(w_i)$$
$$\text{with} \quad sign(w_i) = \begin{cases} 0 & w_i = 0 \\ 1 & w_i \neq 0 \end{cases}$$

Linear l_0-SVM can be formulated as:

$$\begin{aligned} \min_{\boldsymbol{w},b,\boldsymbol{\xi}} \ & \|\boldsymbol{w}\|_0 + C\sum_{i=1}^{n}\xi_i \\ s.t. \ & y_i(\boldsymbol{w}^T\boldsymbol{x_i} + b) \geq 1 - \xi_i \\ & \xi_i \geq 0, i = 1, \cdots, n \end{aligned} \tag{3}$$

Equation 3 is the standard SVM formulation but with the l_2-norm regularization on $\boldsymbol{w}$ replaced by l_0-norm regularization to enforce sparsity on $\boldsymbol{w}$. The components of $\boldsymbol{w}$ corresponds to the features of $\boldsymbol{x}_i$ and can be used to select features. Concretely, a feature can be discarded when the corresponding component of $\boldsymbol{w}$ equals zero. In order to make (3) tractable, the l_0-norm regularization is typically approximated by various non-convex approximations [12]. And Table 1 lists some of the effective non-convex approximations of l_0-norm.

Where for $t \in \mathbb{R}$, $\delta_\theta(t)$ is the step function and $\theta > 0$ is a parameter controlling the tightness of approximation.

As a typical embedded method for feature selection, l_0-SVM can eliminate features based on linear SVM classifier. However, l_0-SVM simply selects features according to the component value of $\boldsymbol{w}$ and ignores complex properties of features. Though Neumann et al. use the concave exponential approximation of l_0-norm to perform feature selection in nonlinear SVM classifiers, the nonlinearities are limited to explicit quadratic feature map or gaussian kernel [16]. When using other feature maps, their methods are no longer applicable. In our method, the feature maps correspond to indefinite kernels and there is a much larger class of kernel functions available.

Table 1 l_0-norm approximation functions δ

Approximation	Function δ
Exp [1, 12, 14, 16, 19]	$\delta_\theta(t) = 1 - e^{-\theta\|t\|}$
Log [2]	$\delta_\theta(t) = \frac{log(1+\theta\|t\|)}{log(1+\theta)}$
Capped-l_1 [10, 18]	$\delta_\theta(t) = \min\{1, \theta\|t\|\}$
SCAD [6]	$\delta_\theta(t) = \begin{cases} \frac{2\theta}{a+1}\|t\| & \|t\| \leq \frac{1}{\theta} \\ \frac{-\theta^2t^2+2a\theta\|t\|-1}{a^2-1} & \frac{1}{\theta} < \|t\| < \frac{a}{\theta} \\ 1 & \|t\| \geq \frac{a}{\theta} \end{cases}$

3 l_0-MKL

The nonlinear SVM maps the training samples from the input space into a higher-dimensional feature space via a mapping function φ and the nonlinear l_0-SVM can be formulated as:

$$\begin{aligned} \min_{\boldsymbol{w},b,\boldsymbol{\xi}} \ & \|\boldsymbol{w}\|_0 + C\sum_{i=1}^{n}\xi_i \\ s.t. \ & y_i(\boldsymbol{w}^T\varphi(\boldsymbol{x_i}) + b) \geq 1 - \xi_i \\ & \xi_i \geq 0, i = 1, \cdots, n \end{aligned} \tag{4}$$

However, due to the non-convexity of l_0-norm, (4) and its dual form are not identical. Equation (2) is not suitable for feature selection when l_0-norm is used. Therefore, we build our model based on the primal form of kernel SVM [9]. In addition, Xue et al. have shown that indefinite kernels can explore complex properties of feature and perform better than positive definite kernels for feature selection [25]. As a result, we use an indefinite base kernel to represent each feature respectively and a l_0-norm constraint to get sparse kernel combination coefficients:

$$\begin{aligned} \min_{\boldsymbol{\beta},b,\boldsymbol{d}} \ & \lambda_1\boldsymbol{\beta}^T\boldsymbol{K}\boldsymbol{\beta} + \lambda_2\|\boldsymbol{d}\|_0 + \sum_{i=1}^{n}\max\left(0, 1 - y_i(\boldsymbol{K}^i\boldsymbol{\beta} + b)\right)^2 \\ s.t \ & d_m \geq 0, m = 1, \cdots, M \end{aligned} \tag{5}$$

In (5), it is worth noting that $\boldsymbol{\beta}$ is unconstrained which is different to the Lagrange multiplier $\boldsymbol{\alpha}$ in (2). The second term is the l_0-norm regularizer related to $\boldsymbol{d}$. If the coefficient d_m equals to zero, it means that the corresponding feature has no effect on the classification and can be discarded. The last term is the smooth quadratic hinge loss function and $\boldsymbol{K}^i$ denotes the ith row of $\boldsymbol{K}$.

Nevertheless, as indicated above, the minimization of l_0-norm is a NP-hard problem. As a result, we use continuous sparse approximations of l_0-norm in Table 1 to approximate (5):

$$\begin{aligned} \min_{\boldsymbol{\beta},b,\boldsymbol{d}} \ & \lambda_1\boldsymbol{\beta}^T\boldsymbol{K}\boldsymbol{\beta} + \lambda_2\sum_{i=1}^{M}\delta_\theta(d_i) + \sum_{i=1}^{n}\max\left(0, 1 - y_i(\boldsymbol{K}^i\boldsymbol{\beta} + b)\right)^2 \\ s.t \ & d_m \geq 0, m = 1, \cdots, M \end{aligned} \tag{6}$$

Compared to l_1-norm based MKL methods, the proposed l_0-MKL has two advantages. Firstly, the non-convexity of l_0-norm can be approximated by various non-convex approximations, which have better sparsity and achieve higher classification accuracies than l_1-norm. Secondly, we construct l_0-MKL on the the primal form of indefinite kernel SVM and avoid the dual gap in the non-convex optimization problem effectively.

Compared to non-convex approximation based linear methods, the proposed l_0-MKL can utilize kernels to

explore complex properties of features and perform nonlinear feature selection effectively.

4 Optimization algorithm

Firstly, we denote the objective function of l_0-MKL as

$$F(\boldsymbol{\beta}, b, \boldsymbol{d}) = \lambda_1 \boldsymbol{\beta}^T \boldsymbol{K} \boldsymbol{\beta} + \lambda_2 \sum_{i=1}^{M} \delta_\theta(d_i) + \sum_{i=1}^{n} \max\left(0, 1 - y_i(\boldsymbol{K}^i \boldsymbol{\beta} + b)\right)^2 \quad (7)$$

We address (7) by developing an iteratively two step optimization problem. Concretely, the SVM parameters $(\boldsymbol{\beta}, b)$ are learnt when the kernel combination is held fixed (Step 4) and then kernel combination is learnt by optimizing over $\boldsymbol{d}$ while the SVM parameters $(\boldsymbol{\beta}, b)$ are held fixed (Step 5). This process is repeated until converges and the complete algorithm for solving (7) is described in Algorithm 1.

Algorithm 1 l_0-MKL.

Inputs:
T: the maximum number of iterations
ϵ: the tolerance value for convergence
Process:
1: set $t = 0$, initialize $\boldsymbol{d}_t$, obj_t;
2: **while** $(t < T)$ **do**
3: set $t = t + 1$;
4: fix $\boldsymbol{d}_{t-1}$ and solve (6) to obtain $(\boldsymbol{\beta}_t, b_t)$;
5: fix $(\boldsymbol{\beta}_t, b_t)$ and solve (6) to obtain a solution $\boldsymbol{d}_t$;
6: calculate the value of objective function obj_t;
7: **if** $|\text{obj}_t - \text{obj}_{t-1}| \leq \epsilon$ **then**
8: algorithm converges and break;
9: **end if**
10: **end while**
11: **return** $\boldsymbol{\beta}_t, b_t, \boldsymbol{d}_t$;

4.1 DC programming and DCA

DC programming plays an important role in non-convex programming and DCA is commonly used to solve the smooth/non-smooth non-convex problems [5, 20, 23]. They address general DC programs of the form:

$$\inf\{f(\boldsymbol{x}) := g(\boldsymbol{x}) - h(\boldsymbol{x}) : \boldsymbol{x} \in R^n\} \quad (8)$$

where the functions g and h are lower semicontinuous proper convex functions defined on R^n and

$$\inf f(\boldsymbol{x}) \geq -\infty \quad (9)$$

DCA is based on local optimality conditions and duality in DC programming. For simplicity, we omit the dual part of DC programming and focus on how DCA is conducted. The main original idea of DCA is simple, it consists in approximating a DC program by a sequence of convex programs: at each iteration, DCA approximates the concave part $-h$ at the current point by its affine majorization and minimizes the resulting convex function to find a new point [11]. The algorithm proceeds as Algorithm 2.

Algorithm 2 DCA [5].

Inputs:
T: the maximum number of iterations
ϵ: the tolerance value for convergence
Process:
1: set $t = 0$, initialize $\boldsymbol{x}_t$;
2: **while** $(t < T)$ **do**
3: calculate $\boldsymbol{y}_t \in \partial h(\boldsymbol{x}_t)$;
4: solve the strongly convex optimization problem
argmin $g(\boldsymbol{x}) - [h(\boldsymbol{x}_t) + < \boldsymbol{x} - \boldsymbol{x}_t, \boldsymbol{y}_t >]$
to obtain $\boldsymbol{x}_{t+1}$
5: **if** $|\boldsymbol{x}_{t+1} - \boldsymbol{x}_t| \leq \epsilon$ **then**
6: DCA converges and break;
7: **end if**
8: set $t = t + 1$;
9: **end while**
10: **return** $\boldsymbol{x}_t$;

Where $\partial h(\boldsymbol{x}_t)$ is the gradient of h at point $\boldsymbol{x}_t$. Algorithm 2 calculates the gradient of the concave part $-h$ (Step 3) and approximates it by its affine majorization to convert the non-convex function to convex function (Step 4). The optimization problem in Step 4 of Algorithm 4 is a convex program and can be solved easily.

4.2 Indefinite kernel SVM solved by DCA (IKSVM-DCA)

When the coefficients $\boldsymbol{d}$ are fixed, (7) degenerates into a non-convex problem of SVM with a single indefinite kernel and its objective function is:

$$f(\boldsymbol{\beta}, b) = \lambda_1 \boldsymbol{\beta}^T \boldsymbol{K} \boldsymbol{\beta} + \sum_{i=1}^{n} \max\left(0, 1 - y_i(\boldsymbol{K}^i \boldsymbol{\beta} + b)\right)^2 \quad (10)$$

According to [24], the non-convex problem of (10) can be reformulated as a DC programming equivalently due to the favorable property of the spectra for indefinite kernel matrices. Concretely, the objective function can be decomposed as

$$\begin{aligned} f(\boldsymbol{\beta}, b) &= g(\boldsymbol{\beta}, b) - h(\boldsymbol{\beta}, b) \\ \text{with} \quad g(\boldsymbol{\beta}, b) &= \lambda_1 \boldsymbol{\beta}^T (\rho \boldsymbol{I}) \boldsymbol{\beta} + \sum_{i=1}^{n} \max\left(0, 1 - y_i(\boldsymbol{K}^i \boldsymbol{\beta} + b)\right)^2 \\ h(\boldsymbol{\beta}, b) &= \lambda_1 \boldsymbol{\beta}^T (\rho \boldsymbol{I} - \boldsymbol{K}) \boldsymbol{\beta} \end{aligned} \quad (11)$$

where the positive number ρ satisfies the condition: $\rho \geq \eta$ and the number η is the maximum eigenvalue of the kernel

matrix $\boldsymbol{K}$. As a result, the functions $g(\boldsymbol{\beta}, b)$ and $h(\boldsymbol{\beta}, b)$ can both be guaranteed to be convex and (10) can be solved by DCA. The detailed steps for solving $(\boldsymbol{\beta}, b)$ in (10) are described in Algorithm 3.

Algorithm 3 IKSVM-DCA [24].

Inputs:

T: the maximum number of iterations

ϵ: the tolerance value for convergence

Process:

1: calculate η, the maximum eigenvalue of $\boldsymbol{K}$;
2: set $\rho = \eta + 10^{-5}$;
3: set $k = 0$, initialize $\boldsymbol{\beta}^k$;
4: **while** $(k < T)$ **do**
5: calculate $\overline{\boldsymbol{\beta}}^k \in \partial h(\boldsymbol{\beta}^k, b^k) = \lambda_1(\rho \boldsymbol{I} - \boldsymbol{K})\boldsymbol{\beta}^k$;
6: calculate $(\boldsymbol{\beta}^{k+1}, b^{k+1}) \in \arg\min\{g(\boldsymbol{\beta}, b) - [h(\boldsymbol{\beta}^k, b^k) + < \boldsymbol{\beta} - \boldsymbol{\beta}^k, \overline{\boldsymbol{\beta}}^k >]\}$;
7: set $\boldsymbol{\alpha}^{k+1} = (\boldsymbol{\beta}^{k+1}, b^{k+1})$ and $\boldsymbol{\alpha}^k = (\boldsymbol{\beta}^k, b^k)$;
8: **if** $\|\boldsymbol{\alpha}^{k+1} - \boldsymbol{\alpha}^k\|_2^2 \le \epsilon$ **then**
9: the algorithm converges and break the loop;
10: **end if**
11: set $k = k + 1$;
12: **end while**
13: **return** $\boldsymbol{\beta}^k, b^k$;

Algorithm 3 firstly performs eigenvalue decomposition on $\boldsymbol{K}$ to find the maximum eigenvalue for DC decomposition (Step 1–2). Within the loop, Algorithm 3 calculates the gradient of the convex function h (Step 5) and then solves nonconvex problem by approximating $h(\boldsymbol{\beta}, b)$ with its affine minorization (Step 6).

4.3 Non-convex approximation solved by DCA (NC-DCA)

When $(\boldsymbol{\beta}, b)$ are fixed, (7) can be reformulated as:

$$f(\boldsymbol{d}) = \sum_{i=1}^{n} d_i \gamma_i + \lambda_2 \sum_{i=1}^{M} \delta_\theta(d_i) + \sum_{i=1}^{n} \max\left(0, 1 - y_i\left(\boldsymbol{\theta}^i \boldsymbol{d} + b\right)\right)^2 \tag{12}$$

where $\boldsymbol{\gamma} = [\lambda_1 \boldsymbol{\beta}^T K_1 \boldsymbol{\beta}, \ldots, \lambda_1 \boldsymbol{\beta}^T K_M \boldsymbol{\beta}]^T$, $\boldsymbol{\theta} = [K_1 \boldsymbol{\beta}, \cdots, K_M \boldsymbol{\beta}]$ and $\boldsymbol{\theta}^i$ represents the ith row of $\boldsymbol{\theta}$.

According to [10], δ_θ is a DC function which can be decomposed as

$$\delta_\theta(d_i) = \varphi_\theta(d_i) - \phi_\theta(d_i) \quad s.t. \quad d_i \ge 0, i = 1, ..., M \tag{13}$$

where $\varphi_\theta, \phi_\theta$ are convex functions. Therefore, (12) can be further decomposed as:

$$\begin{aligned} f(\boldsymbol{d}) &= g(\boldsymbol{d}) - h(\boldsymbol{d}) \\ \text{with} \quad g(\boldsymbol{d}) &= \sum_{i=1}^{n} d_i \gamma_i + \sum_{i=1}^{n} \max\left(0, 1 - y_i\left(\boldsymbol{\theta}^i \boldsymbol{d} + b\right)\right)^2 \\ &\quad + \lambda_2 \sum_{i=1}^{M} \varphi_\theta(d_i) \\ h(\boldsymbol{d}) &= \lambda_2 \sum_{i=1}^{M} \phi_\theta(d_i) \end{aligned} \tag{14}$$

where the functions $g(\boldsymbol{d})$ and $h(\boldsymbol{d})$ are both convex.

Table 2 lists the first decomposition of $\delta_\theta(d_i)$ and the gradients of the second decomposition.

The optimization problem of (12) is a DC programming and can be solved by DCA. The detailed steps described in Algorithm 4.

Algorithm 4 NC-DCA.

Inputs:

T: the maximum number of iterations

ϵ: the tolerance value for convergence

Process:

1: set $k = 0$, initialize $\boldsymbol{d}^k$;
2: **while** $(k < T)$ **do**
3: calculate $\overline{\boldsymbol{d}}^k \in \partial h(\boldsymbol{d}^k)$;
4: calculate $\boldsymbol{d}^{k+1} \in argmin\ \{g(\boldsymbol{d}) - [h(\boldsymbol{d}^k) + < \boldsymbol{d} - \boldsymbol{d}^k, \overline{\boldsymbol{d}}^k >] : \boldsymbol{d} \in \mathbb{R}_+^M\}(P_k)$;
5: **if** $\|\boldsymbol{d}^{k+1} - \boldsymbol{d}^k\|_2^2 \le \epsilon$ **then**
6: the algorithm converges and break the loop;
7: **end if**
8: set $k = k + 1$;
9: **end while**
10: **return** $\boldsymbol{d}^k$;

When $(\boldsymbol{\beta}, b)$ are fixed, Algorithm 4 calculates the gradient $\partial h(\boldsymbol{d}^k)$ of the convex function h according to (14) and Table 2 (Step 3). The optimization problem over the kernel combination coefficients $\boldsymbol{d}$ in Step 4 is a convex program and can be solved easily.

4.4 Convergence and complexity analysis

In this section, we will present a theoretical analysis for the convergence of l_0-MKL.

Proposition 1 *For the sequence* $\{\boldsymbol{d}^k\}$, *we have*

$$(g - h)(\boldsymbol{d}^k) - (g - h)(\boldsymbol{d}^{k+1}) \ge \tau \|\boldsymbol{d}^k - \boldsymbol{d}^{k+1}\|^2,$$

Table 2 The decomposition of functions $\delta = \varphi - \phi$ and $\partial\phi(d_i)$

Approximation	Function φ	$\partial\phi(d_i)$
Exp [1, 12, 14, 16, 19]	$\theta\|d_i\|$	$\text{sign}(d_i)\theta(1-e^{-\theta\|d_i\|})$
Log [2]	$\frac{\theta\|d_i\|}{log(1+\theta)}$	$\text{sign}(d_i)\frac{\theta^2\|d_i\|}{log(1+\theta)(1+\theta\|d_i\|)}$
Capped-l_1 [10, 18]	$\theta\|d_i\|$	$\begin{cases} 0 & d_i \le \frac{1}{\theta} \\ \text{sign}(d_i)\theta & \text{otherwise} \end{cases}$
SCAD [6]	$\frac{2\theta}{a+1}\|d_i\|$	$\begin{cases} 0 & \|d_i\| \le \frac{1}{\theta} \\ \text{sign}(d_i)\frac{2\theta(\theta\|d_i\|-1)}{a^2-1} & \frac{1}{\theta} < \|d_i\| < \frac{a}{\theta} \\ \text{sign}(d_i)\frac{2\theta}{a+1} & \|d_i\| \ge \frac{a}{\theta} \end{cases}$

the equality holds if and only if $\tau\|\boldsymbol{d}^k - \boldsymbol{d}^{k+1}\|^2 = 0$*, where* τ *is a positive parameter to make functions g and h strongly convex.*

Proof This is consequence of DCA's convergence theorem for a general DC program [20]. □

Proposition 2 *For the sequence* $\{(\boldsymbol{\beta}^k, b^k, \boldsymbol{d}^k)\}$ *, we have* $F(\boldsymbol{\beta}^{k+1}, b^{k+1}, \boldsymbol{d}^{k+1}) \le F(\boldsymbol{\beta}^k, b^k, \boldsymbol{d}^k)$,

that is, the objective function $F(\boldsymbol{\beta}^k, b^k, \boldsymbol{d}^k)$ *is strictly monotonic decreasing along the solution sequence.*

Proof From Proposition 1, the objective function F is decreasing when SVM parameters are held fixed. When the coefficients $\boldsymbol{d}$ are fixed, l_0-MKL degenerates into an indefinite kernel SVM and the objective function F is also decreasing according to [24]. Thus, the whole algorithm l_0-MKL is decreasing along the solution sequence. □

When $F(\boldsymbol{\beta}^k, b^k, \boldsymbol{d}^k) = F(\boldsymbol{\beta}^{k+1}, b^{k+1}, \boldsymbol{d}^{k+1})$ comes true, the algorithm l_0-MKL can converge to a stationary point.

As stated in Algorithm 1, l_0-MKL is solved iteratively. When SVM coefficients $(\boldsymbol{\beta}_t, b_t)$ are calculated in each iteration (Step 4), it takes $O(n^2 + T_1 * n^2)$, where n denotes the number of samples and T_1 denotes the number of iterations in Algorithm 3. Similarly, when calculating kernel combination coefficients $\boldsymbol{d}_t$ (Step 5), it takes $O(T_2 * m^2)$, where m denotes the number of features and T_2 denotes the number of iterations in Algorithm 4. In summary, the total complexity is $O(T*(T_1*n^2+T_2*m^2))$, where T is supposed to be the maximum number of iterations of Algorithm 1. Since $\max(T, T_1, T_2) \ll m$ and n, l_0-MKL algorithm's time complexity is $O(m^2 + n^2)$.

Table 3 Datasets description

Datasets	#Num	#Feature
ALLAML	72	7129
Colon	62	2000
Gli_85	85	22283
Prostate_GE	102	5966
Central_Nervous_System	60	7129
Lung_Cancer	181	12533
Leukemia	72	7070
Dbworld	64	4702
Isolet	120	617
Glioma	21	4434
Carcinom	34	9182

5 Experiments

We conduct a series of experiments on several real-world datasets to compare our l_0-MKL to some related state-of-the-art algorithms.

5.1 Experimental setup

We select eleven datasets from three different repositories for experiments: (a) eight datasets from a feature selection repository,[1] namely ALLAML, Colon, Gli_85, Prostate_GE, Leukemia, Isolet, Glioma, Carcinom; (b) two binary datasets from an online repository[2] of high-dimensional biomedical datasets, namely Central_Nervous_System, Lung_Cancer; (c) one dataset Dbworld from UCI Machine Learning Repository. Table 3 lists a brief description of these datasets, including the number of samples and the number of features in each sample.

We randomly divide the samples into two non-overlapping training and testing sets which contain almost half of the samples in each class. The processes are repeated ten times to generate ten independent runs for each dataset and then the average results are reported. Since the three datasets Isolet, Glioma and Carcinom are designed for multi-class classification , we choose the first two classes.

The optimization problems in line 6 of Algorithm 3 and line 4 of Algorithm 4 are convex programs. Both of them

[1] http://featureselection.asu.edu/datasets.php

[2] http://datam.i2r.a-star.edu.sg/datasets/krbd/

Table 4 Classification accuracies and the number of selected features (mean (#dimension)) of each compared algorithm on real-world datasets

	PIE-SVM	CAP-SVM	EP-SVM	MIK-FS	l_0-MKL
ALLAML	93.71 (4)	92.86 (6)	95.71 (93)	97.14 (18)	**98.00 (7)**
Colon	84.84 (6)	85.16 (7)	87.10 (61)	87.74 (215.7)	**87.74(14)**
Gli_85	77.61 (6)	77.61 (6)	81.43 (86)	78.09 (14)	**82.14 (10)**
Prostate_GE	95.88 (7)	95.68 (11)	95.49 (12)	95.88 (8)	**96.28 (8)**
Central_Nervous_System	72.07 (12)	72.07 (12)	70.34 (10)	75.52(17)	**79.31 (20)**
Lung_Cancer	98.67 (12)	98.56 (11)	99.44 (91)	**99.89 (46)**	99.78 (22)
Leukemia	96.00 (11)	96.29 (11)	96.86 (54)	**97.71 (1426)**	**97.71 (7)**
Dbworld	84.52 (14)	85.49 (13)	91.29 (22)	90.00 (11)	**90.97 (9)**
Isolet	98.67 (3)	99.00 (5)	99.00 (10)	**100.00 (9)**	**100.00 (9)**
Glioma	83.00 (2)	82.00 (2)	84.00 (8)	90.00 (3)	**96.00 (4)**
Carcinom	79.41 (2)	80.00 (3)	81.76 (22)	91.70 (11)	**91.77 (7)**

can be solved by CVX [7]. And we conduct our experiments on a Windows 7 machine with 8 GB memory and 3.00 GHz CPU.

We compare the proposed l_0-MKL with the following algorithms:

- PIE-SVM [12]: Feature selection in SVM with the concave exponential approximation regularizer.
- CAP-SVM [10]: Feature selection in SVM with the capped l_1-norm regularizer.
- EP-SVM [11]: An exact penalty approach for feature selection via the l_0-norm regularization problem.
- MIK-FS [25]: l_1-norm based multiple indefinite kernel learning for feature selection.

We use the capped l_1-norm as the approximation of l_0-norm in our experiments. The indefinite sigmoid kernel $k = \tanh(a \cdot \boldsymbol{x_i}^T \boldsymbol{x_j} - r)$ is selected as the base feature kernel for MIK-FS and l_0-MKL. For l_0-MKL and MIK-FS, the hyperparameters and kernel parameters are chosen from the set $\{10^{-2}, 10^{-1}, 1, 10^1, 10^2\}$. For EP-SVM, CAP-SVM and PIE-SVM, the hyperparameters are chosen form the set $\{2^{-6}, \cdots, 2^6\}$ and the parameter θ is chosen from the set the same as l_0-MKL.

In training process, all the features are used for training. Irrelevant features are discarded when the obtained kernel combination coefficients are less than 10^{-5}. In testing process, only selected features are used for testing. In addition, we can get the degree of importance of selected features according to the kernel combination coefficients and use specified number of features for testing.

5.2 Experimental results

Table 4 lists the average classification accuracies and the corresponding number of selected features in each compared algorithm on real-world datasets. The best results are highlighted in bold. In terms of sparsity of solution, the quality of non-convex approximations are comparable. All the three algorithms (PIE-SVM, CAP-SVM and l_0-MKL) reduce considerably the number of selected features. And they outperform l_1-norm (MIK-FS) and auxiliary functions based reformulation (EP-SVM), especially for Leukemia

Table 5 F1 score of each compared algorithm on real-world datasets

	PIE-SVM	CAP-SVM	EP-SVM	MIK-FS	l_0-MKL
ALLAML	95.12	95.18	97.03	97.16	**97.87**
Colon	79.29	79.79	81.58	80.82	**82.93**
Gli_85	36.76	36.76	58.42	64.80	**69.12**
Prostate_GE	95.39	95.43	94.82	94.72	**96.02**
Central_Nervous_System	38.13	38.13	19.61	56.47	**70.95**
Lung_Cancer	96.54	96.54	**98.85**	**98.85**	**98.85**
Leukemia	93.63	94.77	94.65	95.64	**95.77**
Dbworld	82.98	84.16	89.37	90.38	**92.01**
Isolet	98.85	98.85	98.85	**100.00**	**100.00**
Glioma	93.33	90.79	90.45	95.55	**98.00**
Carcinom	88.66	88.66	88.66	**90.91**	90.17

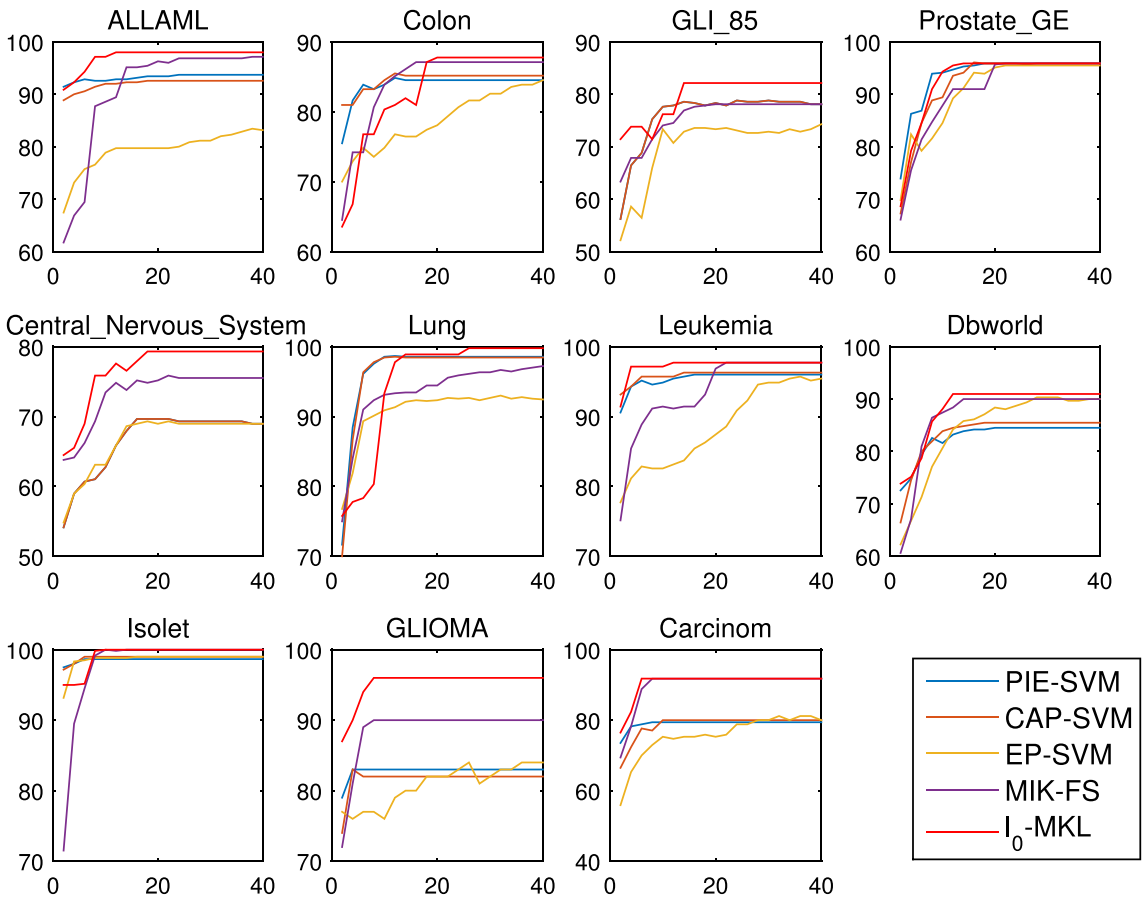

Fig. 1 The above plots show classification accuracy (y-axis) versus number of selected features (x-axis) for our real-world datasets

and Colon. In terms of accuracies, our algorithm is the best. Two nonlinear feature selection methods (MIK-FS and l_0-MKL) obviously outperform the other three linear SVM methods, especially for Glioma and Carcinom. Overall, linear methods with non-convex approximations can select features effectively but tend to achieve low classification accuracies. l_1-norm MIK-FS can achieve high classification accuracies but tends to select too many features. As a result, considering sparsity of features and classification accuracies, our method is superior to these state-of-the-art methods.

Table 5 lists the average F1 score in each compared algorithm on real-world datasets. The best results are highlighted in bold. In terms of F1 score, the proposed method outperforms others on most datasets except for Carcinom. On Gli_85 and Central_Nervous_System, the proposed method can achieve much higher scores than other methods.

Figure 1 shows the classification accuracies corresponding to the specified number of features on real-world datasets. The maximum number of selected feature is set to 40. As shown in Fig. 1, our method is the strongest performer in the large majority of cases, sometimes by a substantial margin as in the case of GLOMA. While our method is occasionally outperformed in the beginning when the number of selected features is small, it either ties or overtakes the leading method by the end in all datasets.

Figure 2 shows the difference value of objective function F in l_0-MKL during each iteration in Algorithm 1. Obviously, l_0-MKL converges rapidly within 10 iterations on all the datasets, in spite of some fluctuations as in ALLAML and Isolet.

In Table 6, the CPU time of each algorithm is listed on every dataset and the last row is the average CPU time. From Table 6, we can see that l_0-MKL takes almost twice the average time than CAP-SVM. This is because the proposed algorithm is an iteratively two-step optimization problem. But the outer loop converges quickly and it still runs faster than EP-SVM.

The best parameters of the proposed method are listed in Table 7, where a, r are the kernel parameters of indefinite kernels and λ_1, λ_2 are the regularization parameters. θ is a

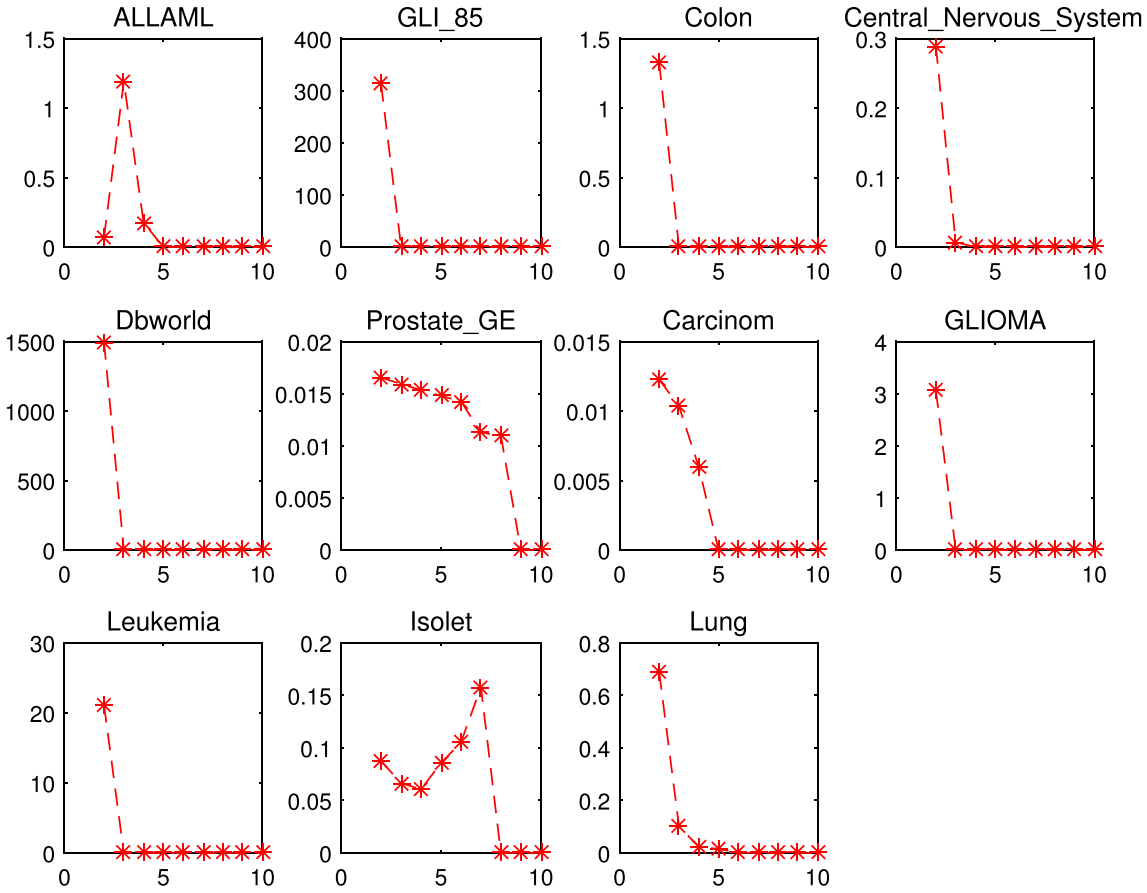

Fig. 2 The above plots show the difference value of objective function (y-axis) versus iteration number (x-axis) for our real-world datasets

parameter controlling the tightness of approximation of l_0-norm and affects the performance in our experiments, so we select it from the set $\{10^{-2}, 10^{-1}, 1, 10^1, 10^2\}$ to improve performance. In fact, the authors set $\theta = 5$ in [1, 16], so we have also conducted some experiments when setting $\theta = 5$. The corresponding results are inferior to the best results in the previous parameter set. Therefore, $\theta = 5$ is not the best parameter for our model.

Table 6 CPU time of each compared algorithm on real-world datasets

CPU time (s)	PIE-SVM	CAP-SVM	EP-SVM	MIK-FS	l_0-MKL
ALLAML	26.06	**6.71**	47.69	16.48	8.79
Colon	2.29	**1.83**	8.34	9.50	4.56
Gli_85	**16.83**	18.40	28.59	29.12	49.41
Prostate_GE	27.53	8.81	59.53	**7.26**	11.17
Central_Nervous_System	3.05	3.62	22.31	**2.78**	10.59
Lung_Cancer	33.28	35.06	211.08	**31.87**	84.85
Leukemia	9.78	**4.49**	65.73	6.18	9.60
Dbworld	2.18	**1.54**	8.24	2.38	6.82
Isolet	2.64	**1.61**	5.83	1.68	8.58
Glioma	3.71	2.27	7.02	**1.72**	2.68
Carcinom	5.71	3.70	24.93	**2.83**	14.38
Average	12.10	**8.00**	44.48	10.16	19.22

Table 7 Best parameters of l_0-MKL on real-world datasets

Best Parameters	a	r	λ_1	λ_2	θ
ALLAML	10	0.01	10	0.1	1
Colon	0.01	1	0.01	10	1
Gli_85	0.01	10	0.1	0.1	0.01
Prostate_GE	0.1	0.01	0.1	0.1	1
Central_Nervous_System	0.01	1	0.1	0.1	1
Lung_Cancer	0.01	1	0.01	10	0.01
Leukemia	0.01	0.1	0.01	0.01	0.01
Dbworld	0.1	0.1	1	1	0.1
Isolet	0.1	1	10	1	0.1
Glioma	1	10	10	1	1
Carcinom	1	0.01	1	10	1

6 Conclusion

We propose a novel non-convex approximation based l_0-norm MKL method for nonlinear feature selection. The proposed method utilizes the advantages of non-convex approximations of l_0-norm to get sparsity in features as well as the advantages of indefinite kernels to capture complex properties of features in order to improve classification performance. An iteratively two step algorithm is further proposed to solve the non-convex optimization problem. Concretely, the SVM parameters are learned by the DCA method when the kernel combination is held fixed and then kernel combination is learned by the DCA method while the SVM parameters are held. Experimental results on real-world datasets have shown that l_0-MKL outperforms some related methods in terms of sparsity of features and classification accuracies.

There are two issues for future work.

- **Choice of indefinite kernel**: We present an overall framework combining the advantages of non-convex approximation and indefinite kernel and design an efficient algorithm to solve the model. But the choice of a good base indefinite kernel is still an open problem and needs to be selected manually. As a result, how to automatically select an appropriate kernel needs more research.
- **Optimization technique**: In the paper, we apply a two-stage algorithm to solve the non-convex optimizations in the proposed models, which optimizes the coefficients of IKSVM and kernel combination by DCA method alternately. However, DCA is time-consuming when both the number of sample and features are large. Therefore, how to accelerate l_0-MKL by refining the solving algorithms needs more systematic research.

Acknowledgments This work was supported by the National Key R&D Program of China (Grant No. 2017YFB1002801) and National Natural Science Foundation of China (Grant Nos. 61375057, 61876091). Furthermore, the work was also supported by Collaborative Innovation Center of Wireless Communications Technology.

References

1. Bradley PS, Mangasarian OL (1998) Feature selection via concave minimization and support vector machines. In: Proceedings of fifteenth international conference on machine learning, vol 98, pp 82–90
2. Candes EJ, Wakin MB, Boyd S (2008) Enhancing sparsity by reweighted L1 minimization. J Fourier Anal Appl 14(5–6):877–905
3. Chen Z, Li J, Wei L (2007) A multiple kernel support vector machine scheme for feature selection and rule extraction from gene expression data of cancer tissue. Artif Intell Med 41(2): 161–175
4. Dileep AD, Sekhar CC (2009) Representation and feature selection using multiple kernel learning. In: Proceedings of the twenty-second international joint conference on neural networks. IEEE, pp 717–722
5. Dinh TP, Le Thi HA (2014) Recent advances in dc programming and dca. In: Transactions on computational intelligence XIII. Springer, pp 1–37
6. Fan J, Li R (2001) Variable selection via nonconcave penalized likelihood and its oracle properties. J Am Stat Assoc 96(456):1348–1360
7. Grant M, Boyd S, Ye Y (2008) Cvx: Matlab software for disciplined convex programming
8. Gribonval R, Nielsen M (2003) Sparse representations in unions of bases. IEEE Trans Inf Theory 49(12):3320–3325
9. Hao Z, Yuan G, Yang X, Chen Z (2013) A primal method for multiple kernel learning. Neural Comput Appl 23(3-4):975–987
10. Le Thi HA, Dinh TP, Le HM, Vo XT (2015) Dc approximation approaches for sparse optimization. Eur J Oper Res 244(1): 26–46
11. Le Thi HA, Le HM, Dinh TP (2015) Feature selection in machine learning: an exact penalty approach using a difference of convex function algorithm. Mach Learn 101(1–3):163–186
12. Le Thi HA, Le HM, Dinh TP et al (2008) A dc programming approach for feature selection in support vector machines learning. Adv Data Anal Classif 2(3):259–278
13. Le Thi HA, Ouchani S et al (2008) Gene selection for cancer classification using dca. In: International conference on advanced data mining and applications. Springer, pp 62–72
14. López J, Maldonado S, Carrasco M (2018) Double regularization methods for robust feature selection and svm classification via dc programming. Inf Sci 429:377–389

15. Luo D, Ding C, Huang H (2010) Towards structural sparsity: an explicit l2/l0 approach. In: 2010 IEEE international conference on data mining, pp 344–353
16. Neumann J, Schnörr C, Steidl G (2005) Combined svm-based feature selection and classification. Mach Learn 61(1–3):129–150
17. Ong CS, An LTH (2013) Learning sparse classifiers with difference of convex functions algorithms. Optim Methods Softw 28(4):830–854
18. Peleg D, Meir R (2008) A bilinear formulation for vector sparsity optimization. Signal Process 88(2):375–389
19. Rinaldi F, Schoen F, Sciandrone M (2010) Concave programming for minimizing the zero-norm over polyhedral sets. Comput Optim Appl 46(3):467–486
20. Tao PD, An LTH (1997) Convex analysis approach to dc programming: theory, algorithms and applications. Acta Math Vietnam 22(1):289–355
21. Tibshirani R (1996) Regression shrinkage and selection via the lasso. J R Stat Soc: Ser B Methodol 58(1):267–288
22. Varma M, Babu BR (2009) More generality in efficient multiple kernel learning. In: Proceedings of the twenty-sixth annual international conference on machine learning. ACM, pp 1065–1072
23. Wang F, Cao W, Xu Z (2018) Convergence of multi-block bregman admm for nonconvex composite problems. Sci China Inf Sci 61(12):122101
24. Xu HM, Xue H, Chen X, Wang Y (2017) Solving indefinite kernel support vector machine with difference of convex functions programming. In: AAAI, pp 2782–2788
25. Xue H, Song Y, Xu HM (2017) Multiple indefinite kernel learning for feature selection. In: Proceedings of the 26th international joint conference on artificial intelligence. AAAI Press, pp 3210–3216
26. Zou H, Hastie T (2005) Regularization and variable selection via the elastic net. J R Stat Soc 67(2):301–320

Hui Xue received the B.Sc. degree in Mathematics from Nanjing Normal University in 2002. In 2005, she received the M.Sc. degree in Mathematics from Nanjing University of Aeronautics & Astronautics (NUAA). And she also received the Ph.D. degree in Computer Application Technology at NUAA in 2008. Since 2009, as an Associate Professor, she has been with the School of Computer Science and Engineering at Southeast University. Her research interests include pattern recognition and machine learning.

Yu Song received the B.S. degree in School of Software Engineering from Southeast University, China, in 2016. He is currently a postgraduate student in the School of Computer Science and Engineering at Southeast University, China. His research interests include pattern recognition, machine learning, and data mining.