Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2016, Article ID 7347986, 7 pages
http://dx.doi.org/10.1155/2016/7347986

Research Article

Feature Scaling via Second-Order Cone Programming

Zhizheng Liang

School of Computer Science and Technology, China University of Mining and Technology, Xuzhou 221116, China

Correspondence should be addressed to Zhizheng Liang; liang@cumt.edu.cn

Received 20 January 2016; Accepted 3 April 2016

Academic Editor: Julien Bruchon

Feature scaling has attracted considerable attention during the past several decades because of its important role in feature selection. In this paper, a novel algorithm for learning scaling factors of features is proposed. It first assigns a nonnegative scaling factor to each feature of data and then adopts a generalized performance measure to learn the optimal scaling factors. It is of interest to note that the proposed model can be transformed into a convex optimization problem: second-order cone programming (SOCP). Thus the scaling factors of features in our method are globally optimal in some sense. Several experiments on simulated data, UCI data sets, and the gene data set are conducted to demonstrate that the proposed method is more effective than previous methods.

1. Introduction

Selecting relevant and important features has been an active research area in statistics and data mining [1–6]. In some real-world applications, one is often confronted with the high-dimensional data such as text data and gene data. One important characteristic of these data sets is that some features of data may contain irrelevant or redundant information. It is shown that a large number of irrelevant or redundant features will degrade the performance of classifiers. In order to improve the comprehensibility of classification modes and their classification performance, it is interesting to explore how to reduce or remove irrelevant features of data.

Owing to their better generalization performance of support vector machines, they have become an effective tool to select relevant features in the past several years. In [1], SVMs are used as a subroutine in the feature selection process and the SVMs accuracy is optimized on the resulting subset of features. In [7], the gradient descent method is used to obtain the weights of features in terms of the SVM criterion. Moreover, the relevance of features can also be measured by their scaling factors. Thus feature scaling is performed to measure the importance of features. In [8], scaling factors of features are tuned by minimizing the standard SVM empirical risk. Further, some estimates of the generalization error of SVMs [9] are used to automatically tune scaling factors of features by using the gradient descent algorithm. In [10], the smooth leave-one-out error is optimized to obtain the scaling factors of features, while an iterative feature scaling method for linear SVM is proposed in [11]. In addition, some methods depend on other criteria to perform feature selection. For example, Maji [12] proposed a rough hypercuboid approach in approximation spaces to select relevant features of data. Liu et al. [13] combined global and local structures of data to perform feature selection. Li et al. [14] developed a stable feature selection algorithm. Li and Tang [15] combined nonnegative spectral analysis and redundancy control to select relevant features in the unsupervised case. Wang et al. [16] minimized global redundancy of data to obtain the optimal features. In order to improve the discriminant power, Tao et al. devised an effective discriminative feature selection method in [17]. Although these algorithms continue to contribute to the development of feature selection, some of these methods are often dependent on the gradient descent method, which may lead to their sensitivity to the choice of the gradient step and often falling into local minima.

In this paper, we propose a novel method for feature scaling in terms of the SVM criteria which avoids the scaling factors of features falling into locally optimal solutions. The proposed method first introduces the scaling factors of features in linear support vector machines and then uses a kind of generalized performance measure to learn the scaling factors of features. To make the generalized performance measure suitable for feature scaling, the measure is modified and formulated as a second-order cone programming problem. Finally, the scaling factors of features are obtained by solving a convex optimization problem. In addition, we also carry out experiments on some data sets to evaluate the proposed method.

2. Linear Support Vector Machines (LSVMs)

In this section, we briefly recall the basic idea of LSVMs. This class of algorithms introduced by in [11] has been shown to perform well in real applications.

Given the training data $\{(x_i, y_i)\}_1^l$ with input data $x_i \in R^n$ and the corresponding binary class label $y_i \in \{-1, 1\}$, the SVM is to find an optimal hyperplane that separates two classes such that the hyperplane is the greatest distance from the closed training vectors of each class. Often, the hyperplane is obtained by solving the following optimization problem:

$$\begin{aligned} \min \quad & \frac{1}{2}\langle w, w\rangle \\ & y_i\left(\langle w, x_i\rangle + b\right) \geq 1, \quad i = 1, \ldots, l, \end{aligned} \tag{1}$$

where $(w, b) \in R^n \times R$ and $\langle \cdot, \cdot \rangle$ is the inner product of two vectors. When data cannot be perfectly separated, a penalty term $C\sum_{i=1}^{l} \xi_i$ is added to the objective function in (1), where C is a positive number. Accordingly, the following optimization problem is constructed:

$$\begin{aligned} \min \quad & \frac{1}{2}\langle w, w\rangle + C\sum_{i=1}^{l} \xi_i \\ & y_i\left(\langle w, x_i\rangle + b\right) \geq 1 - \xi_i, \quad \xi_i \geq 0, \ i = 1, \ldots, l. \end{aligned} \tag{2}$$

It is found that (2) can be solved in the dual space of Lagrange multipliers $\alpha_i \geq 0$, $i = 1, \ldots, l$. In such a case, (2) can be formulated as the following optimization problem:

$$\begin{aligned} \max \quad & W(\alpha) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l} y_i y_j \alpha_i \alpha_j \left\langle x_i, x_j \right\rangle \\ & \sum_{i=1}^{l} y_i \alpha_i = 0, \ \xi_i \geq 0, \ C \geq \alpha_i \geq 0, \ i = 1, \ldots, l. \end{aligned} \tag{3}$$

After α_i and b are obtained, the following decision function is used to classify samples:

$$f(x) = \operatorname{sgn}\left(\sum_{i=1}^{l} \alpha_i y_i \langle x_i, x\rangle + b\right). \tag{4}$$

Note that although there are l training samples in (4), only the samples with $\alpha_i > 0$ play a role in the decision function. The samples with $\alpha_i > 0$ are called support vectors.

3. Feature Scaling Using Second-Order Cone Programming (SOCP)

In this section, we propose a feature scaling method based on the generalized performance measure. First, it would be helpful to introduce the generalized performance measure.

3.1. Generalized Performance Measure. Note that the generalized performance measure [18] is optimized not only over the function space but also over a convex cone of positive semidefinite matrices. In the case of the SVM criteria, the following generalized performance measure is used to choose proper kernel parameters:

$$\begin{aligned} \min_{K} \max_{\alpha} \quad & 2\alpha^T e - \alpha^T \operatorname{diag}(Y) K \operatorname{diag}(Y)\alpha \\ & \alpha^T y = 0, \ \operatorname{trace}(K) = c, \ C \geq \alpha_j \geq 0, \\ & j = 1, \ldots, l, \end{aligned} \tag{5}$$

where K is a linear combination of different kernel matrices such as Gaussian kernels and polynomial kernels, $\operatorname{diag}(Y) = \operatorname{diag}(y_1, y_2, \ldots, y_l)$, $y = (y_1, y_2, \ldots, y_l)^T$, and $e = (1 \cdots 1)^T$.

Instead of adopting multiple kernels, we consider the linear kernel in this paper. It is clear that the Gram matrix K in linear support vector machines can be written as a linear combination of dimensions of features. In other words, the following equation is constructed:

$$\begin{aligned} K &= \begin{bmatrix} \langle x_1, x_1\rangle & \cdots & \langle x_1, x_l\rangle \\ \vdots & \vdots & \vdots \\ \langle x_l, x_1\rangle & \cdots & \langle x_l, x_l\rangle \end{bmatrix} \\ &= \begin{bmatrix} x_{11}x_{11} & \cdots & x_{11}x_{l1} \\ \vdots & \vdots & \vdots \\ x_{l1}x_{11} & \cdots & x_{l1}x_{l1} \end{bmatrix} + \cdots \\ &\quad + \begin{bmatrix} x_{1n}x_{1n} & \cdots & x_{1n}x_{ln} \\ \vdots & \vdots & \vdots \\ x_{ln}x_{1n} & \cdots & x_{ln}x_{ln} \end{bmatrix}. \end{aligned} \tag{6}$$

In [19], each feature of data is associated with an indicator value. In this paper, based on the similar idea, we introduce the scaling factors β_i $(i = 1, \ldots, n)$ of features of data in the Gram matrix K. Accordingly, one can obtain

$$\begin{aligned} \widetilde{K} &= \beta_1 \left(x_{11}, \ldots, x_{l1}\right)\left(x_{11}, \ldots, x_{l1}\right)^T + \cdots \\ &\quad + \beta_n \left(x_{1n}, \ldots, x_{ln}\right)\left(x_{1n}, \ldots, x_{ln}\right)^T = \sum_{i=1}^{n} \beta_i v_i v_i^T, \end{aligned} \tag{7}$$

where $v_i = (x_{1i}, \ldots, x_{li})^T$. From (7), it is observed that the ith feature is removed if $\beta_i = 0$. Further, if the scaling factors β_i $(i = 1, \ldots, n)$ are sparse, this corresponds to selecting part of the features. If the scaling factors of features are not sparse, a large value of β_i indicates a useful and important feature. As a result, feature selection can be performed by removing the features that correspond to small scaling factors [8, 9]. In addition, it should be pointed out that the condition $\beta_i \geq 0$

should be imposed in order that the matrix $\widetilde{K}$ is semidefinite. If the kernel in (5) takes $\widetilde{K}$ in (7), then one has

$$\begin{gathered}\min_{\beta}\max_{\alpha} \quad L(\alpha,\beta) = \min_{\beta}\max_{\alpha} 2\alpha^T e - \alpha^T \operatorname{diag}(Y)\sum_{i=1}^{n}\beta_i v_i v_i^T \operatorname{diag}(Y)\alpha \\ C \geq \alpha_j \geq 0, \quad j = 1,\dots,l, \ \alpha^T y = 0, \ \operatorname{trace}\left(\sum_{i=1}^{n}\beta_i v_i v_i^T\right) = c, \ \beta_i \geq 0, \ i = 1,\dots,n.\end{gathered} \tag{8}$$

Applying an idea in [18], one can recast (8) as a semidefinite programming problem which can be solved using a general-purpose program such as SeDuMi [20] or SDPT3 [21]. Note that $\widetilde{K}$ is a linear combination of rank-one matrices $\beta_i v_i v_i^T$. Equation (8) can also be formulated as the quadratically constrained quadratic programming (QCQP), which is a special form of SDP.

As will be shown in the following, directly solving (8) may not be suitable for feature scaling. To determine why this happens, in the following we first analyze the characteristics of (8). To this end, we start by stating the following definition.

Definition 1. If a pair of variables (α^*, β^*) exists such that for all α and β

$$L(\alpha,\beta^*) \leq L(\alpha^*,\beta^*) \leq L(\alpha^*,\beta), \tag{9}$$

then (α^*, β^*) is a saddle point of (8).

The right-hand side of (9) says that β^* minimizes $L(\alpha^*,\beta)$ for α^*. The left-hand side of (9) says that α^* maximizes $L(\alpha,\beta^*)$ for β^*. Accordingly, it follows that (α^*,β^*) is a saddle point if and only if $\max L(\alpha,\beta^*) = L(\alpha^*,\beta^*) = \min L(\alpha^*,\beta)$ [22]. If there exist saddle points of (8), strong duality holds and the optimal values for β_i $(i = 1,\dots,n)$ and α_j $(j = 1,\dots,l)$ are obtained simultaneously. One can also observe that (8) is a linear programming problem with respect to β if the optimal value of α^* is given. This shows that the optimal value of β can be obtained by searching for extreme points in a linear programming (LP) problem if α^* is known. Although nonextreme points may be optimal values of LP, there always exist extreme points corresponding to optimal values. Based on these facts, one may obtain the extreme points in the linear programming problem as the optimal β. Thus, the solution to β will be overly sparse, since there exists one equation with respect to β in (8). Further speaking, only few scaling factors are not zero in such a case. As a result, one cannot evaluate the importance of most features in such a case.

So now we know that there is a possibility that there might be some cases when there are just few scaling factors of data that are not zero, making it become unsuitable for feature selection in most cases due to the fact that only few features are chosen. To deal with this difficulty, we add the L2-norm of β_i $(i = 1,\dots,n)$ to the objective function of (8). Accordingly, (8) has the following form:

$$\begin{aligned}\min_{\beta}\max_{\alpha} \quad & 2\alpha^T e - \alpha^T \operatorname{diag}(Y)\sum_{i=1}^{n}\beta_i v_i v_i^T \operatorname{diag}(Y)\alpha \\ & + \lambda\sum_{i=1}^{n}\beta_i\beta_i \\ & C \geq \alpha_j \geq 0, \quad j = 1,\dots,l, \ \alpha^T y = 0, \\ & \operatorname{trace}\left(\sum_{i=1}^{n}\beta_i v_i v_i^T\right) = c, \\ & \beta_i \geq 0, \ i = 1,\dots,n.\end{aligned} \tag{10}$$

Likewise, one can transform (10) into a semidefinite programming (SDP) problem.

3.2. Second-Order Cone Programming (SOCP) Problems. Solving SDP remains computationally expensive even with the advances in interior point methods. One way to reduce this computational complexity is to transform (10) into a second-order cone programming (SOCP) problem. Second-order cone programming problems are convex optimization problems in which a linear function is minimized over the intersection of an affine linear manifold with the Cartesian product of second-order cones. Interior point methods for solving SOCP directly have a much better worst complexity than those for SDP. As a result, solving SOCP problems is more efficient than solving SDP problems in practice. It is also noted that SOCP has obtained many applications in machine learning in recent several years [23, 24]. Miyashiro and Takano [25] used mixed integer second-order cone programming formulations for variable selection in linear regression. More interesting, the SOCP technique can be used to solve the problems of support vector machines [26, 27].

Applying the techniques in [28, 29], one can formulate (10) as the following SOCP problem:

$$\begin{aligned}\min_{\beta,\eta,\zeta,b,t_1,t_2,h,\rho} \quad & = \frac{1}{2}t_1 + C\zeta^T e + \lambda t_2 \\ \text{s.t.} \quad & \sum_{i=1}^{n} v_i h_i = -(b_1 - b_2)y + \eta - \zeta + e,\end{aligned}$$

$$t_1 \geq \sum_{i=1}^{n} \rho_i,$$
$$\beta_i \rho_i \geq (h_i)^T h_i, \ t_2 \geq \beta^T \beta,$$
$$\operatorname{trace}\left(\sum_{i=1}^{n} \beta_i v_i v_i^T\right) = c, \ \eta, \zeta, \beta, h, \rho \geq 0, \tag{11}$$

where $K_{(i)} = v_i(v_i)^T$, $r_i = 1$, and $h_i \in R^1$, $i = 1, \ldots, n$, $\eta, \zeta \in R^l$. One can solve (11) by using MOSEK optimization software (http://www.mosek.com/). From (11), one can see that it contains the L1-norm in the constraint. We refer to (11) as L1L2SOCP for clarity. In fact, the constraint $\operatorname{trace}(\sum_{i=1}^{n} \beta_i v_i v_i^T) = c$ may be removed in (11), since there exists one constraint $t_2 \geq \beta^T \beta$. In such a case, we refer to (11) as L2-norm SOCP (L2SOCP). Here it should be pointed out that computational complexity of solving (11) is $O(n^2 l)$, where n is the number of features and l is the number of samples. It is obvious that this method is very effective when the training set contains a large number of samples and becomes less effective when the number of features is very huge. It should be noted that one generally obtains the nonsparse optimal solutions of β_i $(i = 1, \ldots, n)$ in the case of the L2-norm of β_i $(i = 1, \ldots, n)$. It is obvious that scaling factors of features are globally optimal by solving (11), which is different from previous methods, where scaling factors are often locally optimal. In order to obtain the decision function, Lagrange multipliers $\alpha_i \geq 0$ and the bias b can be achieved by the following equations:

$$\alpha = \left(\operatorname{diag}(Y) \sum_{i=1}^{n} \beta_i v_i v_i^T \operatorname{diag}(Y)\right)^{+} \cdot \left[\left(I_l, -I_l, -y, y\right)\left(\eta^T, \zeta^T, b_1, b_2\right)^T + e\right], \tag{12}$$
$$b = b_1 - b_2, \tag{13}$$

where $(\cdot)^{+}$ denotes the pseudoinverse of the matrix and I_l is an $l \times l$ identity matrix.

4. Experimental Results

In this section, we carry out the experiments on simulated data, UCI data sets, and the gene data set to evaluate the proposed optimization model.

4.1. Effect of Irrelevant Features. To evaluate the proposed method, where irrelevant features are present, we generate N-dimensional Gaussian data from two classes, where $\mu_1 = [1, 1, 0, \ldots, 0]^T$, $\mu_2 = [-1, 1, 0, \ldots, 0]^T$, and the covariance matrices are both identity matrices. We compare the proposed method with classical support vector machines (CSVMs), SVM with multiple parameters based on the radius-margin bound (RW), and SVMs with multiple parameters based on the span bound (Span). All the algorithms are trained on a training set consisting of 100 samples of each class and are tested on an independent test of 1000 samples. In order to reduce variations of performance, the experimental results are reported by averaging over 20 runs. Figure 1(a) shows the error rate of each method with the increase of irrelevant features. Figure 1(b) shows the scaling factors of our method.

From Figure 1(a), one can see that the proposed method is superior to CSVM, SVM (RW), or SVM (Span) with the increase of features. This may be because the scaling factors in SVM (RW) and SVM (Span) are locally optimal due to gradient descent and the scaling factors in our method are globally optimal. Note that, in this experiment, we do not carry out feature selection in either SVM (RW) or SVM (Span). These two methods do perform better than CSVM but are not better than the proposed method (L2SOCP or L1L2SOCP). This shows that feature scaling indeed improves the performance of classifiers when irrelevant features are present. From Figure 1(b), it is found that the scaling factors which correspond to the relevant features are significantly larger than those corresponding to the irrelevant features. In other words, one feature scaling factor is obviously bigger than the other scaling factors. In such a case, one can only select the feature with the largest scaling factors. This confirms that it is reasonable to use scaling factors to select relevant features. Overall, the experiments show that feature scaling can improve the classification performance in the presence of irrelevant features and scaling features can be used to select the relevant features.

Table 1: Statistics for the data sets we deal with.

Data sets	Number of samples	Number of dimensions	Number of classes
Australian	690	14	2
Breast	683	10	2
Diabetes	768	8	2
German	270	13	2
Heat	1000	24	2
Ionosphere	351	34	2
Sonar	208	60	2
Liver	345	6	2

4.2. Experimental Results on Benchmark Data Set. To further test the performance of the proposed method, we compared it with CSVM, SVM (RW) and SVM (Span), the iterative method (IM) in [11], and alternative second-order cone programming (ASOCP) formulations of SVMs in [23] on a collection of benchmark data sets which can be obtained from the UCI Machine Learning Repository [30]. These data sets have been widely used in testing and evaluating the performance of learning algorithms. Table 1 shows the statistics for some data sets we use. The attributes of each data set are normalized to the interval of $[-1, 1]$. The parameters of all the support vector machines are estimated using tenfold cross validation on the training set. There exist three parameters, C, c, and λ, in L1L2SOCP. For simplicity, we first perform a grid search over two-dimensional parameter space

Table 2: The error rates (%) of each method on eight binary data sets.

Data set	L2SOCP	L1L2SOCP	CSVMs	ASOCP	SVM (Span)	SVM (RW)	IM
Australian	14.21	**14.01**	15.05	15.01	14.09	14.36	14.28
Breast	3.06	**2.91**	**2.91**	2.92	2.98	3.02	2.93
German	**23.10**	23.90	23.55	23.57	23.70	23.76	23.32
Heart	17.04	**16.76**	17.41	17.48	17.31	17.26	17.38
Ionosphere	**11.11**	11.39	11.85	11.84	11.29	11.56	11.78
Diabetes	**22.40**	22.79	22.79	22.69	22.68	22.54	22.80
Liver	30.67	**30.15**	31.81	31.79	30.55	31.28	31.13
Sonar	**22.18**	23.01	23.48	23.55	22.35	23.58	23.27

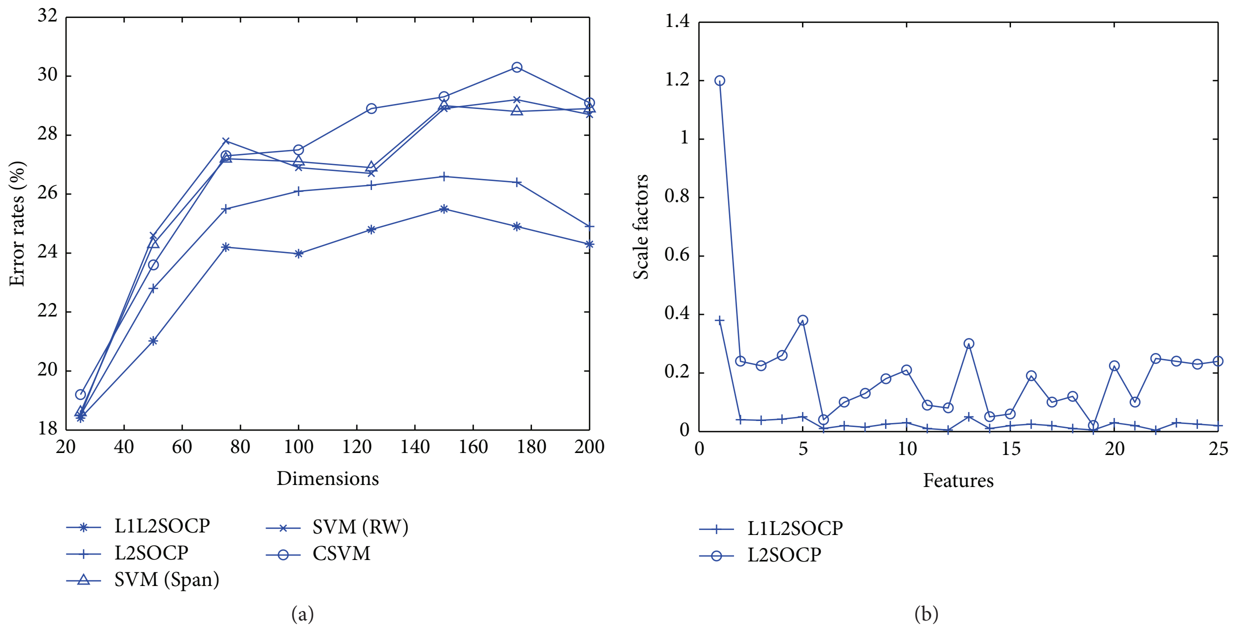

Figure 1: Experimental results on simulated data. (a) Performance comparisons of several featuring scaling methods. (b) Scaling factors with the dimension $d = 25$.

(C, λ) in the case that $c = 1$, where λ ranges from 2^{-10} to 2^6 and C ranges from 2^{-9} to 2^9. Then we fix λ and perform a grid search over two-dimensional parameter space (C, c). In order to evaluate the performance, tenfold cross validation is performed and the average error rate of each method is reported in Table 2.

As can be seen from Table 2, our method is superior to SVM (RM), SVM (Span), and IM in most cases. This may come from the fact that the scaling factors in our method are globally optimal and the scaling factors in other methods are locally optimal. Moreover, L2SOCP is better than CSVM on Ionosphere and Diabetes data sets and L1L2SOCP is better than CSVM on Heart and Liver data sets. This shows the proposed method can use feature scaling to improve the classification performance to some degree. It is found that ASOCP obtains similar performance with CSVM, since these two methods solve SVMs in different optimization algorithms. In addition, we performed a two-tailed t-test with a significance level of 0.05 to determine whether there is a significant difference between the proposed method and other methods. The results show that there is no significant difference among these methods on these data sets. This may be due to the fact that these data sets have few completely irrelevant features. However, we may set a threshold to scaling factors and choose those features as the most relevant features, where the scaling factors are bigger than the threshold. This strategy can effectively reduce the number of features by performing feature selection.

4.3. Gene Expression Data. DNA microarrays [31] measure the expression levels of thousands of genes simultaneously. In general, the gene expression data contains a large number of irrelevant and redundant features. Feature scaling techniques can be used to select relevant genes and contribute to discriminating different genes. In the following experiments, we test our method on the colon data set. This data set contains 62 samples: 22 normal and 40 cancerous colon examples. These two classes are discriminated by the expression profiles of 2000 genes.

In the first set of experiments, the proposed method is evaluated using the leave-one-out cross validation (LOOCV).

Table 3: LOOCV accuracy (%) of the colon data set.

Methods	Accuracy
L1SOCP	**87.1**
L1L2SOCP	**87.1**
SVM	77.4
RVM	80.6
Logistic regression	71.0
JCFO	86.8
MISOCP	86.3

Table 4: Performance comparisons on different selected features.

Selected features	1024	512	256	128	64	32	16	8	4	2	1
Methods	Number of misclassified samples										
RFE + SVM	22	24	22	13	9	4	0	1	1	2	4
L2FS + L2SOCP	22	**14**	**9**	**7**	**4**	**0**	**0**	**1**	**1**	**2**	**4**
L1L2FS + L1L2SOCP	22	22	20	9	8	0	0	1	1	2	4
L2FS + SVM	22	15	10	8	3	0	0	1	1	3	4
L1L2FS + SVM	22	20	18	8	8	0	0	1	1	2	4
MISOCP + SVM	22	20	19	8	5	1	0	1	2	2	4

Table 3 shows the LOOCV results of the proposed method, SVM, RVM [32], logistic regression, and JCFO [32] and mixed integer SOCP (MISOCP) [25] for variable selection. Note that in these methods we only use the linear kernel. From Table 3, it is found that the classification performance of the L2SOCP or L1L2SOCP method is superior to that of SVM, RSVM, and logistic regression. This also shows that the feature scaling can improve the performance of classifiers in the presence of irrelevant features. It is also found that our method is superior to JCFO. This may be because JCFO performs feature selection, while our method does not. It is noted that MISOCP is worse than our method, since MISOCP is only used to select relevant features. Overall, using scaling factors to choose the effective features in the presence of redundant features can improve the performance of algorithms.

In the second set of experiments, we select the features in terms of scaling factors. The larger the scaling factors are, the more important the features are. In general, the features can be sorted based on scaling factors. For comparison purposes, we also perform the recursive feature elimination (REF) method using linear SVMs [2] and MISOCP [25]. Table 4 shows the number of misclassified samples of each method with the change of selected features. For L2FS + L2SOCP and L1L2FS + L1L2SOCP in Table 4, we first use L2SOCP and L1L2SOCP to obtain the scaling factors and then select those features in terms of their scaling factors. We continue to perform L2SOCP and L1L2SOCP on the selected features. For L2FS + SVM and L1L2FS + SVM in Table 4, we first adopt L2SOCP and L1L2SOCP to obtain the scaling factors of features and then select the features in terms of their scaling factors. We continue to perform SVM on the selected features. That is, the results are obtained by using classical SVMs on a subset of genes selected by scaling factors of features based on L2SOCP or L1L2SOCP.

From Table 4, it is found that feature scaling can further improve the performance of classifiers if the number of selected features is high. To be specific, when the number of selected features is bigger than 16, the performance of L2FS + L2SOCP or L1L2FS + L1L2SOCP is superior to that of RFE + SVM. However, as the number of selected features decreases, the feature scaling will result in performance deterioration. That is, when the number of selected features is smaller than 16, the performance of L2FS + L2SOCP or L1L2FS + L1L2SOCP is the same as that of RFE + SVM. However, if we perform SVMs based on selected features from L2SOCP or L1L2SOCP, it is found that the L2FS + SVM or L1L2FS + SVM method is better than the RFE + SVM or MISOCP + SVM method, since our method can obtain the globally optimal scaling factors. This shows that it is better to select those features in terms of L2SOCP or L1L2SOCP and then to perform SVM for classification tasks. It is also observed that the best performance of our method is superior to that of the JCFO method in the first set of experiments if the scaling factors are used to select the features. Overall, the experiments show that it is effective to select the features in terms of scaling factors obtained by L2SOCP or L1L2SOCP.

5. Conclusions and Further Work

When the data in real world contains redundant features, it is important to select those effective features in order to enhance the classification performance of classifiers. Different from previous methods, in this paper, we assign weights to each feature of data and then use the generalized performance measure to construct the optimization model. Fortunately, the proposed optimization model can be transformed into the problem of SOCP. Thus the weight or scaling factors of features can be obtained in globally optimal way. In terms of scaling factors that are nonnegative, the optimal features can be easily obtained. A number of experiments on a toy example, UCI data set, and the gene data set are done and it is found that the proposed method obtains better performance than previous methods. It is also noted that our method is only suitable for linear kernels. It is of interest to extend our optimization model to its nonlinear kernel version, which is our further work in the near future.

Competing Interests

The author declares that there are no competing interests regarding the publication of this paper.

Acknowledgments

This work is partially supported by the FRF for the Central Universities (2015XKMS084).

References

[1] P. S. Bradley and O. L. Mangasarian, "Feature selection via concave minimization and support vector machine," in *Proceedings of the 15th International Conference on Machine Learning (ICML '98)*, pp. 82–90, ACM, 1998.

[2] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik, "Gene selection for cancer classification using support vector machines," *Machine Learning*, vol. 46, no. 1–3, pp. 389–422, 2002.

[3] A. Arauzo-Azofra, J. L. Aznarte, and J. M. Benítez, "Empirical study of feature selection methods based on individual feature evaluation for classification problems," *Expert Systems with Applications*, vol. 38, no. 7, pp. 8170–8177, 2011.

[4] T. Naghibi, S. Hoffmann, and B. Pfister, "A semidefinite programming based search strategy for feature selection with mutual information measure," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 8, pp. 1529–1541, 2015.

[5] L. Laporte, R. Flamary, S. Canu, S. Dejean, and J. Mothe, "Nonconvex regularizations for feature selection in ranking with sparse SVM," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 6, pp. 1118–1130, 2014.

[6] Y. Liu, F. Tang, and Z. Zeng, "Feature selection based on dependency margin," *IEEE Transactions on Cybernetics*, vol. 45, no. 6, pp. 1209–1221, 2015.

[7] J. Weston, S. Mukherjee, O. Chapelle, M. Pontil, T. Poggio, and N. Vapnik, "Feature selection for SVMs," in *Advances in Neural Information Processing Systems 13*, pp. 668–674, 2001.

[8] Y. Grandvalet and S. Canu, "Adaptive scaling for feature selection in SVMs," in *Proceedings of the 16th Annual Neural Information Processing Systems Conference (NIPS '02)*, vol. 15, pp. 553–560, Vancouver, Canada, December 2002.

[9] O. Chapelle, V. Vapnik, O. Bousquet, and S. Mukherjee, "Choosing multiple parameters for support vector machines," *Machine Learning*, vol. 46, no. 1–3, pp. 131–159, 2002.

[10] L. F. Bo, L. Wang, and L. C. Jiao, "Feature scaling for kernel fisher discriminant analysis using leave-one-out cross validation," *Neural Computation*, vol. 18, no. 4, pp. 961–978, 2006.

[11] Z. Liang and T. Zhao, "Feature selection for linear support vector machines," in *Proceedings of the 18th International Conference on Pattern Recognition*, pp. 351–355, Hong Kong, August 2006.

[12] P. Maji, "A rough hypercuboid approach for feature selection in approximation spaces," *IEEE Transactions on Knowledge and Data Engineering*, vol. 26, no. 1, pp. 16–29, 2014.

[13] X. Liu, L. Wang, J. Zhang, J. Yin, and H. Liu, "Global and local structure preservation for feature selection," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 6, pp. 1083–1095, 2014.

[14] Y. Li, J. Si, G. Zhou, S. Huang, and S. Chen, "FREL: a stable feature selection algorithm," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 26, no. 7, pp. 1388–1402, 2015.

[15] Z. Li and J. Tang, "Unsupervised feature selection via nonnegative spectral analysis and redundancy control," *IEEE Transactions on Image Processing*, vol. 24, no. 12, pp. 5343–5355, 2015.

[16] D. Wang, F. Nie, and H. Huang, "Feature selection via global redundancy minimization," *IEEE Transactions on Knowledge and Data Engineering*, vol. 27, no. 10, pp. 2743–2755, 2015.

[17] H. Tao, C. Hou, F. Nie, Y. Jiao, and D. Yi, "Effective discriminative feature selection with nontrivial solution," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 27, no. 4, pp. 796–808, 2016.

[18] G. R. G. Lanckriet, N. Critianini, P. Barteltt, L. El Ghaoui, and M. I. Jordan, "Learning the kernel matrix with semidefinite programming," *The Journal of Machine Learning Research*, vol. 5, pp. 27–72, 2004.

[19] L. Wolf and A. Shashua, "Feature selection for unsupervised and supervised inference: the emergence of sparsity in a weighted-based approach," *Journal of Machine Learning Research*, vol. 6, pp. 1855–1887, 2005.

[20] J. F. Sturm, "Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones," *Optimization Methods and Software*, vol. 11-12, no. 1–4, pp. 625–653, 1999.

[21] K. C. Toh, M. J. Todd, and R. H. Tutuncu, "SDPT3—a MATLAB software package for semidefinite programming, version 1.3," *Optimization Methods and Software*, vol. 11, no. 1–4, pp. 545–581, 1999.

[22] D. P. Bertsekas, A. Nedic, and A. E. Ozdaglar, *Convex Analysis and Optimization*, Athena Scientific, Belmont, Mass, USA, 2003.

[23] J. López and S. Maldonado, "Alternative second-order cone programming formulations for support vector classification," *Information Sciences*, vol. 268, no. 1, pp. 328–341, 2014.

[24] S. Maldonado and J. López, "Imbalanced data classification using second-order cone programming support vector machines," *Pattern Recognition*, vol. 47, no. 5, pp. 2070–2079, 2014.

[25] R. Miyashiro and Y. Takano, "Mixed integer second-order cone programming formulations for variable selection in linear regression," *European Journal of Operational Research*, vol. 247, no. 3, pp. 721–731, 2015.

[26] J. López and S. Maldonado, "Multi-class second-order cone programming support vector machines," *Information Sciences*, vol. 330, no. 10, pp. 328–341, 2016.

[27] M. Carrasco, J. López, and S. Maldonado, "A second-order cone programming formulation for nonparallel hyperplane support vector machine," *Expert Systems with Applications*, vol. 54, no. 15, pp. 95–104, 2016.

[28] F. Alizadeh and D. Goldfarb, "Second-order cone programming," *Mathematical Programming—Series B*, vol. 95, pp. 3–51, 2003.

[29] I. W.-H. Tsang and J. T.-Y. Kwok, "Efficient Hyperkernel learning using second-order cone programming," *IEEE Transactions on Neural Networks*, vol. 17, no. 1, pp. 48–58, 2006.

[30] C. Blake, E. Keogh, and C. J. Merz, *UCI Repository of Machine Learning Databases*, University of California, Irvine, Calif, USA, 1998, http://www.ics.uci.edu/~mlearn/MLRepository.html.

[31] U. Maulik and D. Chakraborty, "Fuzzy preference based feature selection and semisupervised SVM for cancer classification," *IEEE Transactions on Nanobioscience*, vol. 13, no. 2, pp. 152–160, 2014.

[32] B. Krishnapuram, A. J. Hartemink, L. Carin, and M. A. T. Figueiredo, "A Bayesian approach to joint feature selection and classifier design," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 9, pp. 1105–1111, 2004.