Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2014) 22: 1664 – 1685

doi:10.3906/elk-1301-190

A penalty function method for designing efficient robust classifiers with input space optimal separating surfaces

Ayşegül UÇAR[1,*], Yakup DEMİR[2], Cüneyt GÜZELİŞ[3]
[1]Department of Mechatronics Engineering, Firat University, Elazığ, Turkey
[2]Department of Electrical and Electronics Engineering, Firat University, Elazığ, Turkey
[3]Department of Electrical and Electronics Engineering İzmir University of Economics, Balçova, İzmir, Turkey

Received: 25.01.2013 • **Accepted:** 31.03.2013 • **Published Online:** 07.11.2014 • **Printed:** 28.11.2014

Abstract: This paper considers robust classification as a constrained optimization problem. Where the constraints are nonlinear, inequalities defining separating surfaces, whose half spaces include or exclude the data depending on their classes and the cost, are used for attaining robustness and providing the minimum volume regions specified by the half spaces of the surfaces. The constraints are added to the cost using penalty functions to get an unconstrained problem for which the gradient descent method can be used. The separating surfaces, which are aimed to be found in this way, are optimal in the input data space in contrast to the conventional support vector machine (SVM) classifiers designed by the Lagrange multiplier method, which are optimal in the (transformed) feature space. Two types of surfaces, namely hyperellipsoidal and Gaussian-based surfaces created by radial basis functions (RBFs), are focused on in this paper due to their generality. Ellipsoidal classifiers are trained in 2 stages: a spherical surface is found in the first stage, and then the centers and the radii found in the first stage are taken as the initial input for the second stage to find the center and covariance matrix parameters of the ellipsoids. The penalty function approach to the design of robust classifiers enables the handling of multiclass classification. Compared to SVMs, multiple-kernel SVMs, and RBF classifiers, the proposed classifiers are found to be more efficient in terms of the required training time, parameter setting time, testing time, memory usage, and generalization error, especially for medium to large datasets. RBF-based input space optimal classifiers are also introduced for problems that are far from ellipsoidal, e.g., 2 Spirals.

Key words: Classification, gradient methods, penalty approach, spherical/elliptical separation, support vector machines

1. Introduction

Design of robust classifiers, namely determining surfaces separating data from each other and from the noise and outliers with low misclassification error and with low time-memory requirement, is one of the most fundamental problems in pattern recognition. In the last 2 decades, artificial neural networks (ANNs) and support vector machines (SVMs) have offered solutions as alternatives to the conventional classification methods [1,2]. Algebraic ANNs, which define input-output mappings, are used to obtain nonlinear separating surfaces. They are usually designed in a supervised way using a gradient descent algorithm for minimizing the empirical risk, i.e. the training error for a given input-desired output (data-class label) set [2]. The generalization ability is attempted to be gained by some techniques embedded in the learning algorithms, such as the prevention of overfitting, and there is little concern about noise and outliers hidden in the training data.

*Correspondence: agulucar@firat.edu.tr

SVMs have a framework providing a design methodology for classifiers that minimize not only the training error but also the generalization error. An SVM transforms the input data, by a nonlinear mapping defined with a kernel, into a high-dimensional feature space, where the data become more likely linearly separable, and then produces an optimal separating hyperplane that maximizes the margin, i.e. the maximum distance away from the closest (transformed) data belonging to the classes, while ignoring the data falling in the margin to suppress the effect of noise and outliers as much as possible. SVMs use an optimization formulation, where the cost corresponds to the maximization of the margin or equivalently the minimization of a structural risk term for providing the generalization ability, and the linear inequalities in the feature space as constraints define the inclusion or exclusion of the training data by the classes. SVMs employ the Lagrange multiplier method of optimization theory for having an unconstrained optimization problem as augmenting the cost by adding a term for each inequality constraint via a Lagrange multiplier. SVMs reach a quadratic cost in terms of Lagrange multipliers after the elimination of all of the primal parameters, i.e. linear weights defining the separating hyperplane and kernel parameters using the first-order optimality conditions [3]. SVMs constitute quite powerful methods for classification since they have generalization ability, their formulation is independent from the dimension of the data space, and their design requires solving a quadratic minimization problem, which becomes convex when defined by a positive semidefinite kernel, so it can be minimized using one of the many available efficient algorithms. However, SVM classifiers have some problematic features, some of which are outlined below.

I) SVMs are originally developed for 2-class classification problems [4]. Multiclass classification problems need to be solved using 2 or more SVMs, as done in the extensions of SVMs developed in the literature: one versus one, one versus all, and directed acyclic graph [5–7]. However, an optimization formulation enables us to define multiclass classification as a unique optimization problem, so requiring just one routine to be run is more appropriate [8]. One formulation of multiclass classification is presented in this paper.

II) The best appropriate parameter selection of SVM addressed in [9,10] could take a very long time for some problems. The computation and memory needs for SVM grow proportionally according to the training set size; hence, the solution is obtained very slowly for large datasets [11] since a standard SVM has to solve a quadratic programming optimization problem so that the optimization variables are Lagrange multipliers, which are as numerous as the number of training samples. To prevent having an excessive number of optimization variables, inequality constraints might be embedded into the cost in another way, different from the Lagrange multiplier method, which associates an optimization variable for each constraint. The penalty function method employed in this paper provides such a solution.

III) Depending on the choice of kernel and the kernel's parameter, the distance order among the samples may not be preserved while carrying them into the feature space by the nonlinear mapping defined by the chosen kernel [12–15]. This means that the inverse image of the optimal separating hyperplane found in the feature space may be far away from the optimal separating surface in the input data space.

There are many works in the literature that exploit spherical, ellipsoidal, and radial basis surfaces for classification purposes [16–26]. However, some of them are devoted to a single-class classification problem only. Some do not search for minimum volume region classes and ignore the generalization ability and robustness against the noise and outliers. Some search for optimal surfaces in the feature space, not in the input space. Lagrange multiplier and penalty parameter methods are usually used in these works to transform constrained

optimization, defining the classification problem as an unconstrained one. None of them provide a scheme to learn penalty parameters; they just use fixed parameters.

The classifier design method proposed in this paper is different from those of the literature in the following aspects. The multiclass classification problem is posed as a single (constrained) optimization routine. The volumes of the class regions are taken as the main cost in the minimization, which are introduced to provide the generalization ability. The constrained optimization defining the multiclass classification problem is transformed into an unconstrained one by applying the penalty function method of the optimization theory. The penalty parameter, which balances the empirical risk term (i.e. the misclassification error term for the training samples) and the structural risk term (i.e. the sum of squares of the radii or linear connection weights that corresponds to the total volume of the class regions in some sense), is not chosen fixed, but is learned in an iterative way while learning the parameters defining the classes and the separating surfaces.

The remainder of the paper is organized in 5 sections. The method proposed has the same purpose as the SVM and its derivation is also based on the same optimization theory as the SVM, although exploiting a different optimization approach. To see the common and different features, Section 2 is devoted to providing preliminary information about SVMs. Section 3 provides the formulations and solution methods of the spherical and ellipsoidal classifiers and novel classifiers using a convectional radial basis function (RBF) classifier (CRBFC) surface. Numerical results are given in Section 4. Finally, Section 5 concludes the paper with some final remarks.

2. Background of SVMs

Given L samples of training data $(x^1, y^1), ..., (x^L, y^L), x \in \Re^n, y \in \{-1, 1\}$, the SVM first transforms the input samples to a high-dimensional feature space via a nonlinear mapping $\phi(x)$ and then classifies the transformed feature vectors by the optimal separating hyperplane, defined by Eq. (1), in the feature space:

$$\ell(x) = w^T \phi(x) + b, \tag{1}$$

where $w \in \Re^n$ stands for the normal vector and $b \in \Re$ determines a measure for the offset of the separating hyperplane [1,2]. The optimal separating hyperplane is the one with the maximum margin and least error due to the intramarginal misclassified (transformed) training samples. Herein, the margin is the distance between the separating hyperplane and the transformed class sample(s) nearest to the hyperplane (namely support vector(s), SVs). The optimal separating hyperplane defined in this way represents a nonlinear decision surface in the input space, so a nonlinear classifier called the SVM classifier can be found by solving the primal optimization problem given in Eqs. (2) and (3):

$$\min_{w,b,\xi} L_{primal}(w, \xi_i) = C \sum_{i=1}^{L} \xi_i + \frac{1}{2} \|w\|^2, \tag{2}$$

$$\text{subject to } y^i \left[w^T \phi(x^i) + b \right] \geq 1 - \xi_i. \tag{3}$$

The first term in the cost penalizes the sum of absolute errors ξ_i, which are measures of the distances between the intramarginal misclassified (transformed) training samples and the separating hyperplane. The second term in the cost forces the margin to maximum. The constraints correspond to the correct classification of the class samples outside the margin. C creates a tradeoff between the margin size and the error due to the allowed misclassified intramarginal training examples.

Using the Lagrange multiplier method, the constrained problem in Eqs. (2) and (3) is transformed into an unconstrained one:

$$\hat{L}_{primal}(w,b,\xi,\lambda,\beta) = C\sum_{i=1}^{L}\xi_i + \frac{1}{2}\|w\|^2 - \sum_{i=1}^{L}\lambda_i\left\{y^i\left[w^T\phi(x^i)+b\right]-1+\xi_i\right\} - \sum_{i=1}^{L}\beta_i\xi_i, \tag{4}$$

where $\lambda_i \geq 0$ and $\beta_i \geq 0$ are Lagrange multipliers. The solution of the primal problem in Eq. (4) is the optimal saddle point $(w_o, b_o, \xi_o, \lambda_o, \beta_o)$, which can be found by minimizing it with respect to w, b, and ξ, and by maximizing it with respect to λ_i and β_i. The problem can also be solved in the dual space of the Lagrange multipliers. The dual form follows from the Karush–Kuhn–Tucker optimality conditions:

$$\frac{\partial \hat{L}_{primal}}{\partial w_o} = 0 \Rightarrow w_o = \sum_{i=1}^{L}\lambda_i y^i \phi(x^i),$$

$$\frac{\partial \hat{L}_{primal}}{\partial b_o} = 0 \Rightarrow \sum_{i=1}^{L}\lambda_i y^i = 0, \qquad \frac{\partial \hat{L}_{primal}}{\partial \xi_o} = 0 \Rightarrow \lambda_i + \beta_i = C, \tag{5}$$

with $\lambda_i\{y^i[w^T\phi(x^i)+b]-1+\xi_i\}=0$, $\beta_i\xi_i=0$, $\lambda_i \geq 0$, $\beta_i \geq 0$, $\xi_i \geq 0$ for $i=1,...,L$. w, b, and ξ variables in Eq. (4) are calculated in terms of the Lagrange multipliers using Eq. (5). By eliminating all variables other than the Lagrange multipliers to λ_is and by introducing kernel $K(x^i,x^j)$ to represent the inner product by $\phi(x^i)^T\phi(x^j)$, the dual form is obtained as the following quadratic optimization problem, since many efficient solution methods exist.

$$\max_{\lambda} L_{dual}(\lambda) = -\frac{1}{2}\sum_{i,j=1}^{L}\lambda_i\lambda_j y^i y^j K(x^i,x^j) + \sum_{i=1}^{L}\lambda_i \tag{6}$$

$$\text{subject to } \sum_{i=1}^{L} y^i\lambda_i = 0, 0 \leq \lambda_i \leq C, i=1,...,L \tag{7}$$

The separating hyperplane is thus obtained as:

$$\ell(x) = sign\Big(\sum_{support\ vectors} y^i\lambda_i K(x^i,x^j) + b\Big), \tag{8}$$

where the SVs are the data points x^i corresponding to $\lambda_i > 0$.

On the other hand, in the multikernel learning problem, the kernel is given as a convex combination of positive definite classical kernels, K_m:

$$K(x^i,x^j) = \sum_{m=1}^{M} d_m K_m(x^i,x^j) \text{ with } d_m \geq 0, \sum_{m=1}^{M} d_m = 1, \tag{9}$$

where d_m presents the kernel weight. In simple multiple-kernel learning (SimpleMKL) [27], the optimization problem in Eq. (10) is constructed. It uses projected gradient updates and trains the SVMs at each iteration.

$$J(d) = \begin{cases} \min\limits_{w,b,\xi,d} L_{SimpleMKLprimal}(w,\xi_i) = C\sum\limits_{i=1}^{L}\xi_i + \frac{1}{2}\sum\limits_{m=1}^{M}\frac{1}{d_m}\|w_m\|^2 \\ \text{subject to } y^i\sum\limits_{m=1}^{M}\left[w_m^T\phi_m(x^i)+b\right] \geq 1-\xi_i \end{cases} \tag{10}$$

3. Proposed classifiers

The idea behind the proposed design method for spherical classifiers (SCs) and ellipsoidal classifiers (ECs) is to associate 1 region, equivalently 1 surface, for each class so that a region is made large enough to include all of the associated class samples, and so the misclassification error for training samples and the volume of the region are kept at a minimum in order to prevent overfitting the training data and to exclude noise and outliers from the region, thus providing the generalization ability. Below, SCs are presented before presenting the ellipsoidal and radial basis ones, since the proposed method is best understood in the simplest setting with SCs.

3.1. Stage I: spherical classifiers

Let $(x^i, y_m^i) with x^i \in \Re^n, y_m^i \in \{-1, 1\}$ for $i = 1, ..., L$ and $m = 1, ..., M$ define a set of L samples, each of which is associated to one of the M classes $(M \geq 1)$, such that any label $y_m^i = 1$ for a unique m and –1 for the others. M spherical surfaces given in Eq. (11) can be used to define M-class SCs.

$$S_m = \{x \in \Re^n \left| ||(x^i - c_m)||^2 - R_m^2 = 0, i = 1, ..., L, m = 1, ..., M \right.\} \tag{11}$$

Here, $R_m \in \Re$ is the radius of the surface S_m and $c_m \in \Re^n$ is the center. To have zero misclassification error, S_m spheres should include the associated samples, i.e. the following quadratic inequalities should be satisfied:

$$||(x^i - c_m)||^2 - R_m^2 > 0 \quad y_m^i = -1, \tag{12}$$

$$||(x^i - c_m)||^2 - R_m^2 \leq 0 \quad y_m^i = 1. \tag{13}$$

The design of the robust SCs is defined as finding the smallest-volume spheres including the associated class samples:

$$\min_{c_m, R_m} g(R_m) = \sum_{m=1}^{M} R_m^2,$$

$$\text{subject to} \quad \begin{array}{ll} ||(x^i - c_m)||^2 - R_m^2 > 0 & y_m^i = -1 \\ ||(x^i - c_m)||^2 - R_m^2 \leq 0 & y_m^i = 1 \end{array}. \tag{14}$$

Note that the system of inequalities in Eqs. (12) and (13) might be an inconsistent system having no solution, i.e. the classes are spherically nonseparable. In such cases, the above design problem should be read as finding the smallest volume spheres, including the associated class samples as much as possible, and now the minimum error solution to the inequality system is searched for.

Using the penalty function method, the constrained optimization problem in Eq. (14) can be transformed into an unconstrained one as follows:

$$\min_{c_m, R_m} L_{SC} = \sum_{m=1}^{M} \sum_{i=1}^{L} f(y_m^i(||(x^i - c_m)||^2 - R_m^2)) + A \sum_{m=1}^{M} R_m^2, \tag{15}$$

where the penalty function $f(\bullet)$ is chosen as the ramp function:

$$\begin{array}{ll} f(\xi) = \xi \quad \xi > 0 & f'(\xi) = 1 \quad \xi > 0 \\ f(\xi) = 0 \quad \xi \leq 0 & f'(\xi) = 0 \quad \xi < 0. \end{array} \tag{16}$$

Note that ramp function $f(\bullet)$ is not a differentiable function of its ξ argument (the right-hand derivative is different from the left-hand derivative at the origin), but it is a differentiable function of the c_m centers and R_m radii. The reason for choosing such a penalty function, which is differentiable with respect to the class parameters, i.e. the centers and radii, relies on the fact that the gradient descent minimization method, which is chosen in this paper due to its efficiency in time and memory and due to its ability to find acceptable minima without having sensitive dependency on the initial centers and radii, requires the derivative of the cost with respect to the optimization variables. The chosen penalty function has another crucial property: it does not bring any extra variables other than the centers and radii, which are indispensable for defining spherical classes, making the proposed method superior to the Lagrange multiplier-based methods for large datasets where an optimization variable, i.e. a Lagrange multiplier, is introduced for each class sample.

The regularization parameter $A > 0$ is chosen as a variable to balance the volume of the spheres, which are desired to be sufficiently large to include almost all of the class samples and sufficiently small for preventing overfitting, and to exclude noise and outliers.

In the proposed SCs, each class is separately clustered by K-means. Any prototype selection method can be used for the initial cluster selection [28]. The cluster number is automatically adjusted, taking into consideration the quantization error ratio. In this paper, the error ratio is defined as the ratio of the obtained quantization error to the determined quantization error. If the error ratio is more than 3, the cluster number is increased by the nearest integer to the error ratio. Otherwise, the cluster number is increased by the input space dimension minus 1. In this paper, the quantization error is previously determined as $30 \times (n+2)/M$ for all of the datasets. For the SC stage, the initial cluster centers are set as the sphere center and the initial cluster radii are set as half of the distance to the mean of the samples in the cluster.

3.1.1. Gradient descent solution for the spherical classifier design

Several minimization methods, such as standard gradient descent, steepest gradient descent, conjugate gradient descent, scaled conjugate gradient descent, quasi-Newton, and the gradient descent adaptive learning rate (GDALR) including momentum term methods [2], are examined in solving the piecewise quadratic problem in Eq. (15). Among those methods listed, GDALR is the best when considering together all of the computation time and memory requirements, sensitivity to the initial centers and radii, and training and test performances.

To apply any gradient-based method to Eq. (15), it is necessary to calculate the negative gradients of the L_{SC} cost function with respect to the c_m centers and R_m radii:

$$-\frac{\partial L_{SC}}{\partial c_m} = 2\sum_{i=1}^{L} y_m^i (x^i - c_m) f'(y_m^i(||(x^i - c_m)||^2 - R_m^2)), \tag{17}$$

$$-\frac{\partial L_{SC}}{\partial R_m} = 2\sum_{i=1}^{L} y_m^i f'(y_m^i(||(x^i - c_m)||^2 - R_m^2))R_m - 2AR_m. \tag{18}$$

In GDALR, the c_m centers and R_m radii are updated as follows:

$$\Delta c_m(k+1) = \alpha \Delta c_m(k) - (1-\alpha)\eta(k)\frac{\partial L_{SC}}{\partial c_m}, \tag{19}$$

$$\Delta R_m(k+1) = \alpha \Delta R_m(k) - (1-\alpha)\eta(k)\frac{\partial L_{SC}}{\partial R_m}, \tag{20}$$

where α, the momentum constant, is usually chosen as 0.9. The learning rate $\eta(k)$ is updated as in Eq. (21) and is chosen the same in both of the update equations, Eqs. (19) and (20).

$$\eta(k) = \begin{cases} a\eta(k) \text{ if } L_{SC}(x(k)) < L_{SC}(x(k-1)) \\ d\eta(k) \text{ if } L_{SC}(x(k)) \geq eL_{SC}(x(k-1)) \\ \eta(k) \end{cases} \tag{21}$$

with $a = 1.05; d = 0.7; e = 1.04; \eta(0) = 0.01$.

Each $\ell_m(x)$ identifies the class samples, assigning them as 1 when they are in the positive half space of the mth class separating surface and –1 when they are in the negative half space:

$$\ell_m(x) = sign(R_m^2 - ||(x - c_m)||^2). \tag{22}$$

3.2. Learning regularization parameter

As with SVMs and any regularization problem [2,29], the value of the regularization parameter A in Eq. (15) is crucial for the quality of the classifiers obtained as solutions to the posed optimization problem. A value larger than the optimal, say A^*, causes a high misclassification error for the training samples, while a value smaller than A^* causes poor generalization ability. In other words, A should be chosen to balance the $L_{SC}^1 = \sum_{m=1}^{M} \sum_{i=1}^{L} f(y_m^i(||(x^i - c_m)||^2 - R_m^2))$ misclassification cost term and $L_{SC}^2 = \sum_{m=1}^{M} R_m^2$ generalization cost term.

It has been observed in the simulations that choosing A as an optimization variable and then searching for a gradient descent solution does not provide a convergence to an acceptable solution set of class centers and radii due to a piecewise cubic polynomial of the optimization variables for nonconstant As. Thus, there is a need a tool in determining the right value for A by considering the following facts:

1. Equal emphasis should be given to both the misclassification and the generalization errors.

2. A large value for A provides small radii and hence a large misclassification error. On the other hand, a small value for A provides large radii and hence a small misclassification error. As the number L of samples increases, L_{SC}^1 increases as well.

3. For the $y_m^i = 1$ class labels, L_{SC}^1 is a positive definite (piecewise) quadratic with respect to the c_m class center parameters, and it is a negative definite (piecewise) quadratic with respect to the R_m class radii parameters. For the $y_m^i = -1$ class labels, the reverse is true. When trying to minimize L_{SC}^1 by a gradient descent method, for the $y_m^i = 1$ class labels, the negative gradient direction forces the c_m vectors towards the centers of the classes and enlarges R_m; both help to reduce the misclassification error. If c_m and R_m are initialized properly, such that all of the cross-class misclassification terms, i.e. the terms in L_{SC}^1 related to the $y_m^i = -1$ class labels, are 0, then the gradient descent solution provides a movement on the c_m vectors towards the centers of the classes and it provides an enlargement on R_m. It should be noted that the mentioned proper choice would be obtained by choosing the means of the class samples as initial c_m vectors and by choosing a small R_m. The enlargement on R_m will be stopped when the cross-class misclassification terms become active. The second source of controlling the enlargement of R_m is the generalization cost L_{SC}^2, on which the application of the gradient descent reduces R_m. In

light of the above observations, a small value of A is chosen at the beginning and it is then computed iteratively as follows.

if $A < L$
  if $L_{SC}(k+1) < L_{SC}(k)$, $A^{k+1} = 0.01 + A^k$,
elseif $L_{SC}(k+1) > L_{SC}(k)$, $A^{k+1} = A^k - 0.01A^k$, *end*
end

3.3. Stage II: ellipsoidal classifiers

In this stage, the decision surface is defined as ellipsoids. For the $M \geq 1$ class problem, M ellipsoids with a inverse covariance matrix $\Sigma_m \in \Re^{nxn}$ and a center $c_m \in \Re^n$ are defined by:

$$E_m = \{x \in \Re^n | ((x^i - c_m)^T \Sigma_m (x^i - c_m) - 1) = 0, i = 1, ..., L, \quad m = 1, ..., M\}. \tag{23}$$

Similar to the SC stage, it is accepted that the ellipsoid includes the patterns belonging to the class labeled with +1 and the others are excluded:

$$((x^i - c_m)^T \Sigma_m (x^i - c_m) - 1) > 0 \quad y_m^i = -1, \tag{24}$$

$$((x^i - c_m)^T \Sigma_m (x^i - c_m) - 1) \leq 0 \quad y_m^i = 1. \tag{25}$$

In this stage, the minimum volume sphere problem, including the x^i data belonging to the class +1 and taking into consideration the classification error, is:

$$\min_{c_m, \Sigma_m} L_{EC} = \sum_{m=1}^{M} \sum_{i=1}^{L} f(y_m^i((x^i - c_m)^T \Sigma_m (x^i - c_m) - 1)) + \sum_{m=1}^{M} A/\sqrt{|\Sigma_m|}. \tag{26}$$

3.3.1. Solution method for ellipsoidal classifiers

The optimization problem in Eq. (26) can be solved using the GDALR method. First, the negative gradients of the objective function with respect to the c_m and Σ_m variables are taken as:

$$-\frac{\partial L_{EC}}{\partial \Sigma_m} = -\sum_{i=1}^{L} y_m^i (x^i - c_m)(x^i - c_m)^T f'(y_m^i((x^i - c_m)^T \Sigma_m (x^i - c_m) - 1)) + A(\Sigma_m^{-1})^T \Big/ 2\sqrt{|\Sigma_m|}, \tag{27}$$

$$-\frac{\partial L_{EC}}{\partial c_m} = 2\sum_{i=1}^{L} y_m^i \Sigma_m (x^i - c_m) f'(y_m^i((x^i - c_m)^T \Sigma_m (x^i - c_m) - 1)). \tag{28}$$

The obtained centers in the first stage are accepted as the center of the ellipsoids. The covariance matrices are initialized with matrices in which the diagonal elements are $1/R_m^2$ using the radii obtained from the first stage. Next, the Σ_m and c_m parameters are updated by:

$$\Delta c_m(k+1) = \alpha \Delta c_m(k) - (1-\alpha)\eta \frac{\partial L_{EC}}{\partial c_m}, \tag{29}$$

$$\Delta \Sigma_m(k+1) = \alpha \Delta \Sigma_m(k) - (1-\alpha)\eta \frac{\partial L_{EC}}{\partial \Sigma_m}. \tag{30}$$

The final decision is given by:

$$\ell_m(x) = sign(1 - (x^i - c_m)^T \Sigma_m (x^i - c_m)). \tag{31}$$

3.4. Stage III: classifiers based on RBF

For $(x^1, y_m^i), ..., (x^L, y_m^L), x^i \in \Re^n, y_m^i \in \{-1, 1\}$, the output of a CRBFC with r hidden units and 1 layer for M-class classification problems is given by:

$$\ell_m(x) = \sum_{j=1}^{r} w_{jm}\phi_{jm}(x) + b_m, \tag{32}$$

where w_{jm} is the real weight vector, b_m is the distance with respect to the origin, and $\phi_{jm}(x)$ is a spherical or elliptical Gaussian function with the center $c_{jm} \in R^n$ and width $\sigma_{jm} \in \Re$ or inverse covariance Σ_{jm}:

$$\phi_{jm}(x) = e^{\frac{-\left[x^i - c_{jm}\right]^T \left[x^i - c_{jm}\right]}{2(\sigma_{jm})^2}} \text{ or } \phi_{jm}(x) = e^{-0.5\left[x^i - c_{jm}\right]^T \Sigma_{jm} \left[x^i - c_{jm}\right]}. \tag{33}$$

The classification error functions, empirical risks, for conventional spherical RBF classifiers (CSRBFCs) and conventional elliptical RBF classifiers (CERBFCs) are respectively defined by:

$$\min_{w_{jm}, b_m, c_{jm}, \sigma_{jm}} L_{CSRBF} = \frac{1}{2} \sum_{m=1}^{M} \sum_{i=1}^{L} (y_m^i - (\sum_{j=1}^{r} w_{jm}(e^{\frac{-\left[x^i - c_{jm}\right]^T \left[x^i - c_{jm}\right]}{2(\sigma_{jm})^2}}) + b_m)))^2, \tag{34}$$

and

$$\min_{w_{jm}, b_m, c_{jm}, \sum_{jm}} L_{CERBF} = \frac{1}{2} \sum_{m=1}^{M} \sum_{i=1}^{L} (y_m^i - (\sum_{j=1}^{r} w_{jm}(e^{-0.5\left[x^i - c_{jm}\right]^T \sum_{jm} \left[x^i - c_{jm}\right]}) + b_m)))^2. \tag{35}$$

The structure of the CRBFCs is generally determined by different clustering methods. These methods generate one hidden neuron for each cluster [2]. Although they need fewer hidden neurons compared to the methods assigning a hidden neuron for each sample, the decision surfaces of M-class classification problems are not accurately modeled and so their classification performance, especially in test samples, is low.

In this paper, the CSRBFCs and CERBFCs initialized by the parameters obtained from the SC and EC stages are called SC-CSRBFCs and EC-CERBFCs, respectively. On the other hand, the CSRBFCs and CERBFCs initialized by the parameters obtained using the K-means function of MATLAB are called K-means-CSRBFCs and K-means-CERBFCs, respectively. The parameters of all of the CRBFCs are trained by the GDALR algorithm without a momentum term since the GDALR algorithm may increase the error in multiclass classification problems. CRBFCs can also be trained by other methods [30–32].

3.4.1. The proposed robust spherical and elliptical RBF classifiers

As one of the main contributions of the paper, novel classifiers based on RBF surfaces are introduced, which are designed by minimizing both the empirical and structural risk formulated using the penalty function approach for the complex surfaces that are not separated spherically or elliptically. The separating surface of the proposed

robust spherical RBF classifiers (RSRBFCs) is defined as:

$$D = \{x^i \in \Re^n | \sum_{j=1}^{r} w_{jm}(e^{\frac{-\left[x^i - c_{jm}\right]^T \left[x^i - c_{jm}\right]}{2(\sigma_{jm})^2}}) + b_m = 1, \quad j = 1, ..., r, i = 1, ..., L, m = 1, ..., M\}, \tag{36}$$

and the samples are labeled by:

$$\sum_{j=1}^{r} w_{jm}(e^{\frac{-\left[x^i - c_{jm}\right]^T \left[x^i - c_{jm}\right]}{2(\sigma_{jm})^2}}) + b_m \geq 1 \quad y_m^i = 1, \tag{37}$$

$$\sum_{j=1}^{r} w_{jm}(e^{\frac{-\left[x^i - c_{jm}\right]^T \left[x^i - c_{jm}\right]}{2(\sigma_{jm})^2}}) + b_m < 1 \quad y_m^i = -1. \tag{38}$$

The constraints are given by:

$$\left|w^T \phi(x)\right| \leq 1, \tag{39}$$

$$\left|w^T \phi(x)\right| \leq ||w||||\phi(x)|| \leq 1. \tag{40}$$

Since the volume of the enclosed decision surface is restricted by $||w||$, the norm of weights, the constraint problem is likely constructed of the 2-stage formulations:

$$\min_{w_{jm}, b_m, c_{jm}, \sigma_{jm}} L_{RSRBF} = \sum_{m=1}^{M} \sum_{i=1}^{L} f(y_m^i(1 - (\sum_{j=1}^{r} w_{jm}(e^{\frac{-\left[x^i - c_{jm}\right]^T \left[x^i - c_{jm}\right]}{2(\sigma_{jm})^2}}) + b_m))) + A \sum_{m=1}^{M} ||w_{jm}||^2, \tag{41}$$

where the penalty parameter is iteratively computed.

The optimization problem in Eq. (41) is solved using the GDALR method. The centers and radii obtained from the SC stage are initialized as the centers and radii of RBF bases. Next, the negative gradients with respect to w_{jm}, c_{jm}, σ_{jm}, and b_m are taken:

$$-\frac{\partial L_{RSRBF}}{\partial w_{jm}} = \sum_{i=1}^{L} f'(y_m^i(1 - (\sum_{j=1}^{r} w_{jm}(e^{\frac{-[x^i - c_{jm}]^T [x^i - c_{jm}]}{2(\sigma_{jm})^2}}) + b_m)))y_m^i(e^{\frac{-[x^i - c_{jm}]^T [x^i - c_{jm}]}{2(\sigma_{jm})^2}}) - Aw_{jm}, \tag{42}$$

$$-\frac{\partial L_{RSRBF}}{\partial c_{jm}} = \sum_{i=1}^{L} f'(y_m^i(1 - (\sum_{j=1}^{r} w_{jm}(e^{\frac{-[x^i - c_{jm}]^T [x^i - c_{jm}]}{2(\sigma_{jm})^2}}) + b_m)))y_m^i w_{jm}(e^{\frac{-[x^i - c_{jm}]^T [x^i - c_{jm}]}{2(\sigma_{jm})^2}})\frac{[x^i - c_{jm}]}{(\sigma_{jm})^2}, \tag{43}$$

$$-\frac{\partial L_{RSRBF}}{\partial \sigma_{jm}} = \sum_{i=1}^{L} f'(y_m^i(1 - (\sum_{j=1}^{r} w_{jm}(e^{\frac{-[x^i - c_{jm}]^T [x^i - c_{jm}]}{2(\sigma_{jm})^2}}) + b_m)))y_m^i w_{jm}(e^{\frac{-[x^i - c_{jm}]^T [x^i - c_{jm}]}{2(\sigma_{jm})^2}})\frac{[x^i - c_{jm}]^T [x^i - c_{jm}]}{(\sigma_{jm})^3}, \tag{44}$$

$$-\frac{\partial L_{RSRBF}}{\partial b_m} = \sum_{i=1}^{L} f'(y_m^i(1 - (\sum_{j=1}^{r} w_{jm}(e^{\frac{-[x^i - c_{jm}]^T [x^i - c_{jm}]}{2(\sigma_{jm})^2}}) + b_m)))y_m^i. \tag{45}$$

Finally, all of the variables are iteratively updated and the samples are labeled by:

$$\ell_m(x) = sign(-1 + (\sum_{j=1}^{r} w_{jm}(e^{\frac{-[x^i-c_{jm}]^T[x^i-c_{jm}]}{2(\sigma_{jm})^2}}) + b_m)). \tag{46}$$

On the other hand, the cost function for the proposed robust elliptical RBF classifiers (RERBFCs) is defined by:

$$\min_{w_{jm},b_m,c_{jm},\sum_{jm}} L_{RERBF} = \sum_{m=1}^{M}\sum_{i=1}^{L} f(y_m^i(1 - (\sum_{j=1}^{r} w_{jm}(e^{-0.5[x^i-c_{jm}]^T \sum_{jm}[x^i-c_{jm}]}) + b_m))) + A\sum_{m=1}^{M} ||w_{jm}||^2. \tag{47}$$

Here, the Σ_{jm}, w_{jm}, b_m, and c_{jm} parameters are determined by the GDALR algorithm. The centers and inverse covariance matrices obtained from the proposed elliptical classifier stage are used as the initial centers and inverse covariance matrices of RERBFC bases.

In addition, in this paper, robust diagonal elliptical RBF classifiers (RDERBFCs) are also designed with a diagonal inverse covariance matrix. The formulation is the same as the one given in Eq. (47), but the initial centers and covariance matrix parameters are chosen using the parameters obtained from the SC stage. Diagonal components of the inverse covariance are taken as the determined radii.

4. Numerical results

The proposed classifiers are tested on 9 datasets with different kinds of sample distributions, input space dimensions, and class numbers to carry out different evaluation practices: the Satimage, Shuttle (Statlog), Vowel, Liver, Pima, Ionosphere, and Iris plant datasets from the University of California Irvine Knowledge Discovery in Databases Archive [33], and the 2 Spirals and 4 Spirals datasets artificially arranged in [34]. Table 1 lists the characteristics of the 9 benchmark datasets used in the experiments. The training and testing data of the Satimage, Shuttle, Vowel, Iris, 2 Spirals, and 4 Spirals datasets were previously determined in the literature [5,34]. On the other hand, the training data of the Liver, Pima, and Ionosphere datasets are chosen randomly as 70% of all of the sample points and the remaining 30% are used for the testing data according to [27]. The process is repeated 10 times. The average results relating to the time, accuracy, and SV number of the datasets are given in all of the tables. Moreover, the Satimage and Shuttle datasets, rather large datasets, also have validation datasets [35]. The number of features determining the dimension of the input space of the Iris and Ionosphere datasets are 4 and 34, respectively. In this paper, only the third and fourth features of the Iris dataset are used for visualizing aims, similar to [36]. In addition, the second feature of the Ionosphere dataset is removed since it takes only a single value (0), similar to most of the studies in the literature [37].

All of the experiments are run on a personal notebook computer with 2.4-GHz Intel(R) Core(TM)2 Duo processors, 3 GB of memory, and Windows Vista operation system in a MATLAB environment. CRBFCs; MATLAB interface LIBSVM 2.83 [35] software implementing the sequential minimal optimization algorithm, decomposing the overall quadratic programming (QP) problem into QP subproblems; and SimpleMKL [27] using multiple kernels are used for comparison purposes. The optimal parameters of the LIBSVM are determined by 10-fold cross-validation. For the LIBSVM, all of the datasets are first normalized to be in [–1, 1], as in [35]. Using the 225 combinations of $1/(2\sigma^2) = [2^4 2^3 2^2 ... 2^{-10}]$ and $C = [2^{12} 2^{11} 2^{10} ... 2^{-2}]$, the parameters exhibiting the best validation accuracy are accepted as the optimal ones. In SimpleMKL, Gaussian kernels with 10 different

Table 1. Characteristics of the benchmark datasets.

Problem	≠ Input space	≠ Class	≠ Training data	≠ Testing data	≠ Validation data
Satimage	36	6	3104	2000	1331
Shuttle	9	7	30,450	14,500	13,050
Iris	2	3	75	75	-
Two Spirals	2	2	500	500	-
Four Spirals	2	2	1000	1000	-
Vowel	2	10	338	333	-
Liver	6	2	241	104	-
Pima	8	2	538	230	-
Ionosphere	33	2	246	105	-

bandwidths, σ, consisting of [0.5 1 2 5 7 10 12 15 17 20], and polynomial kernels of degree 1 to 3 are used, as in [27]. The best C value of SimpleMKL is searched for in 100 values of the interval [0.01, 1000] on a logarithmic scale for 2-class datasets. For multiclass datasets, the best C value is searched for in 5 different values, consisting of [1, 10, 100, 200, 1000], and the OVA method is used. For SimpleMKL, all of the datasets are normalized to a 0 mean and unity variance.

The results of the proposed classifiers are tabulated. The classifiers are compared in terms of their parameter selection time, training time, testing time, training accuracy, testing accuracy, and SV numbers. In our classifiers, small datasets are normalized to a 0 mean and unity variance but large datasets are normalized into [–1, 1], since this normalization exhibits a small number of clusters. It is reasonable to use the normalization of 0 mean and unity variance as possible for all of the datasets.

In the tables, the results related to 2 kinds of CRBFCs are given. In CRBFCs-1, the cluster parameters obtained by the SC stage, the EC stage, or K-means and the weights and bias parameters determined by the pseudoinverse are trained by GDALR without a momentum term. In CRBFCs-2, the determined cluster parameters are not trained but the weight and bias parameters are solely updated in order to save time. In addition, SC-CDERBFC-1 denotes the conventional diagonal elliptical RBF classifier (CDERBFC) initialized by the cluster parameters obtained by the SC stage in all of the tables.

In this paper, all of the SCs, including the RBF classifiers, are run for 80 epochs. On the other hand, all ECs are run for $2 \times 80/n$ epochs to save time by trusting the efficiency of the initial parameters. If the epoch number was increased, a small increase in accuracy might be observed. However, this case may not always appear due to the conditions causing it to stop the iteration.

Our classifiers have 8 advantages: 1) The parameter setting problem is not available since all of the parameters are automatically adjusted. 2) Even if the parameters obtained from the EC stage or the SC stage are used directly for the CSRBFC or the CERBFC without updating the parameters of c_{jm} and σ_{jm}, good results can be obtained, especially as seen in Tables 2–7. 3) The RSRBFC has a low training time and high testing accuracy, especially as seen in Tables 6 and 7. Tables 6 and 7 show that the parameter selection times of the RSRBFC and RERBFC are much less than those of the LIBSVM and SimpleMKL, even for small datasets, if the grid search is used for the selection of free parameters. 4) The testing time of the RSRBFC is consistently faster than that of the LIBSVM, especially on 2 large datasets, as seen in Tables 8 and 9. SimpleMKL could not be run for the Satimage and Shuttle datasets on our computer without using decomposition methods due to the large memory requirements and lengthy computations. 5) For small datasets, our robust classifiers, or the CRBFCs initialized by the EC stage or the SC stage, have a smaller parameter setting time, training time, and testing time than the other classifiers. 6) The testing accuracies of our robust classifiers are very close to

those of the others for all of the datasets. 7) The proposed surfaces can be visually interpreted. The spherical clusters obtained in the first stage and the elliptical clusters obtained in the second stage are presented in parts a and b of Figures 1–4 for the Vowel, Iris, 4 Spirals, and 2 Spirals datasets, respectively, while the 2D and 3D separating surfaces of the RERBFC are shown in parts c and d. The obtained ellipsoids on the 2D separating surfaces are also plotted, except in Figure 1. These figures exhibit the operation scheme of our classifiers. 8) Our robust classifiers have lower memory requirements. Memory usage can be separately evaluated for the initial parameter selection, training, and testing stages: in the initial parameter selection stage, the SCs (ECs) need to adjust the parameters of c_m, R_m (Σ_m), and A. In the training stage, RSRBFCs (RERBFCs) need to adjust the parameters of A, w_{jm},c_{jm}, σ_{jm} (Σ_{jm}), and b_m, while CSRBFCs (CERBFCs) need to adjust w_{jm}, c_{jm}, σ_{jm} (Σ_{jm}), and b_m. ECs and RERBFCs are more complex than SCs and RSCRBFCs, respectively, due to the need for updating their covariance matrices. RSRBFCs (RERBFCs) have one additional parameter compared to their conventional counterparts. In the update process, all of the parameters take the place of the previous ones at each run, so there is no need to store them. In the initial parameter selection stage of the LIBSVM, it is both trained and tested for 225 combinations of the $1/(2\sigma^2)$ and C parameters. In the training stage, the LIBSVM needs to adjust the λ_i parameters and some columns of the Hessian (kernel) matrix relating to LIBSVM are calculated and stored in the computer memory when needed. Hence, the developed robust classifiers require less memory than the LIBSVM at the initial parameter selection and training stages. On the other hand, in the initial parameter-determining stage of SimpleMKL, it is both trained and tested for many combinations of the C and σ parameters. In the training stage, SimpleMKL needs to adjust the λ_i and d_m parameters. The Hessian (kernel) matrix of SimpleMKL is calculated using the derivative of the dual variable with respect to weights d. Hence, the developed robust classifiers require less memory than SimpleMKL at the initial parameter determining and training stages. For the testing stage, fewer SV or cluster numbers require less memory. All of our classifiers have fewer SV numbers than the LIBSVM and SimpleMKL.

Table 2. Comparative results for the Liver dataset.

Classifiers	Selection time	Training time	Testing time	Training accuracy (%)	Testing accuracy (%)	Cluster/SV number
SC	0.826	2.23	-	-	-	39
EC	-	1.15	-	-	-	39
SC-CSRBFC-1	-	1.76	0	74.27	61.53	39
SC-CSRBFC-2	-	0.01	0	75.51	60.57	39
K-means-CSRBFC-1	-	1.60	0	75.10	66.34	39
K-means-CSRBFC-2	-	0.01	0	74.68	72.11	39
RSRBFC	-	1.62	0	80.91	65.38	39
EC-CERBFC-1	-	11.63	0.03	74.68	61.53	39
EC-CERBFC-2	-	0.01	0	74.68	61.53	39
K-means-CERBFC-1	-	11.65	0.03	60.16	56.73	39
K-means-CERBFC-2	-	0.01	0	57.67	57.69	39
RERBFC	-	9.62	0	77.17	68.26	39
SC-CDERBFC-1	-	10.92	0.03	74.68	60.57	39
RDERBFC	-	8.93	0	74.27	66.34	39
SimpleMKL, C = 86.97	164.10	26.18	0.0394	64.90	75.44	226.40
LIBSVM $1/(2\sigma^2) = 0.5$, C = 64	84.64	0.01	0.004	74.24	67.30	151

Table 3. Comparative results for the Pima dataset.

Classifiers	Selection time	Training time	Testing time	Training accuracy (%)	Testing accuracy (%)	Cluster/SV number
SC	0.1045	5.41	-	-	-	8
EC	-	2.38	-	-	-	8
SC-CSRBFC-1	-	5.36	0.001	74.59	74.19	8
SC - CSRBFC-2	-	0.10	0.001	74.39	74.15	8
K-means-CSRBFC-1	-	4.83	0	78.08	77.40	8
K-means-CSRBFC-2	-	0.05	0	77.22	76.92	8
RSRBFC	-	5.02	0.001	71.22	70.64	8
EC-CERBFC-1	-	11.59	0.004	74.07	74.06	8
EC-CERBFC-2	-	0.09	0.001	74.07	74.02	8
K-means-CERBFC-1	-	11.17	0.003	67.26	65.84	8
K-means-CERBFC-2	-	0.09	0	65.66	65.49	8
RERBFC	-	8.12	0	74.18	74.45	8
SC-CDERBFC-1	-	12.37	0.003	74.09	73.80	8
RDERBFC	-	8.89	0.004	74.13	73.67	8
SimpleMKL, C = 628	1543.60	107.45	0.29	98.92	76.62	456
LIBSVM $1/(2\sigma^2) = 0.0625$, C = 8	161.60	0.04	0.01	78.81	77.82	304

Table 4. Comparative results for the Ionosphere dataset.

Classifiers	Selection time	Training time	Testing time	Training accuracy (%)	Testing accuracy (%)	Cluster/SV number
SC	0.16	3.69	-	-	-	33
EC	0	1.64	-	-	-	33
SC-CSRBFC-1	0	3.82	0.006	97.91	91.69	33
SC-CSRBFC-2	0	0.05	0.004	97.91	91.50	33
K-means-CSRBFC-1	0	4.54	0.004	96.48	94.81	33
K-means-CSRBFC-2	0	0.10	0.003	96.40	95.66	33
RSRBFC	0	1.96	0	98.20	90.75	33
EC-CERBFC-1	0	6.27	0.01	97.95	91.50	33
EC-CERBFC-2	0	0.05	0.01	97.95	91.50	33
K-means-CERBFC-1	0	4.95	0.017	97.14	74.81	33
K-means-CERBFC-2	0	0.04	0.007	93.79	71.79	33
RERBFC	0	5.83	0.012	98.12	91.50	33
SC-CDERBFC-1	0	6.59	0.003	97.91	91.50	33
RDERBFC	0	4.81	0.003	98.16	91.50	33
SimpleMKL, C = 109.7	1375	113.33	0.028	95.10	91.42	80.5
LIBSVM $1/(2\sigma^2) = 0.1250$, C = 4	53.68	0.03	0.006	99.43	93.61	80.20

Table 5. Comparative results for the Iris dataset.

Classifiers	Selection time	Training time	Testing time	Training accuracy (%)	Testing accuracy (%)	Cluster/SV number
SC	0.21	0.32	-	-	-	3
EC	-	0.24	-	-	-	3
SC-CSRBFC-1	-	0.23	0	96.00	97.33	3
SC-CSRBFC-2	-	0	0	96.00	96.00	3
K-means-CSRBFC-1	-	0.32	0	96.00	97.33	3
K-means-CSRBFC-2	-	0	0	96.00	96.66	3
RSRBFC	-	0.43	0	97.33	97.33	3
EC-CERBFC-1	-	0.71	0	96.00	97.33	3
EC-CERBFC-2	-	0	0	96.00	97.33	3
K-means-CERBFC-1	-	0.62	0	81.33	81.33	3
K-means-CERBFC-2	-	0	0	78.66	72.00	3
RERBFC	-	0.60	0	97.33	97.33	3
SC-CDERBFC-1	-	0.73	0	96.00	97.33	3
RDERBFC	-	0.68	0	97.33	97.33	3
SimpleMKL, C = 100	58.89	7.07	0.015	96.00	96.00	25
LIBSVM $1/(2\sigma^2) = 0.5$, C = 4096	2.65	0.0013	9.2e-4	98.66	97.33	11

Table 6. Comparative results for the Vowel dataset.

Classifiers	Selection time	Training time	Testing time	Training accuracy (%)	Testing accuracy (%)	Cluster/SV number
SC	0.34	0.40	-	-	-	17
EC	-	0.43	-	-	-	17
SC-CSRBFC-1	-	0. 34	0	76.92	77.77	17
SC-CSRBFC-2	-	0	0	76.33	77.77	17
K-means-CSRBFC-1	-	0.51	0	76.92	78.57	17
K-means-CSRBFC-2	-	0	0	76.03	77.77	17
RSRBFC	-	0.37	0	76.33	79.07	17
EC-CERBFC-1	-	9.45	0	77.51	80.48	17
EC-CERBFC-2	-	0	0	77.21	80.78	17
K-means-CERBFC-1	-	9.26	0	38.46	41.74	17
K-means-CERBFC-2	-	0	0	47.04	45.34	17
RERBFC	-	0.43	0	76.92	80.48	17
SC-CDERBFC-1	-	9.09	0	77.21	79.57	17
RDERBFC	-	0.92	0	77.21	79.27	17
SimpleMKL, C = 200	217.35	44.94	0.004	80.23	80.48	746
LIBSVM $1/(2\sigma^2) = 0.5$, C = 16	53.07	0.01	0.01	78.40	80.18	226

Table 7. Comparative results for the 4 Spirals dataset.

Classifiers	Selection time	Training time	Testing time	Training accuracy (%)	Testing accuracy (%)	Cluster/SV number
SC	0.15	0.35	-	-	-	10
EC	-	0.42	-	-	-	10
SC-CSRBFC-1	-	0.31	0	91.80	90.40	10
SC-CSRBFC-2	-	0	0	89.60	86.00	10
K-means-CSRBFC-1	-	0.29	0	81.40	76.20	10
K-means-CSRBFC-2	-	0	0	85.60	78.00	10
RSRBFC	-	0.49	0	96.80	94.80	10
EC-CERBFC-1	-	8.25	0	91.80	89.20	10
EC-CERBFC-2	-	0	0	90.00	85.60	10
K-means-CERBFC-1	-	8.23	0	79.00	71.00	10
K-means-CERBFC-2	-	0	0	75.80	69.20	10
RERBFC	-	6.86	0	95.80	94.60	10
SC-CDERBFC-1	-	8.22	0	90.00	85.40	10
RDERBFC	-	0.76	0	92.40	91.40	10
SimpleMKL, C = 1000	373.73	1.82	0.01	0.93	90.67	118
LIBSVM $1/(2\sigma^2) = 16$, C = 4096	82.80	0.02	0.003	99.60	96.80	58

Table 8. Comparative results for the Shuttle dataset.

Classifiers	Selection time	Training time	Testing time	Training accuracy (%)	Testing accuracy (%)	Cluster/SV number
SC	8.14	63.46	-	-	-	17
EC	-	36.72	-	-	-	17
SC-CSRBFC-1	-	76.61	0.28	99.42	99.47	17
SC-CSRBFC-2	-	0.06	0.28	99.44	99.48	17
K-means-CSRBFC-1	-	72.65	0.29	98.71	98.77	17
K-means-CSRBFC-2	-	0.06	0.28	98.45	98.52	17
RSRBFC	-	71.23	0.28	99.57	99.58	17
EC-CERBFC-1	-	355.91	1.49	99.45	99.50	17
EC-CERBFC-2	-	30.54	1.46	99.34	99.39	17
K-means-CERBFC-1	-	353.48	1.51	83.28	83.80	17
K-means-CERBFC-2	-	0.10	1.88	78.44	79.08	17
RERBFC	-	68.68	1.49	99.34	99.39	17
SC-CDERBFC-1	-	247.24	1.31	99.35	99.39	17
RDERBFC	-	62.54	1.38	99.35	99.39	17
LIBSVM $1/(2\sigma^2) = 8$, C = 2048	[5]	7.06	1.99	99.72	99.71	301

Table 9. Comparative results for the Satimage dataset.

Classifiers	Selection time	Training time	Testing time	Training accuracy (%)	Testing accuracy (%)	Cluster/SV number
SC	2.96	24.30	-	-	-	281
EC	-	39.37	-	-	-	281
SC-CSRBFC-1	-	33.94	0.26	95.23	90.25	281
SC-CSRBFC-2	-	0.21	0.18	99.80	89.35	281
K-means-CSRBFC-1	-	33.61	0.26	92.75	90.20	281
K-means-CSRBFC-2	-	0.15	0.38	98.67	90.10	281
RSRBFC	-	15.05	0.26	95.23	89.90	281
EC-CERBFC-1	-	156.39	3.32	94.97	89.75	281
EC-CERBFC-2	-	0.25	125.50	99.80	79.20	281
K-means-CERBFC-1	-	102.69	3.32	46.19	46.10	281
RERBFC	-	48.76	3.32	94.97	89.75	281
SC-CDERBFC-1	-	124.97	2.82	94.97	89.75	281
RDERBFC	-	41.35	2.85	94.97	89.75	281
LIBSVM $1/(2\sigma^2) = 1$, C = 16	551.24	1.35	0.41	99.51	90.35	1219

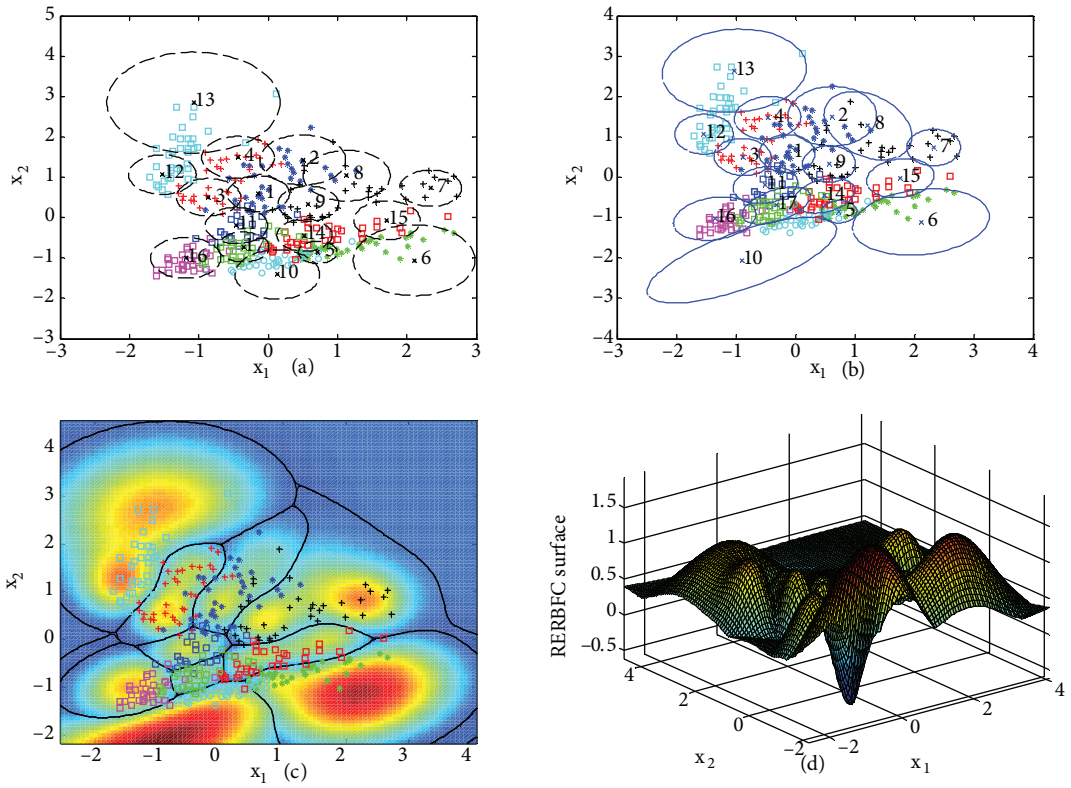

Figure 1. Results relating to the Vowel dataset: a) spherical clusters, b) elliptical clusters, c) 2D decision surface of the RERBFC, d) 3D decision surface of the RERBFC.

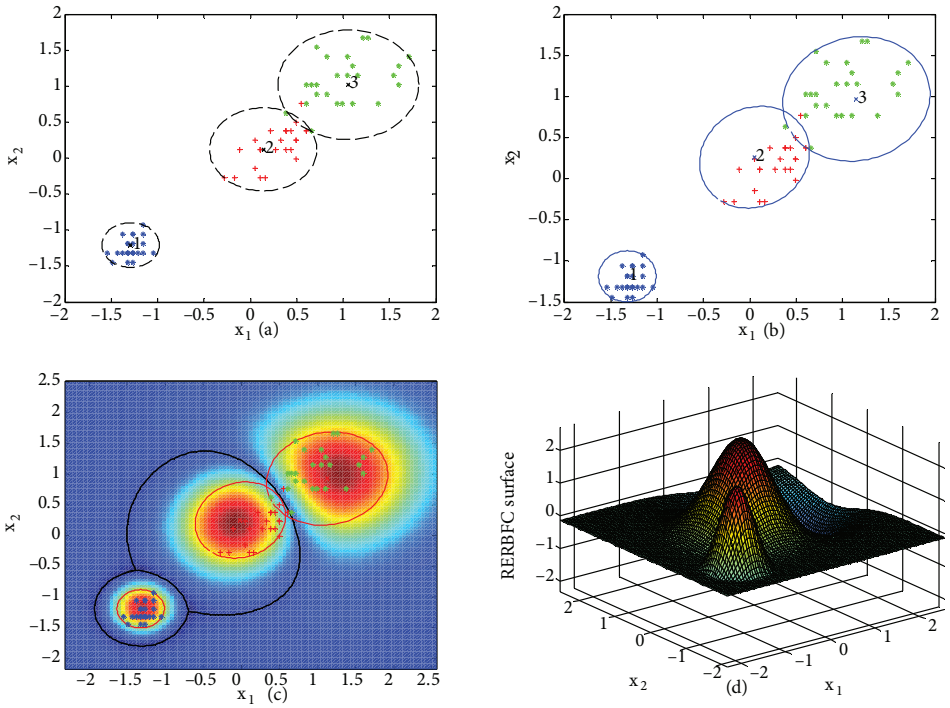

Figure 2. Results relating to the Iris dataset: a) spherical clusters, b) elliptical clusters, c) 2D decision surface of the RERBFC, d) 3D decision surface of the RERBFC.

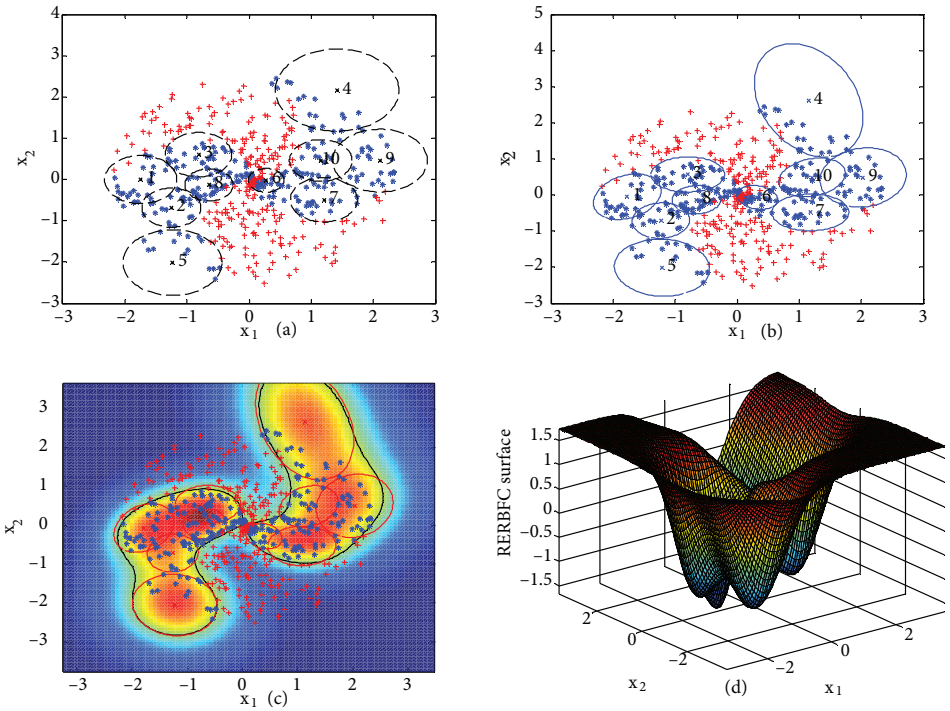

Figure 3. Results relating to the 4 Spirals dataset: a) spherical clusters, b) elliptical clusters, c) 2D decision surface of the RERBFC, d) 3D decision surface of the RERBFC.

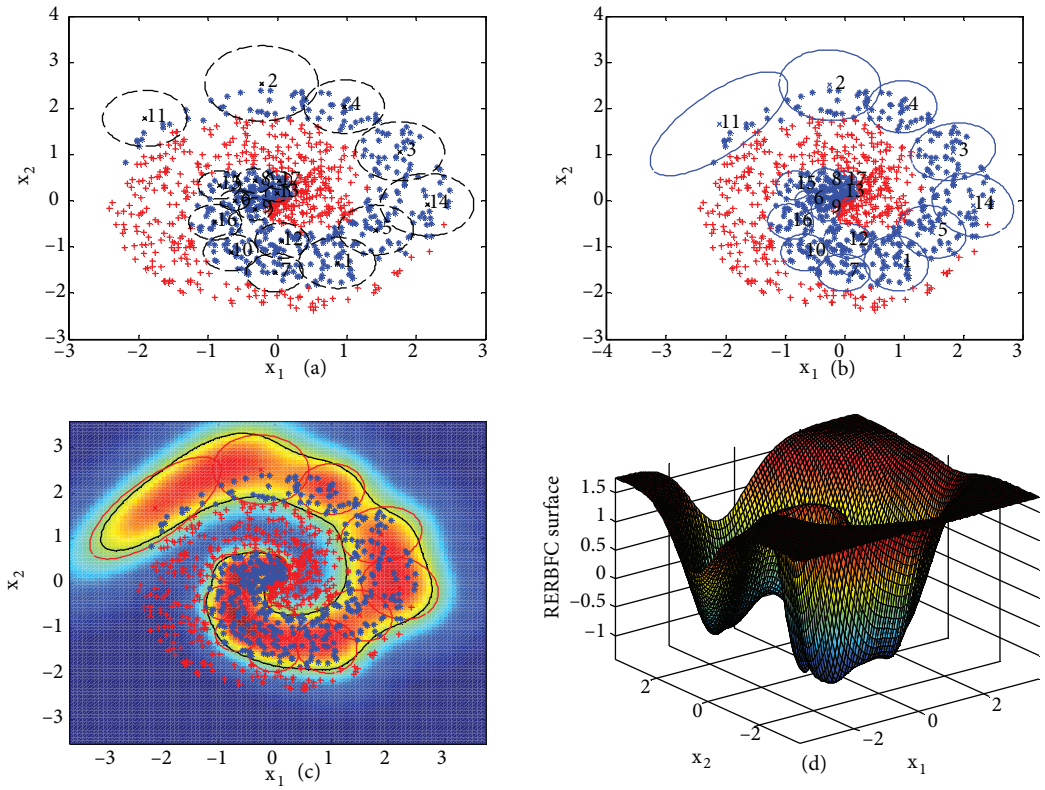

Figure 4. Results relating to the 2 Spirals dataset: a) spherical clusters, b) elliptical clusters, c) 2D decision surface of the RERBFC, d) 3D decision surface of the RERBFC.

On the other hand, our classifiers have some limitations: 1) Determination of the cluster number may not be optimal. In this paper, a predetermined quantization error value produces good results for all of the datasets. The predetermined value can be changed, but the results do not represent a big difference. If the predetermined value is reduced, the cluster number increases. In that case, the testing accuracy might be less as the training accuracy increases and the unsmooth surfaces are generated. On the other hand, if the predetermined value is increased, the same testing accuracy might be obtained as the cluster number reduces. For example, when increasing this value for the 2 Spirals dataset, 10 clusters are generated and similar results to those of 17 clusters given in Table 10 are obtained. 2) The parameter selection stage of the LIBSVM and SimpleMKL, and the training stage of the RERBFCs, need a lot more time and a larger memory for very large datasets. In this paper, very large datasets are not examined due to the increase in the processor heat of our computer. Our algorithms must be specifically developed for very large datasets. On the other hand, the results relating to the parameter selection stage of the LIBSVM could not be obtained using our computer, even for the Shuttle dataset. Hence, the LIBSVM parameters in [5] were used. 3) In addition, our algorithms are sensitive to the initial parameters since the SC stage is initialized by the K-means algorithm. Testing accuracy can vary up to ± 3 according to the tabulated results.

5. Conclusions

In this paper, a set of alternative classifiers to SVMs are designed. The contributions of the proposed classifiers are 6-fold: 1) The data structure is preserved by operating in the input space. 2) Classification is carried out

Table 10. Comparative results for the 2 Spirals dataset.

Classifiers	Selection time	Training time	Testing time	Training accuracy (%)	Testing accuracy (%)	Cluster/SV number
SC	0.24	0.54	-	-	-	17
EC	-	0.70	-	-	-	17
SC-CSRBFC-1	-	0.54	0	95.70	95.90	17
SC-CSRBFC-2	-	0	0	94.90	93.30	17
K-means-CSRBFC-1	-	0.53	0	84.20	83.00	17
K-means-CSRBFC-2	-	0	0	85.00	84.00	17
RSRBFC	-	0.67	0	98.80	97.20	17
EC-CERBFC-1	-	27.08	0	96.10	95.70	17
EC-CERBFC-2	-	0	0	96.10	93.80	17
K-means-CERBFC-1	-	27.17	0	63.60	63.60	17
K-means-CERBFC-2	-	0	0	58.70	56.00	17
RERBFC	-	10.74	0	98.40	97.20	17
SC-CDERBFC-1	-	26.14	0	95.10	93.50	17
RDERBFC	-	2.07	0	99.40	97.50	17
SimpleMKL, C = 394.42	3387.60	21.60	0.11	95.75	94.90	273
LIBSVM $1/(2\sigma^2) = 16$, C = 1024	300.66	0.05	0.008	98.60	97.30	67

by easily interpretable decision surfaces, such as sphere and ellipsoid. 3) The newly proposed classifiers are based on both the structural and empirical risk. 4) Robustness is obtained by softly rejecting outliers and noise, thanks to the penalty approach, without using extra slack variables or Lagrange multipliers. 5) The proposed classifiers have small testing and parameter selection times compared to SVMs. 6) The proposed classifiers can be applied to moderately large datasets without initialization problems and big memory requirements, thanks to the first-order gradient descent methods. Extensions of the algorithms to classify very large datasets are promising areas of future research.

References

[1] Steinwart I, Christmann A. Support Vector Machines. New York, NY, USA: Springer-Verlag, 2008.

[2] Haykin S. Neural Networks and Learning Machines. 3rd ed. Upper Saddle River, USA: Prentice Hall, 2008.

[3] Bertsekas DP. Nonlinear Programming. 2nd ed. Belmont, MA, USA: Athena Scientific, 1999.

[4] Cortes C, Vapnik VN. Support vector networks. Mach Learn 1995; 20: 273–297.

[5] Hsu CW, Lin CJ. A comparison of methods for multi-class support vector machines. IEEE T Neural Networ 2002; 13: 415–425.

[6] Mayoraz E, Alpaydın E. Support vector machines for multi-class classification. In: IWANN Conference; 2–4 June 1999; Alicante, Spain. pp. 833–842.

[7] Platt JC, Cristianini N, Shawe-Taylor J. Large margin DAG's for multiclass classification. In: Solla SA, Leen TK, Müller KR, editors. Advances in Neural Information Processing Systems. Cambridge, MA: MIT Press, 2000. pp. 547–553.

[8] Weston J, Watkins C. Support vector machines for multi-class pattern recognition. In: The 7th European Symposium on Artificial Neural Networks; 21–23 April 1999; Bruges, Belgium. pp. 219–224.

[9] Chapelle O, Vapnik V, Bousquet O, Mukherjee S. Choosing multiple parameters for support vector machines. Mach Learn 2002; 46: 131–159.

[10] Keerthi SS. Efficient tuning of SVM hyperparameters using radius/margin bound and iterative algorithm. IEEE T Neural Networ 2002; 13: 1225–1229.

[11] Joachims T. Making large-scale SVM learning practical. In: Schölkopf B, Burges CJC, Smola AS, editors. Advances in Kernel Methods–Support Vector Learning. Cambridge, MA: MIT Press, 1999. pp. 169–184.

[12] Doğan H, Güzeliş C. Robust and fuzzy spherical clustering by a penalty parameter approach. IEEE T Circuits II 2006; 53: 637–641.

[13] Mao KZ, Huang G. Neuron selection for RBF neural network classifier based on data structure preserving criterion. IEEE T Neural Networ 2005; 16: 1531–1540.

[14] Ayat NE, Cheriet M, Remaki L, Suen CY. KMOD–a new support vector machine kernel with moderate decreasing for pattern recognition. Application to digit image recognition. In: The 6th International Conference on Document Analysis and Recognition; 10–13 September 2001; Quebec, Canada. pp. 1215–1219.

[15] Reilly DL, Cooper LN, Elbaum C. A neural model for category learning. Biol Cybern 1982; 45: 35–41.

[16] Marchand M, Shawe-Taylor J. The set covering machine. J Mach Learn Res 2002; 3: 723–746.

[17] Rosen JB. Pattern separation by convex programming. J Math Anal Appl 1965; 10: 123–134.

[18] Barnes R. An algorithm for separating patterns by ellipsoids. IBM J Res Dev 1982; 26: 759–764.

[19] Tax D, Duin R. Support vector domain description. Pattern Recogn Lett 1999; 20: 1191–1199.

[20] Glineur F. Pattern Separation Via Ellipsoids and Conic Programming. Mons, Belgium: Mémoire de DEA, Faculté Polytechnique de Mons, 1998.

[21] Astorino A, Fuduli A, Gaudioso M. Margin maximization in spherical separation. Comput Optim Appl 2012; 53: 301–322.

[22] Okada Y, Konno H. Failure discrimination by semi-definite programming using a maximal margin ellipsoidal surface. J Comput Financ 2009; 12: 63–77.

[23] Potra FA, Liu X. Pattern separation and prediction via linear and semi-definite programming. Stud Inform Control 2009; 18: 71–82.

[24] Hao PY, Chiang JH, Lin YH. A new maximal-margin spherical-structured multi-class support vector machine. App Intell 2009; 30: 98–111.

[25] Gotoh JY, Takeda A. Conditional minimum volume ellipsoid with application to multiclass discrimination. Comput Optim Appl 2008; 41: 27–51.

[26] Kharechko A, Shawe-Taylor J. Text categorization via ellipsoid separation. In: Learning Methods for Text Understanding and Mining Workshop; 26–29 January 2004; Grenoble, France. pp. 19–20.

[27] Rakotomamonjy A, Bach F, Canu S, Grandvalet Y. SimpleMKL. J Mach Learn Res 2008; 9: 2491–2521.

[28] Duin RPW, Juszczak P, Ridder Dde, Paclík P, Pekalska E, Tax DMJ. PR-Tools, A MATLAB Toolbox for Pattern Recognition. http://www.prtools.org, 2004.

[29] Chen Z, Haykin S. On different facts of regularization theory. Neural Comput 2002; 14: 2791–2846.

[30] Mahdi RN, Rouchka EC. Reduced HyperBF Networks: regularization by explicit complexity reduction and scaled Rprop-based training. IEEE T Neural Networ 2011; 22: 673–686.

[31] Tiantian X, Yu H, Hewlett J, Rozycki P, Wilamowski B. Fast and efficient second-order method for training radial basis function networks. IEEE T Neural Networ 2012; 23: 609–619.

[32] Karayiannis NB. Reformulated radial basis neural networks trained by gradient descent. IEEE T Neural Networ 1999; 10: 657–671.

[33] Blake CL, Merz CJ. UCI Repository of Machine Learning Databases. Irvine, CA, USA: Department of Information and Computer Science, University of California, Irvine, 1998. http://archive.ics.uci.edu/ml/data sets.html.

[34] Duda RO, Hart PE, Stork DG. Pattern Classification. 2nd ed. New York, NY, USA: Wiley, 2000.

[35] Chang CC, Lin CJ. LIBSVM: a library for support vector machines. ACM Trans Intell Syst Technol 2011; 2: 1–27.

[36] Gunn SR. Support Vector Machines for Classification and Regression. Technical Report. Southampton, UK: University of Southampton, 1998.

[37] Chena HPC, Lee KY, Lee TJ, Lee YJ, Huang SY. Multiclass support vector classi?cation via coding and regression. Neurocomputing 2010; 73: 1501–1512.