

Evolutionary strategies for hyperparameters of support vector machines based on multi-scale radial basis function kernels

Tanasanee Phienthrakul · Boonserm Kijisirikul

Published online: 9 June 2009
© Springer-Verlag 2009

Abstract Kernel functions are used in support vector machines (SVM) to compute inner product in a higher dimensional feature space. SVM classification performance depends on the chosen kernel. The radial basis function (RBF) kernel is a distance-based kernel that has been successfully applied in many tasks. This paper focuses on improving the accuracy of SVM by proposing a non-linear combination of multiple RBF kernels to obtain more flexible kernel functions. Multi-scale RBF kernels are weighted and combined. The proposed kernel allows better discrimination in the feature space. This new kernel is proved to be a Mercer's kernel. Furthermore, evolutionary strategies (ESs) are used for adjusting the hyperparameters of SVM. Training accuracy, the bound of generalization error, and subset cross-validation on training accuracy are considered to be objective functions in the evolutionary process. The experimental results show that the accuracy of multi-scale RBF kernels is better than that of a single RBF kernel. Moreover, the subset cross-validation on training accuracy is more suitable and it yields the good results on benchmark datasets.

Keywords Support vector machines · Evolutionary strategies · Kernel methods

1 Introduction

Support vector machines (SVMs) are learning algorithms proposed by Vapnik (1995, 1998), which are based on the idea of empirical risk minimization principle. They have been widely used in many applications such as pattern recognition and function approximation. Basically, SVM performs a linear separation in an augmented space by means of a pre-defined kernel function that satisfies Mercer's condition (Vapnik 1995; Schölkopf et al. 1998; Ayat et al. 2001). This kernel maps the input vectors into a very high dimensional space, possibly of infinite dimension, where a linear separation is more probable (Ayat et al. 2001). Then, a linear separating hyperplane is created by maximizing the margin between two classes in this space. Hence, the complexity of the separating hyperplane depends on the nature and the properties of the chosen kernel (Ayat et al. 2001).

The goal of this paper is to improve the SVM accuracy on classification problems. A kernel function is an important part in SVM that affects classification performance. There are many types of kernel functions such as linear kernels, polynomial kernels, sigmoid kernels, and RBF kernels. Each kernel function is suitable for different tasks, and it must be chosen for the task under consideration by hand or using prior knowledge. The RBF kernel is one of the most effective kernel functions for many problems, but still has restrictions in some complex problems. Therefore, we propose to improve the performance of the classification by using the combination of RBF kernels at different scales. These kernels are weighted and combined. The weights, the widths of the RBF kernels, and the regularization parameter of SVM are called hyperparameters.

In order to obtain an SVM that has good classification accuracy, a large number of the hyperparameters are needed

T. Phienthrakul · B. Kijisirikul (✉)
Department of Computer Engineering, Chulalongkorn
University, 254 Phyathai, Pathumwan, Bangkok 10330, Thailand
e-mail: boonserm.k@chula.ac.th

T. Phienthrakul
e-mail: tanasanee@yahoo.com

to be evaluated. In general, a good set of hyperparameters are usually determined by a grid search. During the search, the hyperparameters are varied by a fixed step-size in the range of possible values, but this kind of search consumes a lot of time. Hence, we propose to use evolutionary strategies (ESs) for choosing these hyperparameters. An objective function is an important part in an evolutionary algorithm, and there are many ways to measure the fitness of the hyperparameters. In this work, we consider training accuracy, the bound of generalization error, and subset cross-validation on training accuracy, as objective functions for evaluating the hyperparameters in the evolutionary process.

We give a short description of support vector machines, evolutionary strategies, and related works in Sect. 2. In Sects. 3 and 4, we propose a multi-scale RBF kernel and apply evolutionary strategies to determine the appropriate hyperparameters, respectively. Three possible objective functions of the evolutionary process are proposed in Sect. 5. We evaluate the proposed kernel and three objective functions for ES in Sect. 6. The results show that ES can effectively identify a good set of hyperparameters when a suitable objective function is used. Finally, the conclusions and the discussions are given in Sect. 7.

2 Background and related works

2.1 Support vector machines

A support vector machine is a classifier which finds an optimal separating hyperplane. It is one of the latest and most successful statistical pattern classifiers that utilizes the kernel technique (Vapnik 1995). For simple pattern recognitions, SVM uses a linear separating hyperplane to create a classifier with the maximum margin (Cristianini and Shawe-Taylor 2000; Kecman 2001; Müller et al. 2001). Consider the problem of binary classification, where a training dataset is denoted as $(x_1, y_1), (x_2, y_2), \dots, (x_l, y_l), x_i \in R^N$, is a sample data and $y_i \in \{-1, 1\}$ is its label (Shawe-Taylor and Cristianini 2004). A linear decision surface is defined by the following equation:

$$w \cdot x + b = 0. \tag{1}$$

Occasionally, there are multiple hyperplanes which can perform the separation. The goal of learning is to find $w \in R^N$ and the scalar b such that the margin between positive and negative examples is maximized. An example of the decision surface and its margin is shown in Fig. 1.

In soft margin SVM, this surface can be achieved by minimizing $\frac{1}{2} \|w\|^2 + C \sum_{i=1}^l \xi_i$ subject to the constraints $y_i((w \cdot x_i) + b) \geq 1 - \xi_i$ where $\xi_i \geq 0$ for all $i = 1, \dots, l$ (Schölkopf and Smola 2002). The width of the soft margin

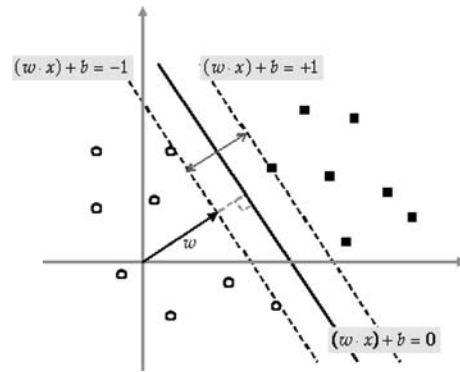


Fig. 1 An example of decision surface and margin

can be controlled by the corresponding regularization parameter C (Kecman 2001). The constant $C > 0$ determines the trade-off between margin maximization and training error minimization (Schölkopf and Smola 2002). These lead to the following decision function

$$f(x) = \text{sign} \left(\sum_{i=1}^l \alpha_i y_i x_i x + b \right). \tag{2}$$

where the α_i for $i = 1, \dots, l$ are the solution of the following quadratic optimization problem:

$$\text{Maximize } \sum_{i=1}^l \alpha_i - \frac{1}{2} \sum_{i,j=1}^l \alpha_i \alpha_j y_i y_j x_i x_j \tag{3a}$$

$$\text{subject to } 0 \leq \alpha_i \leq C \text{ for all } i = 1, \dots, l, \text{ and } \sum_{i=1}^l \alpha_i y_i = 0. \tag{3b}$$

A data example x_i which corresponds to a non-zero α_i value is called support vector.

In most cases, seeking a proper linear hyperplane in the original input space is not always possible. This problem can be solved by enabling these support vector machines to produce complex non-linear boundaries in the original space. This is done by mapping the input space into a higher dimensional feature space through a mapping function Φ . Then a linear separating is performed in the higher dimensional space (Schölkopf and Smola 2002). The mapping is achieved by substituting $\Phi(x_i)$ into each training example x_i .

However, one good property of SVM is that it is not required the explicit form of Φ . Only the inner product in the feature space, called a kernel function $K(x, y) = \Phi(x) \cdot \Phi(y)$, must be defined. The decision function then becomes the following equation:

$$f(x) = \text{sign} \left(\sum_{i=1}^l \alpha_i y_i K(x_i, x) + b \right), \tag{4}$$

Table 1 Common kernel functions

Kernel	Formula
Linear	$K(x, y) = x \cdot y$
Homogeneous polynomial	$K(x, y) = (x \cdot y)^d$
Inhomogeneous polynomial	$K(x, y) = (x \cdot y + c)^d$
Exponential RBF	$K(x, y) = \exp(-\gamma \ x - y\)$
Gaussian RBF	$K(x, y) = \exp(-\gamma \ x - y\ ^2)$
Multi-quadratic	$K(x, y) = -\sqrt{\ x - y\ ^2 + c^2}$

where $\alpha_i \geq 0$ is a coefficient associated with a support vector x_i and b is an offset. A function which can be a kernel must satisfy Mercer’s condition (Shawe-Taylor and Cristianini 2004). Some of the common kernels are shown in Table 1.

Each kernel corresponds to some feature space and because no explicit mapping to the feature space occurs, optimal linear separators can be found efficiently in the feature space with millions of dimensions (Russell and Norvig 2003).

2.2 Evolutionary strategies

The evolutionary strategy (ES) is one of the major branches of evolutionary algorithms, which was developed by Rechenberg and Schwefel (Rechenberg 1965, 1973; Schwefel 1981, 1995; Beyer and Schwefel 2002) at the Technical University of Berlin and has been extensively studied in Europe. ES was developed in order to conduct successive wing tunnel experiments for aerodynamic shape optimization, and it has been used successfully to solve various types of optimization problems. ES is an algorithm that imitates the natural processes (a natural selection and the survival-of-the-fittest principle). ES uses a fixed-length real-valued vector as a representation of a solution, which makes it significantly faster than traditional genetic algorithms that use a binary representation (Beyer and Schwefel 2002; Goldberg 1989; Fogel 1995). The simple ES algorithm is shown in Fig. 2.

Simple ES algorithm:

- Step 1* Randomly generate a parent population of μ solutions.
- Step 2* Evaluate all parents to determine their fitness.
- Step 3* Apply reproduction operators to create λ offspring.
- Step 4* Evaluate and keep the μ fittest individuals.
- Step 5* Repeat step 3 and step 4 until an acceptable solution has been found or a pre-defined number of generations has been produced and evaluated.

Fig. 2 Simple ES algorithm

Every point in the search space is an individual (a potential solution). ES uses a population of μ individuals to conduct the search for possibly better solutions (deDoncker et al. 1996). The initial population of individuals is randomly generated but, ideally, should be uniformly distributed throughout the search space so that all regions can be explored. Each individual in each generation (iteration) is evaluated to determine its fitness. The goal of the search is to find individuals with high fitness values.

The main reproduction operator in evolutionary strategies is Gaussian mutation. Another operator that can be used is intermediate recombination, in which the vectors of two parents are averaged to form a new offspring. Each generation of the ES algorithm takes a population of individuals and modifies them with reproduction and mutation operators to produce offspring (new solutions). Both the parents and the offspring are evaluated but only the fittest individuals (better solutions) survive and become new generation. Poorer solutions are dropped (deDoncker et al. 1996).

In each iteration, individual are recombined and then mutated to produce offspring. This means that ES simultaneously investigates several regions of the search space, which greatly decreases the amount of time required to locate good solutions. ES terminates after a pre-defined number of generations have been produced and evaluated, or earlier if the acceptance criterion is reached (deDoncker et al. 1996).

There are several variations of ES. The $(\mu + \lambda)$ -ES and (μ, λ) -ES are two common variations. In the former, μ parents produce λ offspring. The parents and the offspring compete equally for survival. In the latter, μ parents produce $\lambda > \mu$ offspring, but only the μ best offspring survive. Thus the lifespan of any solution is only one generation (deDoncker et al. 1996). In Sect. 4, we discuss how $(\mu + \lambda)$ -ES is applied in our proposed kernel.

2.3 Related works

There are many research studies that propose new kernel functions for support vector machines. In research of Ayat et al. (2001), a new SVM kernel family, kernel function with moderate decreasing (KMOD), was proposed. This research explained the distinctive properties that allow better discrimination in feature space. Their experimental results showed that their kernel function was better than RBF kernels and exponential RBF kernels on a spiral problem. In addition, a digital recognition task was processed using this new kernel. The results show comparable performance to state-of-the-art kernels. In Zhang et al. (2000), scaling kernels were introduced for SVM. These kernels are multi-dimensional scaling functions with translation vectors. SVMs that use the scaling kernels can

approximate any objective function in some space. In Zhang et al. (2004), a wavelet kernel was proposed. This kernel is a multi-dimensional wavelet function that can approximate arbitrary non-linear functions. The existence of wavelet kernels was proved by the result of theoretic analysis. Computer simulations showed the feasibility and validity of wavelet support vector machines in regression and pattern recognition tasks.

Examples of kernel functions that have been proposed are asymmetric kernel (Koji 1999), time-alignment kernel (Hiroshi et al. 2001), triangular kernel (Fleuret and Sahbi 2002), and hyperkernels (Ong et al. 2005). These kernels are suitable for different applications and datasets. Although these enhanced kernel functions yield better classification results, they are not widely used in practical applications when compared to simple kernels such as linear, polynomial, and RBF kernels. The research of Rameswar and Haruhisa (2004) reviewed that several researchers used the RBF kernel in their applications. According to various experimental results, the RBF kernel has the best performance. Moreover, Rameswar and Haruhisa (2004) tried to improve only the polynomial kernel to compare with the RBF kernel, whereas other kernels were not considered.

However, there are a few research studies that consider combining well-known kernels together. The research of Smits and Jordaan (2002) showed that the RBF kernel has a better interpolation property while the polynomial kernel has a better extrapolation property. Therefore, both kernel functions are combined using mixtures. In Howley and Madden (2005), genetic programming is used to create a kernel for an SVM classifier, but this approach does not guarantee that the resulted kernel obeys Mercer's theorem. In contrast, our approach presented in this paper uses the combination of kernel functions that is represented in the form of the multi-scale RBF kernel which satisfies the Mercer's theorem.

There are also many research studies that attempt to solve the problem of parameter selection for SVM by using meta-heuristic methods. In a work of Chapelle et al. (2002), an algorithm that alternates the SVM optimization with a gradient step was employed. Although this algorithm is useful and accurate, there are a lot of details in the computation that make the algorithm quite complex. The algorithm requires gradient computation which is either not possible or at least very difficult for general kernel functions. Moreover, the gradient descent may get stuck in the local optima.

Most of the research studies on parameter selection use evolutionary algorithms such as a genetic algorithm (GA) and an evolutionary strategy (ES). Eads et al. (2002) proposed the use of the genetic algorithm and support vector machines for time series classification problems. They introduced a hybrid algorithm that employs evolutionary

computation for feature extraction, and a support vector machine for classification. They evaluated the proposed algorithm on a lightning classification task. It yielded better results in terms of a cross-validation fitness measure, though the difference was not large. In Fröhlich et al. (2003), a special genetic algorithm was proposed to solve a feature selection problem which is a difficult combinatorial task in machine learning. They optimized kernel parameters such as the regularization parameter of SVM by means of genetic algorithms. Xuefeng and Fang (2002) and Chunhong and Licheng (2004) proposed other research works that used real-coded GA for SVM parameter selection.

Many research studies use the evolutionary strategies for model selection. In research of deDoncker et al. (1996), the evolutionary strategies were used for computing the solutions of multivariate integration problems. Adaptive integration algorithms and evolutionary strategies can be parallelized easily. Moreover, the covariance matrix adaptation evolution strategy (CMA-ES) in Friedrichs and Igel (2004) was proposed to determine the kernel from a parameterized kernel space and to control the regularization. The ES method proposed in that paper is simpler; the random process is used to find the optimal hyperparameter, and only recombination and mutation methods are used to create new solutions. Their experiments on benchmark datasets showed that ES achieved better results than the grid search and can handle much more kernel parameters.

There are other evolutionary strategy research studies on model selection for support vector machines such as Runarsson and Sigurdsson (2004) and Igel (2005). Both of these studies attempted to use the evolutionary strategies for optimizing parameters of SVMs. Runarsson and Sigurdsson (2004) proposed the asynchronous parallel evolution strategy for SVM model selection, and Igel (2005) proposed the use of the multi-objective in the evolutionary algorithm. The evolutionary strategies were successfully applied to their applications and datasets. Therefore, the evolutionary strategy is an interesting algorithm for parameter selection in our research.

The model-based global optimization (Fröhlich and Zell 2005) was proposed to deal with the model selection problems. This research is based on the idea of learning an online Gaussian process using a sampling technique to search for the solutions in the parameter space. In 2007, differential evolution (DE) was applied to parameter selection of SVM approximation (Zhou et al. 2007). DE was modified by adaptation a time-varying crossover probability strategy. The experiment results demonstrated that SVM with DE parameter selection has better approximation performance than artificial neural network (ANN).

The particle swarm optimization (Guo et al. 2008) is another meta-heuristic algorithm that was used for selecting the hyperparameter of SVM. This method does not

need any prior knowledge about the hyperparameter of SVM and can be used to determine multiple hyperparameters at the same time. Although the concept of particle swarm is different from the evolutionary algorithm, it is a dynamic system that uses a fitness function to evaluate the candidate solutions similar to the evolutionary algorithm. We notice that SVM parameter selection is an optimization problem in a large continuous search space. Among many meta-heuristic algorithms, only the evolutionary algorithms and the dynamic systems were used to solve this problem.

For the fitness function in the evolutionary process, many research studies use a classifier to measure the classification accuracy on the training data. However, Eads et al. (2002) compared the classification accuracy on the training data with the cross-validation accuracy. Their experimental results showed that the accuracies of the cross-validation score are better than those of the simple score. There are some research studies, such as Markatou et al. (2005), Blum et al. (1999) and Kääriäinen and Langford (2005), that analyzed the generalization performance of SVM and proposed to estimate the true error of learned classifiers. However, in Bartlett and Shawe-Taylor (1998), the bound of generalization performance for a large margin linear classifier are clearly explained. Therefore, the generalization performance of SVM is an interesting alternative for estimating the classification performance in the evolutionary process.

3 Multi-scale RBF kernel

There are many kernels that can be used in the SVM. Each kernel is suitable for different kinds of problems. The Gaussian RBF kernel is widely used in many problems. It uses the Euclidean distance between two points in the original space to find the correlation in the augmented space. The points that are very close to each other in the original space are strongly correlated whereas points that are far apart have uncorrelated in the augmented space (Ayat et al. 2001). This correlation function is rather smooth. There is only one parameter for adjusting the width of RBF, which is not powerful enough for some complex problems.

In order to get a better kernel, one possible way is to adjust the velocity of decrement in each range of distance between two points. Moreover, that kernel function should maintain good characteristics of the RBF kernel. To implement this capability, the combination of RBF kernels at different scales is proposed. The analytic expression of this kernel is the following:

$$K(x, y) = \sum_{i=1}^n a_i K(x, y, \gamma_i), \tag{5}$$

where n is a positive integer, $a_i \geq 0$ for $i = 1, \dots, n$ are the arbitrary non-negative weighting constants, and

$$K(x, y, \gamma_i) = \exp(-\gamma_i \|x - y\|^2) \tag{6}$$

is an RBF kernel with a width γ_i for $i = 1, \dots, n$.

In addition, the correlations in the feature space (relations between kernel functions and the distance between two points in the original space) of the multi-scale RBF kernels for $n = 1, 2,$ and 3 are displayed in Fig. 3. This figure shows that the correlations of the RBF kernel are rather smooth, while those of 2-RBF and 3-RBF have more variable shapes. This can be interpreted that the increase in the number of adjustable parameters provides a more adaptive kernel.

In general, the function which maps the input space into the augmented feature space is not explicitly known. However, the existence of such function is assured by Mercer’s theorem (Kecman 2001).

Mercer’s theorem: Any symmetric function $K(x, y)$ in the input space can represent an inner product in the feature space if

$$\iint K(x, y)g(x)g(y) \, dx \, dy \geq 0 \tag{7}$$

is valid for all $g \neq 0$ for which $\int g^2(u)du < \infty$. Then the kernel function K can be expanded in terms of Φ_i

$$K(x, y) = \sum_{i=1}^{\infty} \lambda_i \Phi_i(x) \Phi_i(y) \tag{8}$$

with $\lambda_i \geq 0$ (Schölkopf et al. 1998; Kecman 2001). In this case, the mapping function from the input space to the feature space is expressed as

$$\Phi : x \rightarrow (\sqrt{\lambda_1}\Phi_1(x), \sqrt{\lambda_2}\Phi_2(x), \dots) \tag{9}$$

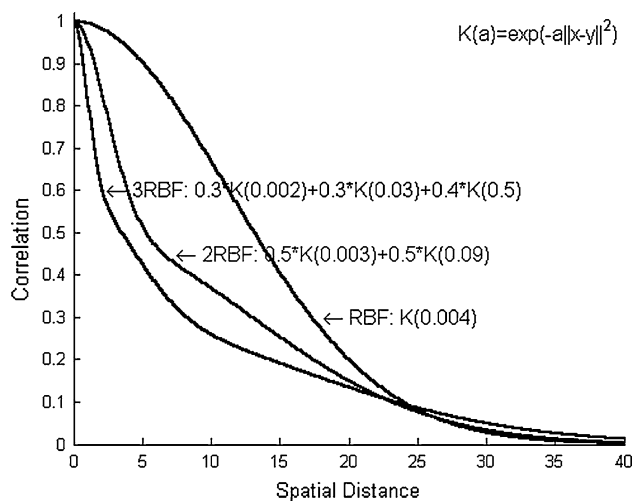


Fig. 3 Graph of single RBF, 2-RBF, and 3-RBF kernels

such that $K(x, y)$ can be the inner product

$$\Phi(x) \cdot \Phi(y) = \sum_{i=1}^{\infty} \lambda_i \Phi_i(x) \Phi_i(y) = K(x, y). \quad (10)$$

In the next lemma, the proposed kernel function is proved to be an admissible kernel by the Mercer's theorem.

Lemma: The non-negative linear combination of Mercer's kernels is a Mercer's kernel.

Proof Let $K(x, y, \gamma_i)$ be the Mercer's kernels with the parameter γ_i , for $i = 1, \dots, n$, and let

$$K(x, y) = \sum_{i=1}^n a_i K(x, y, \gamma_i), \quad (11)$$

where a_i for $i = 1, \dots, n$ are non-negative real values.

According to Mercer's theorem, we know that

$$\iint K(x, y, \gamma_i) g(x) g(y) \, dx \, dy \geq 0, \forall g \quad (12)$$

for $i = 1, \dots, n$.

By taking linear combination with non-negative coefficients a_i , we get

$$\sum_{i=1}^n a_i \iint K(x, y, \gamma_i) g(x) g(y) \, dx \, dy \geq 0 \quad (13)$$

$$\iint \sum_{i=1}^n a_i K(x, y, \gamma_i) g(x) g(y) \, dx \, dy \geq 0 \quad (14)$$

$$\iint K(x, y) g(x) g(y) \, dx \, dy \geq 0, \forall g. \quad (15)$$

Hence, the function $K(x, y) = \sum_{i=1}^n a_i K(x, y, \gamma_i)$ is a Mercer's kernel. \square

The proof of the multi-scale Mercer's kernels is rather obvious. The other combination methods were presented and proved in (Tan and Wang 2004). The RBF is a well-known Mercer's kernel; therefore, the non-negative linear combination of RBFs in (5) can be proved to be an admissible kernel by the Mercer's theorem.

From Eq. 5, there are $2n$ parameters when n terms of RBF kernels are used (n parameters for adjusting weights and n values of the widths of RBFs). However, we notice that the number of parameters can be reduced to $2n - 1$ by fixing the value of the first parameter to 1. The multi-scale RBF kernel that will be used throughout the rest of this paper becomes as follows,

$$K(x, y) = K(x, y, \gamma_0) + \sum_{i=1}^{n-1} a_i K(x, y, \gamma_i). \quad (16)$$

When multiple RBF functions are combined, the separating hyperplane is more flexible than using a single RBF function. However, the multi-scale RBF kernel may lead to the overfitting problem. This problem can be overcome by the objective function proposed in Sect. 5.

4 Evolving hyperparameters of SVM

The approach presented here combines the techniques of SVM and ES together, using ES to evolve hyperparameters of SVM based on the multi-scale RBF kernel. As shown in (16), the proposed kernel has $2n - 1$ parameters when n terms of RBF kernels are used. These values have an influence on the performance of the proposed kernel. In the soft margin SVM, there is a regularization parameter C that affects the performance of the classification. The parameters of the kernel function and the regularization parameter are called hyperparameters.

In order to obtain appropriate values of these hyperparameters, ES is considered. There are several variations of ES. Nevertheless, we choose to use the $(\mu + \lambda)$ -ES where μ parents produce λ offspring. Both parents and offspring compete equally for survival (deDoncker et al. 1996). Therefore, good solutions are preserved until better solutions are found.

Let \bar{v} be a non-negative real-value vector of hyperparameters that has $2n$ dimensions. The vector \bar{v} is represented in the form of:

$$\bar{v} = (C, \gamma_0, a_1, \gamma_1, a_2, \gamma_2, \dots, a_{n-1}, \gamma_{n-1}), \quad (17)$$

where C is a regularization parameter, γ_i for $i = 0, \dots, n - 1$ are the widths of RBFs, a_i for $i = 1, \dots, n - 1$ are the weights of RBFs, and n is the number of terms of RBFs. Our goal is to find \bar{v} that optimizes an objective function $g(\bar{v})$. The $(5 + 10)$ -ES is applied to adjust these hyperparameters.

Although the ES algorithm can be implemented by parallel programming, it is more convenient to implement and run it on a computer. When the population size of ES is large, the algorithm may need a lot of running time to process each generation of ES but may require only a few generations to obtain the optimal solutions. On other hand, when the population size of ES is small, the population may lack diversity and a large number of generations may be required to converge to the optimal solutions. Hence, $(5 + 10)$ -ES is a choice of $(\mu + \lambda)$ -ES which can preserve the diversity of the population and does not require a lot of running time for each generation. The algorithm of $(5 + 10)$ -ES is shown in Fig. 4.

This algorithm starts with the 0th generation ($t = 0$) that selects the non-negative real valued vectors of five solutions $\bar{v}_1, \dots, \bar{v}_5 \in R_+^{2n}$ with standard deviation $\bar{\sigma} \in R_+^{2n}$ using randomization. These five initial solutions are evaluated by their fitness. Then, the global intermediary recombination method is used for creating ten new solutions. Ten pairs of solutions are selected from five conventional solutions. The average of each pair of solutions, element by element, is a new solution.

```

t = 0;
initialization( $\bar{v}_1, \dots, \bar{v}_5, \bar{\sigma}$ );
evaluate  $g(\bar{v}_1), \dots, g(\bar{v}_5)$ ;
while (NOT Terminated_Conditions) do
  for i = 1 to 10 do
     $\bar{v}'_i = \text{recombination}(\bar{v}_1, \dots, \bar{v}_5)$ ;
     $\bar{v}'_i = \text{mutate}(\bar{v}'_i)$ ;
    evaluate  $g(\bar{v}'_i)$ ;
  end
  ( $\bar{v}_1, \dots, \bar{v}_5$ ) = select( $\bar{v}_1, \dots, \bar{v}_5, \bar{v}'_1, \dots, \bar{v}'_{10}$ );
   $\bar{\sigma} = \text{mutate}_\sigma(\bar{\sigma})$ ;
  t = t + 1;
end
    
```

Fig. 4 The (5 + 10)-ES algorithm

$$\bar{v}'_1 = \frac{1}{2}(\bar{v}_1 + \bar{v}_2) \tag{18a}$$

$$\bar{v}'_1 = \frac{1}{2}(\bar{v}_1 + \bar{v}_3) \tag{18b}$$

$$\vdots \tag{18c}$$

$$\bar{v}'_{10} = \frac{1}{2}(\bar{v}_4 + \bar{v}_5)$$

This recombination method is chosen for this research because every individual from the parent population is used for creating new individuals. Most of the offspring individuals are different from their parents because each of them is the average of two different parents, except for the case where there are two parent individuals that are the same. Moreover, no offspring comes from the same parents. Therefore, the diversity of the population can also be preserved by this recombination method.

These solutions are then mutated by the following function:

$$\text{mutate}(\bar{v}) = (C + z_1, \gamma_0 + z_2, \dots, a_{n-1} + z_{2n-1}, \gamma_{n-1} + z_{2n}) \tag{19b}$$

$$z_i \sim N_i(0, \sigma_i^2). \tag{19b}$$

\bar{v}'_i for $i = 1, \dots, 10$ are mutated by adding each of them with $(z_1, z_2, \dots, z_{2n})$, where z_i is a random value from a normal distribution with zero mean and σ_i^2 variation. In each generation, the standard deviation is adjusted by

$$\text{mutate}_\sigma(\bar{\sigma}) = (\sigma_1 \cdot e^{z_1}, \sigma_2 \cdot e^{z_2}, \dots, \sigma_{2n} \cdot e^{z_{2n}}) \tag{20a}$$

$$z_i \sim N_i(0, \tau^2), \tag{20b}$$

where τ is an arbitrary constant.

This mutation is performed in order to vary the solutions. Although the offspring solutions will be different

from their parents and each of them differs from one another by our recombination method, these offspring solutions could be the same as ones on the previous generations. The mutation will make these solutions different from the previous generations. A random number will be added to each component of these offspring solutions. Therefore, the new solutions can be produced unlimitedly.

Then, only the 5 fittest solutions are selected from 5 + 10 solutions to be the parents in the next generation. Then, the process is repeated until a fixed number of generations have been produced or the acceptance criterion is reached. In this research, the maximum number of generations is used as a stop criterion of the ES algorithm. In our experiments, the maximum number of generations is fixed to 1,000. Although high quality solutions can be found with fewer generations for some datasets, we want to ensure that high quality solutions can be found for all datasets in our experiments. A large number of generations does not decrease the performance of the classification. Nevertheless, the maximum number of generations is restricted by the running time allowed to run ES.

5 Objective functions in ES

One of the most important and difficult parts of the evolutionary algorithm is how to define an objective function for the task under consideration. In our case of evaluating the hyperparameters, there are many ways to define an objective function. In this research, we propose three possible objective functions: training accuracy, the bound of generalization error, and subset cross-validation on training accuracy. These objective functions are described and verified in the following sections.

5.1 Training accuracy

The training accuracy is a measure of learning performance. This function indicates the ability of a learning machine as measured by its accuracy on training data. This is the simplest way to define our objective function in the evolutionary algorithm. The formula expression of the training accuracy is depicted by the following equation:

$$g(\bar{v}) = \left(1 - \frac{\sum_{i=1}^l |y_i - f(x_i)|}{2l} \right), \tag{21}$$

where $x_i \in R^N$ is a training data, $y_i \in \{-1, 1\}$ is its label, and $f(x_i)$ is a decision function of x_i for $i = 1, \dots, l$. We expect that a suitable set of hyperparameters should yield a high training accuracy. The optimal hyperparameters

obtained by using the training accuracy as an objective function will be evaluated on the test data.

5.2 Bound of generalization error

The generalization error of a machine learning algorithm is a function that indicates the capacity of the machine in classifying data (Burges 1998). We define the class Q of real-valued functions on the ball of radius R in R^n as

$$Q = \{x \mapsto w \cdot x : \|w\| \leq 1, \|x\| \leq R\}. \tag{22}$$

There is a constant c such that, for all probability distributions, with probability at least $1 - \delta$ over l independently generated examples z , if a classifier $f = \text{sgn}(q) \in \text{sgn}(Q)$ has margin at least γ on all examples in z , then the error of h is no more than

$$\frac{c}{l} \left(\frac{R^2}{\gamma^2} \log^2 l + \log(1/\delta) \right). \tag{23}$$

Furthermore, with probability at least $1 - \delta$, every classifier $h \in \text{sgn}(B)$ has error no more than

$$\frac{k}{l} + \sqrt{\frac{c}{l} \left(\frac{R^2}{\gamma^2} \log^2 l + \log(1/\delta) \right)}, \tag{24}$$

where k is the number of labeled examples in z with margin less than γ (Bartlett and Shawe-Taylor 1998). Hence, we can bound the generalization error of SVM even when the kernel is defined in an infinite dimensional feature space (Bartlett and Shawe-Taylor 1998).

The expression in (24) is called the bound of generalization error. The term $\frac{k}{l}$ is equivalent to the empirical risk that is defined to be just the measured mean error rate on the fixed number of training data. Also, $\frac{R^2}{\gamma^2}$ is the Vapnik–Chervonenkis (VC) dimension. The VC-dimension is a measure of the capacity of a statistical classification algorithm. It was originally defined by Vapnik and Chervonenkis (Vapnik and Chervonenkis 1971; Blumer et al. 1989). Intuitively, the capacity of a classification model is related to how complicated the model can be.

The VC-dimension measures the complexity of the hypothesis space, not by the number of distinct hypotheses, but instead by the number of distinct instances that can be completely discriminated using one hypothesis (Mitchell 1997). As the bound in (24) considers both the training error and the VC-dimension, we expect that the generalization performance of the SVM that uses our kernel can be approximated by this bound. Therefore, the bound of generalization error is considered as one objective function in our ES algorithm. We presume that a suitable set of hyperparameters should provide a lower bound of generalization error.

5.3 Subset cross-validation on training accuracy

Although the training accuracy can be easily calculated, this objective function may overfit the training data. Sometimes, the data may contain a lot of noise. If the decision function is trained by these noisy data, the learned concept may be wrong. Hence, we propose to train the decision function with several sets of data. A good set of parameters should perform well on many training sets. However, as we have only a fixed amount of training data, subset cross-validation is considered.

At the beginning, the training data are divided into five subsets, each of which has almost the same amount of data. For each generation of ES, the five classifiers with the same hyperparameters but with different training and testing sets are evaluated. In the j th iteration ($j = 1, 2, 3, 4, 5$), the classifier is trained on all subsets except for the j th one. Then, its classification accuracy is evaluated on the j th subset. These partitions are displayed in Fig. 5.

Only the real training data sets are used to produce the classifiers with the same set of hyperparameters. Then, the validation sets are used for evaluating the accuracy of the classifiers. The average of these five accuracy values is used as the objective function $g(\bar{v})$.

$$g(\bar{v}) = \frac{\sum_{j=1}^5 \left(1 - \frac{\sum_{i=1}^{l_j} |y_i - f(x_i)|}{2l_j} \right)}{5} \tag{25}$$

For parameter selection, subset cross-validation is a rather good estimate of the generalization accuracy of a learning algorithm. The testing data set is reserved for testing the final classifier with the best parameters identified by the evolutionary strategy. In the same way, subset cross-validation can be applied to the bound of generalization error. However, the bound of generalization

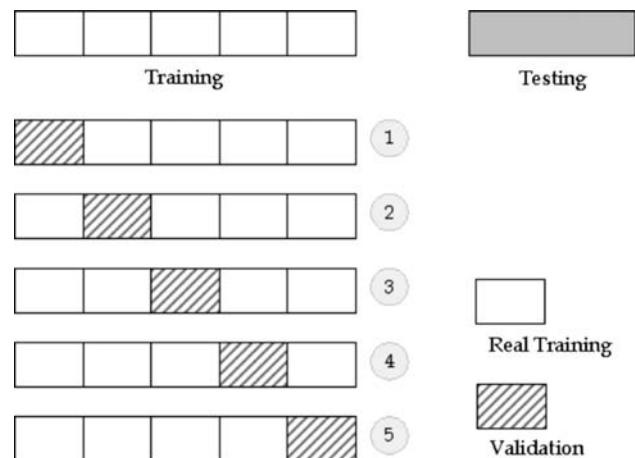


Fig. 5 Partition training data into five subsets

error is already known to overcome the overfitting problem. Therefore, the subset cross-validation on the bound of generalization error may not be necessary. This assumption will be verified in the next section.

6 Experimental results

In order to verify the performance of the proposed method, SVMs with the multi-scale RBF kernels are trained and tested on 15 datasets from the UCI repository (Blake and Merz 1998). These datasets come from real world problems such as game playing, medical inference, and predictions in biology and physics. Each of datasets contains two classes. The name, the number of attributes, and the sample size of each dataset are shown in Table 2.

In our experiments, the proposed method is evaluated by fivefold cross-validation. The evolutionary strategy is used to find the optimal hyperparameters of both the conventional RBF and the proposed kernel. The value of τ in the evaluation process of these experiments is 1.0. The number of RBF terms is a positive integer which is less than or equal to 10. The widths of RBFs (γ_i), the weights of RBFs (a_i), and the regularization parameter (C) are real numbers between 0.0 and 10.0. These hyperparameters are searched within 1,000 generations of ES. A real-coded genetic algorithm (RCGA) and differential evolution (DE) are the other evolutionary algorithms that are applied to hyperparameter selection. These evolutionary algorithms and a grid search are compared to ES.

An algorithm of RCGA presented in Chunhong and Licheng (2004) is implemented and tested on SVM with a

single RBF kernel. The representation of RCGA is the fixed-length real-valued vector. The genetic operators are crossover and mutation that are different from ES. Linear crossover is used as the crossover operator; two parameter vectors are weighted and combined. The mutation operator is performed by random mutation (Herrera et al. 1996). Another difference between RCGA and ES is the mutation of the standard deviation that only appears in the ES algorithm.

For DE, the representation is also the real valued vector of parameters. DE generates a new parameter vector by adding the weight difference between two population vectors to a third vector; this operation is called mutation (Storn and Price 1997). Then, the mutated vector is mixed with the parameters of another predetermined vector; this process is referred to as crossover (Storn and Price 1997).

For each target vector \bar{v}_i , a mutant vector is generated according to

$$\bar{u}_i = \bar{v}_{r_1} + c(\bar{v}_{r_2} - \bar{v}_{r_3}) \tag{26}$$

where $r_1, r_2, r_3 \in \{1, 2, \dots, \mu\}$ are random indices and c is a real and constant factor. Then, the new trial vector:

$$\bar{v}_i = (v_{i,1}, v_{i,2}, \dots, v_{i,m}) \tag{27}$$

is formed by

$$v_{i,j} = \begin{cases} v_{i,j} & \text{if } (\text{rand}(j) \leq \text{cr}) \\ u_{i,j} & \text{if } (\text{rand}(j) > \text{cr}) \end{cases} \tag{28}$$

where $\text{rand}(j)$ is the j th uniform random number generator with outcome in $[0, 1]$, and cr is the crossover constant in $[0, 1]$.

In both RCGA and DE, 5 parents are used for produced 10 new offspring and the number of generations is defined as 1000. These numbers of population and generations are the same with the proposed ES algorithm. The average accuracy from fivefold cross-validation and the ranks on each dataset of the grid search, RCGA, DE, and ES are compared in Table 3. All algorithms used training error as the heuristic for specifying hyperparameters of SVM with the single RBF kernel.

The experimental results show that the average accuracies of DE and ES with a single RBF kernel are higher than the other methods on many datasets. The average accuracy for all datasets and the average rank of ES is the lowest. Therefore, ES is a suitable algorithm for hyperparameter selection. Then, the statistical tests are involved in order to assure the performance of our parameter selection algorithm. An example on the use of the non-parametric statistical tests was appeared in Garcia et al. (2009).

The Friedman test (Friedman 1937) is a statistical method for testing the differences between more than two related sample means, and it can be used to compare multiple classifiers in our experiments. The algorithms are

Table 2 Datasets from UCI Repository

Datasets	No. of attributes	No. of examples
Checkers	2	192
Spiral	2	582
LiverDisorders	6	345
IndiansDiabetes	8	768
ThreeOfNine	9	512
TicTacToe	9	958
BreastCancer	10	699
ParityBits	10	1,024
SolarFlare	10	1,066
ClevelandHeart	13	270
Australian	14	690
German-org	24	1,000
Ionosphere	34	351
Tokyo	44	959
Sonar	60	208

Table 3 Average Accuracy of Single RBF Kernel when Using Training Error as the Heuristic for Hyperparameter Selection (%)

Datasets	Grid search	RCGA	DE	ES
Checkers	83.32 (3)	83.83 (2)	67.65 (4)	89.58 (1)
Spiral	100.00 (2.5)	100.00 (2.5)	100.00 (2.5)	100.00 (2.5)
LiverDisorders	61.74 (4)	64.06 (3)	64.64 (2)	69.57 (1)
IndiansDiabetes	64.97 (3)	64.45 (4)	69.00 (2)	72.52 (1)
ThreeOfNine	53.51 (2.5)	53.51 (2.5)	53.51 (2.5)	53.51 (2.5)
TicTacToe	65.34 (3.5)	68.23 (2)	68.33 (1)	65.34 (3.5)
BreastCancer	86.41 (4)	94.13 (2.5)	94.13 (2.5)	95.28 (1)
ParityBits	48.05 (2.5)	42.08 (4)	56.95 (1)	48.05 (2.5)
SolarFlare	80.87 (2)	80.68 (3)	80.58 (4)	83.30 (1)
ClevelandHeart	55.56 (3)	58.89 (2)	61.48 (1)	55.55 (4)
Australian	55.51 (2)	55.36 (4)	55.51 (2)	55.51 (2)
German-org	70.10 (3)	70.20 (2)	69.70 (4)	70.40 (1)
Ionosphere	66.10 (2.5)	62.74 (4)	69.31 (1)	66.10 (2.5)
Tokyo	81.02 (4)	85.71 (3)	87.69 (2)	91.24 (1)
Sonar	70.67 (4)	87.49 (2)	87.48 (3)	91.35 (1)
Average accuracy	69.54	71.42	72.40	73.82
Average rank	3.0333	2.8333	2.3000	1.8333

Bold values are the best accuracy for each dataset

ranked for each dataset separately, i.e. the best performing algorithm gets the rank of 1, the second best gets the rank of 2, and so on, as shown in the parentheses in Table 3. In case of ties, an average rank is assigned to both algorithms or all tie algorithms. Then, the average ranks across the datasets are compared at the last row of Table 3.

The average ranks of the algorithms are compared under the null-hypothesis, which states that all the algorithms are equivalent so their ranks should be equal (Demšar 2006). Let R_j be the average ranks of the j th of k algorithms on the N datasets. The Friedman statistic

$$\chi_F^2 = \frac{12N}{k(k+1)} \left[\sum_j R_j^2 - \frac{k(k+1)^2}{4} \right] \quad (29)$$

is distributed according to χ_F^2 with $k-1$ degrees of freedom, when N and k are large enough. Iman and Davenport (1980) derived a better statistic from Friedman's χ_F^2

$$F_F = \frac{(N-1)\chi_F^2}{N(k-1) - \chi_F^2} \quad (30)$$

which is distributed according to the F distribution with $k-1$ and $(k-1)(N-1)$ degrees of freedom.

From the experimental results in Table 3, the Friedman test is used to check whether the measured average ranks are significantly different from the mean rank $R_j = 2.5$:

$$\chi_F^2 = \frac{12(15)}{4(4+1)} \left[(3.0333^2 + 2.8333^2 + 2.3000^2 + 1.8333^2) - \frac{4(4+1)^2}{4} \right] = 7.92 \quad (31)$$

$$F_F = \frac{(15-1)7.92}{15(4-1) - 7.92} = 2.9903. \quad (32)$$

With 4 algorithms and 15 datasets, F_F is distributed according to the F distribution with $4-1=3$ and $(4-1) \times (15-1) = 42$ degrees of freedom. The critical value of $F(3,42)$ for $\alpha = 0.05$ is 2.832, so we reject the null-hypothesis.

Then, we use the Bonferroni–Dunn test for pairwise comparison. The performances of two classifiers are significantly different if their corresponding average ranks differ by at least the critical difference

$$CD = q_\alpha \sqrt{\frac{k(k+1)}{6N}}, \quad (33)$$

where critical values q_α are illustrated in Demšar (2006). This research uses $q_{0.10}$ for four classifiers which is 2.128. The corresponding CD is

$$CD = (2.128) \sqrt{\frac{4(4+1)}{6(15)}} = 1.0031. \quad (34)$$

From the CD value, we can say that the performance of ES is significantly better than the grid search. However, it is not sufficient to conclude about the different performance of these evolutionary algorithms.

Holm (1979) is another statistical test; it was used for comparisons of multiple classifiers by adjusting the value of the level of confidence in a step down method (Garcia and Herrera 2008). For Holm's test, the corresponding statistics and probability values (p) are computed and ordered. The ordered p values are denoted by p_1, p_2, \dots , so that $p_1 \leq p_2 \leq \dots \leq p_{k-1}$. Holm's procedure compares

each p_i with $\alpha/(k - i)$. (Demsar 2006). It starts with the most significant p value. If p_1 is below $\alpha/(k - 1)$, the corresponding hypothesis is rejected and p_2 is allowed to compare with $\alpha/(k - 2)$ (Demsar 2006). As soon as a null hypothesis cannot be rejected, all the remaining hypothesis are retained as well (Demsar 2006).

From Table 3, the standard error is

$$SE = \sqrt{\frac{k(k + 1)}{6N}} = \sqrt{\frac{4(4 + 1)}{6(15)}} = 0.4714. \tag{35}$$

The corresponding statistics and p values are computed and displayed in Table 4.

These results correspond to the Bonferroni–Dunn test. The Holm procedure rejects the first null-hypotheses since the corresponding p values are smaller than the adjusted α . Therefore, ES yields the results that are significantly better than grid search at $\alpha = 0.05$. The other null-hypothesis cannot be rejected because their p values are more than the adjusted α , so we would have to retain them. Hence, ES is a better alternative for hyperparameter selection.

Furthermore, the average accuracies of ES can be enhanced by using a suitable objective function. Different objective functions are used in ES for identifying the optimal hyperparameters of SVM. Training accuracy ($TrnAcc$), the bound of generalization error ($BdOfGenErr$), 5-subset cross-validation on training accuracy ($5SubsetOnTrnAcc$), and 5-subset cross-validation on the bound of generalization error ($5SubsetOnBdOfGenErr$) are used as an objective function in the evolutionary process. ES with the various objective functions are compared in terms of the average accuracies and their ranks in Table 5. The statistical paired t test is used for testing the difference of two algorithms in terms of the average accuracy on each dataset.

When different objective functions of ES are used, $TrnAcc$ provides good accuracies on testing and $BdOfGenErr$ yields better results on many datasets. Moreover, the average accuracy on 15 datasets of $BdOfGenErr$ is higher than that of $TrnAcc$. These results show that $TrnAcc$ is not the best objective function, and in fact it may guide ES to select a classifier which overfits the training data. On other hand, $BdOfGenErr$ is an approximation of the generalization performance of SVM which considers both the training error as well as the VC-dimension. Thus, it can avoid the overfitting problem, resulting in better performance.

Although $BdOfGenErr$ can improve the performance of the classification, $5SubsetOnTrnAcc$ is more accurate. When $5SubsetOnTrnAcc$ is used, the average accuracy is higher than those of the other objective functions. The training data is partitioned into five subsets to avoid the overfitting problem. Thus, the hyperparameters which work well for all of the five subsets would have less chance to overfit the data. Hence, $5SubsetOnTrnAcc$ should be a good objective function for our proposed kernel. We also run an experiment using $5SubsetOnBdOfGenErr$, but the average accuracy does not increase from $BdOfGenErr$. This may come from the fact that the bound of generalization error has already taken the overfitting problem into consideration by examining both the training error and the VC-dimension of SVMs. Therefore, subset cross-validation may not be necessary for the bound of generalization error.

Although the average classification accuracies across different datasets can present the performance of our methods, it may be susceptible to outliers. Moreover, the averages could be meaningless if the results on different datasets are not comparable (Demsar 2006). In general, we prefer the classifiers that work well on many problems. Therefore, a ranking method is applied to compare these algorithms. Wilcoxon signed-ranks test is a non-parametric testing that ranks the differences in performances of two classifiers for each dataset, ignoring the signs, and compares the ranks for the positive and the negative differences (Demsar 2006). The Wilcoxon signed-ranks test is applied for comparing between $TrnErr$ and the other objective functions of ES. Table 6 shows the accuracy difference and the ranks of each objective function over the various datasets. These results show that we cannot reject the null-hypothesis at $\alpha = 0.05$ because the value of Z on each pairwise comparison is greater than -1.96 .

Then, the Friedman test is applied for comparing many objective functions. The test is used for testing the difference of multiple algorithms over multiple datasets based on their average ranks. The average ranks provide a fair comparison of these algorithms. On average, ES with $5SubsetOnTrnAcc$ is the best objective function (with the average rank of 2.3333), $TrnAcc$ and $BdOfGenErr$ are the second best, whereas $5SubsetOnBdOfGenErr$ is not better than the other objective functions.

As shown in the experimental results in Table 5, the Friedman test is used to check whether the measured

Table 4 Holm’s Test for Hyperparameter Selection Algorithms

I	Parameter selection methods	$z = (R_0 - R_i)/SE$	p	$\alpha/(k - i)$
1	ES	$(3.0333 - 1.8333)/0.4714 = 2.5456$	0.0109	0.0167
2	DE	$(3.0333 - 2.3000)/0.4714 = 1.5557$	0.1198	0.0250
3	RCGA	$(3.0333 - 2.8333)/0.4714 = 0.4243$	0.6713	0.0500

Table 5 Average Accuracy of ES with Single RBF Kernel that Uses Different Objective Functions (%)

Datasets	Grid search	Objective Function of ES			
		<i>TrnAcc</i>	<i>BdOfGenErr</i>	<i>5SubsetOn TrnAcc</i>	<i>5SubsetOn BdOfGenErr</i>
Checkers	83.32 (3.5)	89.58 (1)	88.56 (2)	83.82 (3.5)	80.19 (5)
Spiral	100.00 (3)	100.00 (3)	100.00 (3)	100.00 (3)	100.00 (3)
LiverDisorders	61.74 (5)	69.57 (1)	68.41 (2)	66.67 (3)	62.61 (4)
IndiansDiabetes	64.97 (5)	72.52 (2)	68.61 (3)	73.30 (1)	65.10 (4)
ThreeOfNine	53.51 (4.5)	53.51 (4.5)	100.00** (2)	100.00** (2)	100.00** (2)
TicTacToe	65.34 (4.5)	65.34 (4.5)	98.96** (3)	99.69** (1)	98.75** (2)
BreastCancer	86.41 (5)	95.28 (1)	94.85 (2)	94.56 (3)	93.85 (4)
ParityBits	48.05 (4)	48.05 (4)	80.46** (1)	75.78** (2)	48.05 (4)
SolarFlare	80.87 (5)	83.30 (1)	82.93 (2)	80.96 (4)	82.93 (3)
ClevelandHeart	55.56 (4)	55.55 (4)	55.55 (4)	78.15 (1)	62.59 (2)
Australian	55.51 (3)	55.51 (3)	55.51 (3)	55.51 (3)	55.51 (3)
German-org	70.10 (3)	70.40 (1)	70.00 (4.5)	70.30 (2)	70.00 (4.5)
Ionosphere	66.10 (4.5)	66.10 (4.5)	95.45** (1)	95.16** (2)	92.31** (3)
Tokyo	81.02 (5)	91.24 (2)	88.32 (4)	91.66 (1)	88.53 (3)
Sonar	70.67 (5)	91.35 (1.5)	91.35** (1.5)	88.92 (3)	88.43 (4)
Average accuracy	69.54	73.82	82.60	83.63	79.26
Average rank	4.2000	2.5333	2.5333	2.3000	3.3667

**Statistically significant at the level of 0.001 when compared to *TrnAcc*

Bold values are the best accuracy for each dataset

Table 6 Comparisons between *TrnAcc* and other objective functions

Datasets	Difference from average accuracy of <i>TrnAcc</i>			Rank		
	<i>BdOfGenErr</i>	<i>5SubsetOn TrnAcc</i>	<i>5SubsetOn BdOfGenErr</i>	<i>BdOfGenErr</i>	<i>5SubsetOn TrnAcc</i>	<i>5SubsetOn BdOfGenErr</i>
Checkers	-1.03	-5.76	-9.39	8	10	12
Spiral	0.00	0.00	0.00	2.5	1.5	2
LiverDisorders	-1.16	-2.90	-6.96	9	9	9
IndiansDiabetes	-3.91	0.78	-7.42	11	6	11
ThreeOfNine	46.49	46.49	46.49	15	15	15
TicTacToe	33.61	34.34	33.40	14	14	14
BreastCancer	-0.43	-0.71	-1.43	7	5	6
ParityBits	32.42	27.73	0.00	13	12	2
SolarFlare	-0.38	-2.34	-0.38	5	7	4
ClevelandHeart	0.00	22.59	7.04	2.5	11	10
Australian	0.00	0.00	0.00	2.5	1.5	2
German-org	-0.40	-0.10	-0.40	6	3	5
Ionosphere	29.35	29.07	26.22	12	13	13
Tokyo	-2.92	0.42	-2.72	10	4	7
Sonar	0.00	-2.43	-2.91	2.5	8	8
$R^+ = \sum_{d_i > 0} \text{rank}(d_i) + \frac{1}{2} \sum_{d_i = 0} \text{rank}(d_i)$				59	76.5	55
$R^- = \sum_{d_i < 0} \text{rank}(d_i) + \frac{1}{2} \sum_{d_i = 0} \text{rank}(d_i)$				61	43.5	65
$T = \min(R^+, R^-)$				59	43.5	55
$Z = \frac{T - \frac{1}{2}N(N+1)}{\sqrt{\frac{1}{32}N(N+1)(2N+1)}}$				-0.0568	-0.93714	-0.28398

average ranks are significantly different from the mean rank $R_j = 3$:

$$\chi^2_F = \frac{12(15)}{5(5+1)} \left[(4.2^2 + 2.5333^2 + 2.5333^2 + 2.3^2 + 3.3667^2) - \frac{5(5+1)^2}{4} \right] = 12.5994 \tag{36}$$

$$F_F = \frac{(15-1)12.5994}{15(5-1) - 12.5994} = 3.7213. \tag{37}$$

With 5 algorithms and 15 datasets, F_F is distributed according to the F distribution with $5 - 1 = 4$ and $(5 - 1) \times (15 - 1) = 56$ degrees of freedom. The critical value of $F(4,56)$ for $\alpha = 0.05$ is 2.54136, so we reject the null-hypothesis.

Then, we use the Bonferroni–Dunn test that is a post-hoc test for pairwise comparisons. This research uses $q_{0.10}$ for five classifiers which is 2.241. The corresponding CD is

$$CD = (2.241) \sqrt{\frac{5(5+1)}{6(15)}} = 1.5538. \tag{38}$$

From the CD value, we can say that the performance of the grid search is significantly worse than those of ES with $TrnAcc$, $5SubsetOnTrnAcc$, and $BdOfGenErr$; however, it is not sufficient to conclude about $5SubsetOnBdOfGenErr$ ($4.2 - 3.3667 < 1.5538$).

The Holm procedure is another post-hoc test that will be performed if the Friedman test rejects the null-hypothesis. Standard error of 5 algorithms and 15 datasets in Table 5 is calculated by the following equation:

$$SE = \sqrt{\frac{k(k+1)}{6N}} = \sqrt{\frac{5(5+1)}{6(15)}} = 0.5774. \tag{39}$$

The corresponding statistics and p values are computed and displayed in Table 7.

When $\alpha = 0.05$, the Holm procedure rejects the first, the second, and then the third hypotheses since the corresponding p values are smaller than the adjusted α . The last hypothesis cannot be rejected, and thus we would have to retain them. Hence, $TrnAcc$, $5SubsetOnTrnAcc$, and $BdOfGenErr$ will be evaluated in the next experiments,

whereas $5SubsetOnBdOfGenErr$ will not be used as an objective function in the rest of our experiments.

For the running time, it is rather obvious that the evolutionary strategy consumes a lot of time when it is compared to a single SVM. However, this process is indispensable, as the accuracy of the learned SVM depends heavily on the quality of the obtained hyperparameters. Furthermore, determining high-quality hyperparameters is an off-line process in most application, and thus this running time can be disregarded. Nevertheless, we found that the running time of each proposed method (ES with a different objective function) is less than that of the grid search with a large number of evaluations. The running time of each method on a fold of Sonar dataset is recorded and illustrated in Table 8.

The proposed methods and the grid search were run on a computer with an Intel Xeon 2.73 GHz CPU and 3.85 GB memory. In the grid search, the regularization parameter, the width of RBF kernel, and the combination weight are varied by a log-scale from 0.0001, 0.0002, ..., 0.001, 0.002, ..., to 10.0. The running time of the grid search on Sonar was about 3 h, whereas the running time of RCGA, DE, and ES with $TrnAcc$ are about eight times shorter than grid search. In addition, the running time of ES with $TrnAcc$ is close to that of $BdOfGenErr$, and the running time of $5SubsetOnTrnAcc$ is also close to that of $5SubsetOnBdOfGenErr$.

However, we found that the running times of $5SubsetOnTrnAcc$ and $5SubsetOnBdOfGenErr$ are about five times longer than those of $TrnAcc$ and $BdOfGenErr$. For the case of five-subsets cross-validation, SVM classifiers are trained and validated five times for each hyperparameter. For other datasets, the running times of these methods have a similar trend; the running times of $5SubsetOnTrnAcc$ and $5SubsetOnBdOfGenErr$ are more than those of $TrnAcc$ and $BdOfGenErr$, and the running time of each proposed methods is less than that of the grid search with a large number of evaluations.

For multi-scale RBF kernels, the average accuracies and the ranks of each objective function using n -RBF when $n = 2, 3, 4,$ and 5 are illustrated in Tables 9, 10, 11, and 12, respectively. Furthermore, the average accuracies on 15 datasets for each objective function are illustrated by the

Table 7 Holm’s test for comparing between Grid search and different objective functions of ES

i	Objective functions	$z = (R_0 - R_i)/SE$	p	$\alpha/(k - i)$
1	$5SubsetOnTrnAcc$	$(4.2000 - 2.3000)/0.5774 = 3.2906$	0.0010	0.0125
2	$TrnAcc$	$(4.2000 - 2.5333)/0.5774 = 2.8866$	0.0039	0.0167
3	$BdOfGenErr$	$(4.2000 - 2.5333)/0.5774 = 2.8866$	0.0039	0.0250
4	$5SubsetOnBdOfGenErr$	$(4.2000 - 3.3667)/0.5774 = 1.4432$	0.1490	0.0500

Table 8 The running time of an SVM with the RBF kernel that uses different parameter selection methods when training on a fold of sonar dataset

Parameter selection methods	Running time (hours)
Grid Search	3:05:35
RCGA with <i>TrnAcc</i>	0:20:20
DE with <i>TrnAcc</i>	0:19:47
ES with <i>TrnAcc</i>	0:23:18
ES with <i>BdOfGenErr</i>	0:22:31
ES with <i>5SubsetOnTrnAcc</i>	1:56:31
ES with <i>5SubsetOn BdOfGenErr</i>	1:58:54

Table 9 Average accuracy of 2-RBF kernel (%)

Datasets	Objective Functions of ES		
	<i>TrnAcc</i>	<i>BdOfGenErr</i>	<i>5SubsetOn TrnAcc</i>
Checkers	90.62 (1)	89.07 (2)	84.35 (3)
Spiral	100.00 (2)	100.00 (2)	100.00 (2)
LiverDisorders	70.73 (2)	71.31 (1)	67.25 (3)
IndiansDiabetes	72.39 (2)	71.35 (3)	74.86 (1)
ThreeOfNine	53.51 (3)	100.00** (1.5)	100.00** (1.5)
TicTacToe	65.34 (3)	98.33** (2)	99.69** (1)
BreastCancer	95.56 (2)	95.42 (3)	95.99 (1)
ParityBits	48.05 (3)	79.49** (2)	80.18** (1)
SolarFlare	83.21 (1)	82.93 (2)	80.96 (3)
ClevelandHeart	55.55 (3)	62.22 (2)	83.34** (1)
Australian	55.51 (2)	55.51 (2)	55.51 (2)
German-org	70.80 (2)	70.00 (3)	73.40 (1)
Ionosphere	66.10 (3)	95.45** (2)	96.29** (1)
Tokyo	89.05 (2)	87.79 (3)	91.66 (1)
Sonar	92.78 (1)	92.30 (2)	88.92 (3)
Average accuracy	73.95	83.41	84.83
Average rank	2.1333	2.1666	1.7000

**Statistically significant at the level of 0.001 when compared to *TrnAcc*

Bold values are the best accuracy for each dataset

graphs in Fig. 6. These results show the performance of each objective function. For all *n*-RBF kernels, the average accuracies of *5SubsetOnTrnAcc* are better than those of *TrnAcc* and *BdOfGenErr*. There are many datasets where *5SubsetOnTrnAcc* yields the highest accuracies. Moreover, the average accuracies of *5SubsetOnTrnAcc* are significantly better than those of *TrnAcc* on some datasets.

When the average rank is considered, we found that *5SubsetOnTrnAcc* performed the best for all *n*-RBF when *n* = 2, 3, 4, and 5, while *TrnAcc* and *BdOfGenErr* gave similar results for all experiments. However, the average accuracies of *BdOfGenErr* are significantly better than those of *TrnAcc* on some datasets, such as ThreeOfNine,

Table 10 Average accuracy of 3-RBF kernel (%)

Datasets	Objective Functions of ES		
	<i>TrnAcc</i>	<i>BdOfGenErr</i>	<i>5SubsetOn TrnAcc</i>
Checkers	90.62 (1)	89.07 (2)	83.82 (3)
Spiral	100.00 (2)	100.00 (2)	100.00 (2)
LiverDisorders	71.02 (1.5)	71.02 (1.5)	68.99 (3)
IndiansDiabetes	73.55 (2)	71.34 (3)	75.26 (1)
ThreeOfNine	53.51 (3)	100.00** (1.5)	100.00** (1.5)
TicTacToe	65.34 (3)	98.54** (2)	99.69** (1)
BreastCancer	94.99 (3)	95.28 (2)	96.42 (1)
ParityBits	48.05 (3)	79.49** (2)	80.18** (1)
SolarFlare	83.21 (1)	82.93 (2)	80.96 (3)
ClevelandHeart	55.55 (3)	71.85 (2)	82.96** (1)
Australian	55.51 (2)	55.51 (2)	55.51 (2)
German-org	72.00 (2)	70.00 (3)	73.70 (1)
Ionosphere	66.10 (3)	96.30** (1)	96.01** (2)
Tokyo	90.19 (2)	89.47 (3)	91.66 (1)
Sonar	92.30 (1.5)	92.30 (1.5)	88.92 (3)
Average accuracy	74.13	84.21	84.94
Average rank	2.2000	2.0333	1.7667

**Statistically significant at the level of 0.001 when compared to *TrnAcc*

Bold values are the best accuracy for each dataset

Table 11 Average Accuracy of 4-RBF Kernel (%)

Datasets	Objective Functions of ES		
	<i>TrnAcc</i>	<i>BdOfGenErr</i>	<i>5SubsetOn TrnAcc</i>
Checkers	90.09 (1)	88.54 (2)	83.82 (3)
Spiral	100.00 (2)	100.00 (2)	100.00 (2)
LiverDisorders	70.15 (1)	68.99 (3)	69.57 (2)
IndiansDiabetes	74.47 (2)	71.98 (3)	75.26 (1)
ThreeOfNine	53.51 (3)	99.80** (2)	100.00** (1)
TicTacToe	65.34 (3)	98.44** (2)	99.69** (1)
BreastCancer	95.85 (2.5)	95.85 (2.5)	96.42 (1)
ParityBits	48.05 (3)	79.49** (2)	80.18** (1)
SolarFlare	83.21 (1)	82.93 (2)	80.96 (3)
ClevelandHeart	55.55 (3)	78.15** (2)	83.33** (1)
Australian	55.51 (1.5)	55.51 (1.5)	55.22 (2)
German-org	73.50 (2)	70.00 (3)	75.40 (1)
Ionosphere	66.10 (3)	95.72** (2)	96.01** (1)
Tokyo	89.15 (3)	90.30 (2)	91.56 (1)
Sonar	92.78 (1.5)	92.78 (1.5)	88.92 (3)
Average accuracy	74.22	84.57	85.09
Average rank	2.1667	2.1667	1.6000

**Statistically significant at the level of 0.001 when compared to *TrnAcc*

Bold values are the best accuracy for each dataset

Table 12 Average Accuracy of 5-RBF Kernel (%)

Datasets	Objective Functions of ES		
	<i>TrnAcc</i>	<i>BdOfGenErr</i>	<i>5SubsetOnTrnAcc</i>
Checkers	90.10 (1)	88.02 (2)	83.82 (3)
Spiral	100.00 (2)	100.00 (2)	100.00 (2)
LiverDisorders	70.15 (1)	69.57 (2)	68.99 (3)
IndiansDiabetes	73.17 (2)	71.99 (3)	74.61 (1)
ThreeOfNine	53.51 (3)	100.00 ** (1.5)	100.00 ** (1.5)
TicTacToe	65.34 (3)	98.44** (2)	99.69 ** (1)
BreastCancer	95.42 (2)	94.99 (3)	96.42 (1)
ParityBits	48.05 (3)	79.49** (2)	80.18 ** (1)
SolarFlare	83.12 (1)	82.93 (2)	80.96 (3)
ClevelandHeart	55.55 (3)	68.89 (2)	83.33 ** (1)
Australian	55.65 (2)	55.51 (3)	56.96 (1)
German-org	71.20 (2)	70.00 (3)	76.70 (1)
Ionosphere	66.10 (3)	96.01 ** (1)	95.73** (2)
Tokyo	90.82 (2)	88.94 (3)	91.66 (1)
Sonar	92.30 (1.5)	92.30 (1.5)	88.92 (3)
Average accuracy	74.03	83.81	85.20
Average rank	2.1000	2.2000	1.7000 ***

**Statistically significant at the level of 0.001 when compared to *TrnAcc*

***Statistically significant by Wilcoxon signed-ranks test at the level of 0.1 when compared to *TrnAcc*

Bold values are the best accuracy for each dataset

TicTacToe, ParityBits, and Ionosphere. As shown in Fig. 6, the average accuracies on 15 datasets of *5SubsetOnTrnAcc* are the highest for all *n*-RBF kernels while those of *BdOfGenErr* are slightly lower. When *5SubsetOnTrnAcc* is used

as an objective function in the evolutionary process, the average accuracy of *n*-RBF increases with the number of RBF terms (*n*). In contrast, the average accuracy of *BdOfGenErr* decreases for some of the multi-scale RBF kernels.

Although *BdOfGenErr* does not provide higher average accuracies than those of *5SubsetOnTrnAcc*, it requires less time to evaluate a set of hyperparameters. *5SubsetOnTrnAcc* has to train a classifier five times when evaluating each set of hyperparameters. Therefore, *BdOfGenErr* is a good choice for an objective function, which yields good results and is easy to implement. In general situations, training an SVM is an off-line process. Therefore, the training time could be disregarded, and in these situations, *5SubsetOnTrnAcc* is the best choice that yields the best accuracies in many datasets. When *5SubsetOnTrnAcc* is used as an objective function, there is a trend that the average accuracy on all 15 datasets increases with the number of terms of RBF kernels.

However, as shown in Table 13, the average accuracies of *5SubsetOnTrnAcc* increase until a specific number of terms of RBF kernels is reached. After that, they are unchanged or slightly decrease. Therefore, increasing the number of terms of the RBF kernels contributes positively to the accuracy. Although it is not always the case that the kernel with the largest number of terms yields the best result, more RBF terms usually provide better outcomes and should be employed when there are no time constraints.

Furthermore, the proposed method was compared to the *k*-nearest neighbors (*k*-NN), where the target class is estimated from the voting among *k* nearest training data points. In this research, SVM with the 10-RBF kernel whose parameters are selected by ES that uses *5SubsetOnTrnAcc*

Fig. 6 Average accuracies on 15 datasets

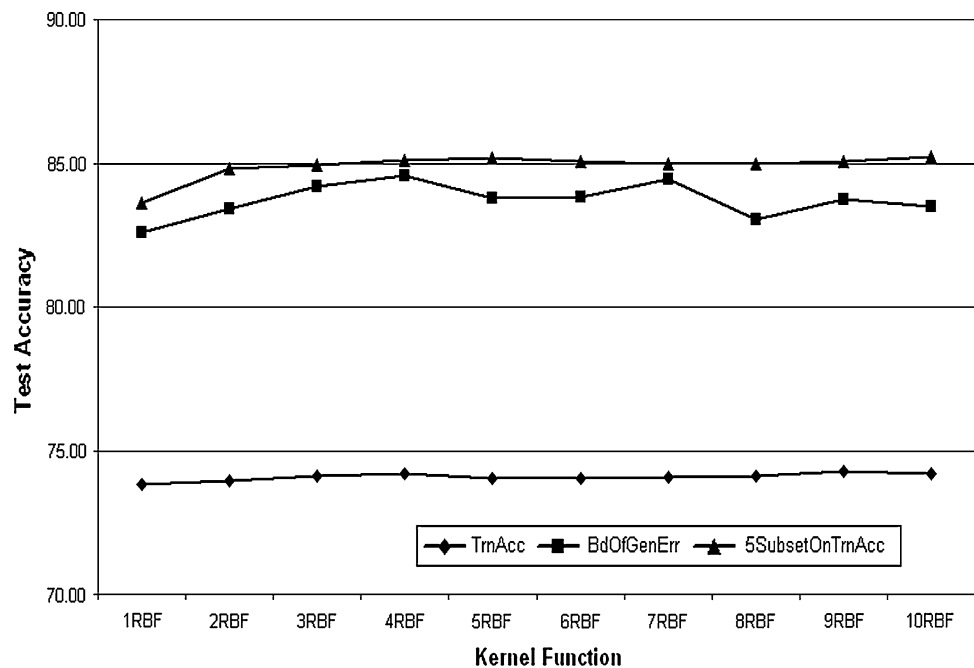


Table 13 Average accuracies of *5SubsetOnTrnAcc* (%)

Datasets	Kernel Function									
	1-RBF	2-RBF	3-RBF	4-RBF	5-RBF	6-RBF	7-RBF	8-RBF	9-RBF	10-RBF
Checkers	83.82	84.35	83.82	83.82	83.82	82.77	82.77	82.77	82.77	82.77
Spiral	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00
LiverDisorders	66.67	67.25	68.99	69.57	68.99	67.83	67.83	67.83	67.83	67.83
IndiansDiabetes	73.30	74.86	75.26	75.26	74.61	75.25	75.25	75.25	75.25	75.25
ThreeOfNine	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00
TicTacToe	99.69	99.69	99.69	99.69	99.69	99.69	99.69	99.69	99.69	99.69
BreastCancer	94.56	95.99	96.42	96.42	96.42	96.42	96.42	96.42	96.42	96.42
ParityBits	75.78	80.18	80.18	80.18	80.18	80.18	80.18	80.18	80.18	80.18
SolarFlare	80.96	80.96	80.96	80.96	80.96	80.87	80.87	80.87	80.87	80.87
ClevelandHeart	78.15	83.34	82.96	83.33	83.33	82.96	81.85	81.48	82.22	83.33
Australian	55.51	55.51	55.51	55.22	56.96	56.96	56.96	56.96	57.39	58.84
German-org	70.30	73.40	73.70	75.40	76.70	76.70	76.70	76.70	76.40	76.40
Ionosphere	95.16	96.29	96.01	96.01	95.73	95.73	95.73	95.73	95.73	95.73
Tokyo	91.66	91.66	91.66	91.56	91.66	91.66	91.66	91.87	92.18	92.18
Sonar	88.92	88.92	88.92	88.92	88.92	88.92	88.92	88.92	88.92	88.92
Average accuracy	83.63	84.83	84.94	85.09	85.20	85.06	84.99	84.98	85.06	85.23

Bold values are the best accuracy for each dataset

as the objective function is compared to k -NN for $k = 1, 3, 5,$ and 7 . The experimental results are shown in Table 14. The average accuracies of the proposed method are better than those of k -NN on many datasets. Furthermore, the result of the Friedman test shows that the average ranks of 5 algorithms, i.e. 1-NN, 3-NN, 5-NN, 7-NN and the proposed method, on 15 datasets are significantly different from the mean rank $R_j = 3$:

$$\chi^2_F = \frac{12(15)}{5(5+1)} \left[(3.6333^2 + 3.1667^2 + 3.1333^2 + 3.3333^2 + 1.7667^2) - \frac{5(5+1)^2}{4} \right] = 12.3468 \tag{40}$$

$$F_F = \frac{(15-1)12.3468}{15(5-1) - 12.3468} = 3.6290. \tag{41}$$

The critical value of $F(4,56)$ for $\alpha = 0.05$ is 2.54136, so we reject the null-hypothesis. Therefore, the average ranks of these five algorithms are significantly different from the mean rank at a significance level of 0.05.

Then, the Bonferroni–Dunn test is used for pairwise comparisons. The corresponding critical difference (CD) for $q_{0.10}$ is

$$CD = (2.241) \sqrt{\frac{5(5+1)}{6(15)}} = 1.5538. \tag{42}$$

Therefore, the average rank of the proposed method (SVM with the 10-RBF kernel and *5SubsetOnTrnAcc*) is significantly better than the average rank of k -NN for $k = 1$

and 7 ($3.6333 - 1.7667$ and $3.3333 - 1.7667$ are more than 1.5538).

For Holm’s test, the standard error of 5 algorithms and 15 datasets is 0.5774. The corresponding statistics and p values are computed and displayed in Table 15. When $\alpha = 0.05$, the Holm procedure rejects all hypotheses since the corresponding p values are smaller than the adjusted α . These results confirm that the results of SVM with the 10-RBF kernel and objective function *5SubsetOnTrnAcc* are also significantly better than the results of k -NN.

7 Conclusions and discussions

The main focus of this paper is to improve the accuracy of SVM on a classification problem. The non-negative weighted linear combination of multiple RBF kernels is proposed for support vector machines in classification tasks. The RBF kernel is the most popular distance-based kernel that has been applied to various applications and yields good results. Here we show that the performance of the RBF kernel can be further enhanced with the combination of several RBF kernels. The proposed kernel is proved to be an admissible kernel by Mercer’s theorem. Moreover, the proposed kernel has more adaptability for complex problems.

Then, the evolutionary strategy is applied to select the hyperparameters of SVM. The optimum values of both the regularization parameter and the parameters of kernel function are searched. Although other methods for

Table 14 Average accuracies of k -nearest neighbors and a proposed method (%)

Datasets	1-NN	3-NN	5-NN	7-NN	SVM with 10-RBF +ES with $5SubsetOnTrnAcc$
Checkers	83.31 (2)	80.70(5)	81.20 (4)	84.32 (1)	82.77 (3)
Spiral	100.00 (2)	100.00 (2)	95.88 (5)	97.59 (4)	100.00 (2)
LiverDisorders	59.71 (3)	64.64 (2)	58.26 (4)	57.39 (5)	67.83 (1)
IndiansDiabetes	71.21 (5)	74.47 (2)	73.43 (3.5)	73.43 (3.5)	75.25 (1)
ThreeOfNine	79.49 (4)	79.29 (5)	85.55 (3)	98.24 (2)	100.00 (1)
TicTacToe	100.00 (1.5)	100.00 (1.5)	99.69 (3.5)	99.17 (5)	99.69 (3.5)
BreastCancer	94.85 (5)	96.14 (3)	96.28 (2)	95.99 (4)	96.42 (1)
ParityBits	50.58 (4)	49.41 (5)	51.27 (3)	51.75 (2)	80.18 (1)
SolarFlare	73.83 (5)	80.02 (4)	81.33 (2)	82.46 (1)	80.87 (3)
ClevelandHeart	76.67 (5)	80.00 (4)	81.48 (3)	81.85 (2)	83.33 (1)
Australian	66.52 (4)	68.41 (1)	67.39 (2)	66.67 (3)	58.84 (5)
German-org	67.40 (5)	71.40 (4)	72.50 (2)	72.30 (3)	76.40 (1)
Ionosphere	87.76 (2)	84.91 (3)	83.48 (4)	82.91 (5)	95.73 (1)
Tokyo	91.14 (5)	92.08 (3)	92.39 (2)	92.07 (4)	92.18 (1)
Sonar	83.17 (2)	82.69 (3)	81.25 (4)	77.36 (5)	88.92 (1)
Average accuracy	79.04	80.28	80.09	80.90	85.23
Average rank	3.6333	3.1667	3.1333	3.3333	1.7667

Bold values are the best accuracy for each dataset

Table 15 Holm’s Test for comparing between 10-RBF SVM with $5SubsetOnTrnAcc$ and k -NN for $k = 1, 3, 5,$ and 7

i	k -NN	$z = (R_0 - R_i)/SE$	p	$\alpha/(k - i)$
1	1-NN	$(3.6333 - 1.7667)/0.5774 = 3.2328$	0.0012	0.0125
2	7-NN	$(3.3333 - 1.7667)/0.5774 = 2.7132$	0.0067	0.0167
3	3-NN	$(3.1667 - 1.7667)/0.5774 = 2.4247$	0.0153	0.0250
4	5-NN	$(3.1333 - 1.7667)/0.5774 = 2.3668$	0.0179	0.0500

optimizing the parameters can also be used such as gradient based methods, these methods are more complicated or cannot be implemented for some kernel functions of SVM. The $(5 + 10)$ -ES is used in this research because it has the ability to escape from local minima. Moreover, its population size of this ES is not so large, and thus it converges to the optimal solution rather quickly. Three possible objective functions in the evolutionary process, training accuracy, the bound of generalization error, and subset cross-validation on training accuracy, are investigated. From our experimental results, we suggest using the subset cross-validation on training accuracy as an objective function since it provides the best accuracy.

Since the multi-scale RBF kernels are flexible according to the number of terms of RBF, a set of hyperparameters may create a decision surface that overfits the training data, and thus produces misclassification on the test data. Subset cross-validation can avoid this overfitting problem because the training data are divided into many subsets. The parameters that work well on all subsets should perform

well on unseen data. Hence, the subset cross-validation on training accuracy is a good choice of objective function for multi-scale RBF kernels.

The experimental results show the performance of the proposed method in the terms of average accuracy on fivefold cross-validation. The multi-scale RBF kernels yield better results. When SVM uses the proposed kernel, it is able to learn from data very well. Furthermore, the evolutionary strategy is effective in optimizing the hyperparameters. Hence, the proposed method is very suitable for the problems where we have no prior knowledge about their parameters. Besides, this non-negative linear combination can be applied to other Mercer’s kernels such as polynomial, Fourier series, and spectrum kernels, as the general form of the linear combination of Mercer’s kernels has already been proved to be a Mercer’s kernel.

Acknowledgments The authors acknowledge the financial support provided by the Thailand Research Fund, the Royal Golden Jubilee Ph.D. Program, and the 90th Anniversary of Chulalongkorn University Fund (Ratchadaphiseksomphot Endowment Fund). The authors also would like to thank Ananlada Chotimongkol for proofreading the paper.

References

Ayat NE, Cheriet M, Remaki L, Suen CY (2001) KMOD—a new support vector machine kernel with moderate decreasing for pattern recognition. In: Proceedings on document analysis and recognition, pp 1215–1219

- Bartlett P, Shawe-Taylor J (1998) Generalization performance of support vector machines and other pattern classifiers. In: Schölkopf B, Burges CJC, Smola AJ (eds) *Advances in kernel methods—support vector learning*. MIT Press, Cambridge
- Beyer H-G, Schwefel HP (2002) Evolution strategies: a comprehensive introduction. *Nat Comput* 1(1):3–52
- Blake CL, Merz CJ (1998) UCI repository of machine learning databases [Online]. University of California, Department of Information and Computer Science, Irvine, CA. <http://www.ics.uci.edu/~mllearn/MLRepository.html>
- Blum A, Kalai A, Langford J (1999) Beating the hold-out: bounds for K-fold and progressive cross-validation. *Computational Learning Theory*, pp 203–208
- Blumer A, Ehrenfeucht A, Haussler D, Warmuth MK (1989) Learnability and the Vapnik–Chervonenkis dimension. *J ACM* 36(4):929–965
- Burges C (1998) A tutorial on support vector machines for pattern recognition. *Data Min Knowl Discov* 2(2):121–167
- Chapelle O, Vapnik V, Bousquet O, Mukherjee S (2002) Choosing multiple parameters for support vector machines. *Mach Learn* 46(1):131–159
- Chunhong Z, Licheng J (2004) Automatic parameters selection for SVM based on GA. In: *Proceeding of the 5th World congress on intelligent control and automation*, Hangzhou, China 2: 1869–1872
- Cristianini N, Shawe-Taylor J (2000) *An introduction to support vector machines and other kernel-based learning methods*. Cambridge University Press, UK
- deDoncker E, Gupta A, Greenwood G (1996) Adaptive integration using evolutionary strategies. In: *Proceedings of 3rd international conference on high performance computing*, pp 94–99
- Demsar J (2006) Statistical comparisons of classifiers over multiple data sets. *J Mach Learn Res* 7:1–30
- Eads DR, Hill D, Davis S, Perkins S, Ma J, Porter R, Theiler J (2002) Genetic algorithms and support vector machines for time series classification. In: *Proceedings of SPIE vol 4787*, pp 74–85
- Fleuret F, Sahbi H (2002) Scale-invariance of support vector machines based on the triangular kernel. *INRIA Research Report*, N 4601, October 2002
- Fogel DB (1995) *Evolutionary computation: toward a new philosophy of machine intelligence*. IEEE Press, Piscataway
- Friedman M (1937) The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *J Am Stat Assoc* 32:675–701
- Friedrichs F, Igel C (2004) Evolutionary tuning of multiple SVM parameters. In: *12th european symposium on artificial neural networks (ESANN 2004)*. pp 519–524
- Fröhlich H, Zell A (2005) Efficient parameter selection for support vector machines in classification and regression via model-based global optimization. In: *IEEE international joint conference on neural networks (IJCNN 2005)* 3:1431–1436
- Fröhlich H, Chapelle O, Schölkopf B (2003) Feature selection for support vector machines by means of genetic algorithms. In: *15th IEEE international conference on tools with AI (ICTAI 2003)*, pp 142–148
- García S, Herrera F (2008) An extension on statistical comparisons of classifiers over multiple data sets for all pairwise comparisons. *J Mach Learn Res* 9:2677–2694
- García S, Fernández A, Luengo J, Herrera F (2009) A study of statistical techniques and performance measures for genetics-based machine learning: accuracy and interpretability. *Soft Comput* 13(10):959–977
- Goldberg DE (1989) *Genetic algorithms in search, optimization and machine learning*. Addison-Wesley, US
- Guo XC, Yang JH, Wu CG, Wang CY, Liang YC (2008) A novel LS-SVMs hyper-parameter selection based on particle swarm optimization. *Neurocomputing* 71(16–18):3211–3215
- Herrera F, Lozano M, Verdegay JL (1996) Tackling real-coded genetic algorithms: operators and tools for behavioural analysis. *Artif Intell Rev* 12(4):265–319
- Hiroshi S, Ken-ichi N, Mitsuru N (2001) Dynamic Time-Alignment Kernel in Support Vector Machine. *Adv Neural Inf Process Syst NIPS2001* 14(2):921–928
- Holm S (1979) A simple sequentially rejective multiple test procedure. *Scand J Stat* 6:65–70
- Howley T, Madden MG (2005) The genetic kernel support vector machine: description and evaluation. *Artif Intell Rev* 24:379–395
- Igel C (2005) Multi-objective model selection for support vector machines. In: *Proceedings of the third international conference on evolutionary multi-criterion optimization vol 3410*, pp 534–546
- Iman L, Davenport JM (1980) Approximations of the critical region of the friedman statistic. *Communications in Statistics A9*: 571–595
- Kääriäinen M, Langford J (2005) A comparison of tight generalization error bounds. In: *Proceedings of the 22nd international conference on machine learning*, Bonn, Germany, pp 409–416
- Kecman V (2001) *Learning and soft computing: support vector machines, neural networks, and fuzzy logic models*. MIT Press, London
- Koji T (1999) Support vector classifier with asymmetric kernel functions. In: *Proceedings of European symposium on artificial neural networks*, Bruges (Belgium), pp. 183–188
- Markatou M, Tian H, Biswas S, Hripcsak G (2005) Analysis of variance of cross-validation estimators of the generalization error. *Journal of Machine Learning Research* 6:1127–1168
- Mitchell TM (1997) *Machine Learning*. McGraw-Hill, New York
- Müller K, Mika S, Rätsch G, Tsuda K, Schölkopf B (2001) An introduction to kernel-based learning algorithm. *IEEE Transactions on Neural Networks* 12(2):181–201
- Ong C, Smola A, Williamson R (2005) Machine learning using hyperkernels. *J Mach Learn Res* 6:1043–1071
- Rameswar D, Haruhisa T (2004) Kernel selection for the support vector machines. *IEICE Trans Inf Syst* E87-D(2)
- Rechenberg I (1965) *Cybernetic solution path of an experimental problem*. Ministry of Aviation, Royal Aircraft Establishment, UK
- Rechenberg I (1973) *Evolutionsstrategie: Optimierung Technischer Systeme Nach Prinzipien der Biologischen Evolution*. Frommann-Holzboog, Stuttgart
- Runarsson TP, Sigurdsson S (2004) Asynchronous parallel evolutionary model selection for support vector machines. *Neural Information Processing—Letter and Reviews*, 3(3)
- Russell S, Norvig P (2003) *Artificial intelligence: a modern approach*. Prentice-Hall, Englewood Cliffs
- Schölkopf B, Smola A (2002) *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT Press, London
- Schölkopf B, Burges C, Smola A (1998) *Advances in kernel methods: support vector machines*. MIT Press, Cambridge
- Schwefel H-P (1981) *Numerical optimization for computer models*. Wiley, Chichester
- Schwefel H-P (1995) *Evolution and optimum seeking*. Wiley, New York
- Shawe-Taylor J, Cristianini N (2004) *Kernel methods for pattern analysis*. Cambridge University Press, UK
- Smits GF, Jordaan EM (2002) Improved SVM regression using mixtures of kernels. In: *Proceedings of the 2002 international joint conference on neural networks vol 3*, pp 2785–2790
- Storn R, Price K (1997) Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces. *J Global Optim* 11:341–359

- Tan Y, Wang J (2004) Support vector machine with a hybrid kernel and minimal Vapnik–Chervonenkis dimension. *IEEE Trans on Know Data Eng* 16(4):358–395
- Vapnik VN (1995) *The nature of statistical learning theory*. Springer, New York
- Vapnik VN (1998) *Statistical Learning Theory*. John Wiley and Sons, New York
- Vapnik V, Chervonenkis A (1971) On the uniform convergence of relative frequencies of events to their probabilities. *Theory Probab Appl* 16(2):264–280
- Xuefeng L, Fang L (2002) Choosing multiple parameters for SVM based on genetic algorithm. In: *International conference on signal processing (ICSP 2002) vol 1*, pp 117–119
- Zhang L, Zhou W, Jiao L (2000) Support vector machines based on scaling kernels. In: *6th international conference on signal processing*, vol 2, pp 1142–1145
- Zhang L, Zhou W, Jiao L (2004) Wavelet support vector machine. *Trans Syst Man Cybern B Cybern* 34:34–39
- Zhou S, Wu L, Yuan X, Tan W (2007) Parameters selection of SVM for function approximation based on differential evolution. In: *Proceedings of the international conference on intelligent systems and knowledge engineering (ISKE 2007)*, Chengdu, China