International Journal on Artificial Intelligence Tools
Vol. 24, No. 4 (2015) 1550013 (29 pages)

DOI: 10.1142/S021821301550013X

World Scientific
www.worldscientific.com

An Empirical Analysis of Evolved Radial Basis Function Networks and Support Vector Machines with Mixture of Kernels

Ch. Sanjeev Kumar Dash and Pulak Sahoo
Silicon Institute of Technology, Silicon Hills
Patia, Bhubaneswar-751024, Odisha, India
sanjeev_dash@yahoo.com, sahoo_pulak@yahoo.com

Satchidananda Dehuri
Department of Systems Engineering, Ajou University
San 5, Woncheon-dong, Yeongtong-gu, Suwon-443-749, South Korea
satchi@ajou.ac.kr

Sung-Bae Cho
Soft Computing Laboratory, Department of Computer Science
Yonsei University, 134 Shinchon-dong, Sudaemoon-gu, Seoul 120-749, South Korea
sbcho@yonsei.ac.kr

Received 24 July 2013
Accepted 11 January 2015
Published 21 August 2015

Classification is one of the most fundamental and formidable tasks in many domains including biomedical. In biomedical domain, the distributions of data in most of the datasets into predefined number of classes is significantly different (i.e., the classes are distributed unevenly). Many mathematical, statistical, and machine learning approaches have been developed for classification of biomedical datasets with a varying degree of success. This paper attempts to analyze the empirical performance of two forefront machine learning algorithms particularly designed for classification problem by adding some novelty to address the problem of imbalanced dataset. The evolved radial basis function network with novel kernel and support vector machine with mixture of kernels are suitably designed for the purpose of classification of imbalanced dataset. The experimental outcome shows that both algorithms are promising compared to simple radial basis function neural networks and support vector machine, respectively. However, on an average, support vector machine with mixture kernels is better than evolved radial basis function neural networks.

Keywords: Classification; imbalanced data; radial basis function neural networks; support vector machine; differential evolution; kernels.

1. Introduction

In medical science, classification has been the subject of extensive study and research since long and also been treated as a formidable task for machine learning researcher.[5]

The reason is that diagnosis and classification of samples are very much accuracy-driven. Techniques which are less accurate could lead to inaccurate diagnosis and subsequent mistreatment leading to disastrous outcomes. In addition, the diagnosis and classification must be done at an early stage for the follow-up treatments to be effective and eventually result in a higher cure and survival rate. For example, liver diseases need to be diagnosed at an early stage by analyzing the levels of different enzymes present in the blood samples. This can result in better cure and survival rates for liver patients. Diabetes is a disease that gets worse progressively. Hence an early detection of diabetes preferably in pre-diabetic state from oral glucose tolerance, blood pressure, skin fold thickness, et cetera can go a long way in decreasing the risks involved.

Many other problems such as, the prediction of the semen quality in men from the data related to various environmental factors and lifestyle is becoming a new challenge for biomedical researchers.[22] Fertility rates have dramatically decreased in the last two decades, especially in men. It has been described that environmental factors, as well as life habits, may affect semen quality.[56] Similarly classification techniques can help to predict whether a previous donor will donate blood in coming times. This will help the patron to manage the functioning of the Blood Transfusion Service Centre in a better manner. It is also common that, these datasets are highly skewed. This means that when the number of instances of certain classes is much lower than the instances of the other classes. The importance of these datasets resides in the fact that usually a minority class represents the concept of interest, for example patients with illness, where healthy patient represents the counterpart of that concept. Additionally, imbalanced data are implicit to meet real world applications such as satellite image classification, risk management, etc.

Several mathematical, statistical, and machine learning techniques[4,7,9,25,27,31,35] have been implemented to model and solve medical related problems. However, they have their own limitations. In this paper, authors studied two kernel based methods such as differential evolution[38,68] based RBFNs (DE-RBFNs) and support vector machine with mixture of kernels (SVM_{mk}) for solving some medical related classification problems. Moreover, these classifiers usually have a bias towards majority class during the learning process,[22] hence the data driven approach is adapted, which preprocess or re-sample the data in order to diminish the effects of their class imbalance. However, in the case of highly imbalanced datasets, to avoid the limitations of data driven approach, both data and method driven approach has been used.

Radial basis function networks (RBFNs)[11,12,49] is not only been studied in disciplines like pattern recognition,[61] multi-media applications,[17] computational finance[55] and software engineering,[32] but also extensively in medical science.[4,7,35,60] It is emerged as a variant in late 1980's, however its root entrenched in much older pattern recognition, numerical analysis, and other related fields.[47] Radial basis function networks have attracted the attention of many researchers because of its: (i) universal approximation[47] (ii) compact topology[75] and (iii) faster learning speed.[43] In the context of universal approximation, it has been proved that "a radial basis function networks can approximate arbitrarily well any multivariate continuous function on a compact domain if a sufficient

number of radial basis function units are given".[69] Note, however that the number of kernels (k) chosen, need not equal to the number of training patterns (n). In general, it is better to have k much less than n i.e., $k << n$. Besides the gain in computational complexity, the reduction in the number of kernels is beneficial for the generalization capability of the resulting model. Hence, to improve the performance of RBFNs, a variety of learning procedure has been developed.[2,13,74,76] It is normally divided into two phases: (1) the adjustment of the connection weight vector; and (2) the modification of parameter of RBF units such as center and spreads.[23,62] To keep this trend intact, in one part, we focus on improvising of RBFNs from two aspects such as devising a new kernel with a suitable modification of existing one and determination of hidden centers, along with spreads by differential evolution.[57–59] The remaining parameters of RBFNs that constitute the final classifier have been optimized in the following way:

$$Y = W\Phi, \tag{1}$$

where Y is a matrix of output value, Φ is the basis matrix, and W is a matrix of second layer weights to be estimated. This is a classical least square estimation problem. A necessary condition for $\|Y - W\Phi\|^2$ to be minimized is that W must satisfy Eq. (2).

$$W = (\Phi^T\Phi)^{-1}\Phi^T Y. \tag{2}$$

The motivation using differential evolution (DE)[44,45] over other evolutionary algorithms (EAs), such as genetic algorithms (GAs)[20,40] is that in DE string encoding are typically represented as real valued vectors, and the perturbation of solution vectors is based on the scaled difference of two randomly selected individuals of the current population. Unlike GA, the resulting step size and orientation during the perturbation process automatically adopt to the fitness function landscape.

Alongside, it has been seen that SVM with its variants for classification[67] is getting popular in machine learning research community.[25,27,31] SVM separates data points into two categories by a decision surface called a hyper plane. The hyper plane maximizes the margin of separation between data points belongs to two classes. This hyper plane is then used to predict the class label of unknown data points based on which side of the plane it falls. In recent years, several methods have been proposed to combine multiple kernels instead of using a single one[24] different kernels may correspond to using different notions of similarity or may be using information coming from multiple sources. Hence to determine the best classification performance, we have used SVM with three traditional kernels (Linear, Polynomial and RBF) as well as three mixture kernels (Linear + Polynomial, Polynomial + RBF and Linear + RBF). The kernel parameters tuned iteratively for each traditional kernel to improve classification accuracy. In case of mixture kernels, the best proportion of each component kernel iteratively determined. The proposed method produces improvement in classification[53] performance. This technique is then compared with DE-RBFNs.

This paper is set out as follows. Section 2 gives overview of RBF network, support vector machine, and differential evolution. In Section 3 the related work is summarized.

The proposed method is discussed in Section 4. Experimental setup, results, and analysis are presented in Section 5. Section 6 concludes the paper with a future line of research.

2. Background

The background of this research work is presented in this section. In Subsection 2.1, RBF network classifier is discussed. Support vector machine and its importance is the focus of Subsection 2.2. Differential evolution, a new meta-heuristic computing paradigm, is discussed in Subsection 2.3.

2.1. *RBF network classifier*

RBF network is a supervised learning algorithm build for classification.[6,65] RBF network is a three layer feed forward network where each hidden unit implements a radial activation function and each output unit implements a weighted sum of hidden units' output. This network is a special class of neural network[26] in which the activation of a hidden neuron is determined by the distance between the input vector and a prototype vector. Prototype vectors refer to centers of clusters formed by the patterns or vectors in the input space. The centers are determined during RBF training. Figure 1 shows a RBF network with m hidden neurons and one neuron in the output layer. Without loss of generality, it can be expressed as follows:

$$f(w;\mu;x) = w_0 + \sum_{i=1}^{M} w_i \phi\left(\left\|x - \vec{\mu}_i\right\|_2\right), \tag{3}$$

where, $\phi(\|\cdot\|)$ is chosen radial basis function; M is the number of hidden neurons; $x \in \Re^n$ is the input; $\|\cdot\|_2$ is the Euclidean norm; $\mu = (\mu_1, \mu_2, \ldots, \mu_m)$ is a vector of the centers of RBFN; and $w = (w_1, w_2, \ldots, w_m)$ is a linear weight vector.

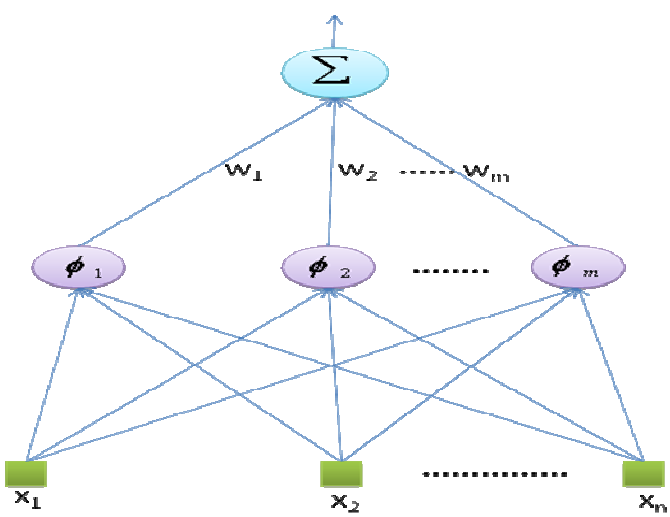

Fig. 1. Pictorial representation of a radial basis function network.

RBFN with it radially symmetric activation function produces a localized response to inputs. There are n number of neurons (c.f., Figure 1) of the input layer that connect the network to the environment. The second layer consists of m kernels units that carry out a non-linear transformation from the input space to the hidden space. Different radial basis functions have been used in the literature such as thin plate spline $\phi(x) = x^2 \log(x)$, Hardy multi-quadrics $\phi(x) = \sqrt{x^2 + c}$, $(c \geq 0)$, the function $\phi(x) = x^k$ (k is odd integer), and Gaussian kernel $\phi(x) = \exp(-x^2/2\sigma^2)$, etc. Usually, the non-linear transformation is based on Gaussian kernel as described in Eq. (4) is considered in this paper with a modification (details given in Section 3).

$$\phi_i(x) = \exp\left(-\frac{\left\|x - \vec{\mu}_i\right\|_2^2}{2\sigma_i^2}\right), \tag{4}$$

where μ_i, σ_i and ϕ_i are center, spread, and the output of ith hidden unit. The inter-connection between the hidden and output layer forms linear weighted connections w_i. The output layer, a summation unit, supplies the response of the network to the outside world.

2.2. *Support vector machine*

Support vector machine (SVM)[37,54] is a supervised machine learning algorithm and is widely used for linear and non-linear classification and non-linear regression. In a nutshell, SVM uses a nonlinear mapping to transform the original training data into a higher dimension. Within this new dimension, it searches for the linear optimal separating hyperplane. With an appropriate non-linear mapping to a sufficiently high dimension, data from two classes can always be separated by a hyperplane (c.f., Figure 2).

The SVM finds this hyperplane using *support vectors* and *margins* (c.f., Figure 3).

SVM is known to be very resistant to the over-fitting problem and achieves a high generalization performance[63]. An important property of SVM is that, training SVM is equivalent to solving a linearly constrained quadratic programming problem and the solution of SVM is unique and globally optimal, unlike neural networks training which requires nonlinear optimization with the danger of achieving local minima.[39] SVM can

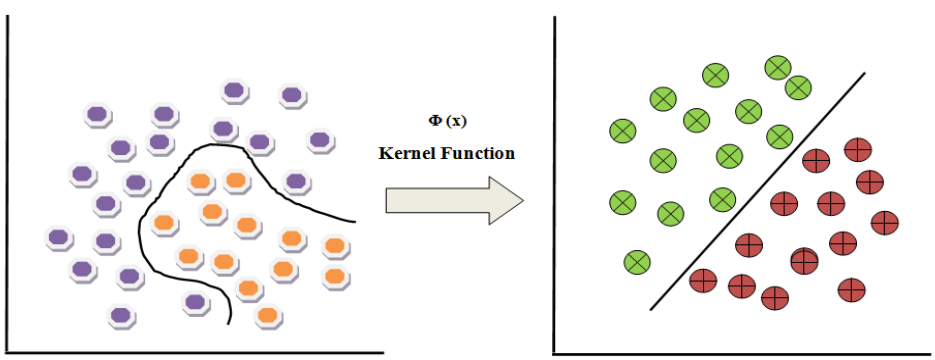

Fig. 2. Mapping by kernel function from non-linearly separable space to linearly separable space.

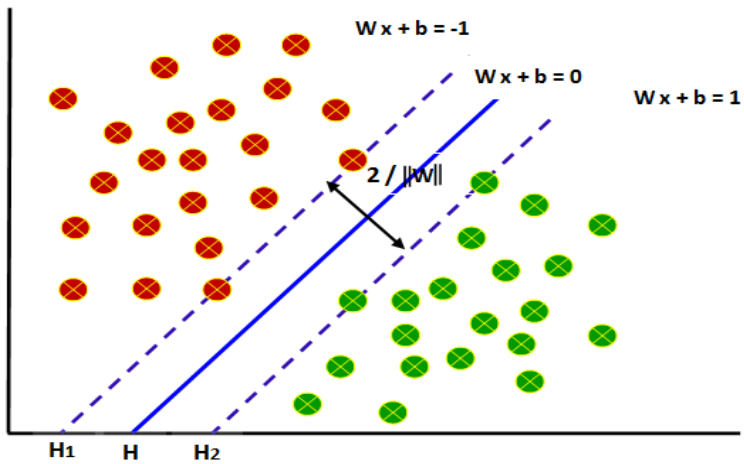

Fig. 3. An optimal hyperplane along with margins and support vectors.

Table 1. Different kernel functions used in SVM.

Name of the Kernel	Mathematical Formula
Linear Kernel	$K(x_i, x_j) = x_i^T \cdot x_j$
Polynomial Kernel	$K(x_i, x_j) = (x_i^T \cdot x_j)^p$ or $K(x_i, x_j) = (x_i^T \cdot x_j + 1)^p$, where $p > 1$ is the degree of the polynomial.
RBF (Gaussian) Kernel	$K(x_i, x_j) = \exp\left(-\|x_i - x_j\|^2 / 2\sigma^2\right)$, where σ is the width parameter.
Sigmoidal Kernel	$\tanh(kx_i^T \cdot x_j - \delta)$, where k and δ are parameters.

perform both linear and non-linear classification[8] using different kernel functions which implicitly map their inputs into high-dimensional feature spaces. The original samples may be stated in a finite dimensional space which is not often linearly separable. Hence the original finite-dimensional space needs to be mapped into a much higher-dimensional space, making the separation easier. This is done by defining them in terms of a kernel function $K(x_i, x_j)$ selected to suit the problem. The commonly used kernel functions[42] are enumerated in Table 1.

Using SVM with the above mentioned traditional kernels (Linear, Polynomial, and Gaussian RBF) may not always yield the best classification performance. In such cases, hybridizing or mixing more than one kernel has potential to improve the classification performance. The hybridization can be achieved by combining two kernels in different proportions. Three such examples are: χ_1·Linear + χ_2·Polynomial, χ_1·Polynomial + χ_2·RBF, and χ_1·Linear + χ_2·RBF. The mixture of different kernels is beneficial because they may complement each other.

2.3. *Differential evolution*

Differential evolution (DE)[57,59] is a population based stochastic search algorithm which typically operates on real valued chromosome encodings and inspired by the *Nelder-Mead* simplex method.[58] Like GAs, DE maintains a population of potential solution encodings which are then perturbed in an effort to uncover yet better solutions to a problem in hand. DE like GAs has also three basic steps: *selection, crossover*, and *mutation*. However in DE, the mutation is a distinct novelty. The perturbation of solution vectors is based on the scaled difference of different individuals of the current population, borrowing ideas from the *Nelder-Mead* simplex method. One of the advantages of this approach is that the resulting 'step' size and orientation during the perturbation process automatically adapts to the fitness function landscape.

Over a decade, several DE algorithms have been reported in literatures[14,50,65] we primarily describe a version of the algorithm based on the DE/rand/1/bin scheme.[59] The different variants of the DE algorithm can be described by the notion DE/*x*/*y*/*z*, where *x* specifies the base vector to be mutated (*rand* if it is randomly selected, or *best* if the best individual in the population is selected), *y* is the number of difference vectors used, and *z* denotes the crossover scheme (*bin* for crossover based on independent binomial experiments, and *exp* for exponential crossover).

A population of *n, d*-dimensional vectors $x_i = (x_{i1}, x_{i2},\ldots, x_{id})$, $i = 1\ldots n$ each of which encode a solution is randomly initialized and evaluated using a fitness function $f(\cdot)$. During the search process, each individual (i) is iteratively refined. The following three steps are required while execution.

(i) **Mutation**: Create a donor vector which encodes a solution, using randomly selected members of the population.
(ii) **Crossover:** Create a trial vector by combining the donor vector with i .
(iii) **Selection:** By the process of selection, determine whether the newly-created trial vector replaces i in the population or not.

In DE/rand/1/bin variant, the mutation operator for each vector $x_i(t)$ of the population, a donor vector $v_i(t + 1)$ is obtained by Eq. (5).

$$v_i(t+1) = x_k(t) + f_m * (x_l(t) - x_m(t)) , \tag{5}$$

where $k, l, m \in 1,\ldots,n$ are mutually distinct, randomly selected indices, i.e., $k \neq l \neq m \neq i$, $(x_k(t))$ is referred to as the base vector and $(x_l(t) - (x_m(t))$ is referred as difference vector. Selecting three indices randomly implies that all members of the current population have the same chance of being selected, and therefore influencing the creation of the difference vector. The difference between vectors x_l and x_m is multiplied by a scaling parameter f_m called mutation factor and the range of the parameter must be associated to it, like $f_m \in [0, 2]$. The mutation factor controls the amplification of the difference between x_l and x_m which is used to avoid stagnation of the search process. There are several alternative versions of the above process for creating a donor vector with different dynamic/static setting of f_m.[14,50,66]

A notable feature of the mutation step in DE is that it is self-scaling. The size/rate of mutation along each dimension stems solely from the location of the individuals in the current population. The mutation step self-adapts as the population converges leading to a finer-grained search. In contrast, the mutation process in GA is typically based on (*or* draws from) a fixed probability density function.

Following the creation of the donor vector, a trial vector $u_i(t+1) = (u_{i1}, u_{i2}, u_{i3},\ldots,u_{id})$ is obtained by Eq. (6).

$$u_i(t+1) = \begin{cases} v_{ip}(t+1), & \text{if } (rand \le c_r) \text{ or } (i = rand(ind)) \\ x_{ip}(t), & \text{if } (rand > c_r) \text{ and } (i \ne rand(ind)) \end{cases}, \tag{6}$$

where $p = 1, 2,\ldots,d$, *rand* is a random number generated in the range (0, 1), C_r is the user-specified crossover constant from the range (0, 1), and rand*(ind)* is a randomly chosen index from the range $(1, 2,\ldots,d)$. The random index is used to ensure that the trial vector differs by at least one element from $x_i(t)$. The resulting trial (child) vector replaces its parent if it has higher fitness (a form of selection), otherwise the parent survives unchanged into the next iteration of the algorithm.

Finally, it is assumed that the fitness of the trial vector exceeds that of its parent fitness then replaces the parent as described in Eq. (7).

$$x_i(t+1) = \begin{cases} u_i(t+1), & \text{if } (f(u_i(t+1)) > f(x_i(t)) \\ x_i(t), & \text{otherwise} \end{cases} \tag{7}$$

This replacement scheme is *de-facto*, a one-to-one tournament selection. Price and Storn[50] provide a comprehensive comparison of the performance of DE with a range of other optimizers, including the GA, and report that the results obtained by DE are consistently as good as the best obtained by other optimizers across a wide range of problem instances. Many variants of DE can be obtained from.[14,18,33,36,37,46,52]

3. Related Work

The problem of imbalanced datasets has been tackled by two different approaches: (i) data-driven or (ii) method driven. The first approach is to pre-process the data by under sampling the majority instances or over sampling the minority instances. The later approach is to use a kind of mechanism while building a model e.g., use biases, so that learned model may further away from the positive class. This is done in order to compensate for the skew associated with imbalanced datasets which pushes the hyper-plane closure to the positive class. In the case of kernel based classifiers, this can be done in various ways e.g., change the kernel function to develop this bias, associate penalty constraints to different classes (i.e., modify the cost metric), etc.

Akbani *et al.*,[1] have discussed the factors behind failure of simple SVM in imbalanced datasets and explained why the common strategy of under sampling the training data may not be the best choice for SVM. Hence, to overcome these problems, they have proposed an algorithm, which is outperformed than under sampling SVM and

regular SVM. Khalilia *et al.*,[34] have presented a method to predict disease risk of individuals based on Healthcare Cost and Utilization Project (HCUP) dataset. They have employed an ensemble learning approach based on repeated random sub-sampling. This technique divides the training data into multiple sub-samples, while ensuring that each sub-sample is fully balanced. They have predicted eight disease categories and concluded that the random forest ensemble learning method outperformed SVM, bagging, and boosting in terms of the area under the receiver operating characteristics curve.

Alongside, the trends of kernel based approaches for classification problem is increasing day by day. Hence, some of the proposals related to this trend are discussed below.

Scholkopf *et al.*,[54] have compared the performance of three machines, namely, a classical RBF machine, an SVM with Gaussian kernel, and a hybrid system with the centers determined by the SVM method. The weights of RBF machine are trained by error back-propagation learning strategies. Feoktiistov *et al.*,[72] have proposed a hybrid method by combining DE with Least-Square Support Vector machine (LS-SVM). Zhang *et al.* (2004), have implemented compactly supported RBF kernels to SVM, least squares SVM, and kernel principal component analysis by proposing two quantitative measures: similarity and sparsity. The linear combination of multiple RBF kernels[19] with including weights is proposed in Ref. 48 for SVM. The RBF kernels is most popular distance based kernel that is applied to various applications and yields good results. Hora *et al.*[28] have used DE-RBFN for classification of various financial and medical datasets. In connection to medical data, they have reported a competitive performance compared to base line classifiers. Huang *et al.*[29] optimize the parameters and feature subset without degrading the SVM classification accuracy. They present a genetic algorithm approach for feature selection and parameters optimization of SVM. Min *et al.*,[41] have a proposed a method for improving SVM performance in two aspects: feature subset selection and parameter optimization. GA is used to optimize both a feature subset and parameters of SVM simultaneously for bankruptcy prediction. Wu *et al.*,[63] have used a real-valued genetic algorithm (GA) to optimize the parameters of SVM for predicting bankruptcy. Further, Wu *et al.*,[64] have developed a model called HGA-SVR, for type of kernel function and kernel parameter value optimization in support vector regression (SVR), which is then applied to forecast the maximum electrical daily load. Their model gives better result than previous model HGA (Hybrid genetic algorithm). Arun *et al.*,[3] have proposed twin support vector machine (TSVM) for binary classification. They attempt to solve two modified primal problems of TSVM, instead of two dual problems. Qiu *et al.*,[51] have used bare bones differential evolution (BBDE) to tune the parameters of SVM. According to them the BBDE is a new optimization algorithm, which is a hybrid of the barebones particle swarm optimization (PSO) and differential evolution (DE). Dash *et al.* have used DE for optimizing different kernel parameters of RBFNs in their earlier work.[15,21] They have reported a good comparative study of the classification performances obtained from DE-RBFN[45] and simple RBFN. In Ref. 16 the performances of DE-RBFN and SVM have been studied for web log data.

4. Our Method

The method of this paper is two folded. First we discuss the integrated framework of DE-RBFN and in second fold we discuss the mixture of kernels based SVM. In our first approach, a novel kernel based on medoid is designed to use in RBFNs for solving classification problem. Unlike mean (or centriod) of the kernel, it is a good representative objects of the dataset i.e., whose average dissimilarity to rest of the points in the dataset is minimal. In subsection 4.1 the modified kernel is described. The learning procedure and pseudo-code are discussed in subsection 4.2.

4.1. Mediod *based Gaussian kernel in RBFNs*

A Gaussian RBFNs with one output neuron and M hidden neurons can be expressed as:

$$f(w;\mu;\sigma;x) = w_0 + \sum_{j=1}^{M} w_j \left(\exp\left(-\frac{\left\| x - \vec{\mu}_j \right\|_2^2}{2\sigma_j^{\,2}} \right) \right),$$

where w_j, w_0, μ_i, and σ_i are the connection weights, bias connection weight, centers, and spreads respectively and x is the input vector to RBF network respectively. The center vectors $\{\mu_j\}_{j=1}^{M}$ are fixed points in d-dimensional input space. Theoretical and numerical studies show that the performance of RBFN depends on the locations of centers and is regardless of which radial basis function is used in hidden neurons. The Gaussian kernel is local in the sense that

$$\lim_{\|x\| \to \infty} f\left(\left\| x - c_i \right\| \right) = 0$$

i.e., changing parameters of one neuron has a small effect for input values that are far away from the centers of that neuron (i.e., such type of case may be arise in the case of noise sample). The algorithm based on centers (i.e., mean vector) is very sensitive to outliers as because an object with an extremely large value may substantially distort the distribution of data. Therefore, to distinguish such sensitivity, we consider a data object which is nearest neighbor to mean vector (i.e., mean vector may not be a data object). Hence, based on the definition of medoid the new Gaussian basis function is defined as:

$$\phi_j(x) = \exp\left(-\frac{\left\| x - \vec{m}_j \right\|_2^2}{2\sigma_j^{\,2}} \right), \tag{8}$$

where $m_j = nn(\mu_j)$ i.e., m_j is the object nearest to the mean μ_j. Hence, the RBFN based on Gaussian radial basis function with modified medoid is expressed as:

$$f(w;m;\sigma;x) = w_0 + \sum_{j=1}^{M} w_j . \phi_j(x). \tag{9}$$

Although our earlier work (Dash *et al.*)[15] has already perform well in web log analysis but studying it in the domain of medical with imbalanced data may further creates an

opportunities of parametric analysis and enlarge its scope of application. To this end, let us examine the dependence of optimal medoid locations on desired outputs.

We know that the design of RBFN in learning of a mapping described by a set $\{x_i, y_i\}_{i=1}^{|T|}$ of $|T|$ input-output patterns can be defined as a minimization problem where the cost is a measure of the distance between actual and desired output sets. The optimal Gaussian RBFNs (mediods, spreads, and linear weights) can minimize the following mean squared output error over the finite set of $|T|$ training samples.

$$E(w;m;\sigma) = \frac{1}{|T|}\sum_{i=1}^{|T|}\left\|y_i - f(w;m;\sigma;x_i)\right\|_2^2$$

$$= \frac{1}{|T|}\sum_{i=1}^{|T|}(y_i - f(w;m;\sigma;x_i))^2 \ , \tag{10}$$

where $w = (w_1, w_2,\ldots,w_M)$, is a vector of linear weights; $m = (\vec{m}_1, \vec{m}_2, ..., \vec{m}_M)$ is a vector of medoids of RBF units; and $\sigma = (\sigma_1, \sigma_2,\ldots,\sigma_M)$ is a vector of spread of RBF units. The output error in (32) can be written in the matrix form as follows:

$$E(w;m;\sigma) = \frac{1}{|T|}\left\|e\right\|_2^2 = \frac{1}{|T|}\left\|Y - W\Phi\right\|_2^2 \tag{11}$$

Substituting the optimal value of the linear weight vector $W = (\Phi^T\Phi)^{-1}\cdot\Phi^T\cdot Y = \Phi^+ Y$, where $\Phi^+ = (\Phi^T\Phi)^{-1}\cdot\Phi^T$ in (33), we get,

$$E(w;m;\sigma) = \frac{1}{|T|}\left\|Y - W\Phi\right\|_2^2 = \frac{1}{|T|}\left\|Y - \Phi^+\cdot Y\cdot\Phi\right\|_2^2$$

$$= \frac{1}{|T|}\left\|(I - \Phi^+\Phi)Y\right\|_2^2 \ , \tag{12}$$

where Φ^+ is the pseudo-inverse of the matrix Φ. From Eq. (12) it is concluded that the performance of the RBFNs not only depends on medoids, spreads, and linear weights but also the desired output vector Y. Therefore, in this paper, during evolution of these parameters, the output vector may also take into account.

4.2. *Learning procedure*

As mentioned there are two phases within the learning procedure.[16] In phase one differential evolution is employed to reveal the centers and spread of the RBFNS. Although centers, spread and weights can be evolved using DE, but here we restrict ourselves with evolving only centers and spreads. This ensures efficient representation of an individual of DE. If we encode all the parameters such as centers, spread and weights into a individual chromosome, the chromosome length is too long and the search space becomes too large, which results in a very slow convergence rate. Since the performance of the RBFNs mainly depends on medoid and spread of the kernel, we just encode the medoids and spread into an individual chromosome for stochastic search.

Suppose the maximum numbers of kernels are set to K_{max}, then the structure of the individual is represented as follows:

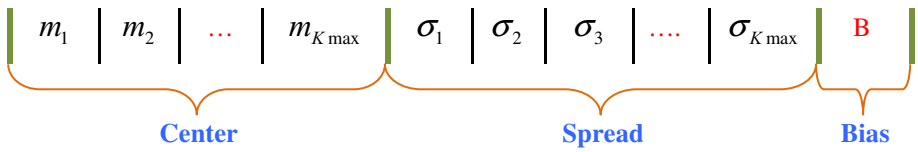

Fig. 4. Structure of the individual.

In other words each individual has three constituent parts such as center, spread and bias. The length of an individual is $2K_{max} + 1$.

The fitness function which is used to guide the search process is defined in Eq. (13).

$$E(w;m;\sigma) = \frac{1}{|T|}\sum_{i=1}^{|T|}(y_i - f(w;m;\sigma;x_i))^2 , \tag{13}$$

where $|T|$ is the total number of training sample, y_i is the actual output and $f(\cdot)$ is the estimated output of RBFNs. Once the centers and spreads are fixed, the task of determining weights in second phase of learning reduces to solving a simple linear system. In this work the pseudo inverse method is adopted to find out a set of optimal weights. Figure 5 illustrate the two phase learning procedure adopted in this work.

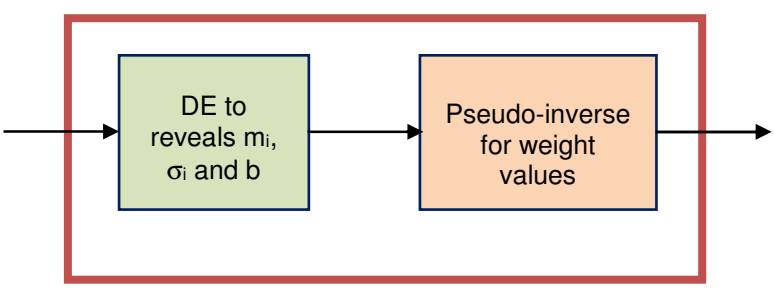

Fig. 5. Two phase learning scheme.

The algorithmic framework of DE-RBFN is described as follows: Initially, a set of n_p individuals (i.e., n_p is the size of the population) pertaining to networks medoids spread and bias is initialized randomly, the individuals have the form:

$$x_i^{(t)} = \left\langle x_{i1}^{(t)}, x_{i2}^{(t)}, ..., x_{id}^{(t)} \right\rangle, \quad i = 1, 2, ..., n_p$$

where $d = 2 \,.\, k_{max} + 1$ and t is the iteration number.

In each iteration, e.g., iteration t, for individual $x_i^{(t)}$ undergoes mutation, crossover and selection as follows:

Mutation. For vector $x_i^{(t)}$ a perturbed vector $V_i^{(t+1)}$ called donor vector is generated according to Eq. (14).

$$V_i^{(t+1)} = x_{r1}^{(t)} + m_f \cdot (x_{r2}^{(t)} - x_{r3}^{(t)}), \tag{14}$$

where m_f is the mutation factor, lies in the interval (0, 2], the indices r_1, r_2 and r_3 are selected randomly from $\{1, 2, 3, \ldots, n_p\}$, such that $r_1 \neq r_2 \neq r_3 \neq i$.

Crossover. The trial vector is generated as given in Eq. (15).

$$u_i^{(t+1)} = u_{i1}^{(t+1)}, u_{i2}^{(t+1)}, \ldots, u_{id}^{(t+1)}$$

$$u_{ij}^{(t+1)} = \begin{cases} v_{ij}^{(t+1)} & \text{if } rand \leq c_r \text{ or } i = rand(1, 2, \ldots, d) \\ x_{ij}^{(t)} & \text{if } rand > c_r \text{ and } i \neq rand(1, 2, \ldots, d) \end{cases} \tag{15}$$

where $j = 1 \ldots d$, *rand* is random number generated in the range (0, 1), c_r is the user specified crossover constant from the range (0, 1) and *rand* $(1, 2, \ldots, d)$ is a random integer from $[1, 2, \ldots, d]$.

The random index is used to ensure that the trial vector differs by, at least one element from $x_i^{(t)}$. The resultant trial (child) vector replaces its parent if it has higher accuracy (a form of selection), otherwise the parent survives unchanged into the next iteration of the algorithm. Figure 6 illustrate this operation in the context of revealing medoid, spread, and bias.

$$\begin{array}{llll}
1 & m_1 \; \hat{m}_1 & rand > cr & m_1 \\
2 & m_2 \; \hat{m}_2 & rand \leq cr & \hat{m}_2 \\
3 & m_3 \; \hat{m}_3 & rand > cr & m_3 \\
4 & m_4 \; \hat{m}_4 & rand > cr & m_4 \\
5 & \sigma_1 \; \hat{\sigma}_1 \Rightarrow & rand \leq cr & \Rightarrow \hat{\sigma}_1 \\
6 & \sigma_2 \; \hat{\sigma}_2 & rand \leq cr & \hat{\sigma}_2 \\
7 & \sigma_3 \; \hat{\sigma}_3 & 7 = rand(1, 2, .., 9) & \hat{\sigma}_3 \\
8 & \sigma_4 \; \hat{\sigma}_4 & rand > cr & \sigma_4 \\
\underbrace{9}_{Index} & \underbrace{b}_{x_i^{(t)}} \; \underbrace{\hat{b}}_{v_i^{(t+1)}} & 9 = rand(1, 2, .., 9) & \underbrace{\hat{b}}_{u_i^{(t+1)}}
\end{array}$$

Fig. 6. Working structure of crossover in DE-RBFNN.

Finally, we use selection operation and obtain the target vector $x_i^{(t+1)}$ as given in Eq. (16).

$$x_i^{(t+1)} = \begin{cases} u_i^{(t+1)} & \text{if } f(x_i^{(t+1)}) \leq f(x_i^{(t)}) \\ x_i^{(t)}, & \text{otherwise} \end{cases}, \qquad j = 1, 2, \ldots, d. \tag{16}$$

After getting medoids, spread and bias from above DE pseudo-code now the weights of the network are computed by pseudo-inverse method as described in Eq. (17).

$$Y = W\Phi \rightarrow W = (\Phi^T \Phi)^{-1} \Phi^T Y. \tag{17}$$

In a nutshell, the pseudo-code for computing spread, medoids, and bias using DE, is as follows:

Begin
$t = 1$;
Initialize a set of n_p *individuals* $x_1^{(t)} = \langle x_{i1}^{(t)}, x_{i2}^{(t)}, \ldots, x_{id}^{(t)} \rangle, \quad i = 1, 2, \ldots n_p$, *randomly*;
For $t = 1$ *to maxG do*
For $i = 1$ *to* n_p *do*
Mutation step: *for each vector* $v_i^{(t)}$ *a perturbed vector* $v_i^{(t+1)}$ *called the donor vector is generated according to Eq. (14).*
Crossover step: *the trial vector* $u_i^{(t+1)}$ *is generated according to Eq. (15).*
Compute fitness function according to Eq. (13) and train dataset.
Selection step: *get the target vector* $x_i^{(t+1)}$ *according to Eq. (16).*
End for
$t = t + 1$
End for
End

After getting medoids, spread, and bias from above DE pseudo-code now the weights of the network are computed by pseudo-inverse method as described in Eq. (17).

Pseudo code of our working model:

Begin
Compute the mediods, spread, bias and weight according to pseudo-code;
Compute the weight according to Eq. (17);
Calculate the basis function (Φ) *using test data, medoid, spread and bias*;
Calculate the output as $Y = W\Phi$;
Calculate the accuracy by comparing actual output with desired output;
End

4.3. *Mixture of kernels in SVM*

In our second approach, SVM is used to separates data points into two categories by a decision surface called a hyperplane. We know that, the hyperplane maximizes the margin of separation between data points belongs to two classes. This hyperplane is then used to predict the class label of unknown data points based on which side of the plane it falls. To determine the best classification performance, we have used SVM with three traditional kernels (Linear, Polynomial, and Gaussian RBF) as well as three mixture kernels (χ_1·Linear + χ_2·Polynomial, χ_1·Polynomial + χ_2·RBF, and χ_1·Linear + χ_2·RBF).

The motivation behind using mixture kernel is described below. Joachims *et al.* (2001) have shown that combining two kernels is beneficial if both of them achieve approximately the same performance and use different data instances as support vectors.

This makes sense because in combination, kernels must be useful by themselves and complimentary. Additionally, the convex combination of two kernels is more powerful than others like linear and cubic. The convex combination of two kernels ($\chi_2 = 1 - \chi_1$) can achieve better classification performance. With this motivation three kernels (c.f., Table 1) like linear, polynomial, and Gaussian have been chosen for the purpose of combination. In general, we can linearly parametrize the combination function as:

$$k_\chi(x_i, x_j) = f_\chi\left(\{k_l(x_i^l, x_j^l)\}_{l=1}^P \middle| \chi\right) = \sum_{l=1}^{P} \chi_l \cdot k_l(x_i^l, x_j^l), \tag{18}$$

where χ denotes the kernel weights. Different versions of this approach differ in the way they put restriction on χ_1. In specific, the combinations are as follows:

$$k_\chi^1(x_i, x_j) = \chi_1(x_i^T \cdot x_j) + \chi_2\left(\exp\left(-\frac{\|x_i - x_j\|_2^2}{2\sigma^2}\right)\right), \tag{19}$$

and

$$k_\chi^2(x_i, x_j) = \chi_1((x_i^T \cdot x_j + 1)^p) + \chi_2\left(\exp-\left(\frac{\|x_i - x_j\|_2^2}{2\sigma^2}\right)\right). \tag{20}$$

Although, it is promising by other combining methods like non-linear and data dependent way, however linear combiner with appropriate setting of combining parameters and complex Gaussian kernel works well.

5. Experimental Study

This section is divided into four Subsections. In Subsection 4.1, the dataset and parameters required for simulation has been studied. Subsection 4.2 discusses the data based approach for handling the class imbalanced problem. The performance matrices and results with analysis is discussed in Subsections 4.3 and 4.4, respectively.

5.1. *Description of dataset and parameters*

The datasets used in this work were obtained from the UCI machine learning repository.[73] We have focused on binary class classification problem with highly imbalanced in nature. Nowadays, the research in imbalanced datasets is receiving more attention because they are present in many real application, often traditional classifiers cannot adequately handle this kind of datasets because they are biased to correctly classify the majority cases, leaving aside minority cases. At the same time kernel based approaches for classification is also used increasingly in bio-medical domain. Hence, to analyze the effect of these approaches (i.e., our developed models- DE-RBFNs and SVM with mixture kernels), we have chosen a few bio-medical domain datasets selectively from UCI repository. Six datasets pertaining to medical domain have been chosen to validate the DE-RBFNs and SVM_{mk}. The details of the six datasets are given below.

Table 2. Attribute information of diabetes dataset.

Serial Number	Description of Attributes
1	Number of times pregnant
2	Plasma glucose concentration of 2 hours in an oral glucose tolerance test
3	Diastolic blood pressure (mm Hg)
4	Triceps skin fold thickness (mm)
5	2-Hour serum insulin (mu U/ml)
6	Body mass index (weight in kg/(height in)^2)
7	Diabetes pedigree function
8	Age (years)
9	Class variable (0 or 1)

Table 3. Attribute information of ILPD dataset.

Serial Number	Description of Attributes
1	Age
2	Gender
3	TB total bilirubin
4	DB direct bilirubin
5	Alkphos alkaline phosphotase
6	Sgpt alamine aminotransferase
7	Sgot Aspartate Aminotransferase
8	TP total proteins
9	ALB Albumin
10	A/G Ratio albumin and globulin tatio
11	Selector field used to split the data into two sets

Diabetes. This dataset is the difficult and imbalanced classification problem. The samples include medical diagnostic reports of 768 individuals belonging to Pima Indian population near Phoenix and Arizona. Women who were at least 21 years old were tested for diabetes according to World Health Organization criteria, which included an oral glucose tolerance test. Diabetes was diagnosed according to WHO Criteria. There are 768 instances, 8 attributes (excluding class attribute) and 2 classes (diabetes positive and negative). Class 1 has 550 negative examples and Class 2 has 218 positive samples. Hence, more than double samples are present in class 1 compared to class 2. Table 2 describes its features.

Indian Liver Patient Dataset (ILPD). In this study, we have used Liver patient dataset from UCI machine learning repository (UCI archieve), sourced from Bendi Venkata Ramana, M. Surendra Prasad Babu.[9] It contains 583 liver patient records from north east part of Andhra Pradesh province of India. The instances have 11 attributes (c.f., Table 3

for description) and two classes labeled by experts. We have considered 292 instances in the training set and 291 instances in the test set. This is also an example of a purely imbalanced dataset, as because, out of 583 records, 416 are belongs to class 1 and 167 are belongs to class 2.

Bupa. This dataset is taken from BUPA Medical Research Ltd. database donated by Richard S. Forsyth. It is also available at UCI machine learning repository. The first five attributes of this dataset are all blood tests which are thought to be sensitive to liver disorders that might arise from excessive alcohol consumption. Each entry in the dataset constitutes the record of a single male individual. The sixth and last attribute, "drinks" contains data about frequency of alcohol consumed. The dataset contains 345 blood tests each with 6 attribute values and one of the two possible outcomes (c.f., Table 4). 216 instances were considered in the training set and remaining 129 instances in the test set. The class distribution of this dataset is not highly imbalanced i.e., out of 345 records, 200 are belongs to class 1 and 145 are belongs to class 2.

Table 4. Attribute information of Bupa dataset.

Serial Number	Description of Attributes
1	Mcv mean corpuscular volume
2	Alkphos alkaline phosphotase
3	Sgpt alamine aminotransferase
4	Sgot aspartate aminotransferase
5	Gammagt gamma-glutamyl transpeptidase
6	Drinks number of half-pint equivalents of alcoholic beverages drunk per day
7	Selector field used to split data into two sets

Table 5. Attribute information of blood transfusion dataset.

Serial Number	Description of Attributes
1	Recency quantitative Months Input
2	Frequency quantitative Times Input
3	Monetary quantitative c.c. blood Input
4	Time quantitative Months Input
5	Whether he/she donated blood in March 2007 binary 1 = yes 0 = no

Blood Transfusion. This dataset is taken from the Blood Transfusion Service Center in Hsin-Chu City in Taiwan. The original donor being I-Cheng Yeh, Department of Information Management, Chung-Hua University, Hsin Chu, Taiwan. The center passes their blood transfusion service bus to one university in Hsin-Chu City to gather blood donated about every three months. This dataset is available at UCI machine learning repository. There are 748 donor instances in the dataset, each has 4 attributes. The two classes are binary values representing whether the donor has donated blood in March

2007 (1 stand for donating blood; 0 stands for not donating blood). The Centre collected different parameters for blood donors. The purpose was to predict whether the donor will donate blood in March 2009. The blood transfusion service uses this prediction to identify and select prospective blood donors. The donor instances were split randomly into two sets, 450 instances were considered in the training set and remaining 298 instances in the test set. In this case 500 instances are belongs to class 1and 248 are belongs to class 2, it indicates that dataset is highly imbalanced.

Fertility. Fertility rates have dramatically decreased in the last two decades, especially in men. It has been described that environmental factors, as well as life habits, may affect semen quality. The purpose is to evaluate their performance in the prediction of the seminal quality from the data about the environmental factors and lifestyle. Data has been collected by a normalized questionnaire from young healthy volunteers and results of a semen analysis to assess the accuracy. 100 volunteers provide a semen sample analyzed according to the WHO 2010 criteria. Sperm concentrations are related to socio-demographic data, environmental factors, health status, and life habits. There are 100 instances, 10 attributes (c.f., Table 6) and 2 classes. Class 1 has 88 and Class 2 has 12 instances. The class distribution in this case is highly imbalanced.

Table 6. Attribute information fertility data set.

Serial Number	Description of Attributes
1	Season in which the analysis was performed
2	Age at the time of analysis
3	Childish diseases
4	Accident or serious trauma
5	Surgical intervention
6	High fevers in the last year
7	Frequency of alcohol consumption
8	Smoking habit
9	Number of hours spent sitting per day
10	Class attribute

Vertebral Column. Biomedical data set built by Dr. Henrique da Mota during a medical residence period in the Group of Applied Research in Orthopaedics (GARO) of the Centre MÃ©dico-Chirurgical de RÃ©adaptation des Massues, Lyon, France. The data have been organized in two different but related classification tasks. There are 310 instances and 8 attributes (c.f., Table 7). Class 1 has 210 and Class 2 has 100 instances, it shows that dataset is imbalanced.

In our experiment, every dataset is divided into two mutually exclusive parts: 50% as training sets and 50% as test sets. The parameters' value used for validating our proposed method is listed in Table 8.

The parameters corresponding to SVM_{mk} is listed in Table 9.

Table 7. Attribute information of vertebral column data set.

Serial Number	Description of Attributes
1	Pelvic incidence
2	Pelvic tilt
3	Lumbar lordosis angle
4	Sacral slope
5	Pelvic radius
6	Grade of spondylolisthesis
7	Class attribute(Abnormal,normal)

Table 8. Parameters used corresponding to DE-RBFN for simulation.

Parameter	Diabetes	ILPD	Bupa	Blood Transfusion	Fertility	Vertebral
Maximum Iteration	100	100	100	100	100	100
Population Size	80	50	60	40	90	60
Mutation Rate	0.2	0.2	0.2	0.2	0.2	0.2
Crossover Rate	0.5	0.5	0.5	0.5	0.5	0.5

Table 9. Dataset specific parameters of SVM_{hk}.

Data Set	Kernel Parameters for Optimal Accuracy	Kernel Hybridization Parameters
Pima Indians Diabetes	$\sigma = 2$	Hybrid of Linear and RBF kernels (0.18 * Linear + 0.82 * RBF)
Indian Liver Patient	$\sigma = 2$	Hybrid of Linear and RBF kernels (0.81 * Linear + 0.19 * RBF)
BUPA Liver Disorders	Polynomial degree, $p = 3$ and $\sigma = 3$	Hybrid of Polynomial and RBF kernels (0.56 * Polynomial + 0.44 * RBF)
Blood Transfusion Service Center	Polynomial degree, $p = 3$ and $\sigma = 3$	Hybrid of Polynomial and RBF kernels (0.71 * Polynomial + 0.29 * RBF)
Fertility	$\sigma = 2$	Hybridization not needed for improving accuracy
Vertebral Column	$\sigma = 2$	Hybridization not needed for improving accuracy

5.2. *Techniques for handling class-imbalanced*

Under the umbrella of data driven, two approaches are used for handling class imbalanced data (Han *et al*., 2009): (i) Over-sampling — It works by resampling the positive tuples so that the resulting training set contains an equal number of positive and negative tuples, and (ii) Under-sampling — It works by decreasing the number of negative tuples. It randomly eliminates tuples from the majority (negative) class until there are an equal number of positive and negative tuples. In the other hand, there are approaches, which uses some methodological changes in the algorithm, so that learned

model may not be influenced by the dominated class. In this work, initially, we have used data driven approaches, however, the inherent limitations of under-sampling and over-sampling ignited us to combine both approaches to tackle highly imbalanced datasets.

5.3. *Performance metrics*

Several criteria may be used to evaluate the performance of a classification algorithm in supervised learning. A confusion matrix is a useful tool for analyzing how well classifier can identify test samples of different classes,[70] which tabulates the records correctly and incorrectly predicted by the model. Although confusion matrix[71] provides the information needed to determine how well a classification model performs, summarizing this information with a single number would make it convenient to compare the performance of different models. This can be done by using the performance metrics such as sensitivity or recall, specificity, precision or positive predictive value, negative predictive value and accuracy.

Sensitivity. It measures the actual members of the class which are correctly identified as such. It is also referred as True Positive Rate (TPR) or recall. It is defined as the fraction of positive examples predicted correctly by the classification mode.

$$Sensitivity(recall) = \frac{TP}{(TP + FN)}$$

Specificity. It is also referred to as true negative rate. It is defined as the fraction of negative examples which are predicted correctly by the model.

$$Specificity = \frac{TN}{(TN + FP)}$$

Precision (Positive Predictive Value). It is also called as positive predicative value and determines the fraction of records that actually turns out to be positive in the group which has been declared as positive Class by the classifier.

$$Precision = \frac{TP}{(TP + FP)}$$

Negative Predictive Value (NPV). It is proportion of the samples which do not belong to the class under consideration and which are correctly identified as non members of the class.

$$NPV = \frac{TN}{TN + FN}$$

Accuracy. It is used as a statistical measure of how well a binary classification test identifies or excludes a condition. It is a measure of proportion of true results.

$$Accuracy = \frac{(TP + TN)}{(TP + FP + TN + FN)}$$

where TP = true positives, TN = true negative, FP = false positives, FN = false negatives.

5.4. *Results and analysis*

Table 10 illustrates the summary of the experimental outcomes obtained from our computer simulation study. Beginning with the analysis it must be highlighted that support vector machine with mixture of kernels is showing better performance in the case of Pima Indian Diabetes, Blood Transfusion, Fertility, and Vertebral Column datasets. On the other hand, DE-RBFN is indicating good performance in the case of Indian Liver Patient and BUPA Liver Disorders.

Table 10. Performance of DE-RBFNs and SVM_{mk}.

Data Set	Accuracy Using DE-RBFNs	Accuracy Using SVM_{mk}
Pima Indians Diabetes	70.31	**82.57**
Indian Liver Patient	**73.70**	71.99
BUPA Liver Disorders	**74.10**	73.64
Blood Transfusion Service Center	75.41	**84.23**
Fertility	61.36	**90.00**
Vertebral Column	62.86	**77.59**
	Average = 69.623	Average = 80.00

In the case of BUPA liver disorder under sampling is not worse than method driven approach, as because the dataset is not highly imbalanced. However, in all other cases, the approximation of under sampling to the true model is unlikely to accept. Hence, method driven approach has been adopted.

Figures 7–12 is demonstrating the errors obtained during different iterations of the experimental study of DE-RBFNs.

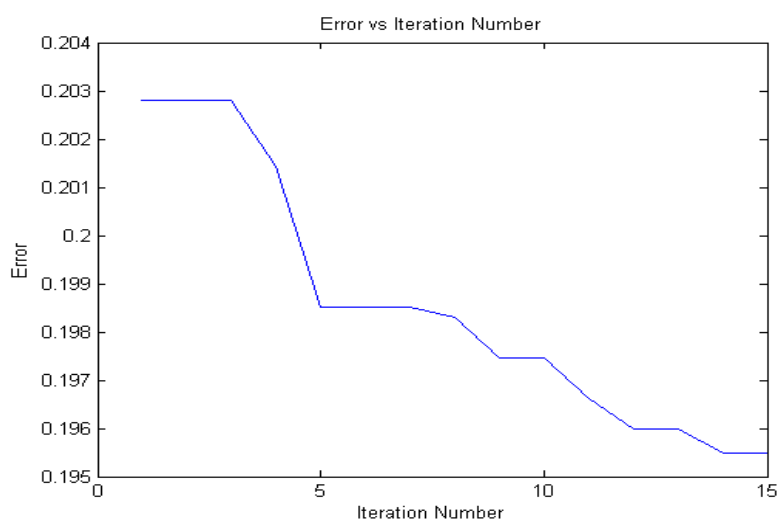

Fig. 7. Error versus iteration — PIMA Indian diabetes dataset.

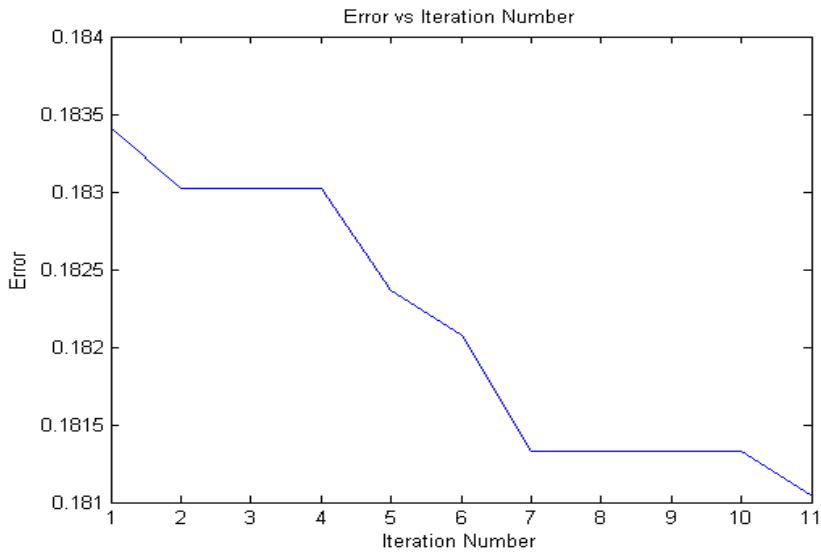

Fig. 8. Error versus iteration — ILPD dataset.

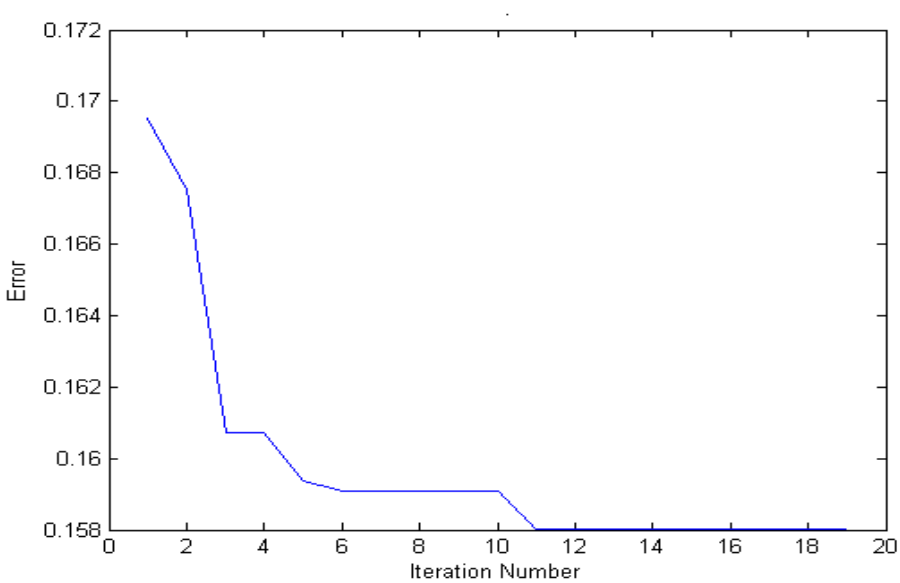

Fig. 9. Error versus Iteration — Blood transfusion dataset.

It is observed from Figure 7 (draws from the best independent run) that, at least 15 iterations are required to reach an error in a stable state. However, due to the highly imbalanced nature of the dataset and the local optimality problem of RBFNs, DE-RBFNs sometimes trigger to escape from the local optima. Moreover, proper tuning of kernel parameters are highly required to reduce the local optimality.

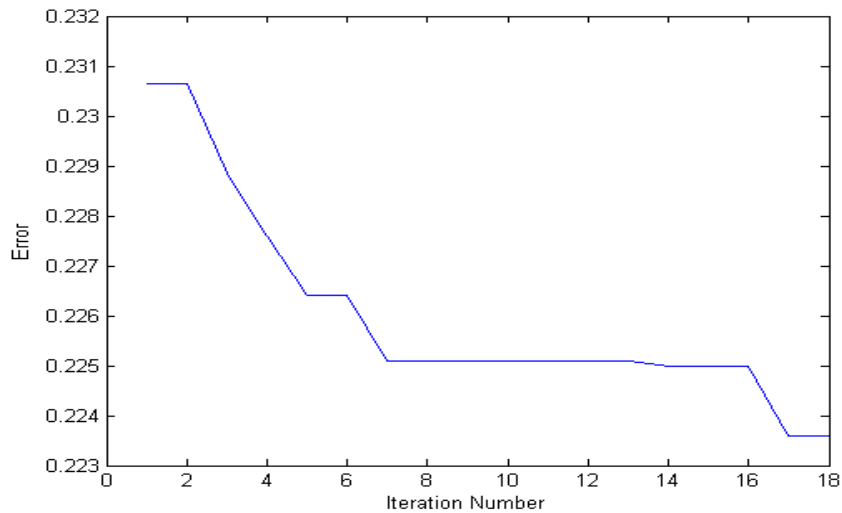

Fig. 10. Error versus iteration — BUPA dataset.

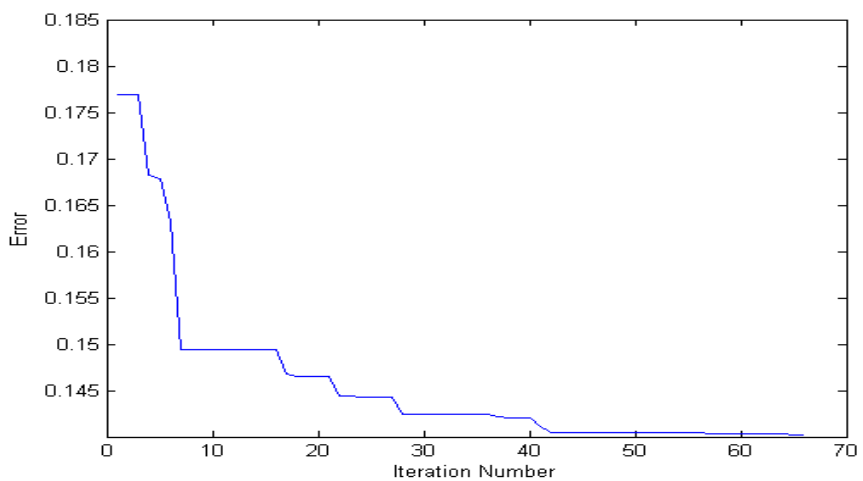

Fig. 11. Error versus iteration — Fertility dataset.

From Figure 8 shows that at least 11 iterations are required to make the error arte stable. Although, the dataset is highly imbalanced, but DE-RBFNs are reducing the local optimal and landed with a promising accuracy compared to SVM with mixture kernels.

In the case of Blood Transfusion dataset (best independent run plotted in Figure 9), DE-RBFNs was very easily trapped with local optimality and could not escape even after iterating more than 18 times. In the other hand, under sampling the majority class instances results a loss of representative samples, therefore, the accuracy did not competitive with SVM with mixture of kernels. The method driven approach adopted

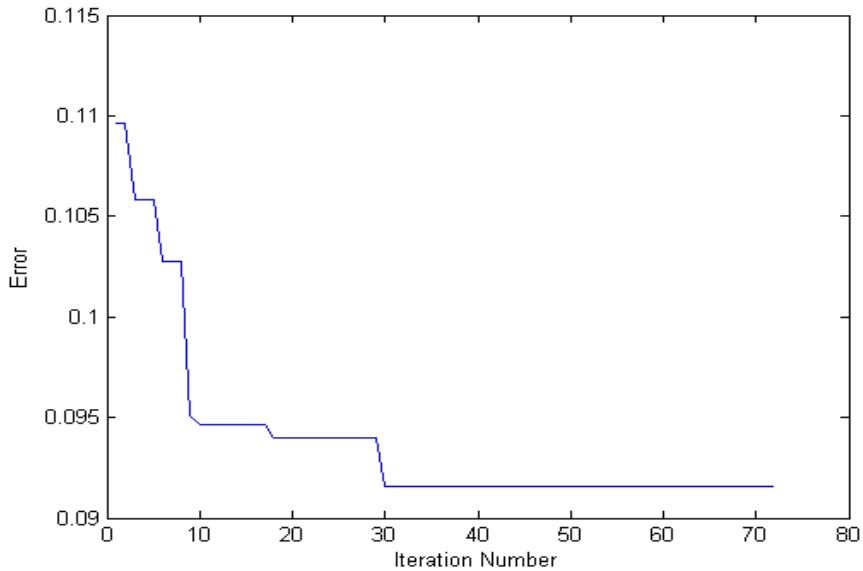

Fig. 12. Error versus iteration — Vertebral dataset.

in DE-RBFNs also fails to achieve the accuracy as discovered by SVM with mixture kernels.

From the Figure 10 (obtained from the best independent run), it is observed that, under sampling of BUPA dataset for training the DE-RBFNs needs at least 17 iterations.

In Fertility dataset, the error rate is reduced after 40 iterations but the local optimality could not be avoided here fully. One of the reasons may be due to highly imbalanced distributions of instances into two different classes.

In the case of vertebral dataset, after 30 iterations the error rate of DE-RBFNs ceases, however, the accuracy is far away from the accuracy derived through SVM with mixture of kernels.

As we know, to evaluate classifiers on highly imbalanced datasets using accuracy as a metric is virtually useless. This is because with an imbalance of 99 to 1, a classifier

Table 11. Outcomes of SVM_{mk} in percentage.

Dataset	Producer Accuracy (Precision) Class-1	Producer Accuracy (Precision) Class-2	User Accuracy (Recall) Class-1	User Accuracy (Recall) Class 2	Overall Accuracy
Pima Indians Diabetes	70.59	90.23	82.19	82.76	82.57
Indian Liver Patient	83.93	56.10	72.31	71.88	72.17
BUPA Liver Disorders	57.41	85.33	73.81	73.56	73.64
Blood Transfusion Service Center	95.19	58.89	84.26	84.13	84.23
Fertility	100.00	50.00	88.89	100.00	90.00
Vertebral Column	80.00	75.00	77.42	77.78	77.59

Table 12. Outcomes of DE-RBFNs in percentage.

Data Set	Producer Accuracy (Precision) Class 1	Producer Accuracy (Precision) Class 2	User Accuracy (Recall) Class 1	User Accuracy (Recall) Class 2	Overall Accuracy
Pima Indians Diabetes	71.43	69.31	67.71	72.92	70.31
Indian Liver Patient	66.18	89.54	93.75	52.74	73.10
BUPA Liver Disorders	53.19	84.78	64.10	78.00	74.10
Blood Transfusion Service Center	50.00	83.70	50.00	83.70	75.41
Fertility	56.82	65.91	62.50	60.42	61.36
Vertebral Column	68.87	55.77	61.35	63.74	62.38

that classifies everything negative will be 99% accurate, but it will be completely useless as a classifier. The medical community and increasingly the machine learning community uses two metrics the producer accuracy and the user accuracy, when evaluating the performance of various tests. The performance of the SVM_{mk} and DE-RBFNs are evaluated using the matrices defined at Section 4.3. Tables 11 and 12 illustrate the accuracy with respect to various measures.

6. Conclusions and Future Line of Research

In this paper, two kernel based algorithms have been studied to analyze their performance on a few bio-medical datasets. In addition, our comparative study reveals that both DE-RBFNs and SVM_{mk} algorithms are producing more promising results compared to simple radial basis function neural networks and support vector machine. However, in majority of cases SVM with mixture of kernels is performing better than evolved radial basis function neural networks. In specific, SVM_{mk} works well for Pima Indian, Blood Transfusion Service Center, Fertility, and Vertebral column datasets, whereas DE-RBFNs work well for Indian Liver Patients and Bupa Liver Disorders datasets. In future, our study will concentrate on more extensive evaluation of these two techniques on many real lives biomedical datasets along with an attempt to analyze the gap between real and UCI repository datasets.[10] In this study, we have focused only on binary imbalanced dataset classification; however, in real life more than two classes with unevenly distributed instances to different classes may pose a challenge. Therefore, our future work will address this issue very seriously.

Further, in this work, we have combined a few selected numbers of kernels in SVM for classification of imbalanced datasets. However, in reality there are many potential kernels available, hence in the absence of prior domain knowledge, it is too difficult to find out potentials pool towards mixture of kernels. Therefore, in our future work, we will develop an algorithm that can automatically determine the useful ones for the purpose of mixtures to use in SVM.

References

1. R. Akbani, S. Kwek and N. Japkowicz, Applying support vector machines to imbalanced datasets, in J. F. Baulicout *et al*. (Eds.), *ECML 2004*, LNAI 3201 (2004), pp. 39–50.
2. A. Alexandridis, E. Chondrodima and H. Sarimveis, Radial basis function network training using a non-symmetric partition of the input space and particle swarm optimization, *IEEE Transactions on Neural Networks and Learning Systems* **24**(2) (2013) 219–230.
3. M. Arun Kumar and M. Gopal, Least squares twin support vector machines for pattern classification, *Expert Systems with Applications* **36**(4) (2009) 7535–7543.
4. K. Aslan, H. Bozdemir, C. Sahin, S. N. Ogulata and R. Erol, A radial basis function neural network model for classification of epilepsy using EEG signals, *Journal of Medical Systems* **32**(5) (2008) 403–408.
5. L. Autio, M. Juhola and J. Laurikkala, On the neural network classification of medical data and an endeavour to balance non-uniform datasets with artificial data extension, *Computers in Biology and Medicine* **37** (2007) 388–397.
6. G. S. Babu and S. Suresh, Sequential projection based metacognitive learning in a radial basis function networks for classification problems, *IEEE Transactions on Neural Networks and Learning Systems* **24**(2) (2013) 194–206.
7. M. Balasubramanian, S. Palanivel and V. Ramalingam, Real time face and mouth recognition using radial basis function neural networks, *Expert Systems with Applications* **36**(3) (2009) 6879–6888.
8. A. I. Belousov, S. A. Verzakov and J. Von Frese, A flexible classification approach with optimal generalization performance: Support vector machines, *Chemometrics and Intelligent Laboratory Systems* **64**(1) (2002) 15–25.
9. V. R. Bendi and M. Surendra, Liver classification using modified rotation forest, *International Journal of Engineering Research and Development* **1**(6) (2012) 17–24.
10. C. Blake, E. Keogh and C. J. Merz, UCI Repository of Machine Learning Database (1998), www.ics.uci.edu/mlearn/MLRepository.html.
11. D. S. Broomhead and D. Lowe, Multivariable functional interpolation and adaptive networks, *Complex Systems* **2** (1988) 321–355.
12. M. D. Buhmann, Radial basis function networks, *Encyclopedia of Machine Learning* (2010) 823–827.
13. S. Chen, C. F. N. Cowan and P. M. Grant, Orthogonal least squares learning algorithms for radial basis function networks, *IEEE Transactions on Neural Networks* **2**(2) (1991) 302–309.
14. S. Das and P. N. Suganthan, Differential evolution: A survey of the state-of-the-art, *IEEE Transactions on Evolutionary Computation* **15**(1) (2011) 4–31.
15. C. S. K. Dash, A. K. Behera, M. K. Pandia and S. Dehuri, Neural networks training based on differential evolution in radial basis function networks for classification of web logs, in *Proc. of ICDCIT 2013* (2013a), pp. 183–194.
16. C. S. K. Dash, A. K. Behera, S. Dehuri and S.-B. Cho, Differential evolution based optimization of kernel parameters in radial basis function networks for classification, *International Journal of Applied Evolutionary Computation* **4** (2013b) 56–80.
17. P. Dhanalakshmi, S. Palanivel, and V. Ramalingam, Classification of audio signals using SVM and RBFNN, *Expert Systems with Applications* **36**(3) (2009) 6069–6075.
18. M. G. Epitropakis, V. P. Plagianakos and M. N. Vrahatis, Multimodal optimization using niching differential evolution with index-based neighborhoods, *IEEE Congress on Evolutionary Computation* (2012) 1–8.
19. A. O. Falcao, T. Langlois and A. Wichert, Flexible kernels for RBF networks, *Neurocomputing* **69** (2006) 2356–2359.

20. S. Forerest, Genetic algorithms: Principles of natural selection applied to computation, *Science* **261**(5123) (1993) 872–888.
21. A. Ghodsi and D. Schuurmans, Automatic basis selection techniques for RBF networks, *Neural Networks* **16** (2003) 809–816.
22. D. Gil, J. L. Girela, J. D. Juan, M. J. Gomez-Torres and M. Johnsson, Predicting seminal quality with artificial intelligence methods. *Expert Systems with Applications* **39** (2012) 12564–12573.
23. J. B. Gomm and D. L. Yu, Selecting radial basis function network centers with recursive orthogonal least squares training, *IEEE Transactions of Neural Networks* **11**(3) (2000) 306–314.
24. M. Gonen and E. Alpaydon, Multiple kernel learning algorithms, *Journal of Machine Learning Research* **12** (2011) 2211–2268.
25. D. Hanbay, An expert system based on least square support vector machines for diagnosis of the valvular heart disease, *Expert Systems with Applications* **36**(3) (2009) 4232–4238.
26. S. Haykin, Neural networks: A comprehensive foundation, *Upper Saddle River* (NJ: Prentice Hall, 1994).
27. M. Homaeinezhad, S. A. Atyabi, E. Tavakkoli, H. N. Toosi, A. Ghaffari and R. Ebrahimpour, ECG arrhythmia recognition via a Neuro-SVM-KNN hybrid classifier with virtual QRS image-based geometrical features, *Expert Systems with Applications* **39**(2) (2012) 2047–2058.
28. B. O. Hora, J. Perera and A. Brabazon, Designing radial basis function networks for classification using differential evolution, in *Proc. of International Joint Conference on Neural Networks* (*IJCNN'06*) (2006), pp. 2932–2937.
29. C. L. Huang and C. J. Wang, GA-based feature selection and parameters optimization for support vector machines. *Expert Systems with Applications* **31**(2) (2006) 231–240.
30. W. Huang, Y. Nakamori and S. Wang, Forecasting stock market movement direction with support vector machine. *Computers & Operations Research* **32**(10) (2004) 2513–2522.
31. M. Hussain, S. K. Wajid, A. Elzaart and M. Berbar, A comparison of SVM kernel functions for breast cancer detection, *2011 Eighth Int. Conf. on Computer Graphics, Imaging and Visualization* (*CGIV*), (2011), pp. 145–150.
32. A. Idri, A. Zakrani and A. Zahi, Design of radial basis function neural networks for software effort estimation, *International Journal of Computer Science* **7**(4) (2010) 11–17.
33. Sk. Islam, S. Das, S. Ghosh, S. Roy and P. Suganthan, An adaptive differential evolution algorithm with novel mutation and crossover strategies for global numerical optimization, *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics* **42**(2) (2012) 482–500.
34. M. Khalilia, S. Chakraborty and M. Popescu, Predicting disease risks from highly imbalanced data using random forest, *BMC Medical Informatics and Decision Making* **11**(1) (2011) 51.
35. M. Korrek and B. Dogan, ECG beat classification using particle swarm optimization and radial basis function neural network. *Expert Systems with Applications* **37**(12) (2010) 7563–7569.
36. D.-C. Li, C.-W. Liu and S. C. Hu, A learning method for the class imbalance problem with medical datasets, *Computers in Biology and Medicine* **40** (2010) 509–518.
37. S. Lin and Z. Liu, Parameter selection in SVM with RBF kernel function, *Journal of Zhejiang University of Technology* **35**(2) (2012) 135.
38. J. Liu and J. Lampinen, A differential evolution based incremental training method for RBF networks, *in Proc. of GECCO 2005* (2005), pp. 881–888.
39. K. Z. Mao and G. B. Huang, Neuron selection for RBF neural network classifier based on data structure preserving criterion, *IEEE Transactions on Neural Networks* **16**(6) (2005) 1531–1540.
40. Z. Michalewicz, *Genetic Algorithm + Data Structure = Evolution Programs* (Springer-Verlag, New York, 1996).

41. S. H Min, J. Lee and I. Han, Hybrid genetic algorithms and support vector machines for bankruptcy prediction, *Expert Systems with Applications* **31**(3) (2006) 652–660.
42. Y. Mo and S. Xu, Application of SVM based on hybrid kernel function in heart disease diagnoses, in *Proc. of Int. Conf. Intelligent Computing and Cognitive Informatics* (*ICICCI*) (2010), pp. 462–465.
43. J. Moody and C. J. Darken, Fast learning networks of locally-tuned processing units, *Neural Computation* **6**(4) (1989) 281–294.
44. N. Naveen, V. Ravi, C. R. Rao and N. Chauhan, Differential evolution trained radial basis function network: Application to bankruptcy prediction in banks, *International Journal of Bio-Inspired Computation* **2**(3) (2010) 222–232.
45. B. O'Hora, J. Perera and A. Brabazon, Designing radial basis function networks for classification using differential evolution, in *Proc. of Int. Joint Conf. on Neural Networks* (2006), pp. 2932–2937.
46. M. A. Panduro, C. A. Brizuela, L. I. Balderas, and D. A. Acosta, A comparison of genetic algorithms, particle swarm optimization and the differential evolution method for the design of scannable circular antenna arrays, *Progress in Electromagnetics Research B***13** (2009) 171–186.
47. J. Park and J. W. Sandberg, Universal approximation using radial basis function networks, *Neural Computation* **3** (1991) 246–257.
48. T. Phienthrakul and B. Kijsirikul, Evolutionary strategies for multi-scale radial basis function rernels in support vector machines, in *Proc. of the 2005 Conference on Genetic and Evolutionary Computation* (2005), pp. 905–911.
49. M. J. D. Powell, Radial basis functions for multi-variable interpolation: A review, *Algorithms for Approximation* (1987) 143–167.
50. K. V. Price, R. M. Storn and J. A. Lampinen, *Differential Evolution: A Practical Approach to Global Optimization* (Springer, 2005).
51. D. Qiu, Y. Li, X. Zhang and B. Gu, Support vector machine with parameter optimization by bare bones differential evolution, in *Proc. of Int. Conf. on Natural Computation* (*ICNC*) (2011), Vol. 1, pp. 263–266.
52. B. Qu, P. N. Suganthan and J. Liang, Differential evolution with neighborhood mutation for multimodal optimization, *IEEE Transactions on Evolutionary Computation* **16**(5) (2012) 601–614.
53. S. N. Quasem and S. M. Shamsuddin, Memetic Elitist pareto differential evolution algorithm based radial basis function network for classification problems, *Applied Soft Computing* **11**(8) (2011) 5565–5581.
54. B. Scholkopf, K. K. Sung, C. J. C. Burges, F. Girosi, P. Niyogi, T. Poggio and V. Vapnik, Comparing support vector machines with Gaussian kernels to radial basis function classifiers, *IEEE Transactions on Signal Processing* **45** (11) (1997) 2758–2765.
55. A. F. Sheta and K. De Jong, Time series forecasting using GA tuned radial basis functions, *Information Sciences* **133** (2001) 221–228.
56. J. W. Smith, J. E. Everhart, W. C. Dickson, W. C. Knowler and R. S. Johannes, Using the ADAP learning algorithm to forecast the onset of diabetes mellitus, in *Proc. of the Annual Symposium on Computer Application in Medical Care* (1988), pp. 261–265.
57. R. Stron and K. Price, Differential evolution — Simple and efficient heuristic for global optimization over continuous spaces, *Journal of Global Optimization* **11** (1997) 341–359.
58. R. Stron, System design by constraint adaption and differential evolution, *IEEE Transactions on Evolutionary Computation* **3** (1999) 22–34.
59. R. Stron and K. Price, Differential evolution — A simple and efficient adaptive scheme for global optimization over continuous spaces, Technical Report TR-05-012: International Computer Science Institute, Berkely (1995).

60. T. S. Subashini, V. Ramalingam and S. Palanivel, Breast mass classification based on cytological patterns using RBFNN and SVM, *Expert Systems with Applications* **36**(3) (2009) 5284–5290.
61. S. Theodoridis and K. Koutroumbas, *Pattern Recognition* (Academic Press San Diego CA 92101-4495, USA, 1998).
62. Z. Uykan, C. Guzelis, M. E. Calebi and H. N. Koivo, Analysis of input-output clustering for determining centers of RBFN. *IEEE Transactions on Neural Networks* **11**(4) (2000) 851–858.
63. C. H. Wu, G. H. Tzeng, Y. J. Goo and W. C. Fang, A real-valued genetic algorithm to optimize the parameters of support vector machine for predicting bankruptcy, *Expert Systems with Applications* **32**(2) (2007) 397–408.
64. C. H. Wu, G. H. Tzeng and R. H. Lin, A novel hybrid genetic algorithm for kernel function and parameter optimization in support vector regression, *Expert Systems with Applications* **36**(3) (2009) 4725–4735.
65. B. Xie, J. Chen, J. Zhang, H. Fang and Z. H. Peng, Hybridizing differential evolution and particle swarm optimization to design powerful optimizers: A review and taxonomy, *IEEE Transactions on Systems, Man, Cybernetics-Part C: Applications and Reviews* **42**(5) (2012) 744–767.
66. T. Xie, H. Yu, J. Hewlett, P. Rozycki and B.Wilamowski, Fast and efficient second order method for training radial basis function Networks, *IEEE Transactions on Neural Networks and Learning Systems* **23**(4) (2012) 609–619.
67. Q. Xu and S. Geng, A fast SVM classification learning algorithm used to large training set, in *Proc. of 2nd Int. Conf. on Intelligent system Design and Engineering Application* (*ISDEA*) (2012), pp. 15–19.
68. B. Yu and X. He, Training radial basis function networks with differential evolution, *World Academy of Science Engineering and Technology* **11** (2005) 337–340.
69. G. L. Zheng and S. A. Billings, Radial basis function network configuration using mutual information and orthogonal least squares algorithm, *Neural Networks* **9**(9) (1996) 1619–1637.
70. J. Han and M. Kamber, *Data Mining Concepts and Techniques*, 2nd edn. (Elsevier Publisher, 2009).
71. T. Sitamahalakshmi, D. A. V. Babu, M. Jagadeesh and K. V. V. Mouli, Performance comparison of radial basis function networks and probabilistic neural networks for telugu character recognition, *Global Journal of Computer Science and Technology* **11**(4) (2011).
72. V. Feoktistov and S. Janaqi, Hybridization of differential evolution with least-square support vector machines, *BENELEARN* (2004) 26–31.
73. Frank and A. Asuncion, Machine learning repository, Irvine CA: University of California School of Information and Computer Science (2010), http://archive.ics.uci.edu/ml.
74. J. Moody and C. J. Darken, Fast learning networks of locally-tuned processing units, *Neural Computation* **6**(4) (1989) 281–294.
75. S. Lee and R. M. Kil, A Gaussian potential function network with hierarchically self organizing learning, *Neural Networks* **4**(2), (1991) 207–224.
76. W. Zhao and D. S. Huang, The structure optimization of radial basis probabilistic neural networks based on genetic algorithms, in *Proc. of Int. Joint Conf. on Neural Networks* (*IJCNN'02*) (2002), pp. 1086–1091.