Pattern Recognition Letters 80 (2016) 76–83

Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

# Design of self-adaptive and equilibrium differential evolution optimized radial basis function neural network classifier for imputed database☆

Ch. Sanjeev Kumar Dash[a], Amitav Saran[a], Pulak Sahoo[a], Satchidananda Dehuri[b,*], Sung-Bae Cho[c]

[a] Silicon Institute of Technology, Silicon Hills, Patia, Bhubaneswar 751024, Odisha, India
[b] Department of Systems Engineering, Ajou University, San 5, Woncheon-dong, Yeongtong-gu, Suwon 443-749, South Korea
[c] Soft Computing Laboratory, Department of Computer Science, Yonsei University, 50 Yonsei-ro, Sudaemoon-gu, Seoul 120-749, South Korea

ARTICLE INFO

*Article history:*
Received 1 January 2015
Available online 13 May 2016

*Keywords:*
Data mining
Imputation
Classification
Radial basis function neural networks
Differential evolution
k-nearest neighbor

ABSTRACT

The occurrence of missing values is not uncommon in real life databases like industrial, medical, and life science. The imputation of these values has been realized through the mean/mode of known values (for a quantitative/qualitative attribute) or nearest neighbors. Mean based imputation considerably underestimates the population variance and tends to weaken the attribute relationships. Similarly, the nearest neighbor approach uses only information of the nearest neighbors and leaving other observations aside. Hence to overcome the shortcomings of these methods, we have introduced a method known as medoid based imputation to impute missing values. Further, to achieve better performance, we have devised a novel classifier for imputed datasets, by using the self-adaptive control parameters of differential evolution (DE) with equilibrium of exploitation and exploration optimized radial basis function neural networks (RBFNs). By newly associating a weight parameter with target vector during mutation, we maintain equilibrium on the exploration and exploitation mechanism of DE. The self-adaptive equilibrium DE (SAEDE) is used to explore and exploit the suitable kernel parameters of RBFNs along with bias and then used for classifying unknown samples. The performance of the proposed classifier named as SAEDE-RBFN has been extensively evaluated on seven datasets retrieved from University of California, Irvine (UCI) and KEEL machine learning repositories after imputation by mean, nearest neighbor, and proposed method. The average performance of classifiers has been listed based on the imputation by K-nearest neighbor (Knn = 1, Knn = 3, Knn = 5, and Knn = 7), mean, and medoid, respectively. Outcome of the experimental study shows that the performance of SAEDE-RBFN on medoid based imputed dataset is relatively better than DE-RBFN.



## 1. Introduction

In Data Mining [1] and Big Data analysis [2], classification, prediction, association rule mining, and clustering are identified as fundamental activities in dredging knowledge from the real life databases. Over the years, for each of these tasks, many models have been proposed with their associated pros and cons [3]. However, the common consensus is that the accuracy of the model depends highly on the quality of the data being mined. In many application areas, quality datasets are difficult to obtain and are small in size. It is not uncommon to encounter industrial and life science databases with half of the entries of a particular instance missing [4,5]. It has been observed that some of the datasets contain only a few samples with missing values in useful attributes. For example, in patient diagnosis data, we often have missing values [6–8] corresponding to a particular feature that would have helped in predicting the disease and subsequent treatment. There are various reasons for these missing values, such as human errors, machine errors, and incorrect measurements.

In the last decades, many approaches have been developed to impute missing values [4]. The simplest technique used in many literatures is to remove the samples with missing attributes [9] and use the remaining samples for modeling. But, there can be substantial loss of information due to simple deletion of samples with missing attributes, especially in the case of small and imbalanced datasets. Moreover, this method is practical only when the data

☆ This paper has been recommended for acceptance by L. Heutte.

* Corresponding author. Tel.: +91 966 832 1964; fax: +82 2365 2579.
*E-mail addresses:* sanjeev_dash@yahoo.com (Ch.S.K. Dash), amitavsaran@yahoo.com (A. Saran), sahoo_pulak@yahoo.com (P. Sahoo), satchi@ajou.ac.kr (S. Dehuri), sbcho@yonsei.ac.kr (S.-B. Cho).

http://dx.doi.org/10.1016/j.patrec.2016.05.002
0167-8655/

contain relatively small number of samples with missing values and the analysis of the complete data will not lead to serious bias during the inference.

Many researchers [10–14] are working on how to handle these missing values (Farhangfar et al., and [15]) in both training and testing datasets. Imputation is a very popular approach to replace missing attribute values. There are two main types of imputation approaches: (1) Value imputation and (2) Distribution-based imputation. Value imputation estimates a value to be used by the model in place of the missing feature [16, 17] and is common in statistical community. In contrast distribution-based imputation estimates the conditional distribution of the missing value ([18]; Grzymala-Busse and Hu, 2001; [19, 20]) and the prediction will be based on this estimated distribution.

In this paper, we propose a new way of imputing the missing values called medoid based imputation. In addition, we have tried multiple imputation approaches to handle missing values and avoid simple removal of samples which will result in useful information being lost. We have applied imputation by k-nearest neighbor (Knn = 1, Knn = 3, Knn = 5, and Knn = 7) [21] and mean techniques to study their effect on classification performances [22–26]. After imputing the datasets, our self-adaptive equilibrium DE optimized RBFN is used to study the influence of the imputed techniques such as k-nearest neighbor, mean, and medoid. We have conducted the simulation on seven benchmark datasets obtained from University of California, Irvine (UCI) machine learning repository [27] and KEEL repository.

## 2. Background

The background of this research work (i.e., missing data imputation and classification, the salient features of RBFN, and DE) are discussed distinctly in Sections 2.1–2.3, respectively.

### 2.1. Missing data imputation and classification

Missing data [28, 29] are source of common problems in all type of research work. Imputation techniques are based on the idea that any subject in a study can be replaced by a new randomly chosen subject from the same source. Imputation of missing data on a variable is replaced by a value that is drawn from an estimate of the distribution of this variable. In methods like complete or available case analysis, the missing indicator, and overall mean imputation lead to inefficient analysis and more seriously, produce severely biased estimates of the association(s) investigated. Hence, to reduce the biased-ness or increase the quality of analysis, we have proposed the medoid based imputation. Additionally, the Knn based analysis is also been examined. The problem of classification [30] is basically the core of partitioning the feature space into regions, one region for each category of inputs. Classifiers are generally, but not always designed with labeled data, in which case these problems are sometimes referred to as supervised classification (where the parameters of a classifier are learnt). In general, supervised classification with missing data [31] focuses on two distinct tasks: handling missing values (i.e., imputing values) and pattern classification [12]. Table 1 illustrates a few imputation methods and usages of some of the machine learning classifiers on imputed database. However, it is evident that not a single literature has been suggested the medoid based imputation.

**Table 1**
Imputation method and usage of classifiers on imputed dataset.

| Authors | Imputation method | Classifiers |
|---|---|---|
| [46] | Mode, C4.5, LERS ML | LERS |
| [31] | 10-NNI, Mean, Mode | C4.5, CN2 |
| [30] | Case Deletion, Mean, Median, Knn | LDA, Knn |
| [24] | GA, Mean, HD | MLP + RBF |
| [4] | KSOP | KSOP |
| [47] | Mean, HD, FNB NB, FHD | RIPPER, SVM, C4.5, Knn, NB |
| [48] | MI-Knn | Knn, MI-Knn |
| [21] | Knn, SKnn, IKnn, K-Means, EAC | MLP, NB, J4.8, Knn |

### 2.2. Radial basis function networks

The RBFN is a two layered network [32], where each hidden unit of hidden layer implements a radial activation function and each output unit of output layer implements a weighted sum of hidden units' output. This network is a special class of neural network in which the activation of a hidden neuron is determined by the distance between the input and a prototype vector. Prototype vectors refer to centers of clusters formed by the patterns in the input space. The centers are determined during RBFN training. Two parameters are associated with each RBFN node, the center and the width. In the input layer, the number of input neurons is determined based on the input features that connect the network to the environment. As stated, the second layer (hidden layer) consists of a set of kernel units that carry out a nonlinear transformation from the input to the hidden space. Usually, a nonlinear transformation is made based on Gaussian kernel as described in Eq. (1).

$$\varphi_i(x) = \exp\left(-\frac{\|x-\mu_i\|^2}{2\sigma_i^2}\right) \tag{1}$$

where ||...|| represents Euclidean norm, $\mu_i$, $\sigma_i$, and $\varphi_i$ are center, spread, and the output of $i$th hidden unit, respectively.

Radial basis function networks use in literature is extensive and its application varies from pattern classification to time series prediction. Training of RBF network involves two steps: (1) the basis function parameters corresponding to hidden neurons are determined by clustering and or heuristic method; (2) the final-layer weights are determined by least square which reduces to solve a simple linear system. This stage requires the solution of a linear problem, which is very fast.

The problem of selecting suitable number of basis functions is an important issue for RBFN [33]. The number of basis functions controls the complexity and the generalization ability of RBFN. Too few basis functions give poor predictions on new data due to limited flexibility [34]. It results in a high bias and low variance estimator. Too many basis functions also result in poor generalization as it is too flexible and fits the noise in the training data. It causes a low bias but high variance estimator. The best generalization performance is obtained by a compromise between conflicting requirements of reducing bias as well as variance.

The center of gravity and width is of particular importance for the improvement of the performance of RBFN. There are many approaches along the line with their own pros and cons. This paper proposes the use of differential evolution to find hidden centers and spreads. The motivation using differential evolution over other EAs such as GAs ([35].) 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.

Regarding missing values treatment in RBFNs there are some contributions using the RBFNs to predict the missing values [36] or obtaining the Radial Basis Function from a Vector Quantization of the dataset with missing values [37]. Also, the impact of missing values in the RBFNs has been considered in [38], but only in a case study.

### 2.3. Differential evolution

Differential evolution [39] is a population based optimization technique, which tries to improve candidate solutions in an iterative manner. In DE, a population at a generation G consists of a set of *d*-dimensional parameter vectors, where each vector refers to a possible solution generated randomly at the first generation. DE is coming under the umbrella of EA ([40]; Michalewicz, 1996), it has many similarities with GAs [35]. However, DE differs significantly from GAs e.g., trial vector generation process uses the information of distance and direction from the current population. Further, in GA crossover is applied first and then mutation operations are carried out to generate improved candidate solutions for the next generation. But in DE, mutation is applied first to create a donor vector and then the trial vector is generated through crossover. Then the selection process selects the vectors for the next generation depending on the fitness values. In Price et al. (2007) a comprehensive comparison of the performance of DE with a range of other optimizers, including GA has been reported and results shows that DE is consistently as good as the best obtained by other optimizers across a wide range of problem instances. In [41] show that DE performs better than the GA or the particle swarm optimization [42].

Many variants of DE algorithms [39, 43, 44] have been developed over the years due to the continuous effort of researchers. But, we develop a new version of the algorithm based on the DE/rand/1/bin scheme (Storn and Price, 1995) with adaptation of control parameters and an equilibrium parameter in mutation. Each variant of the DE algorithm is described using the shorthand DE/*b*/*dv*/*z* (*b* specifies how the base vector (of real values) is chosen (rand if it is randomly selected, or best if the best individual in the population is selected), *dv* 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)) and the adjustment of control parameters [45].

## 3. Proposed method

The proposed method consists of two major phases: (i) imputing missing values, and (ii) designing a suitable optimal model based on SAEDE [44] optimized RBFN to classify unknown sample. In first phase, we impute the missing attribute values by three techniques: (i) Knn, (ii) mean, and (iii) medoid. The first two methods are well known in imputing literatures. But, the third method is introduced by us and is discussed as follows.

Medoid is a representative data point of a given data set, selected in such a manner that it's average distance to all other data points in the dataset is minimal. It is also defined as the nearest neighbor of mean. The medoid based imputation shares the properties of mean based imputation. But, unlike mean, medoid has the property that the centers are located among the data points themselves. It is considered to be a more robust estimate of a representative point than the mean. We have illustrated through a simple example as given below. Fig. 1 illustrates the data points along with mean and medoid. The mean of the data is represented as a blue color diamond, whereas the asterisk inside a circle is the medoid of the dataset.

From this example, it is imperative to note that due to noise the mean may not be the good representative as compared to medoid. In this case the average distance of the points from the medoid is comparatively less than mean. Hence, medoid based imputation (in the case of dataset with outliers) can be chosen as the suitable alternative to mean based imputation although it is computationally expensive than mean based imputation.

In second phase, we have designed a novel self-adaptive equilibrium differential evolution (SAEDE) algorithm for better

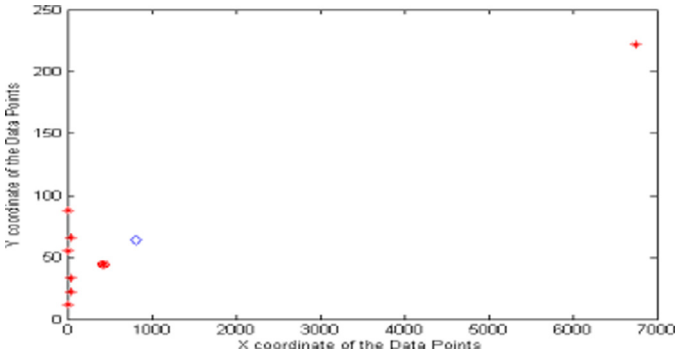


**Fig. 1.** Illustration of best representatives. (For interpretation of the references to colour in the text, the reader is referred to the web version of this article.)

exploration and exploitation of the search space to reveal optimal values of center, spread, and bias for RBFN. Thereafter, the optimized RBFN is used for classification of unknown samples. Let us discuss the SAEDE for optimizing the parameters of RBFN.

***SAEDE:*** In order to get rid of the shortcomings of basic DE, researchers have improved DE in many ways. The potential where DE can be improved is broadly classified into two categories: (i) hybridization of DE with other population based probabilistic or deterministic algorithms and (ii) a new mechanisms for controlling the evolution, which may require new parameters and/or fine tuning of DE controlling parameters *N*, $f_{\mathrm{m}}$, and $c_{\mathrm{r}}$.

Our proposed self-adaptive equilibrium DE is focused on evolution of control parameters such as $f_{\mathrm{m}}$, $c_{\mathrm{r}}$, and a new self confidence weight factor *w*. This new approach falls under the category two of the above classification.

***Motivations:*** The inherent drawback with most of the population based stochastic algorithms including DE is premature convergence. Any population based algorithm is regarded as an efficient algorithm if it is fast in convergence and able to explore the maximum area of search space. In other words, if a population based algorithm is capable of balancing between exploration and exploitation of the search space, then the algorithm is regarded as an efficient. Hence the problem of pre-mature convergence and stagnation (i.e., DE sometimes stop proceedings toward the global optima even though the population has not converged to local optima or any other point) is a matter of serious consideration for designing a comparatively efficient DE algorithm. The proposed self-adaptive equilibrium DE combines the idea of self adapting the control parameters such as $f_{\mathrm{m}}$, $c_{\mathrm{r}}$ and a weight *w* associated with target vector.

Exploration of the whole search space and exploitation of the near optimal solution region may be balanced by maintaining the diversity in early and later iteration of any random number based search algorithm. In DE suitable values of three control parameters namely amplification factor of the difference vector $f_{\mathrm{m}}$, crossover rate $c_{\mathrm{r}}$, and population size (*N*) are generally set using trial and error method. On the other hand, in SADE [45], self adaptive control mechanism is used for each population vector to determine the parameters $f_{\mathrm{m}}$ and $c_{\mathrm{r}}$ during the execution of the program. The third parameter *N* is kept constant throughout the run. Like SADE, we have made self-adaptive to the parameters $f_{\mathrm{m}}$ and $c_{\mathrm{r}}$ as follows:

$$f_m^i(t+1) = \begin{cases} f_m^l + U_1(0,1).f_m^u & if \quad U_2(0,1) < \tau_1 \\ f_m^i(t) & otherwise \end{cases} \tag{2}$$

and

$$c_r^i(t+1) = \begin{cases} U_3(0,1) & if \quad U_4(0,1) < \tau_2 \\ c_r^i(t) & Otherwise \end{cases} \tag{3}$$

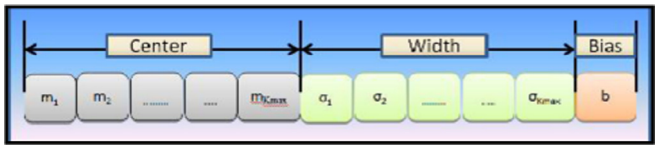


**Fig. 2.** Structure of the individual.

Here $\tau_1$ and $\tau_2$ represent the probabilities to adjust the parameters $f_m$ and $c_r$ respectively. It is suggested that $\tau_1 = \tau_2 = 0.1$ and the values of $f^l_m$ and $f^u_m$ are fixed at 0.1 and 0.9. Therefore, the new $f_m$ takes value from [0.1, 1.0] and $c_r$ from [0, 1], randomly. The newly introduced parameter $w$ can modify the mutation Eq. (4) in DE as follows.

$$u_i(t) = w \times x_k(t) + f_m \times (x_i(t) - x_m(t)) \tag{4}$$

The donor vector $u_i(t)$ is the weighted sum of target vector and the difference vector. Here $w$ is the weight to the target vector and $f_m$ is the weight to the difference vector. In canonical DE, $w$ is set to 1 and $f_m \in (0, \infty)$. However, in this work, $w$ is a value varies from 0.1 to 0.9 during the evolution along with self-adaptive strategies of $f_m$ and $c_r$. Since these three parameters are adaptive and make a balance of exploration and exploitation, therefore, its name is self-adaptive equilibrium DE (SAEDE).

It is clear from the Eq. (4) that small $w$ and large $f_m$ increases the exploration capability, as the weight for individual's current position is low. On the other hand, larger $w$ and smaller $f_m$ can increase the capability of exploitation. Therefore, a suitable value of $w$ and $f_m$ can balance the diversity in the population. The value of $w$ is linearly increasing with the iterative generations as follows:

$$w = w_{\min} + (w_{\max} - w_{\min}) \cdot (t/T) \tag{5}$$

where, $t$ is the iteration index, representing the evolutionary generations. Here, the maximal and minimal weights $w_{\max}$ and $w_{\min}$ are usually set as 0.9 and 0.1, respectively.

Notice that, in the second phase, we are focusing on the learning of the classifier. However, learning RBFNs itself involves two steps procedure. In step one SAEDE is employed to reveal the centers, width, and bias of the RBFNs. Although centers, widths, bias, and weights that connect kernel nodes and output node simultaneously be evolved using DE, but here we restrict ourselves with evolving only centers, spreads, and bias. This ensures efficient representation of an individual of DE. If all these parameters are encoded, then the length of the individual would be too long and hence the search space size becomes too large, which results in a very slow convergence rate. Since the performance of the RBFNs mainly depends on center and width of the kernel, we just encode the centers and widths into an individual for stochastic search.

Suppose the maximum number of kernel nodes is set to $K_{max}$, then the structure of the individual is represented as follows (cf., Fig. 2):

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

The fitness function which is used to guide the search process is defined in Eq. (6).

$$f(x) = \frac{1}{N_t} \sum_{i=1}^{N_t} (o_i - \hat{\Phi}(\hat{x}_i))^2, \tag{6}$$

where, $N_t$ is the total number of training sample, $o_i$ is the actual output and $\hat{\Phi}(\vec{x}_i)$ is the estimated output of RBFNs. Given that the centers, widths, and bias which are computed from training vectors, the weight of the learning network are computed by the following pseudo-inverse matrix manipulation method.

$$Y = W\Phi$$

**Table 2**
Description of datasets.

| Dataset | #Instances | #Attributes | #Classes |
|---|---|---|---|
| Diabetes | 768 | 8 | 2 |
| Mammographic | 961 | 5 | 2 |
| Wisconsin | 699 | 50 | 2 |
| House-votes | 435 | 16 | 2 |
| Hepatitis | 155 | 19 | 2 |
| Post-operative | 90 | 8 | 3 |
| Horse Colic | 368 | 8 | 2 |

$$\Rightarrow W = (\Phi^T\Phi)^{-1}\Phi^T Y \tag{7}$$

**Algorithm for SAEDE-RBFN**

```
BEGIN
Initialize the control parameters like f_m, c_r, N, t, T, and w.
Generate randomly an initial population of N solutions.
  WHILE (t < T)
    For each individual x_i(t) ∈ P(t) do
      Evaluate the fitness using Eq. (6)
      Create the donor vector.
      Create the trail vector.
      If (f(u_i(t)) > f(x_i(t)))
        For each member of the next generation
          Add u_i(t) ∈ P(t)
      Else
          Add x_i(t) ∈ P(t + 1)
      End if
    End for
    Replace P(t) by P(t + 1)
    Update the parameters using , 3) and (4)
  END WHILE
```

## 4. Experimental study

The characteristics of the datasets and parameters used for simulation along with results obtained by two different methods DE-RBFN and SAEDE-RBFN are presented here.

### 4.1. Description of datasets and parameters

Table 2 describes eight datasets obtained from the UCI [27] and KEEL repository (http://sci2s.ugr.es/keel).

In the case of Horse Colic and House-votes datasets, the percentage of missing values is comparatively more than other datasets considered in this work. In our experiment, every dataset is divided into two mutually exclusive parts: training set and test set. The experimental results of our proposed method SAEDE optimized RBFN and DE optimized RBFN has been illustrated under different imputation with 95% and 98% of confidence intervals. The parameters such as maximum iteration, size of the population, $f_m$, $c_r$ and $w$ are fixed for all datasets are 200, 50, [0.1, 1.0], [0, 1], and [0.1, 0.9], respectively. The experimental study of both classifiers has been carried out in MATLAB 6.5.

### 4.2. Results and analysis

The average results of the experiment obtained from 10 fold-cross validation of 20 independent runs are presented in Tables 2–7 with confidence level of 95% and 98%.

In Table 3, the one nearest neighbor imputed method has an influence on the newly developed classifier (SAEDE-RBFN). For Diabetes, Wisconsin, and post-operative datasets, the SAEDE-RBFN has clear dominated performance over DE-RBFN.

From Table 4, it is clear that the influence of three-nearest neighbor imputation has not made any improvement in SAEDE-RBFN compared to DE-RBFN. In other words, except the Horse colic dataset, SAEDE-RBFN and DE-RBFN are equally good.

**Table 3**
Results obtained from (K = 1 nn) DE-RBFN and SAEDE-RBFN with confidence levels 95% and 98%.

| Dataset | DE-RBFN 95% | DE-RBFN 98% | SAEDE-RBFN 95% | SAEDE-RBFN 98% |
|---|---|---|---|---|
| Diabetes | 72.6563 ± .044 | 72.6563 ± .053 | 77.0291 ± .042 | 77.0291 ± .049 |
| Mammographic | 81.2890 ± .035 | 81.2890 ± .041 | 81.2276 ± .035 | 81.2276 ± .041 |
| Wisconsin | 83.4286 ± .039 | 83.4286 ± .046 | 89.3411 ± .032 | 89.3411 ± .038 |
| House-votes | 93.1193 ± .033 | 93.1193 ± .040 | 94.2889 ± .031 | 94.2889 ± .037 |
| Hepatitis | 91.0256 ± .063 | 91.0256 ± .075 | 91.0256 ± .063 | 91.0256 ± .075 |
| Post-operative | 66.6667 ± .138 | 66.6667 ± .164 | 73.8949 ± .129 | 73.8949 ± .154 |
| Horse Colic | 79.8913 ± .058 | 79.8913 ± .069 | 81.1502 ± .056 | 81.1502 ± .067 |

**Table 4**
Results obtained from (K = 3 nn) DE-RBFN and SAEDE-RBFN with confidence levels 95% and 98%.

| Dataset | DE-RBFN 95% | DE-RBFN 98% | SAEDE-RBFN 95% | SAEDE-RBFN 98% |
|---|---|---|---|---|
| Diabetes | 78.6458 ± .041 | 78.6458 ± .049 | 78.1999 ± .041 | 78.1999 ± .049 |
| Mammographic | 83.7500 ± .033 | 83.7500 ± .039 | 83.3537 ± .033 | 83.3537 ± .039 |
| Wisconsin | 90.8571 ± .031 | 90.8571 ± .037 | 90.6865 ± .031 | 90.6865 ± .037 |
| House-votes | 91.2844 ± .037 | 91.2844 ± .044 | 91.2844 ± .037 | 91.2844 ± .044 |
| Hepatitis | 91.0256 ± .063 | 91.0256 ± .146 | 91.0256 ± .063 | 91.0256 ± .075 |
| Post-operative | 75.5556 ± .126 | 75.5556 ± .15 | 70.3136 ± .133 | 70.3136 ± .159 |
| Horse Colic | 78.8043 ± .059 | 78.8043 ± .069 | 77.6790 ± .060 | 77.6790 ± .072 |

**Table 5**
Results obtained from (K = 5 nn) DE-RBFN and SAEDE-RBFN with confidence levels 95% and 98%.

| Dataset | DE-RBFN 95% | DE-RBFN 98% | SAEDE-RBFN 95% | SAEDE-RBFN 98% |
|---|---|---|---|---|
| Diabetes | 79.4271 ± .040 | 79.4271 ± .048 | 78.5662 ± .041 | 78.5662 ± .049 |
| Mammographic | 81.4969 ± .027 | 81.4969 ± .039 | 82.5919 ± .034 | 82.5919 ± .040 |
| Wisconsin | 83.1429 ± .039 | 83.1429 ± .046 | 84.8240 ± .038 | 84.8240 ± .045 |
| House-votes | 96.7890 ± .026 | 96.7890 ± .030 | 96.0792 ± .026 | 96.0792 ± .030 |
| Hepatitis | 89.7436 ± .069 | 89.7436 ± .082 | 90.7710 ± .066 | 90.7710 ± .078 |
| Post-operative | 66.6667 ± .138 | 66.6667 ± .163 | 67.1813 ± .137 | 67.1813 ± .163 |
| Horse Colic | 75.5435 ± .062 | 75.5435 ± .072 | 82.5846 ± .055 | 82.5846 ± .065 |

**Table 6**
Results obtained from (K = 7 nn) DE-RBFN and SAEDE-RBFN with confidence levels 95% and 98%.

| Dataset | DE-RBFN 95% | DE-RBFN 98% | SAEDE-RBFN 95% | SAEDE-RBFN 98% |
|---|---|---|---|---|
| Diabetes | 74.4792 ± .043 | 74.4792 ± .051 | 73.7417 ± .044 | 73.7417 ± .051 |
| Mammographic | 81.4969 ± .035 | 81.4969 ± .039 | 81.5412 ± .035 | 81.5412 ± .041 |
| Wisconsin | 87.7143 ± .035 | 87.7143 ± .039 | 85.2482 ± .037 | 85.2482 ± .044 |
| House-votes | 93.1193 ± .033 | 93.1193 ± .040 | 93.7366 ± .033 | 93.7366 ± .040 |
| Hepatitis | 91.0256 ± .063 | 91.0256 ± .074 | 90.7710 ± .066 | 90.7710 ± .078 |
| Post-operative | 75.5556 ± .126 | 75.5556 ± .149 | 73.8949 ± .129 | 73.8949 ± .154 |
| Horse Colic | 80.4348 ± .057 | 80.4348 ± .067 | 80.0335 ± .057 | 80.0335 ± .068 |

**Table 7**
Results obtained from (mean imputation) DE-RBFN and SAEDE-RBFN with confidence levels 95% and 98%.

| Dataset | DE-RBFN 95% | DE-RBFN 98% | SAEDE-RBFN 95% | SAEDE-RBFN 98% |
|---|---|---|---|---|
| Diabetes | 73.9583 ± .044 | 73.9583 ± .051 | 72.7235 ± .044 | 72.7235 ± .053 |
| Mammographic | 81.0811 ± .035 | 81.0811 ± .039 | 81.7373 ± .035 | 81.7373 ± .041 |
| Wisconsin | 80.4574 ± .041 | 80.4574 ± .048 | 86.3797 ± .036 | 86.3797 ± .043 |
| House-votes | 95.4128 ± .028 | 95.4128 ± .032 | 95.6779 ± .028 | 95.6779 ± .034 |
| Hepatitis | 74.3590 ± .097 | 74.3590 ± .114 | 87.6482 ± .074 | 87.6482 ± .088 |
| Post-operative | 66.6667 ± .138 | 66.6667 ± .163 | 66.7544 ± .138 | 66.7544 ± .163 |
| Horse Colic | 84.7826 ± .052 | 84.7826 ± .062 | 84.3219 ± .052 | 84.3219 ± .062 |

From Table 6, it is noted that, DE-RBFN performs better than the newly developed SAEDE-RBFN in Diabetes, Wisconsin, Hepatitis, and Post-Operative. However, it is interesting to note that SAEDE-RBFN is competitive with the datasets containing large missing values e.g., Horse Colic and House Votes. From Table 5 it is clear that the proposed method is comparatively better than DE-RBFN.

The mean and medoid based imputations (cf., Tables 7 and 8) have a greater influence in classification compared to k = 3 and 7 nearest neighbor imputation by using the proposed model. However, SAEDE-RBFN classification on medoid based imputed databases is depicting very promising and dominating results.

From Table 9, it is clear that on an average, irrespective of datasets, medoid and three nearest neighbor based imputation are giving better results in the case of DE-RBFN. However, the newly proposed classifier SAEDE-RBFN has demonstrated dominating results in medoid imputed database compared to other imputed methods. Fig. 3 illustrates the pictorial view of the performance of DE-RBFN on imputed databases.

**Table 8**
Results obtained from (medoid imputation) DE-RBFN and SAEDE-RBFN with confidence levels 95% and 98%.

| Dataset | DE-RBFN 95% | DE-RBFN 98% | SAEDE-RBFN 95% | SAEDE-RBFN 98% |
|---|---|---|---|---|
| Diabetes | 78.5901 ± .041 | 78.5901 ± .048 | 79.2029 ± .040 | 79.2029 ± .062 |
| Mammographic | 84.8233 ± .032 | 84.8233 ± .037 | 84.9011 ± .032 | 84.9011 ± .038 |
| Wisconsin | 90.8571 ± .031 | 90.8571 ± .037 | 90.9221 ± .031 | 90.9221 ± .037 |
| House-votes | 95.4128 ± .026 | 95.4128 ± .030 | 96.0930 ± .026 | 96.0930 ± .030 |
| Hepatitis | 89.7436 ± .069 | 89.7436 ± .082 | 90.6983 ± .066 | 90.6983 ± .078 |
| Post-operative | 66.6667 ± .138 | 66.6667 ± .163 | 74.6103 ± .128 | 74.6103 ± .152 |
| Horse Colic | 83.6957 ± .054 | 83.6957 ± .062 | 83.7783 ± .054 | 83.7783 ± .064 |

**Table 9**
Classification accuracy obtained through DE-RBFN.

| Dataset | K = 1 nn | K = 3 nn | K = 5 nn | K = 7 nn | Mean | Medoid |
|---|---|---|---|---|---|---|
| Diabetes | 72.65 | 78.65 | 79.43 | 74.48 | 73.96 | 78.59 |
| Mammographic | 81.29 | 83.75 | 81.50 | 81.50 | 81.08 | 84.82 |
| Wisconsin | 83.43 | 90.86 | 83.14 | 87.71 | 80.46 | 90.86 |
| House Votes | 93.12 | 91.28 | 96.79 | 93.12 | 95.41 | 95.41 |
| Hepatitis | 91.03 | 91.03 | 89.74 | 91.03 | 74.36 | 89.74 |
| PostOperatives | 66.67 | 75.56 | 66.67 | 75.56 | 66.67 | 66.67 |
| Horse Colic | 79.90 | 78.80 | 75.54 | 80.43 | 84.78 | 83.70 |
| Average | 81.15 | 84.27 | 81.83 | 83.40 | 79.53 | 84.26 |
| Maximum | 93.12 | 91.28 | 96.79 | 93.12 | 95.41 | 95.41 |
| Minimum | 66.67 | 75.56 | 66.67 | 74.48 | 66.67 | 66.67 |

**Table 10**
Classification accuracy obtained through SAEDE-RBFN.

| Dataset | K = 1 nn | K = 3 nn | K = 5 nn | K = 7 nn | Mean | Medoid |
|---|---|---|---|---|---|---|
| Diabetes | 77.03 | 78.20 | 78.57 | 73.74 | 72.72 | 79.20 |
| Mammographic | 81.23 | 83.35 | 82.59 | 81.54 | 81.74 | 84.90 |
| Wisconsin | 89.34 | 90.69 | 84.82 | 85.25 | 86.38 | 90.92 |
| House Votes | 94.29 | 91.28 | 96.08 | 93.74 | 95.68 | 96.09 |
| Hepatitis | 91.03 | 91.03 | 90.77 | 90.77 | 87.65 | 90.70 |
| PostOperatives | 73.89 | 70.31 | 67.18 | 73.89 | 66.75 | 74.61 |
| Horse Colic | 81.15 | 77.68 | 82.58 | 80.03 | 84.32 | 83.78 |
| Average | 83.99 | 83.22 | 83.23 | 82.71 | 82.18 | 85.74 |
| Maximum | 94.29 | 91.28 | 96.08 | 93.74 | 95.68 | 96.09 |
| Minimum | 73.89 | 70.31 | 67.18 | 73.74 | 66.75 | 74.61 |

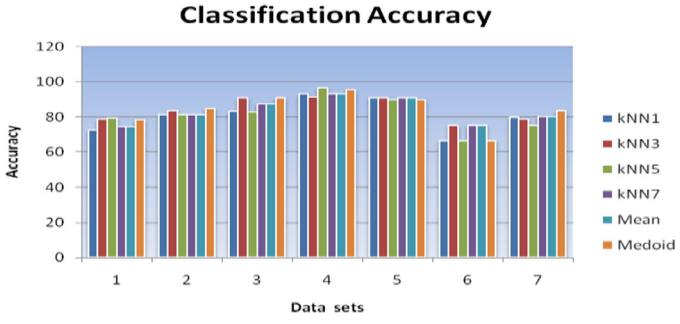


**Fig. 3.** Classification accuracy of DE-RBFN w.r.t different imputed methods [the x-axis represents the dataset numbered as 1 = Diabetes, 2 = Mammographic, 3 = Wisconsin, 4 = House Votes, 5 = Hepatitis, 6 = Post Operatives, and 7 = Horse Colic].

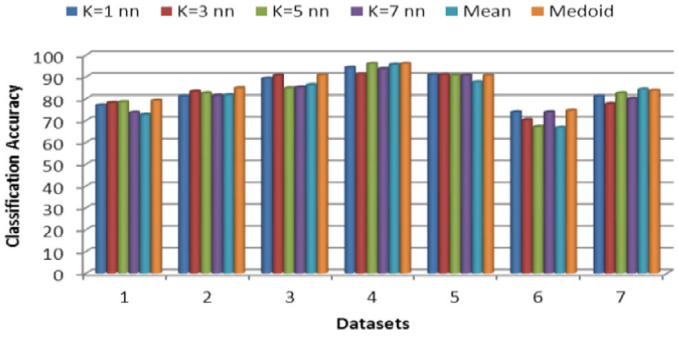


**Fig. 4.** Classification accuracy of SAEDE-RBFN w.r.t. different imputed methods [the x-axis represents the dataset numbered as 1 = Diabetes, 2 = Mammographic, 3 = Wisconsin, 4 = House Votes, 5 = Hepatitis, 6 = Post Operatives, and 7 = Horse Colic].

Table 10 and Fig. 4 displays classification accuracies of SAEDE-RBFN approach with different imputed methods for Diabetes, Mammographic, Wisconsin, House Votes, Hepatitis, Post Operatives and Horse Colic datasets.

Table 11 illustrates the CPU time in seconds for DE-RBFN and SAEDE-RBFN approaches. It is observed that the proposed approach computational time is not better than DE-RBFN. This is due to the balancing of exploitation and exploration natures of the proposed method. As we know, for better accuracy, we need to pay more cost. Similarly, Table 12 shows the computational complexity of k-nn based imputation.

**Table 11**
CPU time of DE-RBFN and SAEDE-RBFN.

| Dataset | DE-RBFN | SAEDE-RBFN |
|---|---|---|
| Diabetes | 37.3281 | 38.54 |
| Mammographic | 12.5313 | 20.12 |
| Wisconsin | 41.2969 | 45.16 |
| House-votes | 191.547 | 193.89 |
| Hepatitis | 44.625 | 43.127 |
| Post-operative | 20.7656 | 25.73 |
| Horse Colic | 271.594 | 251.43 |

**Table 12**
CPU time of imputed methods.

| Dataset | K = 1 nn | K = 3 nn | K = 5 nn | K = 7 nn |
|---|---|---|---|---|
| Diabetes | 3.15625 | 2.46875 | 2.71875 | 2.4375 |
| Mammographic | 2.46875 | 3.14063 | 2.60938 | 3.04688 |
| Wisconsin | 1.15625 | 1.4375 | 1.51563 | 1.375 |
| House Votes | 1.4375 | 1.92188 | 1.79688 | 1.75 |
| Hepatitis | 0.671875 | 0.890625 | 0.78125 | 0.765625 |
| PostOperatives | 0.46875 | 0.328125 | 0.34375 | 0.359375 |
| Horse Colic | 1.20313 | 1.28125 | 1.26563 | 1.375 |

**Table 13**
CPU time of mean and medoid based imputation.

| Dataset | Medoid | Mean |
|---|---|---|
| Diabetes | 1.01563 | 0.265625 |
| Mammographic | 0.984375 | 0.53125 |
| Wisconsin | 1.03125 | 0.5625 |
| House Votes | 0.984375 | 0.640625 |
| Hepatitis | 1.09375 | 0.46875 |
| PostOperatives | 1.03125 | 0.203125 |
| Horse Colic | 1.00 | 0.6875 |

Table 13 illustrates the computation time in seconds of mean and medoid based imputation methods. The medoid based method is more costly than mean based method. However, the proposed algorithm is performing well in medoid imputed database.

## 5. Conclusions

Occurrence of missing values is very common in many datasets belonging to various domains. By removing the samples with missing attribute(s), we end up with missing valuable information resulting in poor classification accuracy. In this paper, we have used imputation based methods to find appropriate replacements for missing values in various datasets. The techniques used to find replacements include Knn, mean, and a newly proposed medoid based imputation. With resulting imputed datasets, our proposed SAEDE-RBFN method has been used to explore the optimal kernel parameters and bias for RBFNs, which is then used to classify unknown samples. Extensive simulation studies on seven datasets obtained from UCI and KEEL machine learning repositories, shows that the average performance of proposed SAEDE-RBFN classifier based on the imputation by medoid is better than that by mean and K = 1, 3, 5, and 7 nearest neighbor methods. Moreover, the medoid based imputation is drawing a clear edge between SAEDE-RBFN and DE-RBFN.

## References


[1] J. Han, M. Kamber, J. Pei, Data Mining Concepts and Techniques, third ed., Morgan Kaufmann, 2011.
[2] M. Hilbert, P. Lopez, The World's technological capacity to store, communicate, and compute information, Science 322 (6025) (2011) 60–65.
[3] S. Dehuri, A. Ghosh, Revisiting evolutionary algorithms in feature selection and non fuzzy/fuzzy rule based classification, Wiley Interdisc. Rev.: Data Mining Knowl. Discov., 3, 2013, pp. 83–108.
[4] R. Rustum, A.J. Adeloye, Replacing outliers and missing values from activated sludge data using Kohonen self-organizing map, J. Environ. Eng. 133 (9) (2007) 909–916.
[5] H. Xian Wang, Q. Bing Zhang, B. Luo, S. Wei, Robust mixture modelling using multivariate t-distribution with missing information, Pattern Recognit. Lett. 25 (6) (2004) 701–710.
[6] M.S.B. Sehgal, I. Gondal, Laurence S. Dooley, Collateral missing value imputation: a new robust missing value estimation algorithm for microarray data, Bioinformatics 21 (10) (2005) 2417–2423.
[7] O. Troyanskaya, M. Cantor, G. Sherlock, P. Brown, T. Hastie, R. Tibshirani, D. Botstein, R.B. Altman, Missing value estimation methods for DNA microarrays, Bioinformatics 17 (6) (2001) 520–525.
[8] G.E. Batista, M.C. Monard, A study of k-nearest neighbors as an imputation method, HIS, 87, 2002, pp. 251–260.
[9] J.L. Schafer, Multiple imputation: a primer, Stat. Methods Med. Res. 8 (1) (1999) 3–15.
[10] Y. Qin, S. Zhang, X. Zhu, J. Zhang, C. Zhang, Expert systems with applications POP algorithm: kernel-based imputation to treat missing values in knowledge discovery from databases, Expert Syst. Appl. 36 (2) (2009) 2794–2804.
[11] O. Kabak, D.A. Ruan, A cumulative belief degree-based approach for missing values in nuclear safeguards evaluation, IEEE Trans. Knowl. Data Eng. 23 (10) (2011) 1441–1454.
[12] P.J. Garcia-Laencina, J.-L. Sancho-Gomez, A.R. Figueiras-Vidal, Pattern classification with missing data: a review, Neural Comput. Appl. 19 (2) (2010) 263–282.
[13] X. Huang, Q. Zhu, Pseudo-nearest-neighbor approach for missing data recovery on Gaussian random data sets, Pattern Recognit. Lett. 23 (13) (2002) 1613–1622.
[14] B. Walczak, D.L. Massart, Dealing with missing data: part I, Chemomet. Intell. Lab. Syst. 58 (1) (2001) 15–27.
[15] B. Twala, M.C. Jones, D.J. Hand, Good methods for coping with missing data in decision trees, Pattern Recognit. Lett. 29 (7) (2008) 950–956.
[16] T.E. Raghunathan, J.M. Lepkowski, J. Van Hoewyk, P. Solenberger, A multivariate technique for multiply imputing missing values using a sequence of regression models, Surv. Methodol. 27 (1) (2001) 85–96.
[17] P.R. Houck, S. Mazumdar, T. Koru-Sengul, G. Tang, B.H. Mulsant, B.G. Pollock, C.F. Reynolds III, Estimating treatment effects from longitudinal clinical trial data with missing values: comparative analyses using different methods, Psychiatr. Res. 129 (2) (2004) 209–215.
[18] N. Friedman, et al., Learning belief networks in the presence of missing values and hidden variables, ICML 97 (1997) 125–133.
[19] E. Pesonen, M. Eskelinen, M. Juhola, Treatment of missing data values in a neural network based decision support system for acute abdominal pain, Artif. Intell. Med. 13 (3) (1998) 139–146.
[20] A.D. Preece, A new approach to detecting missing knowledge in expert system rule bases, Int. J. Man Mach. Stud. 38 (4) (1993) 661–688.
[21] J. de Andrade Silva, E.R. Hruschka, An experimental study on the use of nearest neighbors-based imputation algorithms for classification tasks, Data Knowl. Eng. 84 (2013) 47–58.
[22] P. Royston, Multiple imputation of missing values, Stata J. 4 (2004) 227–241.
[23] H. Wang, S. Wang, Discovering patterns of missing data in survey databases: an application of rough sets, Expert Syst. Appl. 36 (3) (2009) 6256–6260.
[24] M. Abdella, T. Marwala, The use of genetic algorithms and neural networks to approximate missing data in database, in: Proceedings of IEEE Third International Conference on Computational Cybernetics, 2005, pp. 207–212.
[25] C.H. Mallinckrodt, T.M. Sanger, S. Dube, D.J. DeBrota, G. Molenberghs, R.J. Carroll, W.Z. Potter, G.D. Tollefson, Assessing and interpreting treatment effects in longitudinal clinical trials with missing data, Biol. Psychiatr. 53 (8) (2003) 754–760.
[26] J.L. Schafer, M.K. Olsen, Multiple imputation for multivariate missing-data problems: a data analyst's perspective, Multivar. Behav. Res. 33 (4) (1998) 545–571.
[27] A. Frank, A. Asuncion, UCI Machine Learning Repository, University of California, School of Information and Computer Science, Irvine, CA, 2010 http://archive.ics.uci.edu/ml .
[28] H.-Y. Shum, K. Ikeuchi, R. Reddy, Principal component analysis with missing data and its application to polyhedral object modeling, IEEE Trans. Pattern Anal. Mach. Intell. 17 (9) (1995) 854–867.
[29] B. Walczak, D.L. Massart, Dealing with missing data: part II, Chemomet. Intell. Lab. Syst. 58 (1) (2001) 29–42.
[30] E. Acuna, C. Rodriguez, The Treatment of Missing Values and Its Effect on Classifier Accuracy. Classification, Clustering, and Data Mining Applications, 2004, pp. 639–647.
[31] G.E. Batista, M.C. Monard, An analysis of four missing data treatment methods for supervised learning, Appl. Artif. Intell. 17 (5–6) (2003) 519–533.
[32] S. Haykin, Neural Networks: A Comprehensive Foundation, Prentice Hall, Upper Saddle River, NJ, 1994.
[33] N. Naveen, V. Ravi, C.R. Rao, N. Chauhan, Differential evolution trained radial basis function network: application to bankruptcy prediction in banks, Int. J. Bio-Inspired Comput. 2 (3) (2010) 222–232.
[34] C.S.K. Dash, A.P. Dash, S. Dehuri, S.-B. Cho, G.-N. Wang, DE+ RBFNs based classification: a special attention to removal of inconsistency and irrelevant features, Eng. Appl. Artif. Intell. 26 (10) (2013) 2315–2326.
[35] D.E. Goldberg, Genetic Algorithm in Search Optimization and Machine Learning, Addison-Wesley, Reading, MA, 1989.
[36] M. Uysal, Reconstruction of time series data with missing values, J. Appl. Sci. 7 (2007) 922–925.
[37] A. Lendasse, D. Francois, V. Wertz, M. Verleysen, Vector quantization: a weighted version for time-series forecasting, Fut. Gener. Comput. Syst. 21 (2005) 1056–1067.
[38] C.W. Morris, L. Boddy, M.F. Wilkins, Effects of missing data on RBF neural network identification of biological taxa: discrimination of microalgae from flow cytometry data, Int. J. Smart Eng. Syst. Des. 3 (2001) 195–202.
[39] R. Storn, K. Price, Differential evolution—simple and efficient heuristic for global optimization over continuous spaces, J. Global Optim. 11 (1997) 341–359.
[40] S. Forerest, Genetic algorithms: principles of natural selection applied to computation, Science 261 (1993) 872–888.
[41] J. Vesterstorm, R. Thomsen, A comparative study of differential evolution, particle swarm optimization, and evolutionary algorithms on numerical benchmark problems, in: Proceedings of IEEE Congress on Evolutionary Computation, 2, 2004, pp. 1980–1987.

[42] J. Kennedy, R. Eberhart, Particle swarm optimization, in: Proceedings of IEEE International Conference on Neural Networks, 4, 1995, pp. 1942–1948.
[43] S. Das, P.N. Suganthan, Differential evolution: a survey of the state-of-the-art, IEEE Trans. Evol. Comput. 15 (1) (2011) 4–31.
[44] A. Ghosh, A. Datta, S. Ghosh, Self-adaptive differential evolution for feature selection in hyperspectral image data, Appl. Soft Comput. 13 (4) (2013) 1969–1977.
[45] J. Brest, S. Greiner, B. Boskovice, M. Mernik, V. Zumer, Self-adaptive control parameters in differential evolution: a comparative study on numerical benchmark problems, IEEE Trans. Evol. Comput. 10 (6) (2006) 646–657.
[46] J.W. Grzymala-Busse, M.A. Hu, A comparison of several approaches to missing attribute values in data mining, in: Proceedings of the Second International Conference on Rough Sets and Current Trends in Computing, 2000, pp. 378–385.
[47] A. Farhangfar, L. Kurgan, J. Dy, Impact of imputation of missing values on classification error for discrete data, Pattern Recognit. 41 (12) (2008) 3692–3705.
[48] P.J. GarcıA-Laencina, J.-L. Sancho- Gomez, A.R. Figueiras-Vidal, M. Verleysen, K nearest neighbors with mutual information for simultaneous classification and missing data imputation, Neurocomputing 72 (7) (2009) 1483–1493.