MEDICAL DIAGNOSIS CLASSIFICATION USING MIGRATION BASED DIFFERENTIAL EVOLUTION ALGORITHM

Htet Thazin Tike Thein[1] and Khin Mo Mo Tun[2]

[1]Ph.D Student, University of Computer Studies, Yangon, Myanmar
[2]Department of Computational Mathematics,
University of Computer Studies, Yangon, Myanmar
[1]htetthazintikethein@ucsy.edu.mm,[2]khinmo2tun@gmail.com

ABSTRACT

Constructing a classification model is important in machine learning for a particular task. A classification process involves assigning objects into predefined groups or classes based on a number of observed attributes related to those objects. Artificial neural network is one of the classification algorithms which, can be used in many application areas. This paper investigates the potential of applying the feed forward neural network architecture for the classification of medical datasets. Migration based differential evolution algorithm (MBDE) is chosen and applied to feed forward neural network to enhance the learning process and the network learning is validated in terms of convergence rate and classification accuracy. In this paper, MBDE algorithm with various migration policies is proposed for classification problems using medical diagnosis.

KEYWORDS

Artificial Neural Network, Differential Evolution, Island Model, Classification, Medical Diagnosis

1. INTRODUCTION

An objective function describing the artificial neural network (ANN) training problem is going to be multimodal. Therefore, algorithm based on gradient methods can easily get stuck in local minima. To avoid this problem, it is possible to use a global optimization technique, such as, for example, the differential evolution (DE) algorithm [1], [2], which is a variant of the evolutionary algorithm [3], [4], or other techniques, such as: a particle swarm optimization algorithm [5], a continuous ant colony optimization algorithm [6], a bee colony optimization algorithm [7], or an evolutionary strategy [8]. The DE algorithm is a heuristic algorithm for global optimization. It was introduced several years ago (in 1997) and has been developed intensively in recent years [9]. Its advantages are as follows: the possibility of finding the global minimum of a multimodal function regardless of the initial values of its parameters, quick convergence, and the small number of parameters that needs to be set up at the start of the algorithm's operation [10]. Since 1997, the DE algorithm has been modified to increase its effectiveness

.

To speed up an evolutionary computation it is a common practice to use multiple machines. Island models behave qualitatively different from standard EAs and even using them on a single

Natarajan Meghanathan et al. (Eds) : NeTCoM, CSIT, GRAPH-HOC, SPTM - 2014
pp. 153–163, 2014. © CS & IT-CSCP 2014 DOI : 10.5121/csit.2014.41313

machine may produce different, possibly better solutions for many problems, especially complex ones (for example in engineering). Separating individuals spatially from each other results in slowing down the information flow between individuals, which may have both desired and undesired results. A slower information flow may stop temporarily best solution from dominating the population and allow different building blocks or solutions to be discovered and later confronted, which is important in the context of engineering design and creativity. On the other hand one can prevent successful mixing, which could otherwise lead to constructing a novel solution.

In this paper, a latest optimization algorithm called DE with island model is applied in feed forward neural network to improve neural network learning mechanism. Island based model works by running multiple algorithms and shares the results at regular interval promoting the overall performance of the algorithm. This paper proposes the migration based differential evolution algorithm for classification medical diagnosis.

2. DIFFERENTIAL EVOLUTION ALGORITHM

Having developed an ANN-based process model, a DE algorithm is used to optimize the N-dimensional input space of the ANN model. Conventionally, various deterministic gradient-based methods are used for performing optimization of the phenomenological models. Most of these methods require that the objective function should simultaneously satisfy the smoothness, continuity, and differentiability criteria. Although the nonlinear relationships approximated by an ANN model can be expressed in the form of generic closed-form expressions, the objective function(s) derived thereby cannot be guaranteed to satisfy the smoothness criteria. Thus, the gradient-based methods cannot be efficiently used for optimizing the input space of an ANN model and, therefore, it becomes necessary to explore alternative optimization formalisms, which are lenient towards the form of the objective function.

In the recent years, Differential Evolution (DE) that are members of the stochastic optimization formalisms have been used with a great success in solving problems involving very large search spaces. The DEs were originally developed as the genetic engineering models mimicking the population evolution in natural systems. Specifically, DE like genetic algorithm (GA) enforces the “survival-of-the-fittest” and “genetic propagation of characteristics” principles of biological evolution for searching the solution space of an optimization problem. The principal features possessed by the DEs are: (i) they require only scalar values and not the second- and/or first-order derivatives of the objective function, (ii) the capability to handle nonlinear and noisy objective functions, (iii) they perform global search and thus are more likely to arrive at or near the global optimum and (iv) DEs do not impose pre-conditions, such as smoothness, differentiability and continuity, on the form of the objective function.

Differential Evolution (DE), an improved version of GA, is an exceptionally simple evolution strategy that is significantly faster and robust at numerical optimization and is more likely to find a function’s true global optimum. Unlike simple GA that uses a binary coding for representing problem parameters, DE uses real coding of floating point numbers. The mutation operator here is the addition instead of bit-wise flipping used in GA. And DE uses non-uniform crossover and tournament selection operators to create new solution strings. Among the DEs advantages are its simple structure, ease of use, speed and robustness. It can be used for optimizing functions with real variables and many local optima.

This paper demonstrates a successful application of DE with island model. As already stated, DE in principle is similar to GA. So, as in GA, we use a population of points in our search for the optimum. The population size is denoted by NP. The dimension of each vector is denoted by D.

The main operation is the NP number of competitions that are to be carried out to decide the next generation. To start with, we have a population of NP vectors within the range of the objective function. We select one of these NP vectors as our target vector. We then randomly select two vectors from the population and find the difference between them (vector subtraction). This difference is multiplied by a factor F (specified at the start) and added to the third randomly selected vector. The result is called the noisy random vector. Subsequently, the crossover is performed between the target vector and noisy random vector to produce the trial vector. Then, a competition between the trial vector and target vector is performed and the winner is replaced into the population. The same procedure is carried out NP times to decide the next generation of vectors. This sequence is continued till some convergence criterion is met. This summarizes the basic procedure carried out in differential evolution. The details of this procedure are described below.

Steps performed in DE

Assume that the objective function is of D dimensions and that it has to be optimized. The weighting constants F and the crossover constant CR are specified.

Step 1. Generate NP random vectors as the initial population: generate (NP×D) random numbers and linearize the range between 0 and 1 to cover the entire range of the function. From these (NP×D) numbers, generate NP random vectors, each of dimension D, by mapping the random numbers over the range of the function.

Step 2. Choose a target vector from the population of size NP: first generate a random number between 0 and 1. From the value of the random number decide which population member is to be selected as the target vector (Xi) (a linear mapping rule can be used).

Step 3. Choose two vectors from the population at random and find the weighted difference: Generate two random numbers. Decide which two population members are to be selected (Xa,Xb).Find the vector difference between the two vectors (Xa - Xb).Multiply this difference by F to obtain the weighted difference. Weighted difference = F (Xa - Xb)

Step 4. Find the noisy random vector: generate a random number. Choose the third random vector from the population (Xc). Add this vector to the weighted difference to obtain the noisy random vector (X'c).

Step 5. Perform the crossover between Xi and X'c to find Xt, the trial vector: generate D random numbers. For each of the D dimensions, if the random number is greater than CR, copy from Xi into the trial vector; if the random number is less than CR, copy the value from X'c into the trial vector.

Step 6. Calculate the cost of the trial vector and the target vector: for a minimization problem, calculate the function value directly and this is the cost. For a maximization problem, transform the objective function f(x) using the rule $F(x) = 1 / [1 + f(x)]$ and calculate the value of the cost. Alternatively, directly calculate the value of f(x) and this yields the profit. In case the cost is calculated, the vector that yields the lower cost replaces the population member in the initial population. In case the profit is calculated, the vector with the greater profit replaces the population member in the initial population.

Steps 1–6 are continued till some stopping criterion is met. This may be of two kinds. One may be some convergence criterion that states that the error in the minimum or maximum between two previous generations should be less than some specified value. The other may be an upper bound on the number of generations. The stopping criterion may be a combination of the two. Either

way, once the stopping criterion is met, the computations are terminated. Choosing DE key parameters NP, F, and CR is seldom difficult and some general guidelines are available. Normally, NP ought to be about 5 to 10 times the number of parameters in a vector. As for F, it lies in the range 0.4 to 1.0. Initially, F= 0.5 can be tried and then F and/or NP is increased if the population converges prematurely. A good first choice for CR is 0.1, but in general CR should be as large as possible (Price and Storn, 1997). DE has already been successfully applied for solving several complex problems and is now being identified as a potential source for the accurate and faster optimization.

3. ISLAND MODEL

The main difference between the island model and the single population model is the separation of individuals into islands. As against the master-slave model the communication to computation ratio of the island model approach is low, owing to the low communication frequency between the islands. Also, separating individuals separately from each other results in a qualitative change in the behaviour of the algorithm.

In the island model approach, each island executes a standard sequential evolutionary algorithm. The communication between sub-population is assured by a migration process. Some randomly selected individuals (migration size) migrate from one island to another after every certain number of generations (migration interval) depending upon a communication topology (migration topology). The two basic and most sensitive parameters of island model strategy are: migration size, which indicates the number of individuals migrating and controls the quantitative aspect of migration; and migration interval denoting the frequency of migration. Although different aspects of migration size and interval were studied in the past, we are unaware of any work studying directly the influence of these parameters on the behaviour of island model based differential evolution, though [15] presents a similar study on a set of 8 standard functions.

3.1. Migration Topology

The migration topology describes which islands send individuals to which islands. There are many topologies. This system investigates the fully connected topology. In this paper, simulations were run with setups of five islands.

3.2. Migration Policy

A migration policy consists of two parts. The first part is the selection of individuals, which shall be migrated to another island. The second part is to choose which individuals are replaced by the newly obtained individuals. Four migration policies are proposed in this system:

- Select the best individuals replace the worst individuals.
- Select random individuals, replace the worst individuals.
- Select the best individuals replace random individuals.
- Select random individuals, replace random individuals.

This system experiments all of the above migration policies and compare their results.

3.3. Migration Interval

In order to distribute information about good individuals among the islands, migration has to take place. This can either be done in synchronous way every n^{th} generation or in an asynchronous way, meaning migration takes place at non-periodical times. It is commonly accepted that a more

frequent migration leads to a higher selection pressure and therefore a faster convergence. But as always with a higher selection pressure come the susceptibility to get stuck in local optima. In this system, various migration intervals are experimented to find the best solution for the neural network training.

3.4. Migration Size

A further important factor is the number of individuals which are exchanged. According to these studies the migration size has to be adapted to the size of a subpopulation of an island. When one migrates only a very small percentage, the influence of the exchange is negligible but if too much individuals are migrated, these new individuals take over the existing population, leading to a decrease of the global diversity. In this system, the migration sizes were chosen to be approximately 10% of the size of a subpopulation as suggested in [19].

4. THE PROPOSED MODEL

As shown in figure 1, the training patterns of medical dataset are used as input data. Attributes are scaled to fall within a small specific range by using min-mix normalization. At the start of the algorithm, dataset were loaded from the database. In the next step, each chromosome or vector is randomly initialized with random neural network weight. Fitness of each chromosome is evaluated using following step. Fitness defined how well a chromosome solves the problem in hand. The first step converts chromosome's genes into neural network and fitness is calculated for each individuals. Mutation operator produce the trial vector from parent vector and randomly selected three vectors. Crossover recombines the parent vector and trial vector to produce offspring. By using mutation and crossover, some genes are modified that mean weights are updated. Fitness of offspring is calculated and compare with fitness of parent vector, the chromosome with high fitness survive and next generation begin. Choose individuals according to migration policy. Migrate and replace individuals according to migration topology. Figure 1 presents the flow of the proposed system.

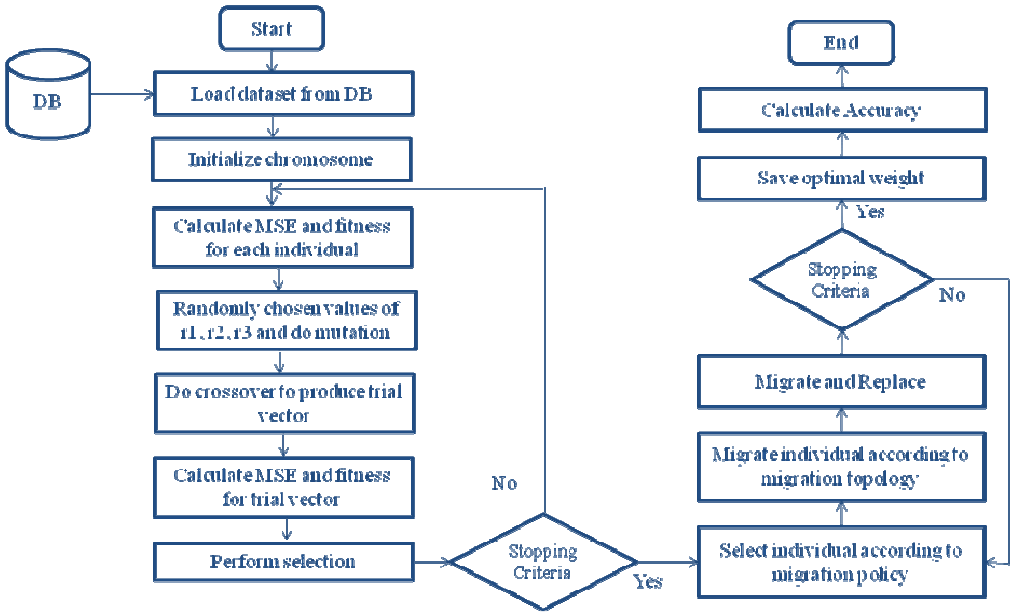

Figure 1. ANNs-MBDE algorithm training process

5. DESCRIPTION OF DATASETS

This paper presents four medical datasets such as Breast Cancer, Heart, Liver and Pima Indian Diabetes from UCI machine learning repository [18]. The size and number of attributes are different for each dataset. The size means number of medical dataset records for training.

5.1. Breast Cancer

The cancer dataset requires the decision maker to correctly diagnose breast lumps as either benign or malignant based on data from automated microscopic examination of cells collected by needle aspiration. The dataset includes 9 inputs and 2 outputs. A total of 345 instances are available in breast cancer data set. 231 instances are used for training and 114 instances are used for testing.

5.2. Heart

The network architecture used for Heart dataset consists of 13 continuous and 2 classes. The attribute are age, sex, chest pain type, resting blood pressure, serum cholestoral, fasting blood pressure, serum cholestoral, fasting blood sugar > 120 mg/dl, resting electrocardiographic results, maximum heart rate achieved, exercise induced angina, oldpeak, the slope of the peak exercise ST segment, number of major vessels and thal. The classes are absent (1) and present (2).. A total of 303 instances are available in heart disease data set. 201 instances are used for training and 100 instances are used for testing.

5.3. Liver

In this paper, we used liver dataset from UCI machine learning repository. There are 345 instances, 6 continuous attributes and 2 classes. The attributes are mean corpuscular volume, alkaline phosphatase, alamine aminotransferase, aspartate, aminotransferase, gamma-glutamyl transpeptidase and number of half-pint equivalents of alcoholic beverages drunk per day. The classes are absents (1) and present (2). The first 5 attributes are all blood tests which are thought to be sensitive to liver disorders that might arise from excessive alcohol consumption. Each record is a single male individual.

5.4. Pima Indian Diabetes Database (PIDD)

There are 768 instances, 8 continuous attributes and 2 classes. PIDD includes the following attributes (1-8 attributes as input and last attributes as target variable) number of times pregnant, plasma glucose concentration a 2 hours in an oral glucose tolerance test, diastolic blood pressure (mm Hg), triceps skin fold thickness (mm), 2 hours serum insulin (mu U/ml), body mass index (weight in kg/ (height in m) ^ 2), diabetes pedigree function and age (years). Class to be predicted is patient is suffering from tested-positive or test-negative.

6. EXPERIMENTAL RESULT

Java programming language, which is the platform independent and a general-purpose development language, is used to implement the proposed system. First of all, the medical datasets are accessed. And then, normalization is made for pre-processing steps according to the different medical data sets. Experiments are performed with four migration policies. Currently the system experiment the island model with fully connected topology and four migration policies. In this system, five islands are used. The island model used the iteration as the migration interval and one-third of the old population is used to migrate and replace. Four medical datasets are used

from the UCI, namely Breast Cancer, Heart, Liver and Pima Indian Diabetes. The results for each dataset are compared and analysed based on the convergence rate and classification performance.

6.1. Parameters of Datasets

The table 1 below shows the parameter of four datasets.

Table 1. Datasets Information

Parameter	Medical Datasets			
	Breast cancer	***Heart***	***Liver***	***Pima Diabetes***
Train Data	456	201	231	509
Test Data	227	100	114	259
Output Neuron	2	2	2	2

6.2. Data Normalization

The data normalization is considered as the most important pre-processing step using neural networks. To improve the performance of multilayer neural networks, it is better to normalize the data entry such that will be found in the interval of [0 1]. To transform the data into digital form, and use it as inputs of the neural network, scaling or normalization should be realized for each attribute. The nine numerical attributes, in the analog form, are scaled with a range of 0 and 1. There are many types of normalization that are found in the literature. The new values obtained after normalization, follow this equation:

$$New\ value\ (after\ normalization) = \frac{current - min}{\max - min} \quad (1)$$

6.3. Classifier Accuracy

Estimating classifier accuracy is important since it determines to evaluate how accurately a given classifier will label future data, data on which the classifier has not been trained. Accuracy estimates also help in the comparison of different classifiers. The following classification features are used to train and test the classifier.

Given: A collection of labeled records (training set). Each record contains a set of features (attributes), and the true class (label).

Find: A model for the class as a function of the values of the features.

Gold: Previously unseen records should be assigned a class as accurately as possible. A test set is used to determine the accuracy of the model. Usually, the given data set is divided into training and test sets, with training set used to build the model and test set used to validate it. The Sensitivity and Specificity measures can be used to determine the accuracy measures.

Precision may also be used to access the percentage of samples labeled as for example, “cancer” that actually are “cancer” samples. These measures are defined as

$$Specificity = \frac{t\_neg}{neg} \tag{2}$$

$$Precision = \frac{t\_pos}{t_{pos} + f\_pos} \tag{3}$$

$$Sensitivity = \frac{t\_pos}{pos} \tag{4}$$

Where,

t_pos = the number of true positives ("medical dataset class" samples that were correctly classified as such class),

pos = the number of positive ("medical dataset class") samples

t_neg = the number of true negative ("not medical dataset class" samples that were correctly classified as such class)

neg = the number of negative samples

f_pos = number of false positive ("not medical dataset class" samples that were incorrectly labeled as such class)

$$accuracy = sensitivity \frac{pos}{pos + neg} + specificity \frac{negs}{pos + neg} \tag{5}$$

6.4. Accuracy Comparisons for UCI Medical Datasets

Four medical datasets are tested with MBDE neural network classifier for this system. Below the tables show the accuracy of training and testing of MBDE neural network on medical datasets.

Table 2. Results of classification accuracy on breast cancer dataset

Migration Policies	Error Convergence	Convergence Time (sec)	Classification Accuracy	
			Training Accuracy (%)	*Testing Accuracy (%)*
Best-Worst	0.0011	9	97.82	100
Best-Random	0.0021	10	99.31	99.43
Random-Worst	0.0037	13	97.35	98.46
Random-Random	0.0043	13	98.76	97.32

Table 3. Results of classification accuracy on heart dataset

Migration Policies	Error Convergence	Convergence Time (sec)	Classification Accuracy	
			Training Accuracy (%)	*Testing Accuracy (%)*
Best-Worst	0.0021	11	97.32	99.89
Best-Random	0.0107	12	98.09	99.32
Random-Worst	0.0130	13	99.14	98.72
Random-Random	0.0160	14	99.04	98.04

Table 4. Results of classification accuracy on liver dataset

Migration Policies	Error Convergence	Convergence Time (sec)	Classification Accuracy	
			Training Accuracy (%)	*Testing Accuracy (%)*
Best-Worst	0.0011	13	99.72	100
Best-Random	0.0167	11	98.62	98.45
Random-Worst	0.0203	11	98.14	97.68
Random-Random	0.0264	12	97.68	97.36

Table 5. Results of classification accuracy on PIDD dataset

Migration Policies	Error Convergence	Convergence Time (sec)	Classification Accuracy	
			Training Accuracy (%)	*Testing Accuracy (%)*
Best-Worst	0.0035	12	98.23	99.01
Best-Random	0.0159	13	98.97	98.34
Random-Worst	0.0159	13	98.35	97.78
Random-Random	0.0178	14	97.65	97.78

The MBDE is successfully applied in neural network and has been tested using Breast Cancer, Heart, Liver and Pima Indian Diabetes datasets.

7. ACCURACY COMPARISON ON MIGRATION POLICIES

This analysis is carried out to compare the results of migration policy. To do this, the learning patterns for the proposed system is compared using all four medical datasets. The comparative correct classification percentage for all datasets is shown in figure 2.

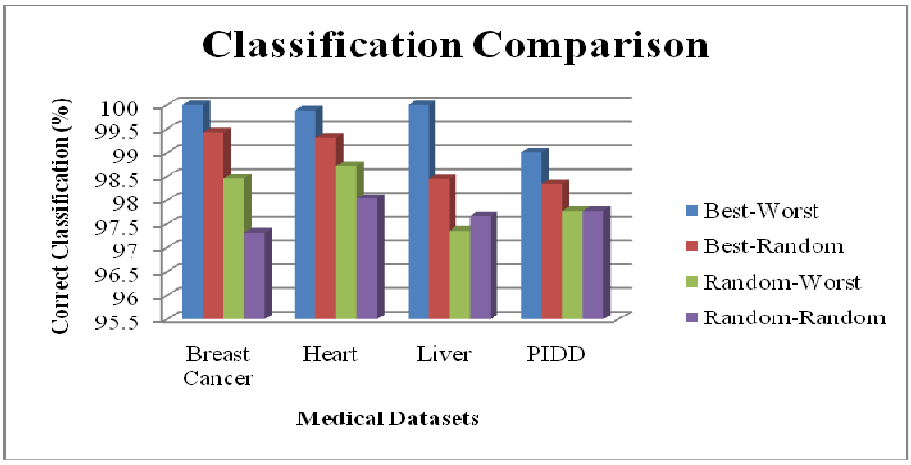

Figure 2. Comparative of correct classification percentage of migration policies

In this paper, we have introduced the various migration policies with fully connected topology and compared their results. Figure 2 shows the accuracy comparison with four migration policies by using medical datasets. Medical datasets are used to implement this proposed system which shows better experiments with higher accuracy. The proposed system also reduces the computing time. For all medical datasets, the experiments show that best-worst migration policy has better results on convergence time and correct classification percentage. The proposed algorithm converges in a short time with high correct classification percentage.

8. CONCLUSIONS

The proposed system (MBDE) algorithm is successfully applied in neural network and has been tested using breast cancer, heart, liver and pima indian diabetes database datasets. The analysis is done by comparing the results for each dataset. The results produced by ANNs are not optimal before using migration based differential evolution algorithm. Therefore, this paper improves the performance of ANNs by using the proposed algorithm, MBDE. The main difficulty of neural network is to adjust weight in order to reduce the error rate. This system presents the neural network training algorithm using migration based differential evolution algorithm. By exploiting the global search power of differential evolution algorithm in conjunction with island model will boost the training performance of the algorithm. The system will converge quickly to the lower mean square error. Island model encourage the diversity among the individual among islands which increase search capability and by migration island model can share the best experiences of each other. By using island model rather than single DE, it can get advantages from parallel problem solving and information sharing which lead to faster global search. This system improves the performance of medical datasets in feed forward neural network and reduces the computing time.

REFERENCES

[1] R. Storn and K. Price, "Differential evolution—A simple and efficient heuristic for global optimization over continuous spaces," J. Glob. Optim., vol. 11, no. 4, pp. 341–359, Dec. (1997).

[2] K. Price, "An introduction to differential evolution," in New Ideas in Optimization, D. Corne, M. Dorigo, and F. Glover, Eds. London, U.K.: McGraw-Hill, pp. 79–108 (1999).

[3] D. Goldberg, Genetic Algorithms in Search, Optimization, and Machine Learning. Reading, MA: Addison-Wesley, (1989).

[4] Z. Michalewicz, Genetic Algorithms+Data Structures=Evolution Programs. Berlin, Germany: Springer-Verlag, (1998).
[5] J. Kennedy, R. C. Eberhart, and Y. Shi, Swarm Intelligence. San Francisco, CA: Morgan Kaufmann, (2001).
[6] K. Socha and M. Doringo, "Ant colony optimization for continous domains,"Eur. J. Oper. Res., vol. 185, no. 3, pp. 1155–1173, Mar. (2008).
[7] D. T. Pham, A. Ghanbarzadeh, E. Koç, S. Otri, S. Rahim, and M. Zaidi, "The bees algorithm—A novel tool for complex optimization problems," in IPROMS 2006. Oxford, U.K.: Elsevier, (2006).
[8] H. G. Beyer and H. P. Schwefel, "Evolution strategies: A comprehensive introduction," Nat. Comput., vol. 1, no. 1, pp. 3–52, May (2002).
[9] K. V. Price, R. M. Storn, and J. A. Lampinen, Differential Evolution: A Practical Approach to Global Optimization. Berlin, Germany: Springer Verlag, (2005).
[10] R. L. Becerra and C. A. Coello Coello, "Cultured differential evolution for constrained optimization," Comput. Methods Appl. Mech. Eng., vol. 195, no. 33–36, pp. 4303–4322, Jul. 1, (2006).
[11] A. Slowik and M. Bialko, "Adaptive selection of control parameters in differential evolution algorithms," in Computational Intelligence: Methods and Applications, L. Zadeh, L. Rutkowski, R. Tadeusiewicz, and J. Zurada, Eds. Warsaw, Poland: EXIT, pp. 244–253, (2008).
[12] J. Liu and J. Lampinen, "A fuzzy adaptive differential evolution algorithm," Soft Comput.—A Fusion of Foundations, Methodologies and Applications, vol. 9, no. 6, pp. 448–462, Jun. (2005).
[13] M. M. Ali and A. Torn, "Population set-based global optimization algorithms: Some modifications and numerical studies," Comput. Oper. Res., vol. 31, no. 10, pp. 1703–1725, Sep. (2004).
[14] E. Mezura-Montes, C. A. Coello Coello, J. Velázquez-Reyes, and L. Munoz-Dávila, "Multiple trial vectors in differential evolution for engineering design," Eng. Optim., vol. 39, no. 5, pp. 567–589, Jul. (2007).
[15] Z. Skolicki and K. De Jong, "The influence of migration sizes and intervals on island models," in Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2005), ACM Press, (2001).
[16] R. Storn, "On the usage of differential evolution for function optimization," in Proc. of the 1996 Biennial Conference of the North American Fuzzy Information processing society- NAFIPS, Edited by: M. H. Smith, M. A. Lee, J. Keller and J. Yen, June 19-22, Berkeley, CA, USA, IEEE Press, New York, pp 519-523, (1996).
[17] F. Amato, A. Lopez, E. Maria, P. Vanhara, A. Hampl, " Artificial neural networks in medical diagnosis ", J Appl Biomed.11, 2013, DOI 10.2478/v10136-012-0031-x ISSN 1214-0287, pp.47-58, (2013).
[18] https://archive.ics.uci.edu/ml/datasets.html
[19] Z. Skolicki and K. De Jong. The influence of migration sizes and intervals on island models. In GECCO'05: Proceedings of the 2005 conference on Genetic and evolutionary computation, pages 1295-1302, New York, NY, USA, ACM, (2005).