Artificial Immune System in Doing 2-Satisfiability Based Reverse Analysis Method via a Radial Basis Function Neural Network

Authors:

Shehab Abdulhabib Alzaeemi, Saratha Sathasivam

Date Submitted: 2021-04-29

Keywords: 2-satisfiability based reverse analysis, radial basis functions neural network, Particle Swarm Optimization, artificial bee colony, Genetic Algorithm, differential evolution, artificial immune system

Abstract:

A radial basis function neural network-based 2-satisfiability reverse analysis (RBFNN-2SATRA) primarily depends on adequately obtaining the linear optimal output weights, alongside the lowest iteration error. This study aims to investigate the effectiveness, as well as the capability of the artificial immune system (AIS) algorithm in RBFNN-2SATRA. Moreover, it aims to improve the output linearity to obtain the optimal output weights. In this paper, the artificial immune system (AIS) algorithm will be introduced and implemented to enhance the effectiveness of the connection weights throughout the RBFNN-2SATRA training. To prove that the introduced method functions efficiently, five well-established datasets were solved. Moreover, the use of AIS for the RBFNN-2SATRA training is compared with the genetic algorithm (GA), differential evolution (DE), particle swarm optimization (PSO), and artificial bee colony (ABC) algorithms. In terms of measurements and accuracy, the simulation results showed that the proposed method outperformed in the terms of Mean Absolute Error (MAE), Mean Absolute Percentage Error (MAPE), Root Mean Square Error (RMSE), Schwarz Bayesian Criterion (SBC), and Central Process Unit time (CPU time). The introduced method outperformed the existing four algorithms in the aspect of robustness, accuracy, and sensitivity throughout the simulation process. Therefore, it has been proven that the proposed AIS algorithm effectively conformed to the RBFNN-2SATRA in relation to (or in terms of) the average value of training of RMSE rose up to 97.5%, SBC rose up to 99.9%, and CPU time by 99.8%. Moreover, the average value of testing in MAE was rose up to 78.5%, MAPE was rose up to 71.4%, and was capable of classifying a higher percentage (81.6%) of the test samples compared with the results for the GA, DE, PSO, and ABC algorithms.

Record Type: Published Article

Submitted To: LAPSE (Living Archive for Process Systems Engineering)

Citation (overall record, always the latest version):	LAPSE:2021.0288
Citation (this specific file, latest version):	LAPSE:2021.0288-1
Citation (this specific file, this version):	LAPSE:2021.0288-1v1

DOI of Published Version: https://doi.org/10.3390/pr8101295

processes

MDPI

Article

Artificial Immune System in Doing 2-Satisfiability Based Reverse Analysis Method via a Radial Basis Function Neural Network

Shehab Abdulhabib Alzaeemi and Saratha Sathasivam *

School of Mathematical Sciences, Universiti Sains Malaysia, 11800 USM, Penang, Malaysia; shehab_alzaeemi@yahoo.com

* Correspondence: saratha@usm.my

Received: 22 July 2020; Accepted: 22 September 2020; Published: 16 October 2020

Abstract: A radial basis function neural network-based 2-satisfiability reverse analysis (RBFNN-2SATRA) primarily depends on adequately obtaining the linear optimal output weights, alongside the lowest iteration error. This study aims to investigate the effectiveness, as well as the capability of the artificial immune system (AIS) algorithm in RBFNN-2SATRA. Moreover, it aims to improve the output linearity to obtain the optimal output weights. In this paper, the artificial immune system (AIS) algorithm will be introduced and implemented to enhance the effectiveness of the connection weights throughout the RBFNN-2SATRA training. To prove that the introduced method functions efficiently, five well-established datasets were solved. Moreover, the use of AIS for the RBFNN-2SATRA training is compared with the genetic algorithm (GA), differential evolution (DE), particle swarm optimization (PSO), and artificial bee colony (ABC) algorithms. In terms of measurements and accuracy, the simulation results showed that the proposed method outperformed in the terms of Mean Absolute Error (MAE), Mean Absolute Percentage Error (MAPE), Root Mean Square Error (RMSE), Schwarz Bayesian Criterion (SBC), and Central Process Unit time (CPU time). The introduced method outperformed the existing four algorithms in the aspect of robustness, accuracy, and sensitivity throughout the simulation process. Therefore, it has been proven that the proposed AIS algorithm effectively conformed to the RBFNN-2SATRA in relation to (or in terms of) the average value of training of RMSE rose up to 97.5%, SBC rose up to 99.9%, and CPU time by 99.8%. Moreover, the average value of testing in MAE was rose up to 78.5%, MAPE was rose up to 71.4%, and was capable of classifying a higher percentage (81.6%) of the test samples compared with the results for the GA, DE, PSO, and ABC algorithms.

Keywords: artificial immune system; differential evolution; genetic algorithm; artificial bee colony; particle swarm optimization; radial basis functions neural network; 2-satisfiability based reverse analysis

1. Introduction

Radial basis function neural network (RBFNN) has been widely used in many fields due to its simpler network structure, faster learning speeds, and better approximation capabilities [1,2]. RBFNN is a feed-forward neural network, which was first utilized by Moody and Darken [3]; they confirmed that the RBFNN has faster learning speed than the multilayer perceptron neural network (MLP). Moreover, RBFNN is simpler than MLP network, which may contain more than three layers of the structure; thus, the process of training in RBFNN is generally faster than MLP [4,5]. The difficulty of applying the traditional RBFNN lies in training the network, which should include selecting the proper input variables, the number of hidden neurons, and estimating the parameters (centers, widths) of the RBFNN [4]. The majority of the traditional RBFNN only focuses on determining the parameters

of RBFNN, while leaving the input variables and the number of hidden neurons fixed, and then the trial-and-error method may be adopted to choose the number of hidden neurons [6]. Moreover, 2 Satisfiability (2SAT) logic will help finding the input data that is used to estimate the parameters of the hidden layer and the logical rule to approximate the number of hidden neurons in RBFNN. In RBFNN training, optimal results can be guaranteed by using the training algorithm to finding the RBFNN output weights while solving the linear output of RBFNN-2SAT in a less time-consuming manner [2]. In this study, we used different algorithms to train RBFNN-2SATRA.

RBFNN is one of the most popular feed-forward neural networks, which has three layers only (i.e., output, hidden, and input). The value of the neurons moves from the input layer, and passes the hidden layer to the output layer. The basis of including three layers strives for minimizing classification and forecast errors in RBFNN [1]. The appropriate operation of RBFNN primarily relies on the adequate parameter choice of its basic functions. The simplest approach to train an RBFNN involves assuming fixed radial basis functions, which define the activation of the hidden units. Recently, researchers have begun to train RBFNN by using different methods and algorithms. Yu et al. proposed a new method for training RBFNN called an error correction algorithm. However, the introduced algorithm focused on the hidden neuron parameters. The output weight has not been considered in the feature extraction process by using the proposed algorithm [7]. Dash et al. used differential evolution (DE) to optimize RBFNN by adaptively controlling the hidden parameter in the hidden layers [8]. This method, nevertheless, lacks interpretability in the hidden layer of RBFNN [8]. Yang and Ma [9] tried to apply the Sparse Neural Network (SNN) algorithm to optimize the hidden neuron number. The SNN core mechanism is to reduce the errors through the trial and error tactic, to identify the hidden neuron number expressly from the crowd of neurons. The SNN system restriction can be observed in the aspect of high computation time in search of the best number of hidden neurons in the computational process. Inspired by various works in [10–12], 2-satisfiability (2SAT) representation logic has been used with RBFNN for identifying the relevant parameters as the center. Moreover, 2SAT has been chosen because it can comply with the RBFNN representations and structure. There were several studies that used logic programming as a symbolic rule in RBFNN. The first effort was to implement logic programming in RBFNN by [13], where they proposed a new model by embedding higher-order logic programming into RBFNN as a single network. The quest for finding the optimal model was continued by Hamadneh et al. [14]. In their paper, they embedded HornSAT logic programming in RBFNN to improve the performance of RBFNN. The result of the new method showed the HornSAT logic is able to improve the performance of RBFNN. Unfortunately, the proposed RBFNN does not integrate HornSAT to process real dataset. The final classification of the dataset via RBFNN only capitalizes standard classification of RBFNN. No attempt to extract the knowledge from the dataset via logical rule has been done.

Logic mining has been formally introduced by Sathasivam and Abdullah [15]. In this paper, the proposed logic mining managed to extract the HornSAT logical rule in student's dataset. One of the limitations of the proposed logic mining is the ability to generalize the induced logical rule that represents the dataset. The development of logic mining has been continued by the work of Kho et al. [16], where they proposed a 2-satisfiability based reverse analysis method (2SATRA). In 2SATRA, 2SAT represents explicit information of the datasets in terms of trend. The proposed 2SATRA utilizes systematic 2SAT in the Hopfield Neural Network (HNN) by extracting the optimal logical rule in electronic games. The application of 2SATRA was reported in several applications, such as palm oil price extraction [17], Amazon human resources [18], medical datasets [19], and social media analysis [20]. The proposed 2SATRA managed to achieve acceptable accuracy and generalize the behavior of the dataset. It is worth mentioning that, the induced logical rule is always converged to global minimum energy. Although Hamadneh [21] has formally introduce logic programming in RBFNN, but there is no attempt to extract the 2SAT logical rule by using RBFNN. Hence, by implementing metaheuristics, such as AIS, the proposed RBFNN is able to reduce the training error that leads to suboptimal induced logic.

Another main ingredient in integrating 2SAT into the reverse analysis method in RBFNN is the traineeship system, which exerts a considerable effect on the RBFNN performance. In this regard, an overabundance of global improvement techniques has been extensively applied to train RBFNN because of its global search aptitude. The algorithm metaheuristics are global improvement techniques—a popularly used algorithm to seek the near-optimal solution for RBFNN [13,22]. Numerous, naturally inspired, and latterly developed optimization algorithms include artificial immune systems (AIS) [1], artificial bee colony (ABC) [23], particle swarm optimization [24], differential evolution (DE) [25], genetic algorithm [26], etc. Some of these algorithms verified their appropriateness to numerous problems of engineering optimization [27]. Each algorithm seeks a resolution within a specific solution space via the movement towards the best solution in all the iterations, in most cases.

The theoretical basis for the genetic algorithm (GA) was developed in 1973 by Holland [26]. Goldberg and Holland [28] were the first researchers to implement GA in a problem that involved controlling gas pipeline transmission. Other attempts were done by Hamadneh et al. [21], who utilized GA for training the hybrid RBFNN model with higher-order SAT logic by managing a full-training paradigm. In another study, a genetic algorithm and multiple linear regression approaches were compared to predict temporal scour depth near the circular pier in non-cohesive sediment. The results showed that prediction via utilizing GA is more accurate than that of multiple linear regression [29]. In up-to-date publications, whereby GA is integrated with RBFNN to develop a reliability analysis method [30], GA has been used to optimize RBFNN to solve the constrained optimization problem. The results confirmed the robustness, accuracy, and efficiency of GA in RBFNN.

Storn and Price were the first researchers who introduced the DE algorithm as a means for solving numerous problems of global optimization. DE is a flexible algorithm; it is a powerful evolutionary algorithm with the advantages of fast convergence, fewer parameters, and superlative simplicity [25]. DE has been merged into numerous neural nets, such as feed-forward neural networks [31] and Hopfield neural net [32]. Other attempts were done by other scholars to utilize the DE algorithm for training Wavelet Neural Network toward bankruptcy prediction in banks [33]. The results showed that DE with Wavelet Neural Network outperformed in terms of sensitivity and accuracy. Recently, Tao et al. [34] have developed a prediction model by integrating the DE algorithm and RBFNN for the coking energy consumption process. The results showed that DE has improved RBFNN in terms of stability and higher accuracy.

PSO was suggested by Kennedy and Eberhart in 1995 as one of the evolutional algorithms [24]. PSO is inspired by nature and mimics the effect of bird migration behavior [35]. In another work, Qasem and Shamsuddin [36] proposed PSO to enhance RBFNN training by optimizing the hidden layer and the output layer's parameters. Another study by Alexandridis et al. [37] utilized the PSO algorithm for optimizing the structure of RBFNN. Their model proved competence in solving the function approximations and classification problems by enhancing generalization abilities and accuracy.

ABC is inspired by bee collective behaviors while gathering their honey in an optimum pattern [23]. ABC is proposed to acquire a computational advantage in optimizing the aptitude of global and local search [23]. Accordingly, ABC is utilized to train numerous nets, such as RBFNN [12], Hermite Neural Net [38], and Hopfield Neural Net [39]. ABC has been used to estimate the main parameters of RBFNN, such as the centers, width, and output weights by Kurban and Besdok [40]. Yu and Duan [41] introduced the hybrid ABC combined with Fuzzy C means Clustering into RBFNN to improve the image fusion accuracy. There are many studies that used the hybrid ABC with RBFNN as a model in many applications [42,43], including solving well-known datasets [44]. Jiang et al. [45] employed ABC to optimize parameters in RBFNN and projected the ecological pressure. In another improvement, ABC with RBFNN has been used to predict the solubility of CO2 in brine [46]. The performance analysis confirmed that ABC in RBFNN showed higher accuracy compared to other proposed models.

AIS is inspired by immune systems, which utilize the immunomodulatory properties to develop adaptive systems for accomplishing a wide range of tasks in different research areas, such as supervised classification, intrusion detection, improvement, and aggregation [47,48]. In theory, the binary AIS

has produced a plethora of works, ranging from combinatorial optimization to real-life applications. In 1996, an AIS was described as a natural immune system [49]. In 2012, AIS was developed by integrating the affinity-based interaction for AIS with the Tabu search mechanism [50]. Such a prospect has been expanded by Valarmathy and Ramani [51] when they introduced a hybrid AIS with RBFNN for improving the classification accuracy of all images of the magnetic resonance. From the perspective of the logic rule in RBFNN, there has not been an extensive study, so far, on optimizing the parameter of RBFNN by using AIS.

From the viewpoint of the satisfiability logic rule in RBFNN, very limited research has been done to utilize the metaheuristics algorithm for optimizing the parameter of RBFNN. Kasihmuddin et al. successfully introduced 2SAT as the best logical rule in an artificial neural network system with different metaheuristic algorithms [39]. The metaheuristics algorithm is a popular algorithm, which can be used for searching a semi-optimum solution to RBFNN [52]. The work of Mansor et al. [53] identified AIS as the best training model in the 3-SAT neural network system compared to another metaheuristics algorithm. This paper aims to examine the impact of AIS on the training phase of the network by constructing RBFNN, integrated with 2SAT. The adopted approach in this work has been inspired by the work of Hamadneh et al. [14,21], whereby the emphasis is on establishing an ideal logic model of RBFNN and reverse analysis (RA) by utilizing the comprehensive training process.

In this paper, the hidden neurons, and their parameters in the hidden layer, and the output weight in the output layer of RBFNN, were trained with the help of the idea of 2SAT and the metaheuristics algorithm. Once the RBFNN parameters are fixed by logic programming 2SATRA, the optimum set of output weights and the optimum output of RBFNN-2SATRA can be directly determined by utilizing the metaheuristics algorithm. To the best knowledge of the researchers, none of the existing studies proposed merging 2SATRA in RBFNN with AIS.

Therefore, this study has several contributions. First, this work aims to investigate another perspective in dealing with tacit knowledge, utilizing an explicit training model. Secondly, this study is the first attempt to integrate 2SATRA into the feed-forward neural network; 2SATRA was inserted in RBFNN as an alternative system for extracting information from real data set in the logical symbol form. Third, this work aims to create a modified RBFNN-2SATRA system with AIS to improve the training aspect of RBFNN-2SATRA, as wide-range experiments with numerous performances have been carried out. This is to measure the effect of AIS on RBFNN-2SATRA. Finally, this study aims to propose RBFNN-2SATRAIS to achieve a promising possibility of comparison with other models for all types of datasets.

To evaluate the effectiveness and efficiency of the AIS algorithm, the proposed algorithm has been applied to five popular real benchmark datasets namely: German Credit Dataset, Hepatitis Dataset, Congressional Voting Records Dataset, Car Evaluation Dataset, and Postoperative Patient Dataset, chosen from the University of California, Irvine (UCI) machine learning repository [54]. The outcomes of the AIS algorithm were then compared with GA, DE, PSO, and ABC.

2. 2 Satisfiability Logic Representation

The representation is defined as a logic rule of determining the satisfiability of clause sets, which consist of two literals in each clause [55]. The properties of 2SAT can be summarized as follows:

i. A set of m logical variables, $x_1, x_2, \ldots, x_m$. Each variable stores a binary value of $x_i \in \{1, 0\}$ that exemplify TRUE and FALSE, respectively.
ii. Each variable in x_i can be set of literals, where positive literal and negative literal is defined as x_m and $\neg x_m$, respectively.
iii. Consisting of a set of n distinct clauses, $C_1, C_2, \ldots, C_n$. Each C_i is connected by logical AND ($\wedge$). Every k literals will form a single C_i and connected by logical OR ($\vee$).

By using property (i) until (iii), the explicit definition of the 2SAT formulation can be defined or P_{2SAT}:

$$P_{2SAT} = \bigwedge_{i=1}^{n} l_i, \text{ where } l_i = \bigvee_{i=1}^{m} C_i \bigvee_{j=1}^{m}, \; n \in \mathbb{Z}, \; m = 2 \tag{1}$$

The example of P_{2SAT} is:

$$P_{2SAT} = (C \vee D) \wedge (E \vee \neg F) \wedge (K \vee \neg L) \tag{2}$$

The formulation for P_{2SAT} formulation must be represented in Conjunctive Normal Form (2CNF) because the Satisfiability nature of CNF can be conserved compared to other forms, such as Disjunctive Normal Form (DNF). In this paper, the information of the datasets is represented in the form of attributes. The attributes are defined as variables in P_{2SAT} and become the symbolic rule for Artificial Neural Network (ANN). In this study, 2SAT is considered as the main driver because the focus of logical programming involves ensuring that the program considers only two letters per item per implementation. It has been proven in previous studies that many of the combinatorial problems can be formulated using the 2SAT logic [56–58]. A key reason that makes the 2SAT logic an appropriate approach of representing logical rules in the neural network involves choosing two literals per clause in satisfiability logic, which can reduce the logic complexity of disclosing the relation between the variables in the neural network.

3. Radial Basis Function Neural Network (RBFNN)

RBFNN is a feed-forward neural network, which was first utilized by Moody and Darken [3]. Compared to other networks, RBFNN has more integrated structure and faster learning speed. In terms of composition, RBFNN involves three layers as the input layer, the hidden layer, and the output layer [59]. According to [60], the Gaussian activation function was chosen due to the differentiability of the function, and capability of establishing the non-linear relationship between the input neuron in the input layer, and the output neuron in the output layer. The following equation shows the Gaussian activation function $\varphi_i(x)$ [60]:

$$\varphi_i(x) = e^{\frac{\| \sum_{j=1}^{N} w'_{ji} x_j - c_i \|^2}{2\sigma_i^2}} \tag{3}$$

We set $w'_{ji} = 1$ because other values of w'_{ji} will result in the biased selection, which leads to weighted 2SAT [4,13,61]. c_i is the center and σ_i is the width as shown in the following equations:

$$c_i = \frac{1}{m} \sum_{i=1}^{N} x_i \tag{4}$$

$$\sigma_i^2 = \frac{1}{m} \sum_{i=1}^{N} \|x_i - c_i\|^2 \tag{5}$$

where m is the number of neuron per clause as shown in the logic programming P_{2SAT} in Equation (1) [10,13], x_i is a binary input value for N input neurons, and Euclidean norm ||space|| as follows:

$$\| \sum_{i=1}^{N} x_i - c_i \| = \sqrt{\sum_{i=1}^{N} (x_i - c_i)^2} \tag{6}$$

The final output of RBFNN $f(w_i)$ is given as follows [62]:

$$f(w_i) = \sum_{i=1}^{j} w_i \varphi_i(x) \tag{7}$$

where $f(w_i) = (f(w_1), f(w_2), f(w_3), \ldots, f(w_k))$ are the RBFNN output value and $w_i = (w_1, w_2, w_3, \ldots, w_N)$ is the output weight. Figure 1 illustrates the structure of Satisfiability RBFNN.

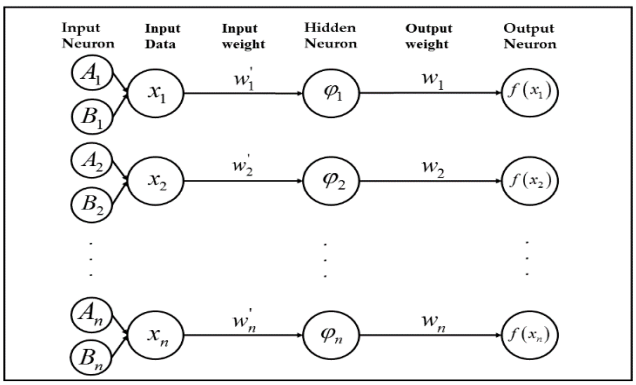

Figure 1. Structure of radial basis function neural network (RBFNN).

Figure 1 shows the structure of RBFNN in dealing with satisfiability logic programming. The RBFNN process works as follows, firstly, the input neuron gets the input data to enter the network through the input layer. After that, each neuron in the hidden layer calculates the center and the width between the input data and the prototype stored inside it by using the Gaussian activation function that helps to obtain the optimal output weight for the output layer. In this study, we have established a new approach to determine the best RBFNN structure for 2-satisfiability reverse analysis (2 SATRA).

4. 2-Satisfiability Based Reverse Analysis Method (2SATRA) in RBFNN

In this study, 2SAT enhanced the RA method (abbreviated as 2SATRA) [16] and is proposed to extract the optimum 2SAT logic rule to explain the behavior of the real datasets. In this regard, 2SATRA is a logic mining tool that utilizes RBFNN-2SAT models for extracting the useful logic rule from the dataset. The 2SAT logical rule is utilized to represent and map the datasets due to flexibility and simplicity. Thus, the attributes in the datasets are transformed into a binary form {0, 1}. Specifically, the 2SATRA method extracts the optimal logical rule, which represents the relationship between the attributes of a specific real data set. Accordingly, the hidden information in the data set is extracted to be utilized in classification or prediction. In this study, 2SATRA has been carried out in the RBFNN to describe an intelligence system in doing data mining, and each attribute has been transformed into the atoms inside the clauses. Therefore, six attributes from the datasets were selected to form the 2SAT logical rule. The implementation of the 2SATRA method in the RBFNN networks is demonstrated in the following algorithm:

Step 1: convert all raw dataset to binary, split into a training dataset, and test dataset with the outcome P_{learn} (60%) and P_{test} (40%) [16,18].

Step 2: initialize the input data, width, and center of the neurons, and designation of all the neurons with binary data from Step 1.

Step 3: segregate the collection of two neurons per clause $L_1, L_2, \ldots, L_n$ that leads to $P_{learn} = 1$.

Step 4: obtain P_{best} by comparing the frequency of the 2SAT clauses in the overall learning dataset.

Step 5: check the output weight of the clauses in the hidden layer of P_{best} by using GA, DE, PSO, ABC, and AIS.

Step 6: save the best output weight W_i of P_{best}.

Step 7: find the final state of neurons by computing the corresponding output of RBFNN-2SAT according to [63] as shown below:

$$\operatorname{sgn}(f(w_i)) = \begin{cases} 1, & f(w_i) \geq 0 \\ 0, & Otherwise \end{cases} \tag{8}$$

where w_i is the output weights and $f(w_i)$ is the RBFNN output value.

$$f(w_i) = \sum_{i=1}^{j} w_i \varphi_i(x) \tag{9}$$

where φ_i is the activation function of input x_i in the hidden layer and W_i is the weight between the input data in the hidden layer and the output data in the output layer.

Step 8: induce all possible 2SAT logic $P_1^B, P_2^B, \ldots, P_n^B$ from the neuron states.

Step 9: examine all of the induced logic P_i^B by comparing the outcome of P_i^B with P_{test}.

Step 10: obtain all of the performance evolution and calculation of accuracy.

It should be noted that 2SATRA is a method that utilizes the beneficial feature of RBFNN and 2-satisfiability logic, or RBFNN-2SATRA. Furthermore, 2SATRA is regarded as a feasible approach to help extract the best logical rule, which governs the behavior of the data set [16].

The complete flowchart of Figure 2 shows the methodology of this work in steps to train RBFNN-2SATRA.

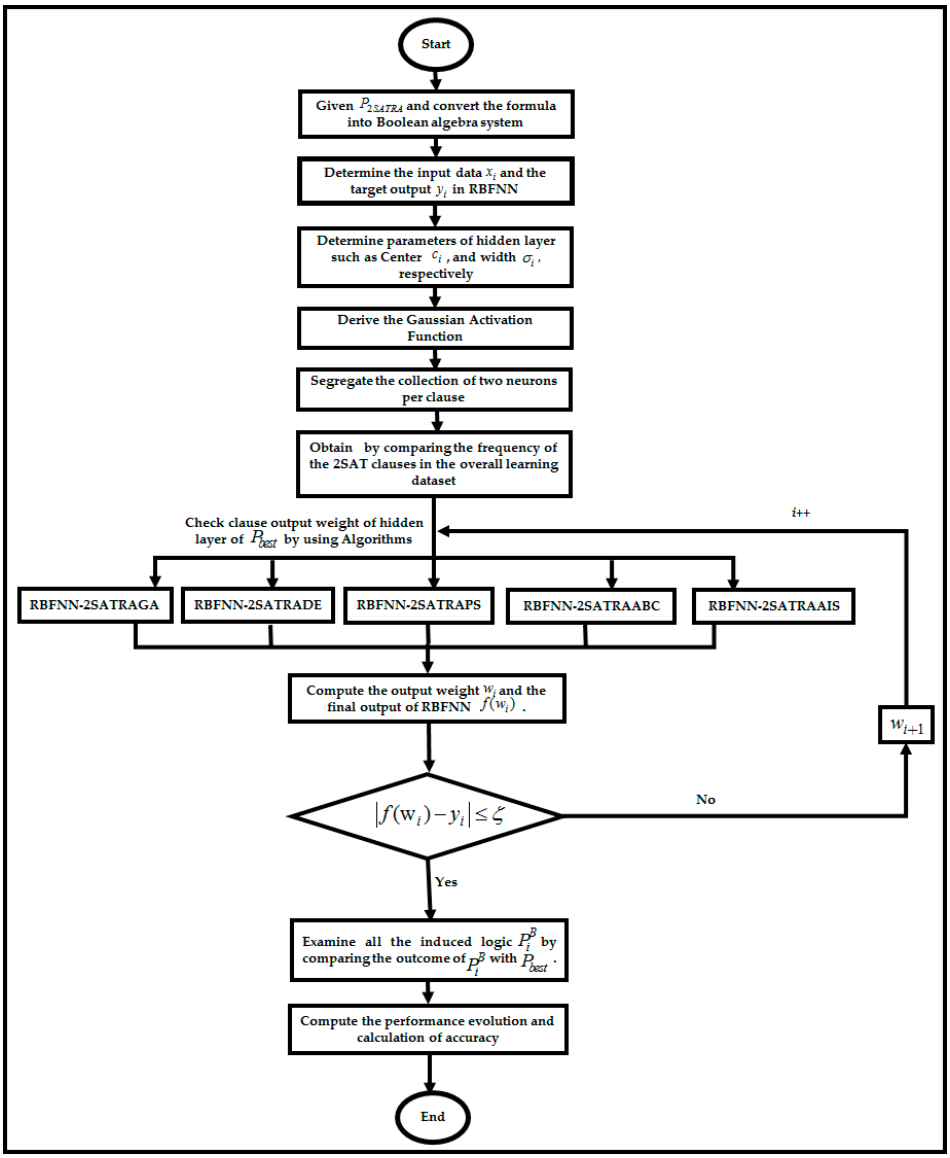

Figure 2. Flowchart of the methodology.

4.1. Genetic Algorithm in RBFNN-2SATRA

GA was developed in the 1970s as a popular metaheuristic algorithm. Since then, it has been widely implemented to solve numerous optimization problems. The structure of GA can be separated into local searches and global searches [64] using crossover, selection, and mutation for adaptive and optimization, artificial systems, and other problem-solving strategies [65]. The implementation of GA in RBFNN-2SATRA is defined as RBFNN-2SATRAGA. The steps involved in RBFNN-2SATRAGA are shown in Figure 3 as follows:

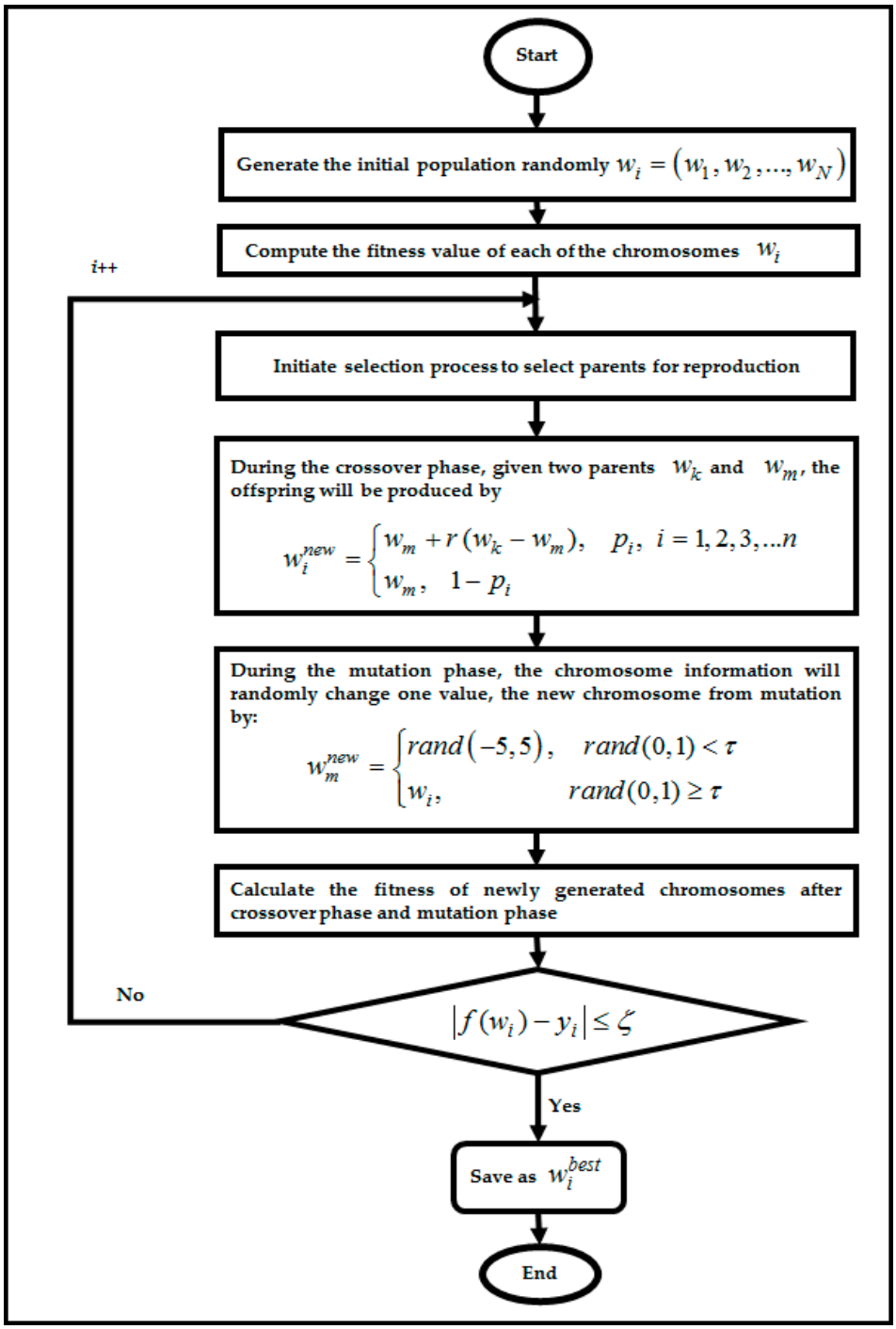

Figure 3. The implementation of genetic algorithm (GA) in radial basis function neural network-based 2-satisfiability reverse analysis (RBFNN-2SATRA).

4.2. Differential Evolution Algorithm in RBFNN-2SATRA

DE is a new evolutionary population-based algorithm that has been typically utilized in numerical optimization [66]. In DE, each individual (solution) of the population competes with its parents, and the fittest wins [67]. The implementation of DE in RBFNN-2SATRA is defined as RBFNN-2SATRADE. The algorithm steps in RBFNN-2SATRADE are shown in Figure 4 as follows:

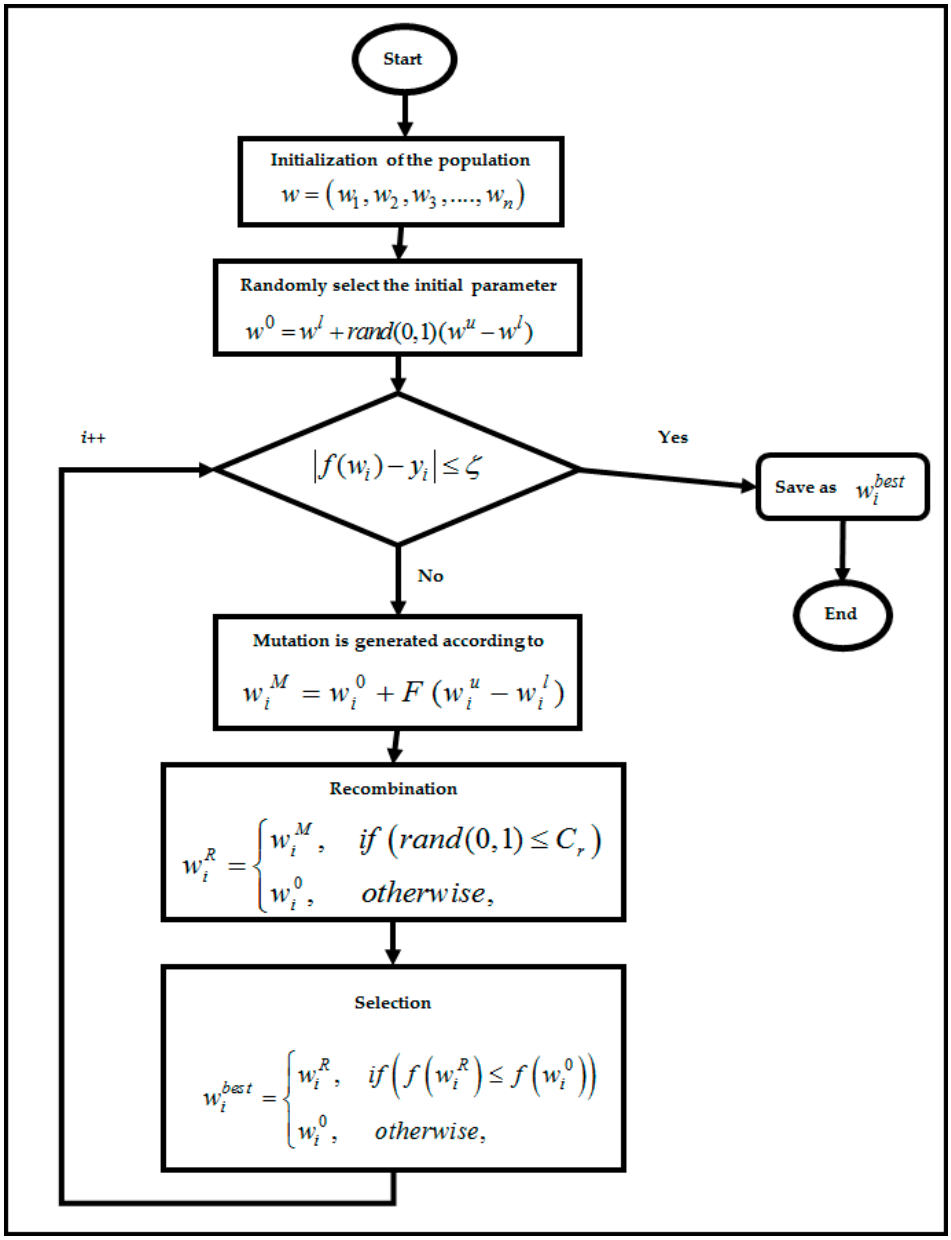

Figure 4. Flowchart of RBFNN-2SATRADE. Note: the implementation of differential evolution (DE) in RBFNN-2SATRA is defined as RBFNN-2SATRADE.

4.3. Particle Swarm Optimization Algorithm in RBFNN-2SATRA

The PSO algorithm is a popular swarm computation algorithm. It is utilized for solving global optimization in continuous search space. It has been successfully applied to solve different types of real-world optimization problems due to its simplicity in implementation, alongside its remarkable features, such as the presence of flexible free parameters [68]. The main steps of the procedure in the RBFNN-2SATPSO model are shown in Figure 5 as follows:

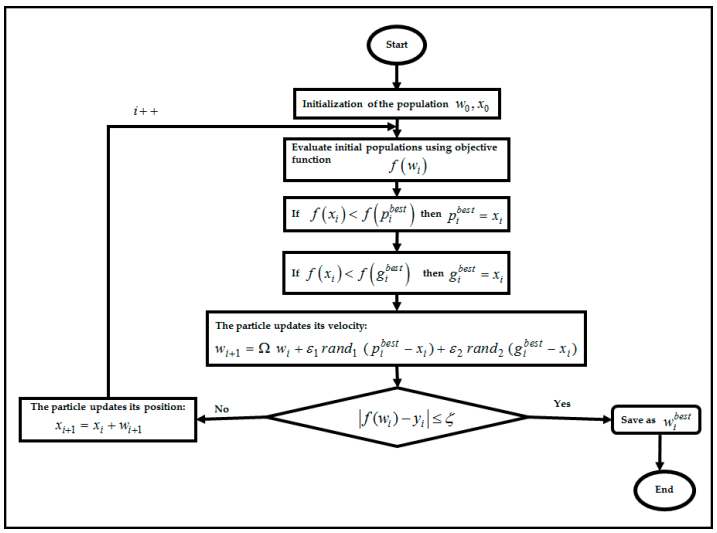

Figure 5. Flowchart of RBFNN-2SATRAPSO. PSO = particle swarm optimization.

4.4. Artificial Bee Colony Algorithm in RBFNN-2SATRA

The ABC algorithm is inspired by the social behavior of the natural bees. It is utilized to solve numerous optimization problems [69]. ABC society consists of three swarms called employed bees, scout bees, and onlooker bees that help improve the solution. The algorithm involved in RBFNN-2SATRAABC is shown in Figure 6 as follows:

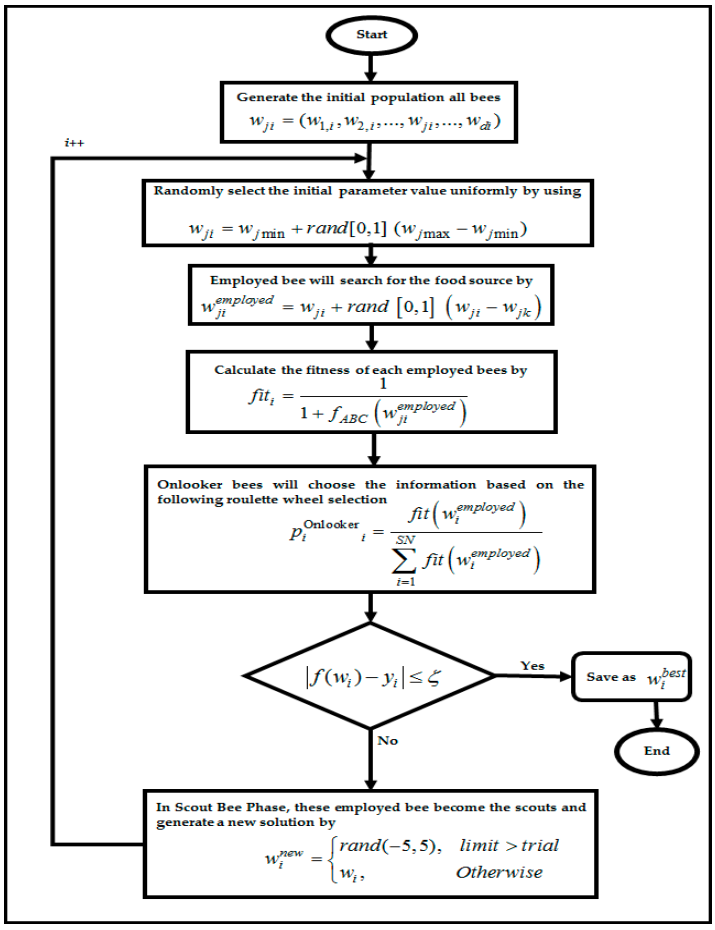

Figure 6. The artificial bee colony (ABC) algorithm implemented in RBFNN-2SATRA.

4.5. Artificial Immune System Algorithm in RBFNN-2SATRA

In recent years, non-traditional, nature-inspired optimization techniques have been growing in popularity in the combinatorial optimization field. The AIS algorithm is one of these techniques, which is enthused by the human body's immune system. The AIS algorithm is known as an adaptive system stimulated by the theoretical immunology and observed immune functions, which are applied to complex problem fields [70]. The AIS algorithm application exists in fields, such as computer network security, biological modeling, virus detection, robotics, data mining, scheduling, classification, and clustering [53,70]. The AIS implementation is defined in RBFNN-2SATRA as RBFNN-2SATRAAIS. The algorithm, which is involved in RBFNN-2SATRAAIS, is shown in Figure 7 as follows:

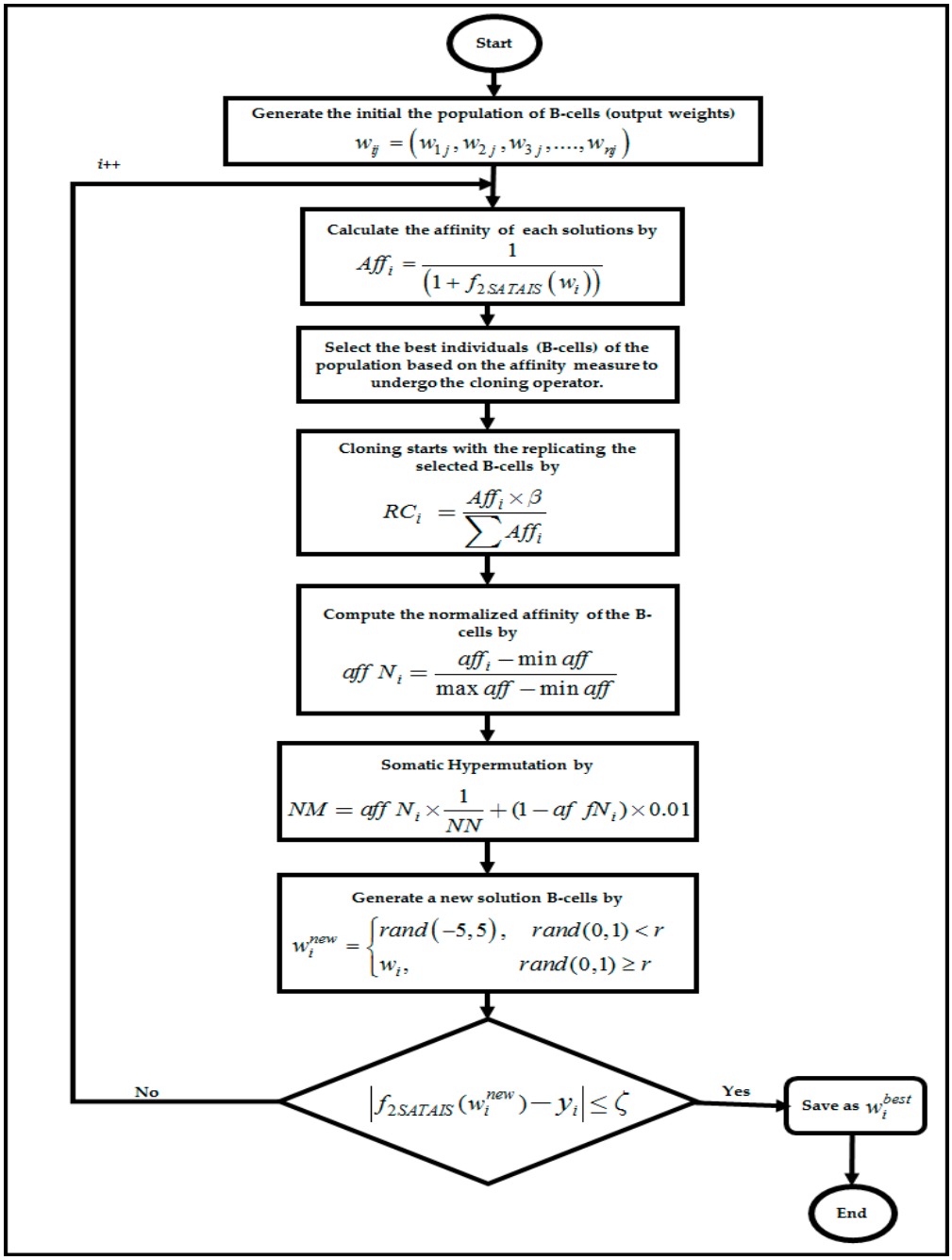

Figure 7. The implementation of the artificial immune system (AIS) algorithm in RBFNN-2SATRA.

5. Experimental Setup

The experimental simulation is developed to assess the capacity of the metaheuristic algorithms to train RBFNN in doing 2SATRA. In every dataset, 60% of the data points in the datasets will be utilized for training, and 40% will be used for testing. The k Mean clustering [18] will be used to convert the dataset into binary representation. As for missing data in the dataset, the neuron state will be defined randomly. All of the 2SATRA models were implemented in Microsoft Visual C++ software with Microsoft Windows 7, in 64-bit, with 3.40 GHz processor, 4096 MB RAM, and 500 GB hard drive specification. The use of C++ is to help the user have full control over the memory management. Note that all simulations will be conducted in the same device to avoid any possible biases. The total CPU time for both training and testing is 24 h [71]. If the model exceeds the recommended CPU time threshold, it means the structure of the recommended algorithm does not have the capability to train RBFNN based 2SATRA by using real-life datasets. In terms of choice of activation function, we utilized the Gaussian activation function due to association properties for each radial unit as the center and width. Other activation functions, such as Hyperbolic Activation Function [72], Bipolar Activation Function [73], and McCulloch–Pitts Activation Function [73] are not compatible with the proposed RBFNN due to the non-compatible classification interval. The use of the activation function will result in overfitting nature of the RBFNN. The classification outcome will utilize the same tolerance value, which is $Tol = 0.001$, proposed by Sathasivam [74]. The choice of the Tol value is to ensure the reduction of the possible statistical error between the target output and the output. In the aspect of the 2SAT logical rule, we only utilize the satisfiable logical rule, where the $Min\left|f(w_i) - y_i\right|$ is always zero. The use of other non-satisfiable logic, such as maximum satisfiability [75], is only compatible for $P_{learn} = 0$. The lists of parameters used in each RBFNN-2SATRA model are summarized in Table 1.

Table 1. Optimal parameters in RBFNN-2SATRA models.

AIS		**ABC**		**PSO**	
Parameter	**Value**	**Parameter**	**Value**	**Parameter**	**Value**
Number of iteration	10,000	No_Employed_bees	50	Ω	0.6
β	200	No_Onlooker_bees	50	ε_1	2
Population size	100	No_Scout_bees	1	ε_2	2
r	[0, 1]	Limit	1000	$rand_1 = rand_2$	[0,1]
		Trial	10,000	Number of iteration	10,000
DE		**GA**			
Parameter	**Value**	**Parameter**		**Value**	
Number of iteration	10,000	Number of iteration		10,000	
C_r	[0,1]	Selection type		Wheel selection	
F	[0,2]	Number of individuals		50	
Population	50	Mutation ratio		1	
Number of iteration	10,000	Mutation type		Uniform	
		Crossover ratio		1	

6. Datasets Description

The evaluation of the proposed AIS algorithm has been performed by utilizing five well-known diverse real datasets chosen from the UCI Repository [54,76,77]. It is widely used as the benchmark dataset by neural network practitioners. Table 2 illustrates these datasets in terms of many features, training samples, and test samples per data set.

Table 2. List of Benchmark Datasets Information from UCI Repository.

Benchmark Datasets	Field	Attributes	Instances	Training Samples	Testing Samples
German Credit Dataset (GCR)	Finance	Duration of Credit (month) Payment Status of Previous Credit Amount Value Savings/Stocks Length of current employment Installment percent Creditability Sex	1000	600	400
Hepatitis Dataset (HR)	Medical	Steroid Antiviral Fatigue Malaise Anorexia Die or live	155	93	62
Congressional Voting Records Dataset (CVR)	Social Science	Handicapped infant Water-project-cost-sharing El-Salvador-aid Religious-groups-in-schools Aid-to-Nicaraguan-contras Immigration Rep/demo (P)	435	261	174
Car Evaluation Dataset (CE)	Business	Buying Price Maintenance Doors Number Person Number Size Boot Safety Values	1728	1037	691
Postoperative Patient Dataset (PP)	Medical	L-CORE (patient's internal temperature in C) L-SURF (patient's surface temperature in C) L-O2(oxygen saturation in %) L-BP (last measurement of blood pressure) CORE-STBL (stability of patient's core temperature) BP-STBL (stability of patient's blood pressure) Decision ADM-DECS(discharge decision)	90	54	36

7. Results and Discussion

Based on the experiments, the performance of the training algorithms has been assessed based on a different number of neurons $6 \leq NN \leq 120$. Five various measurements have been used to assess the RBFNN-2SATRA models with metaheuristic algorithms, including Accuracy and Schwarz Bayesian Criterion (SBC) to assess the prediction accuracy, while Mean Absolute Error (MAE), Mean Absolute Percentage Error (MAPE), Root Mean Square Error (RMSE), and Central Process Unit time (CPU time) showed the structure complexity of RBFNN-2SATRA network based on the rising neuron numbers as shown in the following equation:

$$RMSE = \sum_{i=1}^{n} \sqrt{\frac{1}{n}(f(w_i) - y_i)^2} \tag{10}$$

RMSE [10] is a standard error estimator, which has been commonly used in predictions and classifications. During the learning phase, RMSE measured the deviation of the error between the current value $f(w_i)$ and y_i vis-à-vis mean $\bar{f}$. Lower RMSE refers to the better accuracy of our model.

$$\text{MAE} = \sum_{i=1}^{n} \frac{1}{n} \left| f(w_i) - y_i \right| \tag{11}$$

MAE is one of the loss function type of error, which evaluates the straightforward difference between the expected value and the current value. During the learning phase, MAE measured the

absolute difference between the current value $f(w_i)$ and y_i [70]. In addition, the smaller value of MAE refers to the best fitness of the method.

$$MAPE = \frac{100\%}{n}\sum_{i=1}^{n}\left|\frac{f(w_i) - y_i}{f(w_i)}\right| \tag{12}$$

MAPE [10] measured the size of the error in the form of percentage terms. During the learning phase, MAPE measured the percentage difference between the current value $f(w_i)$ and y_i. Then, the lower MAPE leads to better accuracy in terms of percentage for the model.

$$SBC = n\ \ln(\frac{\sum_{i=1}^{n}(f(w_i) - y_i)^2}{n}) + pa\ \ln(n) \tag{13}$$

where pa is the number of centers, the widths, and the output weights. For SBC values, the lower values are better. When the value of the errors is small, indicate the accuracy is better. The accuracy is defined as follows:

$$Accuracy = \frac{Number\ of\ the\ correct\ induced\ logic}{Total\ number\ of\ testing\ data} \times 100\% \tag{14}$$

The accuracy will determine the ability of the system for training the dataset. Meanwhile, when the CPU time is lower, the efficiency of the algorithm will be increased.

The results of the RBFNN-2SATRA with GA, DE, PSO, ABC, and AIS are summarized in Table 3 and Figures 8–27. Based on the experimental results, the following findings are concluded: (1) the proposed model RBFNN with 2SATRA can receive more input data and can deal with the hidden neuron with a fixed value of the parameters of the hidden layer as width and center. In this situation, the RBFNN-2SATRA with AIS established the best model, which classified datasets based on the logic rule 2SATRA with a minimal value of errors (Mean Absolute Error (MAE), Mean Absolute Percentage Error (MAPE), Root Mean Square Error (RMSE), Schwarz Bayesian Criterion (SBC), and Central Processing Unit (CPU) time. (2) The model RBFNN-2SATRAAIS showed the best performance in terms of RMSE, SBC, and CPU time, although the number of neurons increased. The important features of AIS, such as variation, recognition, memory, learning, and self-organizing influenced performance capability. (3) RBFNN-2SATRAAIS showed the best performance in terms of SBC, although the number of neurons increased. According to Hamadneh et al. [21], the lowest value of SBC indicates that the model can be classified as the best model. (4) In terms of the CPU time, the model RBFNN-2SATRAAIS has been reported as a faster performance than other RBFNN-2SATRA models. When the number of neurons exceeded 40, the possibility for GA, DE, and PSO trapped in trial and error state increased. Trial and error caused GA, DE, and PSO to reach pre-mature convergence. On the other hand, RBFNN-2SATRA with ABC had a relative training error because, during the employed bee phase, the time of the algorithm was wasted without achieving significant improvement. The scout bee phase allowed the algorithm from being trapped at the local minima after a certain count "limit" of unsuccessful improving attempts. Several iterations were required for ABC to produce solutions (output weight) with high quality. These experiments have shown that the AIS algorithm can be successfully applied to train RBFNN-2SATRA due to new generations being formed through cloning. In AIS, the number of the search agents has not been constant and increased due to cloning operations. Even the clone itself moved to the neighboring nodes, which led to fewer iterations required for RBFNN-2SATRAAIS to produce a solution (output weight) with high quality.

Table 3. Testing Error for Real Datasets.

Data Set	Metric	Algorithms				
		GA	**DE**	**PSO**	**ABC**	**AIS**
German Credit Dataset	MAE	0.2575	0.215	0.2125	0.1625	0.12
	MAPE%	0.064375	0.05375	0.053125	0.040625	0.03
Hepatitis Dataset	MAE	0.290323	0.193548	0.177419	0.145161	0.064516
	MAPE%	0.468262	0.312175	0.28616	0.234131	0.104058
Congressional Voting Records Dataset	MAE	0.298851	0.275862	0.224138	0.218391	0.183908
	MAPE%	0.171753	0.158541	0.128815	0.125512	0.104694
Car Evaluation Dataset	MAE	0.204052	0.193922	0.188133	0.086831	0.081042
	MAPE%	0.02953	0.028064	0.027226	0.012566	0.011728
Postoperative Patient Dataset	MAE	0.388889	0.361111	0.25	0.138889	0.111111
	MAPE%	1.080246	1.003086	0.694444	0.385802	0.308642

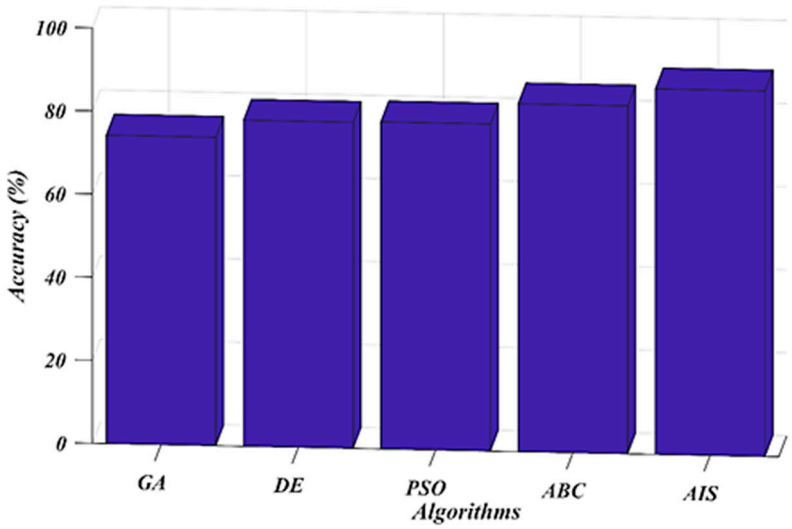

Figure 8. Accuracy Evaluation for German Credit.

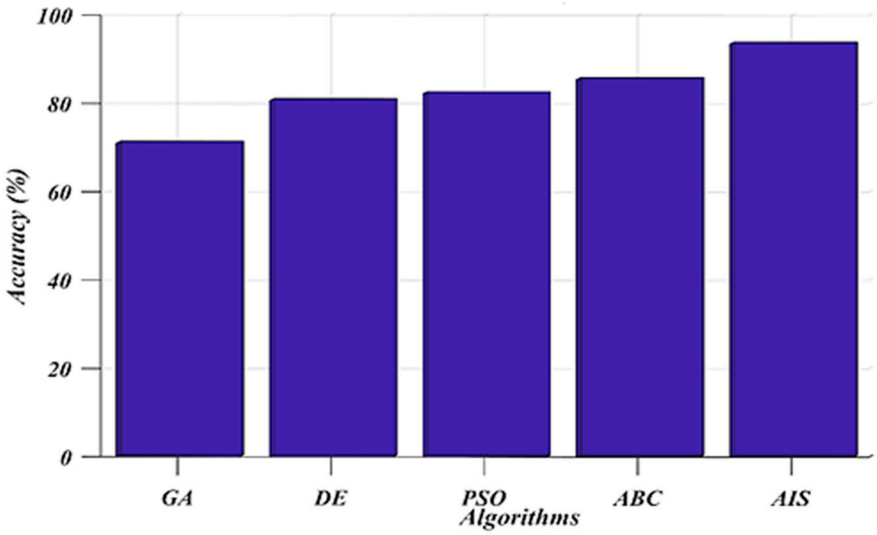

Figure 9. Accuracy Evaluation for Hepatitis.

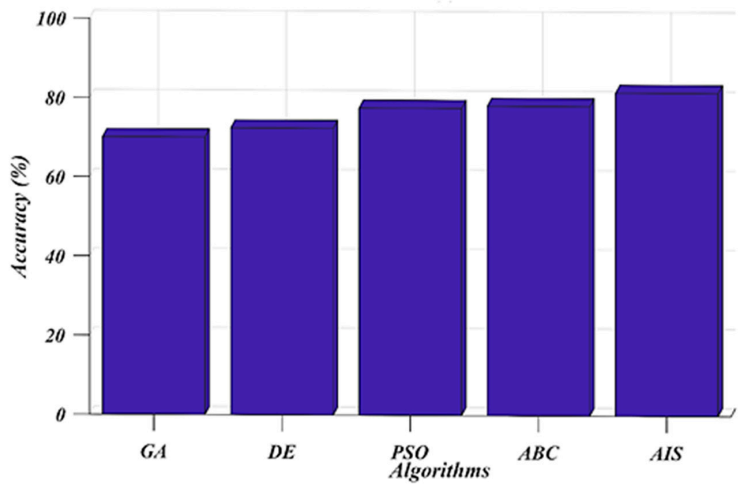

Figure 10. Accuracy Evaluation for Congressional Voting Records.

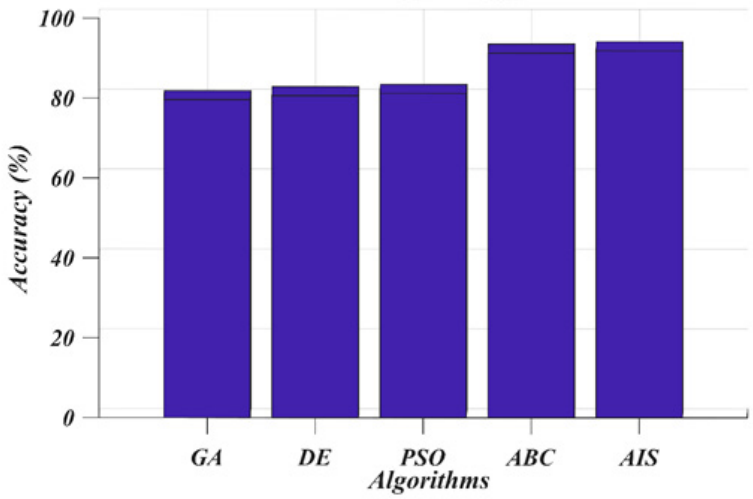

Figure 11. Accuracy Evaluation for Car Evaluation.

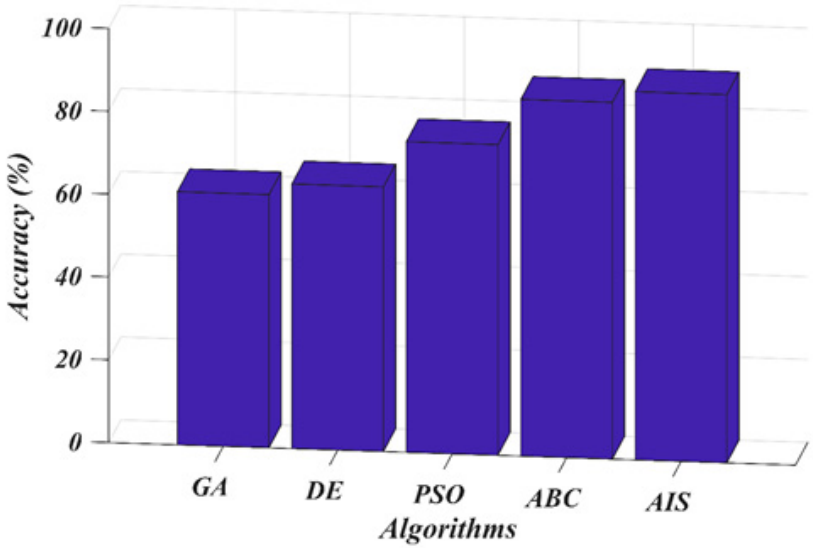

Figure 12. Accuracy Evaluation for Postoperative Patient.

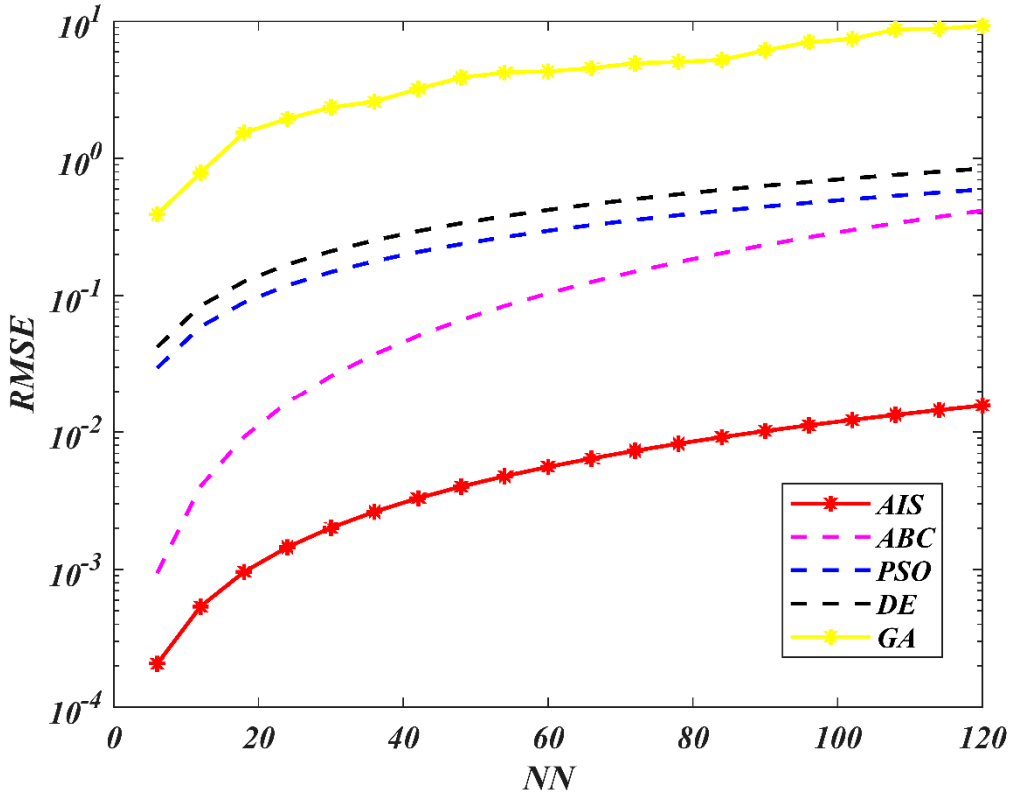

Figure 13. Root Mean Square Error (RMSE) Evaluation for German Credit.

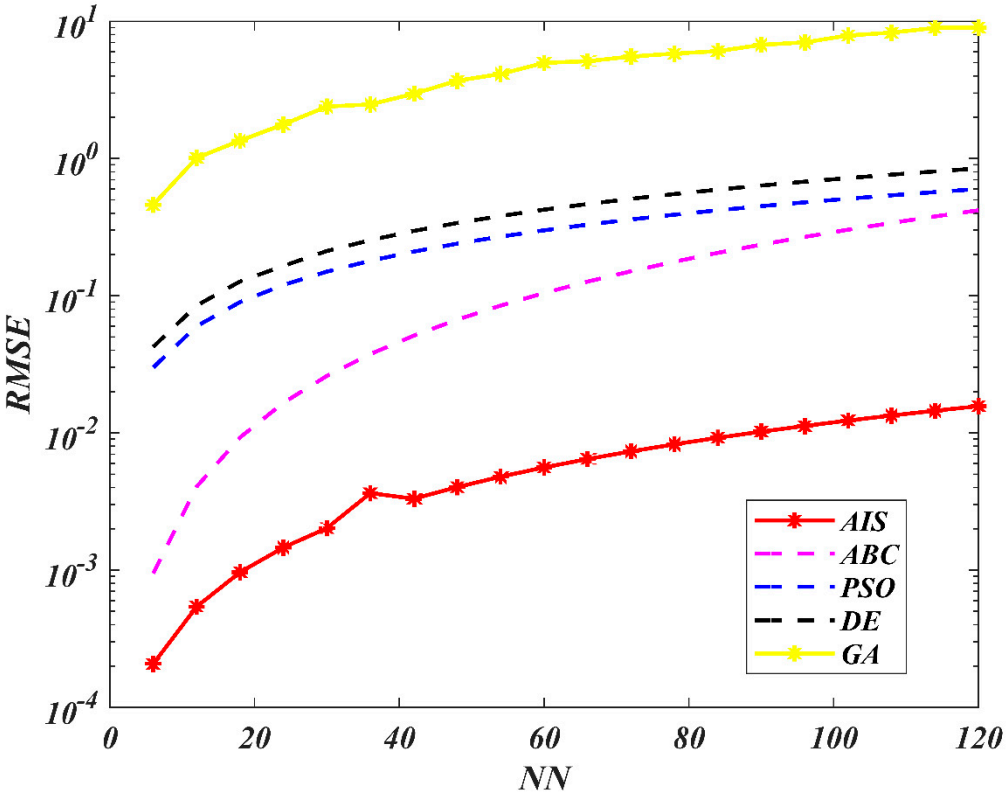

Figure 14. RMSE Evaluation of Hepatitis.

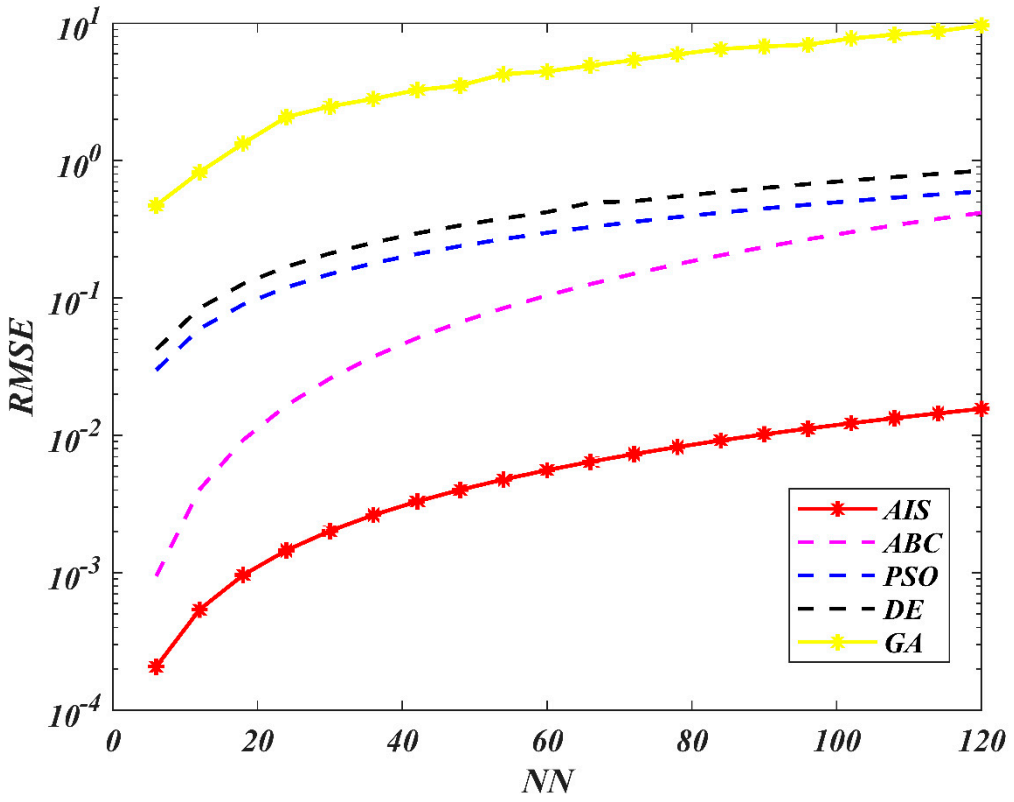

Figure 15. RMSE Evaluation for Congressional Voting Records.

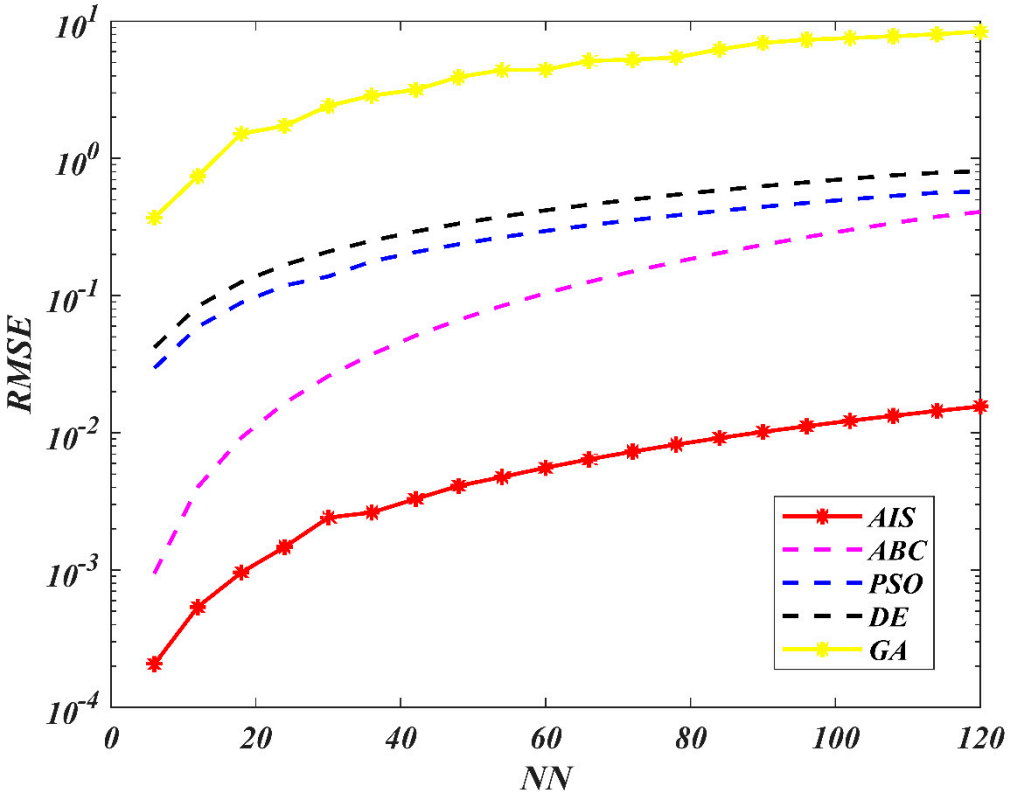

Figure 16. RMSE Evaluation for Car Evaluation.

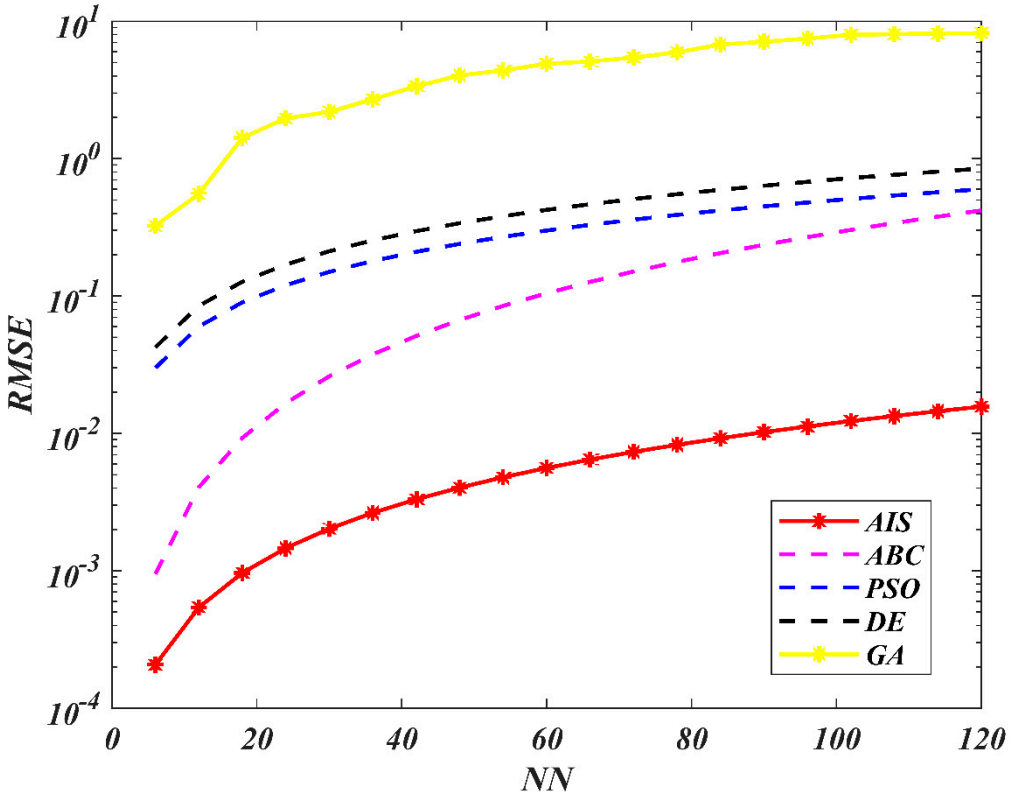

Figure 17. RMSE Evaluation for Postoperative Patient.

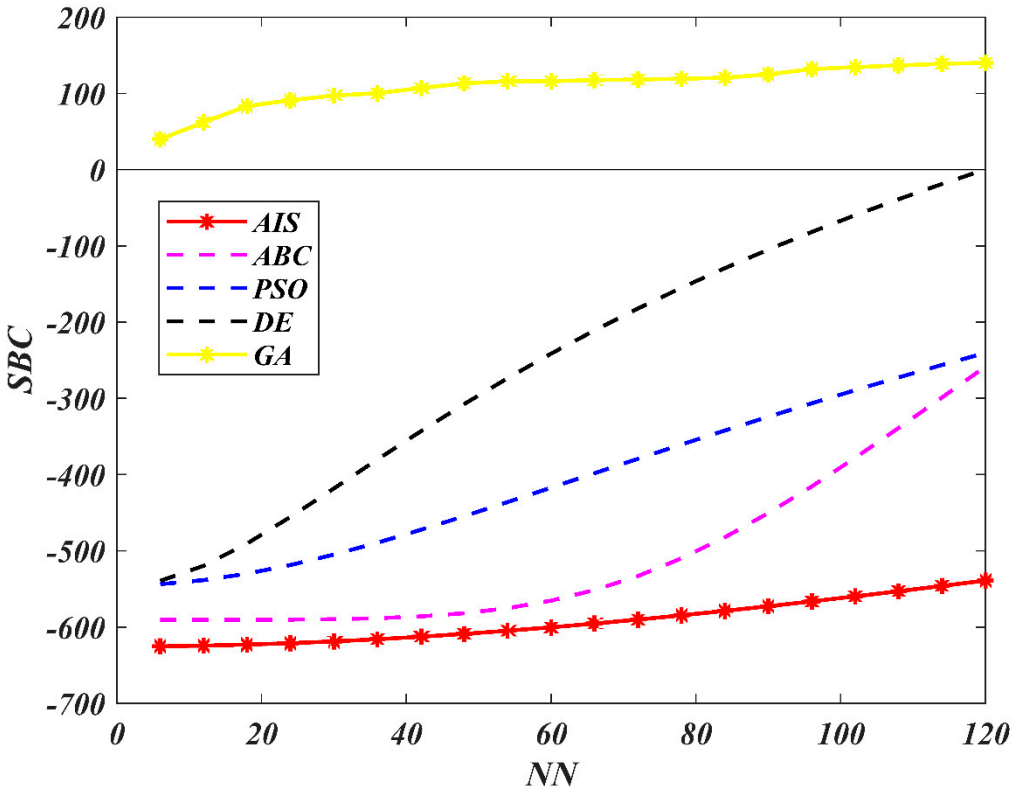

Figure 18. Schwarz Bayesian Criterion (SBC) Evaluation for German Credit.

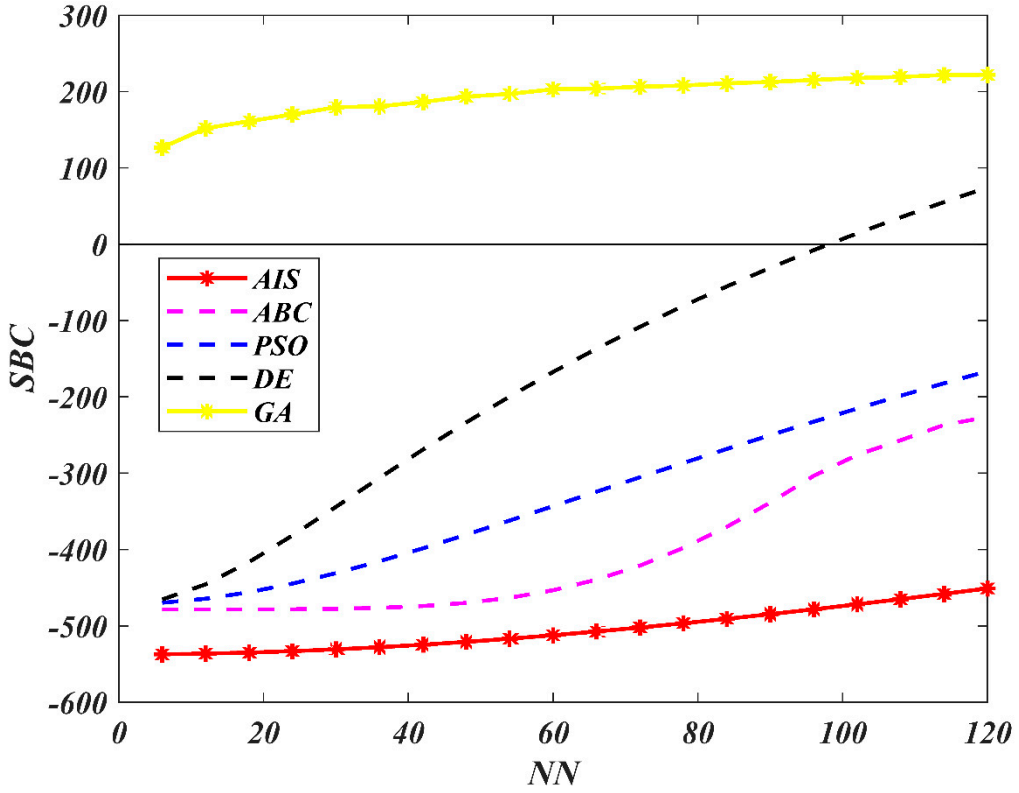

Figure 19. SBC Evaluation for Hepatitis.

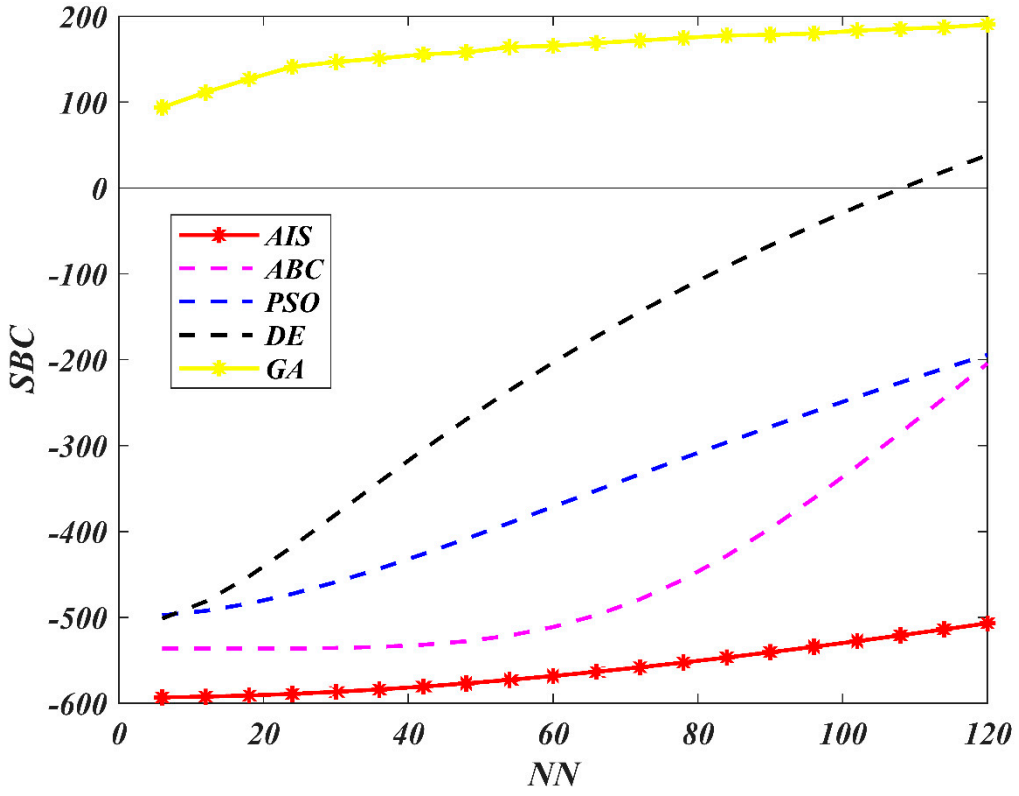

Figure 20. SBC Evaluation for Congressional Voting Records.

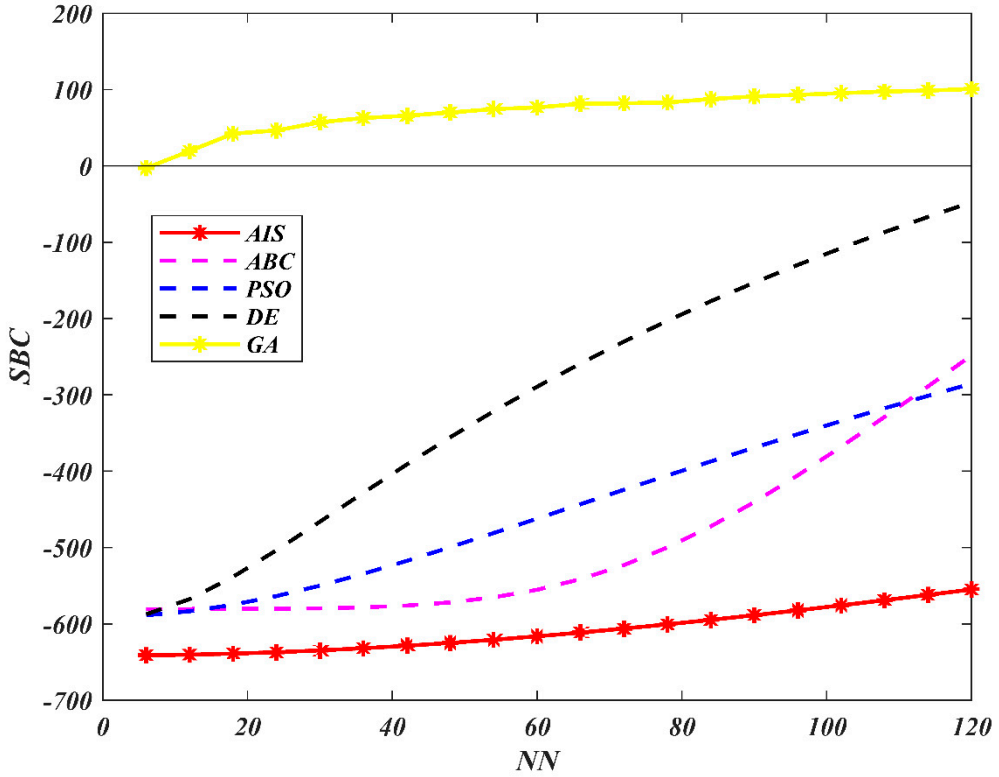

Figure 21. SBC Evaluation for Car Evaluation.

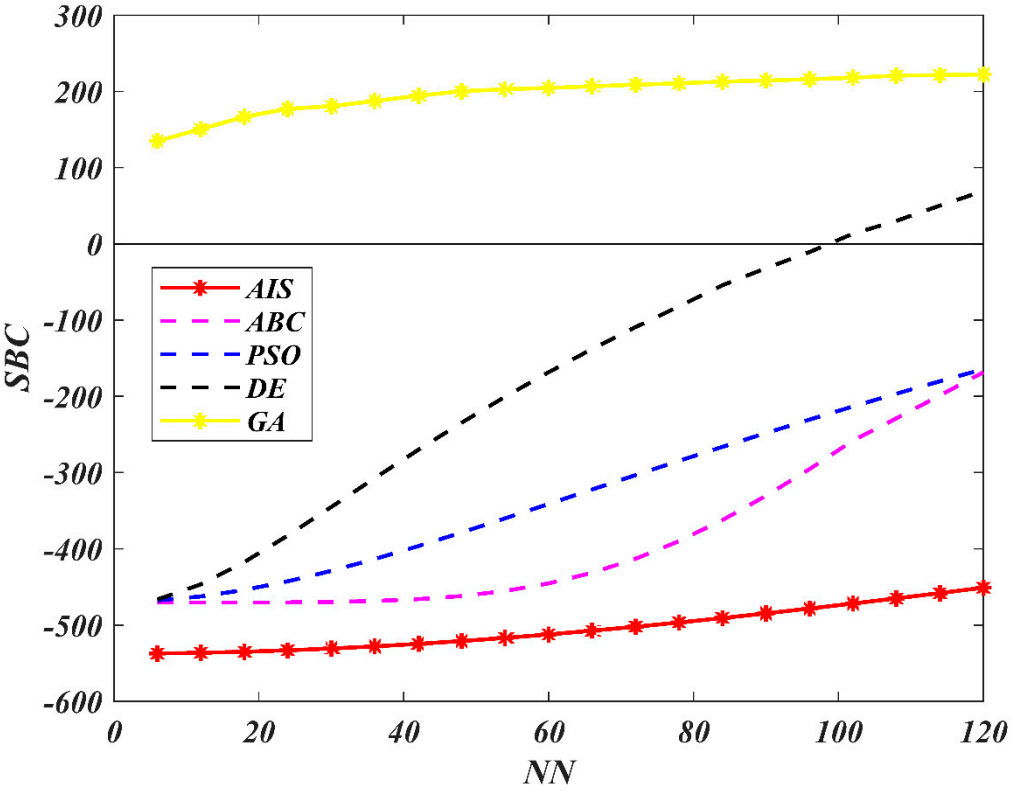

Figure 22. SBC Evaluation for Postoperative Patient.

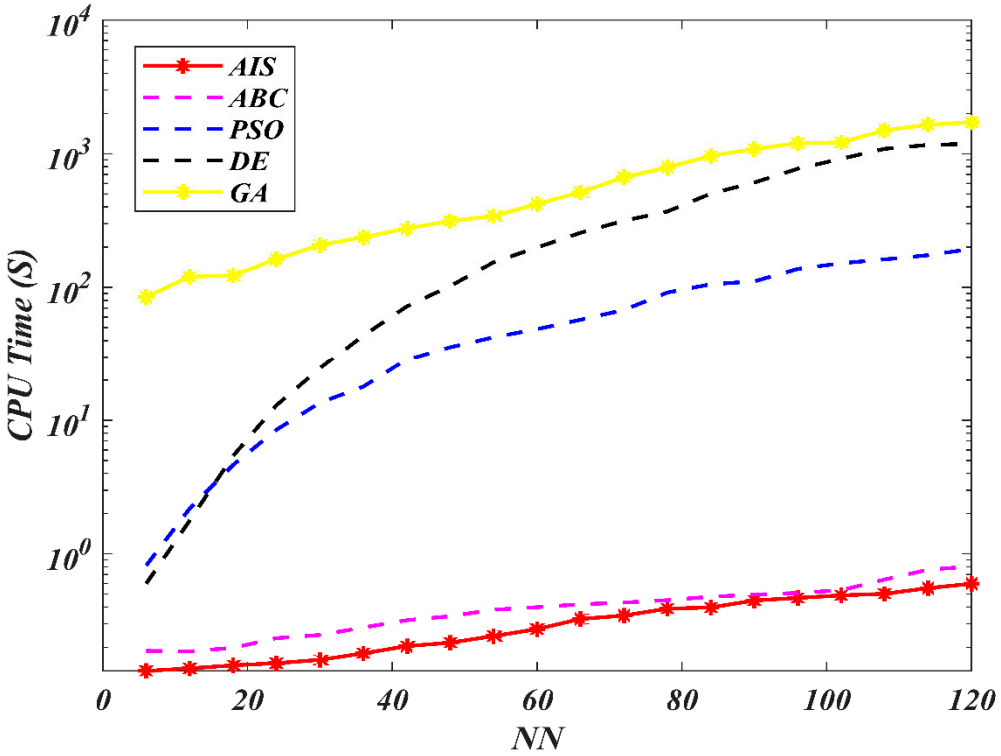

Figure 23. Central Process Unit (CPU) time Evaluation for German Credit.

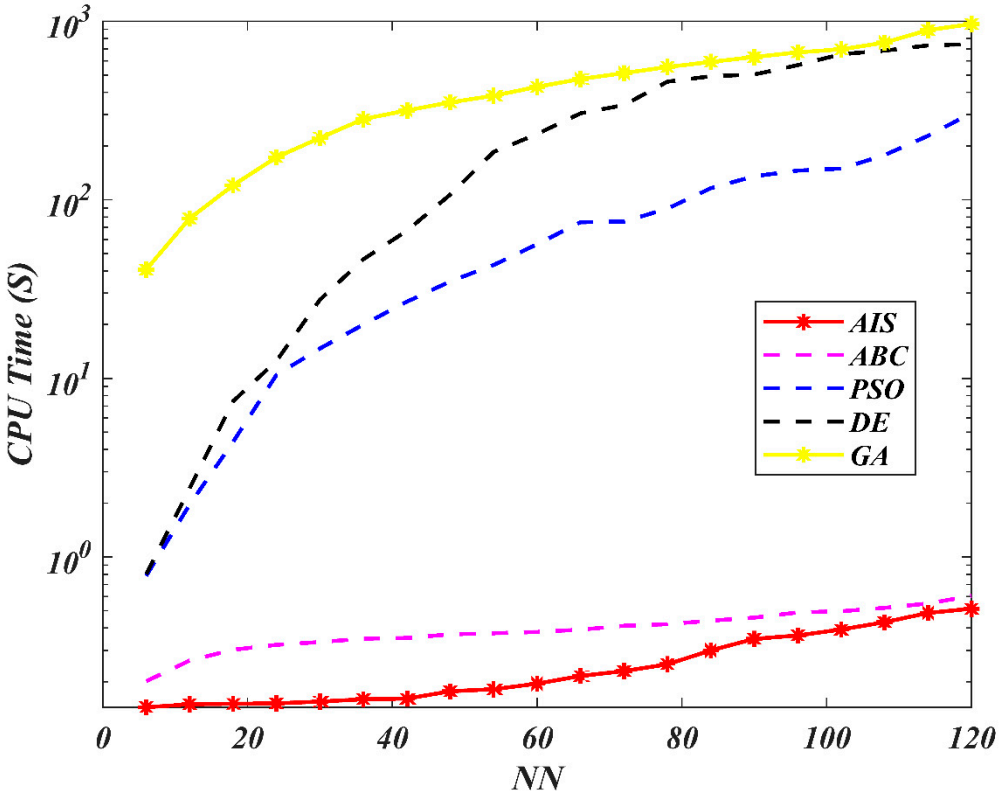

Figure 24. CPU time Evaluation for Hepatitis.

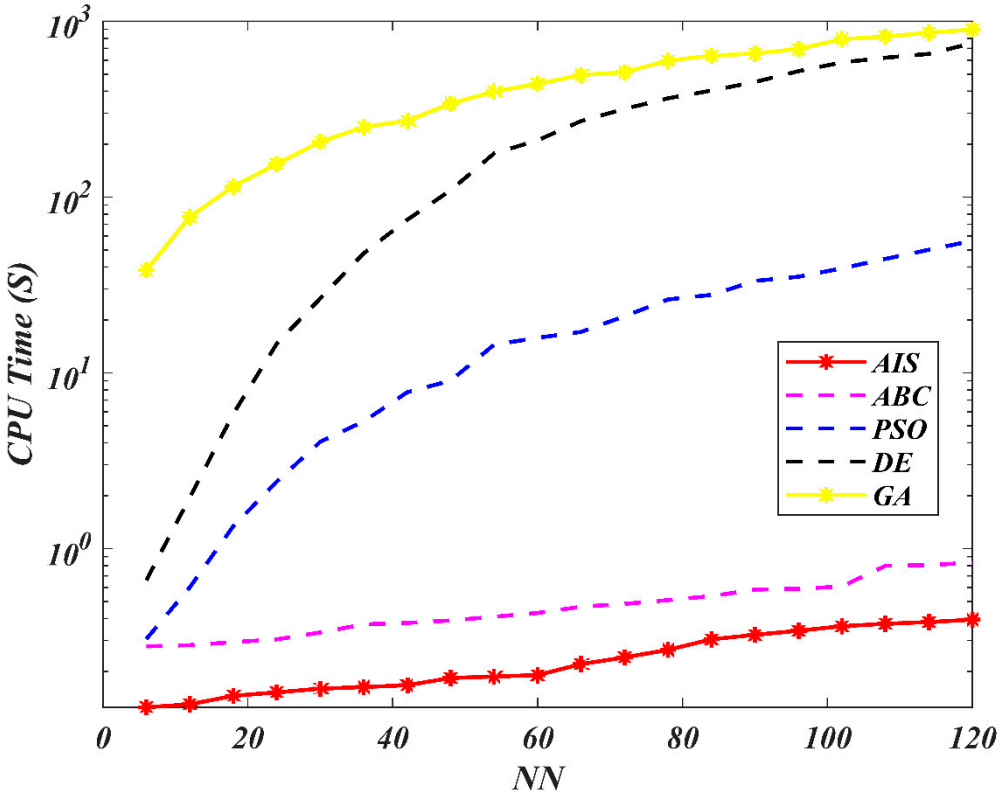

Figure 25. CPU time Evaluation for Congressional Voting Records.

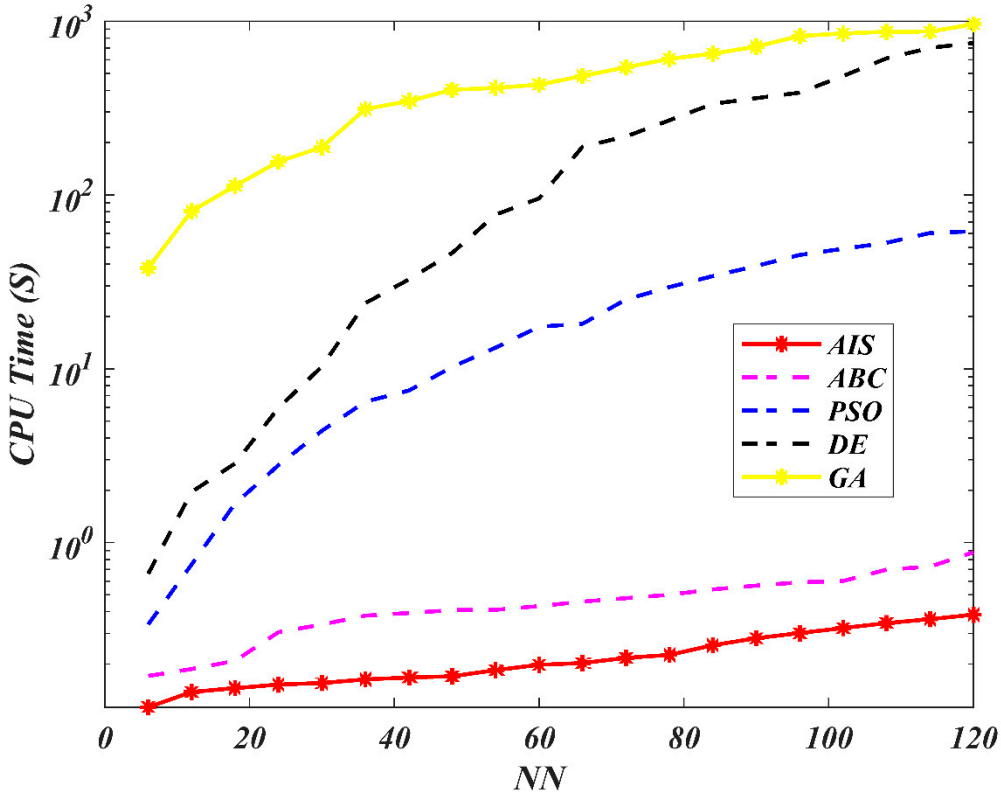

Figure 26. CPU time Evaluation for Car Evaluation.

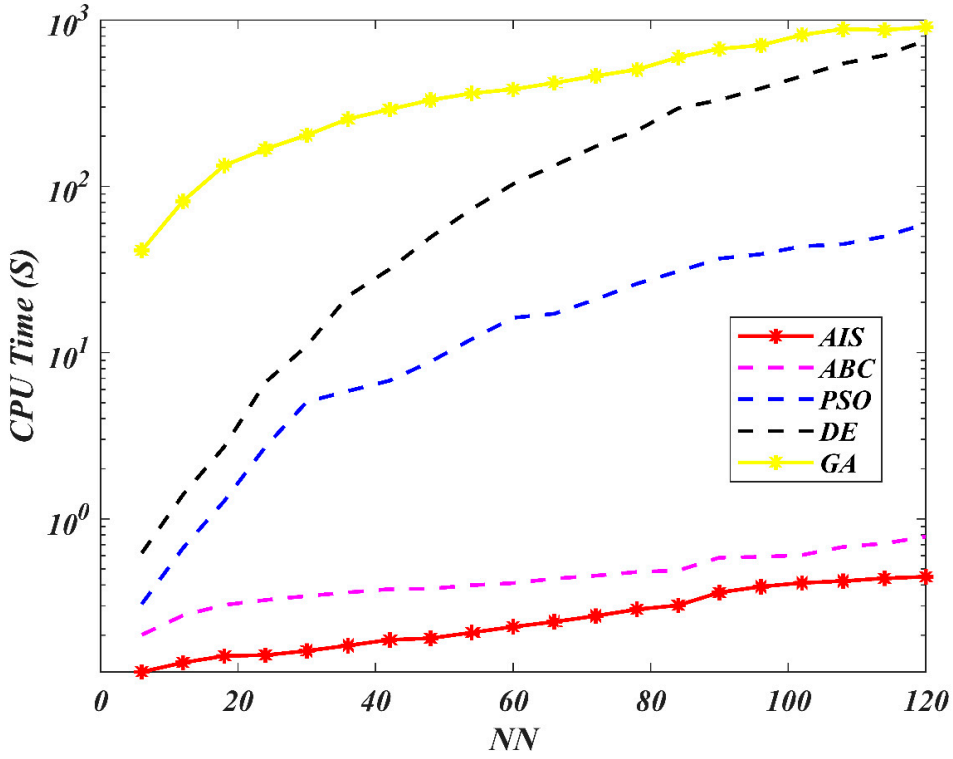

Figure 27. CPU time Evaluation for Postoperative Patient.

The simulation results have authenticated that the AIS algorithm complied efficiently with RBFNN based on 2SATRA in terms of the average value of training, where RMSE rose up to 97.5%, SBC rose up to 99.9%, CPU time by 99.8%, and the average value of testing in MAE rose up to 78.5%, MAPE rose up to 71.4%, and was capable of classifying a higher percentage of 81.6% of the test samples compared to the results of the GA, DE, PSO, and ABC algorithms. These experiments also showed that the AIS algorithm can be strongly applied for training the RBFNN-2SATRA model. Another observation involves the efficacy of AIS, which can be clearly observed when increasing the number of neurons. Furthermore, AIS with RBFNN-2SATRA achieved promising performance based on RSME, MAPE, MAE, SBC, and CPU time. This confirmed that AIS in RBFNN-2SATRA can be utilized in the pursuit of achieving better forecasting results for the 2SATRA logic rule.

8. Conclusions

The findings of the study confirmed the significant improvement of the paradigm RBFNN model via utilizing the AIS algorithm in performing 2SATRA to assist the best logical rule, which governs the behavior of the dataset. Upon introducing the new training method utilizing AIS, it has been used to train five recognized datasets, compared with four training algorithms, including ABC, PSO, DE, and GA. To affirm the performance of the proposed algorithm, all algorithms were compared through analytical tests on RBFNN-2SATRA with different numbers of neurons. Based on the results, the analysis, and discussion in this study, the following conclusions can be drawn. AIS showed a faster convergence rate with superior accuracy results. AIS achieved a lower value of RMSE, MAE, MAPE error, a lower value of SBC, and faster Central Process Unit time for training RBFNN-2SATRA. Therefore, AIS proved to be an effective approach to train RBFNN-2SATRA to classify different datasets with a diverse number of features and training samples. AIS also proved to be an effective approach to train RBFNN-2SATRA for classifying various datasets with a varied number of features and training samples. AIS can generally train RBFNN-2SATRA with a differing number of neurons. The simulation results have proven that AIS complied efficiently with RBFNN-2SATRA in relation to terms of the average value of training RMSE rose up to 97.5%, SBC rose up to 99.9%, and CPU time by 99.8%,

and the average value of testing in MAE rose up to 78.5%, MAPE rose up to 71.4%, alongside its capability of classifying 81.6% of the test samples, which is a higher percentage, compared to the results of the GA, DE, PSO, and ABC algorithms. The results confirmed that AIS significantly outperformed other contemporary technologies by substantially overwhelmingly large datasets.

For future work, it is recommended that further studies pursue two key aspects. First, the proposed RBFNN-2SATRA can be investigated for other data mining tasks, such as time series prediction and regression. Second, further studies are recommended to examine the efficiency of RBFNN-2SATRAAIS to be utilized in the future to solve traditional optimization methods, such as the N-Queen's problem and the Traveling Salesman problem.

Author Contributions: Methodology, software, resources, S.A.A.; conceptualization, validation, writing original draft preparation, project administration, S.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Fundamental Research Grant Scheme (FRGS)(203/PMATHS/6711689) by the Ministry of Higher Education Malaysia and Universiti Sains Malaysia.

Acknowledgments: We would like to thank Mohd Shareduwan Mohd Kasihmuddin and Mohd. Asyraf Mansor for their extraordinary support for this research.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Li, Z.; He, G.; Li, M.; Ma, L.; Chen, Q.; Huang, J.; Cao, J.; Feng, S.; Gao, H.; Wang, S. RBF neural network based RFID indoor localization method using artificial immune system. In Proceedings of the 2018 Chinese Control And Decision Conference (CCDC), Shenyang, China, 9–11 June 2018; pp. 2837–2842.
2. Yu, B.; He, X. Training radial basis function networks with differential evolution. In Proceedings of the Institute of Electrical and Electronics Engineers (IEEE) International Conference on Granular Computing, Atlanta, GA, USA, 10–12 May 2006; pp. 157–160.
3. Moody, J.; Darken, C.J. Fast Learning in Networks of Locally-Tuned Processing Units. *Neural Comput.* **1989**, *1*, 281–294. [CrossRef]
4. Yu, H.; Xie, T.; Paszczynski, S.; Wilamowski, B.M. Advantages of Radial Basis Function Networks for Dynamic System Design. *IEEE Trans. Ind. Electron.* **2011**, *58*, 5438–5450. [CrossRef]
5. Qadir, A.; Gazder, U.; Choudhary, K.U.N. Artificial Neural Network Models for Performance Design of Asphalt Pavements Reinforced with Geosynthetics. *Transp. Res. Rec. J. Transp. Res. Board* **2020**, 0361198120924387. [CrossRef]
6. Gan, M.; Peng, H.; Dong, X.-P. A hybrid algorithm to optimize RBF network architecture and parameters for nonlinear time series prediction. *Appl. Math. Model.* **2012**, *36*, 2911–2919. [CrossRef]
7. Yu, H.; Reiner, P.D.; Xie, T.; Bartczak, T.; Wilamowski, B.M. An Incremental Design of Radial Basis Function Networks. *IEEE Trans. Neural Networks Learn. Syst.* **2014**, *25*, 1793–1803. [CrossRef] [PubMed]
8. Dash, C.S.K.; Saran, A.; Sahoo, P.; Dehuri, S.; Cho, S.-B. Design of self-adaptive and equilibrium differential evolution optimized radial basis function neural network classifier for imputed database. *Pattern Recognit. Lett.* **2016**, *80*, 76–83. [CrossRef]
9. Yang, J.; Ma, J. Feed-forward neural network training using sparse representation. *Expert Syst. Appl.* **2019**, *116*, 255–264. [CrossRef]
10. Mansor, M.A.; Jamaludin, S.Z.M.; Kasihmuddin, M.S.M.; Alzaeemi, S.A.; Basir, F.M.; Sathasivam, S. Systematic Boolean Satisfiability Programming in Radial Basis Function Neural Network. *Processes* **2020**, *8*, 214. [CrossRef]
11. Alzaeemi, S.A.; Mansor, M.A.; Kasihmuddin, M.S.M.; Sathasivam, S.; Mamat, M. Radial basis function neural network for 2 satisfiability programming. *Indones. J. Electr. Eng. Comput. Sci.* **2020**, *18*, 459–469. [CrossRef]
12. Kasihmuddin, M.S.B.M.; Bin Mansor, M.A.; Alzaeemi, S.A.; Sathasivam, S. Satisfiability Logic Analysis Via Radial Basis Function Neural Network with Artificial Bee Colony Algorithm. *Int. J. Interact. Multimedia Artif. Intell.* **2020**. [CrossRef]
13. Hamadneh, N.; Sathasivam, S.; Choon, O.H. Higher order logic programming in radial basis function neural network. *Appl. Math. Sci.* **2012**, *6*, 115–127.

14. Hamadneh, N.; Sathasivam, S.; Tilahun, S.L.; Choon, O.H. Satisfiability of logic programming based on radial basis function neural networks. In Proceedings of the 21ST NATIONAL SYMPOSIUM ON MATHEMATICAL SCIENCES (SKSM21): Germination of Mathematical Sciences Education and Research towards Global Sustainability, Penang, Malaysia, 6–8 November 2013; pp. 547–550. [CrossRef]
15. Sathasivam, S.; Abdullah, W.A.T.W. Logic mining in neural network: Reverse analysis method. *Computing* **2010**, *91*, 119–133. [CrossRef]
16. Kho, L.C.; Kasihmuddin, M.S.M.; Mansor, M.; Sathasivam, S. Logic Mining in League of Legends. *Pertanika J. Sci. Technol.* **2020**, *28*, 211–225.
17. Alway, A.; Zamri, N.E.; Kasihmuddin, M.S.M.; Mansor, M.A.; Sathasivam, S. Palm Oil Trend Analysis via Logic Mining with Discrete Hopfield Neural Network. *Pertanika J. Sci. Technol.* **2020**, *28*, 967–981.
18. Zamri, N.E.; Mansor, M.A.; Kasihmuddin, M.S.M.; Alway, A.; Jamaludin, S.Z.M.; Alzaeemi, S.A. Amazon Employees Resources Access Data Extraction via Clonal Selection Algorithm and Logic Mining Approach. *Entropy* **2020**, *22*, 596. [CrossRef]
19. Kasihmuddin, M.S.M.; Mansor, M.A.; Jamaludin, S.Z.M.; Sathasivam, S. Systematic Satisfiability Programming in Hopfield Neural Network-A Hybrid Expert System for Medical Screening. *Comput. Appl. Math.* **2020**, *2*, 1–6.
20. Mansor, M.A.; Sathasivam, S.; Kasihmuddin, M.S.M. Artificial immune system algorithm with neural network approach for social media performance metrics. In Proceedings of the 25TH NATIONAL SYMPOSIUM ON MATHEMATICAL SCIENCES (SKSM25): Mathematical Sciences as the Core of Intellectual Excellence, Pahang, Malaysia, 27–29 August 2017. [CrossRef]
21. Hamadneh, N.; Sathasivam, S.; Tilahun, S.L.; Choon, O.H. Learning Logic Programming in Radial Basis Function Network via Genetic Algorithm. *J. Appl. Sci.* **2012**, *12*, 840–847. [CrossRef]
22. Ayala, H.V.H.; Coelho, L.D.S. Cascaded evolutionary algorithm for nonlinear system identification based on correlation functions and radial basis functions neural networks. *Mech. Syst. Signal Process.* **2016**, *68*, 378–393. [CrossRef]
23. Karaboga, D.; Kaya, E. Training ANFIS by Using an Adaptive and Hybrid Artificial Bee Colony Algorithm (aABC) for the Identification of Nonlinear Static Systems. *Arab. J. Sci. Eng.* **2018**, *44*, 3531–3547. [CrossRef]
24. Poli, R.; Kennedy, J.; Blackwell, T. Particle swarm optimization. *Swarm Intell.* **2007**, *1*, 33–57. [CrossRef]
25. Storn, R.; Price, K. Differential Evolution—A Simple and Efficient Heuristic for global Optimization over Continuous Spaces. *J. Glob. Optim.* **1997**, *11*, 341–359. [CrossRef]
26. Holland, J.H. Genetic Algorithms and the Optimal Allocation of Trials. *SIAM J. Comput.* **1973**, *2*, 88–105. [CrossRef]
27. Dandagwhal, R.D.; Kalyankar, V.D. Design Optimization of Rolling Element Bearings Using Advanced Optimization Technique. *Arab. J. Sci. Eng.* **2019**, *44*, 7407–7422. [CrossRef]
28. Goldberg, D.E.; Holland, J.H. Genetic Algorithms and Machine Learning. *Mach. Learn.* **1988**, *3*, 95–99. [CrossRef]
29. Pandey, M.; Zakwan, M.; Sharma, P.K.; Ahmad, Z. Multiple linear regression and genetic algorithm approaches to predict temporal scour depth near circular pier in non-cohesive sediment. *ISH J. Hydraul. Eng.* **2018**, *26*, 1–8. [CrossRef]
30. Jing, Z.; Chen, J.; Li, X. RBF-GA: An adaptive radial basis function metamodeling with genetic algorithm for structural reliability analysis. *Reliab. Eng. Syst. Saf.* **2019**, *189*, 42–57. [CrossRef]
31. Ilonen, J.; Kamarainen, J.-K.; Lampinen, J. Differential Evolution Training Algorithm for Feed-Forward Neural Networks. *Neural Process. Lett.* **2003**, *17*, 93–105. [CrossRef]
32. Saha, A.; Konar, A.; Rakshit, P.; Ralescu, A.L.; Nagar, A. Olfaction recognition by EEG analysis using differential evolution induced Hopfield neural net. In Proceedings of the 2013 International Joint Conference on Neural Networks (IJCNN), Dallas, TX, USA, 4–9 August 2013; pp. 1–8. [CrossRef]
33. Chauhan, N.; Ravi, V.; Chandra, D.K. Differential evolution trained wavelet neural networks: Application to bankruptcy prediction in banks. *Expert Syst. Appl.* **2009**, *36*, 7659–7665. [CrossRef]
34. Tao, W.; Chen, J.; Gui, Y.; Kong, P. Coking energy consumption radial basis function prediction model improved by differential evolution algorithm. *Meas. Control.* **2019**, *52*, 1122–1130. [CrossRef]
35. Eberhart, R.; Kennedy, J. A new optimizer using particle swarm theory. In Proceedings of the MHS'95. Sixth International Symposium on Micro Machine and Human Science, Nagoya, Japan, 4–6 October 1995; pp. 39–43.

36. Qasem, S.N.; Shamsuddin, S.M.H. Improving performance of radial basis function network based with particle swarm optimization. In Proceedings of the 2009 IEEE Congress on Evolutionary Computation, Trondheim, Norway, 18–21 May 2009; pp. 3149–3156. [CrossRef]
37. Alexandridis, A.; Chondrodima, E.; Sarimveis, H. Cooperative learning for radial basis function networks using particle swarm optimization. *Appl. Soft Comput.* **2016**, *49*, 485–497. [CrossRef]
38. Tsekouras, G.E.; Trygonis, V.; Maniatopoulos, A.; Rigos, A.; Chatzipavlis, A.; Tsimikas, J.; Mitianoudis, N.; Velegrakis, A. A Hermite neural network incorporating artificial bee colony optimization to model shoreline realignment at a reef-fronted beach. *Neurocomputing* **2018**, *280*, 32–45. [CrossRef]
39. Kasihmuddin, M.S.M.; Mansor, M.; Sathasivam, S. Robust Artificial Bee Colony in the Hopfield Network for 2-Satisfiability Problem. *Pertanika J. Sci. Technol.* **2017**, *25*, 453–468.
40. Kurban, T.; Besdok, E. A Comparison of RBF Neural Network Training Algorithms for Inertial Sensor Based Terrain Classification. *Sensors* **2009**, *9*, 6312–6329. [CrossRef] [PubMed]
41. Yu, J.; Duan, H. Artificial Bee Colony approach to information granulation-based fuzzy radial basis function neural networks for image fusion. *Optik* **2013**, *124*, 3103–3111. [CrossRef]
42. Jafrasteh, B.; Fathianpour, N. A hybrid simultaneous perturbation artificial bee colony and back-propagation algorithm for training a local linear radial basis neural network on ore grade estimation. *Neurocomputing* **2017**, *235*, 217–227. [CrossRef]
43. Satapathy, S.K.; Dehuri, S.; Jagadev, A.K. ABC optimized RBF network for classification of EEG signal for epileptic seizure identification. *Egypt. Informatics J.* **2017**, *18*, 55–66. [CrossRef]
44. Aljarah, I.; Faris, H.; Mirjalili, S.; Al-Madi, N. Training radial basis function networks using biogeography-based optimizer. *Neural Comput. Appl.* **2016**, *29*, 529–553. [CrossRef]
45. Jiang, S.; Lu, C.; Zhang, S.; Lu, X.; Tsai, S.-B.; Wang, C.-K.; Gao, Y.; Shi, Y.; Lee, C.-H. Prediction of Ecological Pressure on Resource-Based Cities Based on an RBF Neural Network Optimized by an Improved ABC Algorithm. *IEEE Access* **2019**, *7*, 47423–47436. [CrossRef]
46. Menad, N.A.; Hemmati-Sarapardeh, A.; Varamesh, A.; Shamshirband, S. Predicting solubility of CO2 in brine by advanced machine learning systems: Application to carbon capture and sequestration. *J. CO2 Util.* **2019**, *33*, 83–95. [CrossRef]
47. Dasgupta, D. (Ed.) Chapter Title. In *Artificial Immune Systems and Their Applications*; Springer Science and Business Media LLC: Berlin, Germany, 1999.
48. De Castro, L.; Von Zuben, C.J.; De Castro, L.N. Learning and optimization using the clonal selection principle. *IEEE Trans. Evol. Comput.* **2002**, *6*, 239–251. [CrossRef]
49. Hunt, J.E.; Cooke, D.E. Learning using an artificial immune system. *J. Netw. Comput. Appl.* **1996**, *19*, 189–212. [CrossRef]
50. Layeb, A. A Clonal Selection Algorithm Based Tabu Search for Satisfiability Problems. *J. Adv. Inf. Technol.* **2012**, *3*, 138–146. [CrossRef]
51. Valarmathy, S.; Ramani, R. Evaluating the Efficiency of Radial Basis Function Classifier with Different Feature Selection for Identifying Dementia. *J. Comput. Theor. Nanosci.* **2019**, *16*, 627–632. [CrossRef]
52. Hamadneh, N. Grey Optimization Problems Using Prey-Predator Algorithm. In *Advances in Data Mining and Database Management*; IGI Global: Hershey, PA, USA, 2018; pp. 31–46.
53. Mansor, M.; Kasihmuddin, M.S.M.; Sathasivam, S. Artificial Immune System Paradigm in the Hopfield Network for 3-Satisfiability Problem. *Pertanika J. Sci. Technol.* **2017**, *25*, 1173–1188.
54. Dua, D.; Graff, C. UCI Machine Learning Repository. 2017. Available online: http://archive.ics.uci.edu/ml (accessed on 24 September 2018).
55. Kasihmuddin, M.S.M.; Mansor, M.; Sathasivam, S. Artificial Bee Colony in the Hopfield Network for Maximum k-Satisfiability Problem. *J. Inform. Math. Sci.* **2016**, *8*, 317–334.
56. Miyashiro, R.; Matsui, T. A polynomial-time algorithm to find an equitable home–away assignment. *Oper. Res. Lett.* **2005**, *33*, 235–241. [CrossRef]
57. Even, S.; Itai, A.; Shamir, A. On the complexity of time table and multi-commodity flow problems. In Proceedings of the 16th Annual Symposium on Foundations of Computer Science (sfcs 1975), Berkeley, CA, USA, 13–15 October 1975; pp. 184–193. [CrossRef]
58. Mukherjee, S.; Roy, S.; Shyamapada, M. Multi terminal net routing for island style FPGAs using nearly-2-SAT computation. In Proceedings of the 2015 19th International Symposium on VLSI Design and Test, Ahmedabad, India, 26–29 June 2015; pp. 1–6. [CrossRef]

59. De Leon-Delgado, H.; Praga-Alejo, R.J.; González-González, D.S.; Cantú-Sifuentes, M. Multivariate statistical inference in a radial basis function neural network. *Expert Syst. Appl.* **2018**, *93*, 313–321. [CrossRef]
60. Idri, A.; Zakrani, A.; Zahi, A. Design of radial basis function neural networks for software effort estimation. In Proceedings of the 11th International Design Conference—DESIGN 2010, Zagreb, Croatia, 17–20 May 2010; pp. 513–519.
61. Kopal, I.; Harničárová, M.; Valíček, J.; Krmela, J.; Lukáč, O. Radial Basis Function Neural Network-Based Modeling of the Dynamic Thermo-Mechanical Response and Damping Behavior of Thermoplastic Elastomer Systems. *Polymers* **2019**, *11*, 1074. [CrossRef]
62. Hamadneh, N.; Sathasivam, S. Solving Satisfiability Logic Programming Using Radial Basis Function Neural Networks. *J. Eng. Appl. Sci.* **2017**, *4*, 1–7. [CrossRef]
63. Friedrichs, F.; Schmitt, M. On the power of Boolean computations in generalized RBF neural networks. *Neurocomputing* **2005**, *63*, 483–498. [CrossRef]
64. Awad, M. Optimization RBFNNs parameters using genetic algorithms: Applied on function approximation. *IJCSS* **2010**, *4*, 295–307.
65. Eshelman, L.J.; Schaffer, J.D. Real-Coded Genetic Algorithms and Interval-Schemata. In *Foundations of Genetic Algorithms*; Whitley, L.D., Ed.; Elsevier B.V.: Amsterdam, The Netherlands, 1993; Volume 2, pp. 187–202.
66. Wang, S.L.; Morsidi, F.; Ng, T.F.; Budiman, H.; Neoh, S.C. Insights into the effects of control parameters and mutation strategy on self-adaptive ensemble-based differential evolution. *Inf. Sci.* **2020**, *514*, 203–233. [CrossRef]
67. Opara, K.R.; Arabas, J. Differential Evolution: A survey of theoretical analyses. *Swarm Evol. Comput.* **2019**, *44*, 546–558. [CrossRef]
68. Fukuyama, Y.; Yoshida, H. A particle swarm optimization for reactive power and voltage control in electric power systems. In Proceedings of the 2001 Congress on Evolutionary Computation (IEEE Cat No 01TH8546), Seoul, Korea, 27–30 May 2001; Volume 1, pp. 87–93. [CrossRef]
69. Karaboga, D. *An Idea Based on Honey Bee Swarm for Numerical Optimization*; Technical Report-TR06; Computer Engineering Department, Engineering Faculty, Erciyes University: Kayseri, Turkey, 2005.
70. De Castro, L.; Jos, F.; Von Zuben, A.A. *Artificial Immune Systems: Part II–A Survey of Applications*; Technical Report for University of Campinas School of Electrical and Computer Engineering: Campinas, Brazil, February 2000.
71. Kasihmuddin, M.S.M.; Sathasivam, S.; Mansor, M.A. Hybrid genetic algorithm in the Hopfield network for maximum 2-satisfiability problem. In Proceedings of the 24th National Symposium on Mathematical Sciences (SKSM24), Terengganu, Malaysia, 27–29 September 2016; p. 050001. [CrossRef]
72. Mansor, M.A.; Sathasivam, S. Accelerating Activation Function for 3- Satisfiability Logic Programming. *Int. J. Intell. Syst. Appl.* **2016**, *8*, 44–50. [CrossRef]
73. Mansor, M.A.; Sathasivam, S. Performance analysis of activation function in higher order logic programming. ADVANCES IN INDUSTRIAL AND APPLIED MATHEMATICS. Proceedings of 23rd Malaysian National Symposium of Mathematical Sciences (SKSM23), Johor Bahru, Malaysia, 24–26 November 2015. [CrossRef]
74. Sathasivam, S. Upgrading logic programming in Hopfield network. *Sains Malays* **2010**, *39*, 115–118.
75. Kasihmuddin, M.S.M.; Mansor, M.A.; Sathasivam, S. Discrete Hopfield Neural Network in Restricted Maximum k-Satisfiability Logic Programming. *Sains Malays* **2018**, *47*, 1327–1335. [CrossRef]
76. Lichman, M. UCI Machine Learning Repository. 2013. Available online: http://archive.ics.uci.edu/ml (accessed on 4 April 2013).
77. Hamadneh, N. An improvement of radial basis function neural network architecture based on metaheuristic algorithms. *Appl. Math. Sci.* **2020**, *14*, 489–497. [CrossRef]