*Turkish Journal of Computer and Mathematics Education* *Vol.12 No.12 (2021), 2591-2600*

*Research Article*

# Artificial Immune System Algorithm for Training Symbolic Radial Basis Function Neural Network Based 2 Satisfiability Logic Programming

**Shehab Abdulhabib Saeed Alzaeemi[1], Saratha Sathasivam[2]*, Muraly Velavan[3], Mustafa Mamat[4]**

[1]School of Mathematical Sciences, Universiti Sains Malaysia, 11800 USM, Penang, Malaysia
[2]School of Mathematical Sciences, Universiti Sains Malaysia, 11800 USM, Penang, Malaysia
[3]School of General & Foundation Studies, AIMST University, 08100 Bedong, Kedah, Malaysia
[4]Faculty of Informatics and Computing, Universiti Sultan Zainal Abidin, 21300 Kuala Terengganu, Terengganu

**Corresponding author:**
Saratha Sathasivam
School of Mathematical Sciences, Universiti Sains Malaysia, 11800 USM, Penang, Malaysia
saratha@usm.my

**Article History:** Received: 11 January 2021; Revised: 12 February 2021; Accepted: 27 March 2021; Published online: 23 May 2021

**Abstract**: The process of radial basis function neural network based 2 Satisfiability logic programming (RBFNN-2SAT) depends mainly upon an adequate obtain the linear optimal output weights with the lowest iteration error. In this paper, the capability and effectiveness of the Artificial Immune System algorithm beside RBFNN-2SAT approach are investigated to improve the linear output by find the best output weights. In this paper, AIS algorithm is presented for enhancing the weights during training RBFNN-2SAT. The performance analysis of the presented technique RBFNN-2SATAIS is compared to three techniques, namely the no-training method, which is incorporated with radial basis function neural network 2SAT (RBFNN-2SATNT), the half-training method, which is incorporated into radial basis function neural network 2SAT (RBFNN-2SATHT), in addition to a genetic algorithm incorporated into radial basis function neural network 2SAT (RBFNN-2SATGA). The simulated results showed the paradigm performance vis-à-vis mean absolute error (MAE) and Root Mean Square Error (RMSE), as well as Schwarz Bayesian Criterion (SBC), along with the CPU Time. Accordingly, the introduced approach, i.e., RBFNN-2SATAIS outperformed the corresponding conventional approaches regarding robustness, accuracy, as well as sensitivity throughout simulation. The simulation established that the artificial immune system algorithm has effectively complied in tandem with radial basis function neural network 2SAT.

**Keywords:** Artificial Immune System Algorithm; Radial Basis Functions Neural Network; Genetic algorithm; 2 Satisfiability**.**

## 1. Introduction

The radial basis function neural network (RBFNN) in Artificial Neural Network (ANN) has been broadly implemented in many fields owing to the simpler structure of the network, better approximation capabilities, as well as faster speed learning [1]. RBFNN represents a neural network, which is feedforward, having three layers of the neuron, including an input layer, an output layer, and a hidden layer. Neurons that belong to the same layer receive inputs only from the neurons in the previous layer and send their values only to neurons in the next layer. The goal of the existence of these three layers involves minimizing the classification, as well as the RBFNN error of prediction [2]. The proper function of the radial basis function neural network (RBFNN) is primarily reliant upon the adequate parameters' choice of basis functions. A simple approach for training RBFNN assumes the fixed radial basis functions in defining the activation of the hidden units. The best set of RBFNN-2SAT of output weights can be determined directly via using a metaheuristics algorithm when the RBFNN parameters are fixed via logic programming 2SAT. 2SAT was successfully presented in 2017 as the most optimum logic programming in the system of the artificial neural network, including other metaheuristics algorithm [3]. This metaheuristics algorithm has a wide-ranging implementation to locate the nearest optimum solution for RBFNN [4, 5]. AIS, which is enthused by the immune system, uses immunological properties to develop adaptive systems to carry out various tasks in different research fields. These include supervised classification, as well as intrusion detection, in addition to clustering and optimization [6, 7]. The improved binary artificial immune system, which is based on the process of the clonal selection is presented. The binary artificial immune system, in theory, generated a plethora of works that involve combinatorial optimization, in addition to applications of real-life. In 1996, AIS was defined in accordance with the models of natural immune system [8]. Later, Valarmathy and Ramani [9] expanded this perspective when a hybrid AIS along with RBFNN was proposed to advance the classification's accuracy of the entire magnetic resonance images. However, as for the perspective of logic programming in RBFNN, extensive studies were lacking

on the optimization of the RBFNN's parameter via AIS. Mansor et al. [3] acknowledged that AIS is the ideal training model in a 3SAT neural network's system compared with the remaining metaheuristics algorithms. Through constructing RBFNN combined with 2SAT, the AIS influences on the network's training phase are examined in this study. The presented approach has been inspired by Hamadneh et al. [10], whereby the main objective is to establish the RBFNN's ideal logical model using an inclusive training method. A number of contributions were made in this paper as follows:

1. A novel perspective to approach an implicit knowledge is investigated using the model of the explicit learning. A real-life problem (an implicit representation) is learnable using an explicit mathematical representations' specified set (2SAT logical rule).

2. This study is a pioneering work to embed a 2SAT logical rule (i.e., knowledge) in a feedforward neural network (i.e., learner), whereby 2SAT logical rule has been embedded in the RBFNN via achieving the ideal parameters' value systematically, i.e., center and width.

3. Due to training the introduced RBFNN has always converged to the suboptimal output weight, two major metaheuristics are investigated in this work, including evolutionary (GA), (AIS), and two methods, including the no-training (NT) method, as well as the half training (HT) method. The RBFNN training model aims at obtaining the lowest error of iteration and the ideal output weight. It is worth mentioning that wide-ranging experimentations were performed using different performance metrics. The aims of these experiments involved revealing the AIS effectiveness in the introduced RBFNN-2SAT.

4. The introduced RBFNN is expected to provide a new perspective as RBFNN obtained the 2SAT output weight via diminishing the objective function with the structurally systematic parameter. Therefore, the presented method is a different approach compared to [11], in which Wan Abdullah's method has been used to find the accurate synaptic weight (the output weight). Both paradigms have utilized the AIS for improving the introduced methods. However, the introduced method in this work has tackled non-binary optimization in comparison with the existing approach. The presented method has, therefore, advanced a considerable potential for logic programming in the neural network.

## 2. Logic 2 Satisfiability Representation

The Logic of 2 Satisfiability (i.e., 2SAT) involves determining the Satisfiability of specific sets of clauses, containing strictly two literals for each clause [12]. This represents a given general form of Satisfiability problem, which is classified to randomized Satisfiability, as well as the maximum Satisfiability. The problem of 2SAT is described in a 2CNF form, whereby the 2SAT problem's three components include:

(a) Comprise a specific set of the *m* variables, these are $v_1, v_2, ..., v_m$

(b) Comprise a specific literals' set, whereby a specific literal signifies a specified variable or specified variable negation.

(c) A specific set of *n* distinct clauses, these are $l_1, l_2, ..., l_n$ . Each of the clauses only comprises literals, which are combined by $\wedge$ only, that is logical AND.

Every single variable takes only a bipolar value, these are 1 or 0, to exemplify true/false idea. 2SAT logic aims to identify if there exists an assignment of true values to the variables, making $P$ Satisfiable. 2 Satisfiability (2SAT) comprises a clauses' set, containing two literals. The following is the general formula of the 2SAT logic:

$$P = \bigwedge_{i=1}^{n} \left( \bigvee_{i=1}^{k} C_i \bigvee_{j=1}^{m} D_j \right), \; k = 2 \tag{1}$$

In this paper, 2SAT is the key impetus as logic programming necessitates that the program can consider 2 literals only for each clause for each execution. Many studies provided evidence that numerous combinatorial problems are formed via 2SAT logic [13- 15]. A good justification for the 2SAT logic appropriateness in representing logical rules in a given neural network is the ability to choose two literals for every clause in the given Satisfiability logic, thereby diminishing logical complexity of defining the relationships among variables in a specific neural network.

## 3. Radial Basis Function Neural Network (RBFNN)

Radial Basis Function Neural Network (RBFNN) is referred to as a feed-forward neural network. It was first used by Moody and Darken [16]. In comparison with other networks, RBFNN possesses a more integrated topology, as well as faster learning speed. Regarding the structure, RBFNN encompasses 3 neuron layers for purposes of computation. *m* neurons, in input layer, indicate transported input data to a system. These parameters, in a training phase, (center and width) can be calculated in a hidden layer; the achieved parameter is used to calculate a given output weight in a given output layer. Gaussian activation function is presented for diminishing dimensionality from a given input layer to a specific output layer. Thus, the Gaussian activation function $\varphi_i(x)$ for a hidden neuron in RBFNN [17] is as follows:

$$Q(x)=\frac{\left\|\sum_{j=1}^{N} w_{ji}^{'} x_j - c_i\right\|^2}{2\sigma_i^2} \tag{2}$$

$$\varphi_i(x)=e^{-Q(x)} \tag{3}$$

whereby $w_{ji}^{'}$ signifies a given input weight in the middle of an input neuron $j$ and a given hidden neuron $i$. Thus, structurally, $c_i$, $\sigma_i$ signify the hidden neuron's center and width, respectively. Herein, $x_j$ indicates a specific binary input value of $N$ input neurons, as well as Euclidean norm $\|\ \|$ from a specific neuron *i* to the *j* is:

$$\left\|\sum_{j=1}^{N} w_{ji}^{'} x_j - c_i\right\| = \sqrt{\sum_{m=1}^{N}\left(\sum_{j=1}^{N} w_{ji}^{'} x_j - c_i\right)^2} \tag{4}$$

The final output of RBFNN $F(x_k)$ is given by the following equation:

$$F(x_k)=\sum_{i=1}^{j} w_i\ \varphi_i(x_k) \tag{5}$$

whereby $F(x_i)=\left(F(x_1),F(x_2),...,F(x_k)\right)$ indicate an output value of the RBFNN; an output weight indicates $w_i=(w_1,w_2,...,w_N)$, RBFNN aims to obtain the ideal $w_i$, which satisfies the favorite output value. Thus, in the model above, a set of functions is provided by the hidden neuron, representing an input pattern, which is spanned by the hidden neuron [18].

**4. Logic Programming 2SAT in RBFNN**

Logic programming was presented by Kasihmuddin et al. [19] by assimilating the 2 SAT rule with the neural network. The network's weight has been determined by the Wan Abdullah's method [20], whereby the 2 Satisfiability logical rule's inconsistencies are reduced. However, there is only one issue with the introduced network, involving the weight calculation rigidness. The 2SAT is embedded into RBFNN when the variable is represented as an input neuron. Consequently, every single input neuron $x_j$ forms $\{0,1\}$, showing True/False. Upon using the value of the given input neuron, these parameters $c_i$, as well as $\sigma_i$ can be computed to achieve the ideal hidden neuron's number. Therefore, embedding 2SAT in a form of a given logical rule allows the RBFNN to accept additional input data with the fixed (center and width) value. Such a combination generates a model of RBFNN. This model can classify data in comformity with 2SAT logical rule. The representation of 2SAT in RBFNN is given in this formula:

$$P_{2SAT}=\bigvee_{i=1}^{k} C_i \bigvee_{j=1}^{n} D_j \tag{6}$$

whereby $k,n\in\mathbb{N}$. $C_i$, as well as $D_j$ represent the atoms. When applying the method of embedding RBFNN, Equation (6) is transformed into:

$$x=\sum_{i=1}^{k} I(C_i)+\sum_{j=1}^{n} I(D_j) \tag{7}$$

$$I(C_i)\ or\ I(D_j)=\begin{cases}1, & when\ C\ or\ D\ is\ True\\ 0, & when\ C\ or\ D\ is\ False\end{cases} \tag{8}$$

Both equations 7 and 8 are essential for computing the training data of every clause of 2SAT. Thus, applying 2SAT in the RBFNN is abbreviated to the RBFNN-2SAT. The RBFNN-2SAT input data is provided in Table 1.

$$P_{2SAT}=C,D\leftarrow,E\leftarrow F,K\leftarrow L \tag{9}$$

**Table. 1** Input data form/output target data form in the training data of the logic programming $P_{2SAT}$

| Clause | $C,D\leftarrow$ | | | $E\leftarrow F$ | | | $K\leftarrow L$ | | |
|---|---|---|---|---|---|---|---|---|---|
| DNF | $C\vee D$ | | | $E\vee\neg F$ | | | $K\vee\neg L$ | | |
| The input value of the data form $x_i$ | $x=C+D$ | | | $x=E-F$ | | | $x=K-L$ | | |
| The input data in a training set $x_i$ | 0 | 1 | 2 | -1 | 0 | 1 | -1 | 0 | 1 |
| Output target data $y_i$ | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 |

When center and width of the hidden layer is identified, RBFNN implemented Gaussian function in the provided equation (3) to calculate the given output weight. The increased number of clauses can make RBFNN-2SAT require a more effective learning method to locate the accurate output weight. The metaheuristics algorithm has been implemented in this work to find the most optimum output weight, which reduces this objective function:

$$f\left(w_i\right)=\sum_{i=1}^{j} w_i\,\varphi_i\left(x\right) \tag{10}$$

whereby $w_i \in \mathbb{R}$ signifies output weight (chromosome) amid a specific hidden neuron in a specific hidden layer and a given output neuron in a given output layer. Thus, $\varphi_i\left(x\right)$ refers to the provided Gaussian Activation Function in RBFNN-2SAT. $y_i$ signifies the target output value in the RBFNN-2SAT. $j$ signifies a number in the given hidden neuron.

## 5. Artificial Immune System Algorithm in RBFHNN-2SAT

Non-traditional optimization techniques have been recently enthused by nature. These techniques have increasingly attracted attention and become popular in the combinatorial optimization field. Among these techniques is the artificial immune system algorithm, stimulated by the human body's immune system. AIS is referred to as the adaptive system, enthused by the theoretical immunology, as well as the observed immune functions, principals, in addition to the models that are applied to complicated domains [21]. Moreover, AIS has been applied in various fields, namely computer network security and biological modeling, as well as virus detection, in addition to data mining and robotics, along with scheduling, classification, and clustering [21, 22]. More importantly, AIS can be demonstrated as a distributed network; it has the capability of performing parallel processing. Technically, binary AIS has been introduced based on the understanding of the immune clonal selection. This work focuses on the implemented clonal selection in binary AIS, which is utilized to optimize the output weight of RBFNN-2SAT by diminishing the given training error. Applying AIS in RBFNN is referred to as RBFNN-2SATAIS. The function to be optimized in this context is as follows:

$$f_{AIS}\left(w_i\right)=\sum_{i=1}^{j} w_i\,\varphi_i\left(x\right) \tag{11}$$

whereby $w_i \in \mathbb{R}$ signifies output weight (denoted as an antibody) amid the given hidden neuron in the given hidden layer, in addition to the given output neuron in the specified output layer. $\varphi_i\left(x\right)$ indicates the Gaussian Activation Function in RBFNN-2SAT. $y_i$ is the target output value in RBFNN-2SAT. $j$ is the hidden neuron's number and the algorithm, which is involved in RBFNN-2SATAIS, can be illustrated as follows:

Step 1

**Initialization Phase**: Following Layeb et al. [23], initialize the given population of the 100 B-cells (the given output weights) in the specified system. The B-cells representations are:

$$w_{ij}=\left(w_{1j}, w_{2j}, w_{3j}, ...., w_{nj}\right) \tag{12}$$

whereby $nj$ refers to the weight output number. The presented problem's objective function minimizes the objective function value:

$$f_{AIS}\left(w_i\right)=\sum_{i=1}^{j} w_i\,\varphi_i\left(x\right) \tag{13}$$

whereby $w_i \in \mathbb{R}$ designates the given output weight (the antibody) amid the given hidden neuron in the given hidden layer, as well as the given output neuron in the specified output layer. $\varphi_i\left(x\right)$ designates the presented Gaussian Activation Function in the RBFNN-2SAT. $y_i$ refers to the target output value in the RBFNN-2SAT. $j$ is the hidden neuron number.

Step 2

**Affinity Evaluation:** The term affinity is utilized to assess the achievable solution value for the presented objective problem. The term, which is the B-cells affinity, designates the given objective function in the specified algorithm. Each of the presented solutions' affinity value in this population can be assessed to respond to applying RBFNN-2SAT. The basis function, which can calculate each solution's affinity, in this paper, is presented in following equation [1]:

$$Aff_i=\frac{1}{\left(1+ f_{AIS}\left(w_i\right)\right)},\ 0 \le fit_i \le 1 \tag{14}$$

Based on equation (14), the lower the $f_{AIS}(w_i)$, the larger the given affinity value, whereby $f_{AIS}(w_i)$ designates the presented objective function. In case $f_{AIS}(w_i) \to \infty$, $Aff_i \to 0$. On the contrary $Aff_i = 1$ *if* $f_{AIS}(w_i) = 0$ , and thus, $Aff_i \in [0,1]$ following Li et al. (2018), feasible solutions have larger affinity.

Step 3

**Selection Phase**: Select the most optimum fifty population individuals, i.e., (B-cells) following the given affinity measure to undertake the given cloning operator for further diversification of the given population so that more efficient affinity can be achieved.

Step 4

**Cloning Phase:** This phase is key for cloning the most optimum selected B-cells. It starts with replicating the chosen B-cells via the provided classical roulette wheel selection to a system [24]. Following [24], $\beta$ designates the population's clone number presented by the given program to the given search space. Because it is efficient consistency with the study of [23], $\beta = 200$ is selected.

$$RC_i = \frac{Aff_i \times \beta}{\sum Aff_i} \tag{15}$$

where $RC_i$ is the rate of cloning or the number of clones allowed, $Aff_i$ is the affinity value of a solution, and $\sum Aff_i$ designates all solution values of affinity in the given population, whereby this procedure gives additional clones of the strings of the lower $f_{AIS}(w_i)$ compared with the higher $f_{AIS}(w_i)$, $\beta$ = 200 is chosen as a fixed given parameter.

Step 5

**Normalization Phase:** As a mechanism process before being enhanced by the given hypermutation process, this phase is key in the provided algorithm. The B-cells normalized affinity is computed, namely an affinity maturation process. The B-cells normalization's standard formulation is presented in following equation [24]:

$$affN_i = \frac{aff_i - \min aff}{\max aff - \min aff} \tag{16}$$

whereby $affN_i$ designates normalizing the B-cells affinity. $\min aff$ designates the affinity of B-cells minima value. $\max aff$ designates the B-cells maxima value.

Step 6

**Somatic Hypermutation Phase:** Calculating the number of mutations is key for AIS all through the given optimization process, which is an essential event in the enhanced binary AIS. The somatic hypermutation core impetus involves enhancing the B-cells to accomplish a specific feasible solution. According to De Castro and Von Zuben [25], a specific mechanism of the selective pressure optimizes the B-cells ability (the output weight) in obtaining the most optimum affinity. The rate of the somatic hypermutation is conversely proportional to cell affinity, whereby the higher affinity the cell receptor possesses with an antigen, the lower the mutation rate will be or vice versa. By using such a strategy, the immune system can keep in hand the higher-affinity offspring cells, as well as ensuring larger mutations for the lower-affinity ones so that more efficient affinity cells are provided [26]. Accordingly, the given mutation formula number has been presented by Layeb et al. [23] in this equation:

$$NM = aff\ N_i \times \frac{1}{NN} + (1 - af\ fN_i) \times 0.01 \tag{17}$$

$NM$ is the given Number of the specified Mutation, $NN$ designates the neuron's number, $affN_i$ designates normalizing the B-cells affinity (the output weights). After that, produce a new B-cells solution ( $w_i^{new}$ ) according to this equation:

$$w_i^{new} = \begin{cases} rand(-5,5), & rand(0,1) < r \\ w_i^*, & rand(0,1) \geq r \end{cases} \tag{18}$$

whereby $w_i^{new}$ designates the novel B-cell in case $r \in [0,1]$. After that, the B-cells new generation affinity is calculated.

Step 7

**Termination Phase:** When the condition of termination is achieved,

$$\left| f(w_i^{new}) - y_i \right| \leq tolerance \tag{19}$$

Stop, then the ideal B-cells (the ideal output weights) is memorized, otherwise the algorithm will go back to the second phase. The given tolerance value designates termination or the stopping criterion, therefore, we choose 0.001 for the analysis to decrease the statistical error [24].

## 6. Experimental Setup

All the introduced RBFNN-2SAT model has been utilized in Microsoft Visual Dev C++ software, along with Microsoft Window 7, in 64-bit, the specification of 500 GB hard drive, and 4096 MB RAM, in addition to 3.40 GHz processor. The simulated data sets have been randomly obtained by generating the input data. The data selection has reduced any possible data bias, covering more wide-ranging search space. The $NN$ , i.e., the utilized number of the neurons in the experiment, varied between $6 \le NN \le 108$ .

## 7. Results and Discussion

For a fair assessment of AIS performance when it is applied to the training RBFNN-2SAT, several experimental tests are performed, and a comparison is conducted with the remaining algorithms. Radial Basis Function Neural Network 2SAT has been compared with the untrained (i.e., RBFNN-2SATNT), and Radial Basis Function Neural Network 2SAT with the half-trained (i.e., RBFNN-2SATHT), as well as Radial Basis Function Neural Network 2SAT with the genetic algorithm (i.e., RBFNN-2SATGA), in addition to Radial Basis Function Neural Network 2SAT with the artificial immune system algorithm (i.e., RBFNN-2SATAIS). Hamadneh and Sathasivam [27] use one metric to evaluate the performance of trained RBFNN with Satisfiability logic programming using different algorithms called Mean Squares Error (MSE). In this paper used four performance metrics are calculating their respective process time (Computation time in seconds), Schwarz Bayesian Criterion (SBC), Mean Square Error (MSE), as well as Root Mean Square Error (RMSE) as in this equation:

$$RMSE = \sum_{i=1}^{n} \sqrt{\frac{1}{n}\left( f_{AIS}(w_i) - y_i \right)^2} \tag{20}$$

$$SBC = n \ln\left(\frac{\sum_{i=1}^{n}\left( f_{AIS}(w_i) - y_i \right)^2}{n}\right) + pa \ln(n) \tag{21}$$

whereby $pa$ signifies the centers' number, the widths, as well as output weights. $n$ signifies the target data's number and $f_{AIS}(w_i)$ signifies the value of the actual output, $y_i$ signifies the value of the target output.

The CPU time represents the needed time by the RBFNN-2SAT models to complete a single execution. It involves the ability, as well as the stability of RBFNN-2SAT models.

$$CPU\ time = Treaning\ Time + Testing\ Time \tag{22}$$

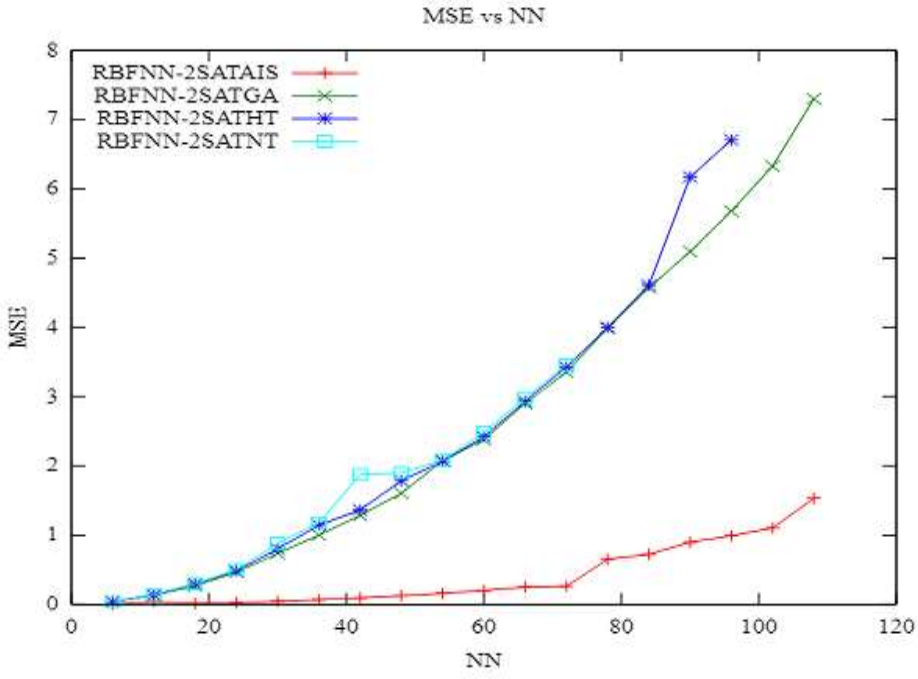


Figure 1. MSE value of the entire models of RBFNN-2SAT

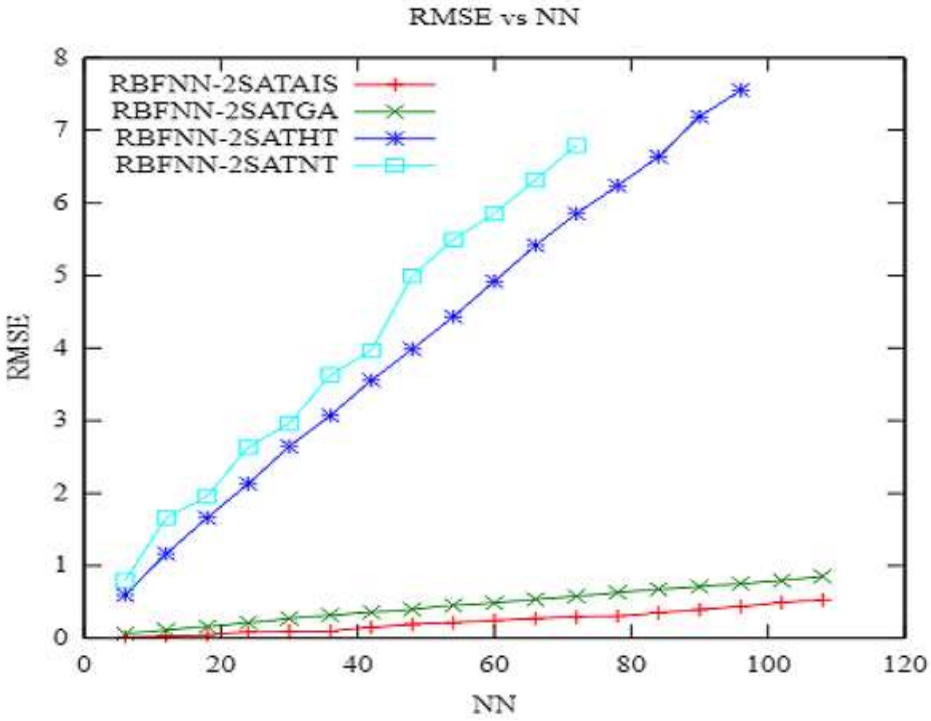


Figure 2. RMSE value of the entire models of RBFNN-2SAT

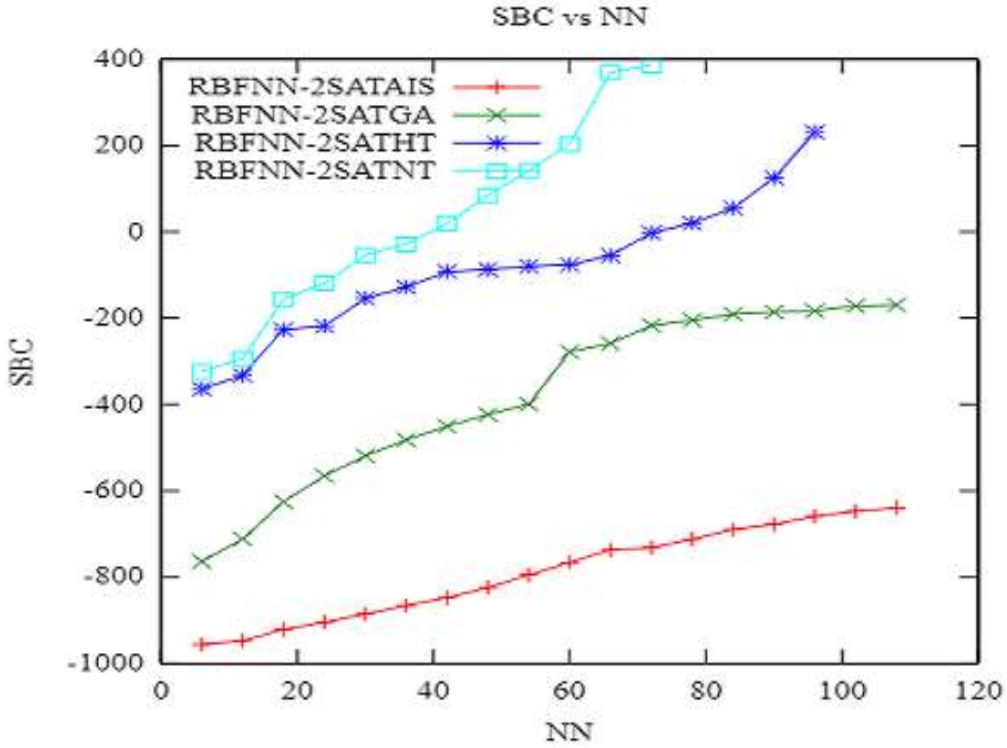


Figure 3. SBC value of the entire models of RBFNN-2SAT

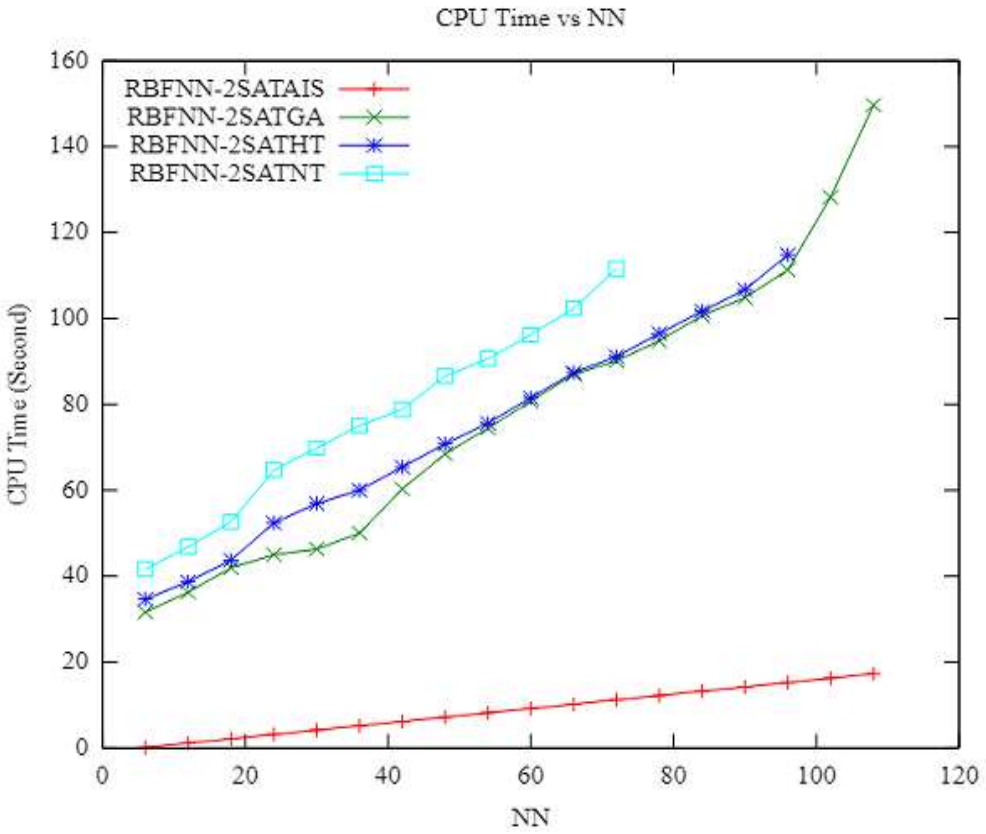


Figure 4. CPU time (seconds) models of RBFNN-2SAT

The aim of the 2SAT logical rule, in this work, is to perform much better in the neural network compared to other forms of SAT such as HornSAT [28] and generalized kSAT [12, 18], Random SAT [29] and Maximum SAT [3]. This can be attributed to the variation of the variables number in each of the clauses, which has caused RBFNN to alter the hidden layer's dimension. The imbalanced signal of the hidden layer towards the output later has led to the imbalanced value of the parameters (the center, as well as the width) and the high computation error. Thus, the results of RBFNN-2SATNT, RBFNN-2SATHT, RBFNN-2SATGA, as well as RBFNN-2SATAIS are illustrated in summary from Figure 1 until Figure 4. The results in Figure 1 until 4 showed the following findings:

1. RBFNN-2SAT is capable of receiving further input data with a fixed value of the center and the width. RBFNN-2SATAIS, herein, can create a specific model that is capable of classifying data depending upon the 2SAT logical rule using the RMSE, the SBC, and the CPU time minimum value.

2. RBFNN-2SATAIS achieved the best performance in relation to errors when the number of the neurons increased owing to the features, which made the AIS algorithm more superior compared to other methods as the AIS significant features are recognition, variation, memory, distributed perception, learning, in addition to self-organizing.

3. RBFNN-2SATAIS achieved the most optimum performance with regard to the Schwarz Bayesian Criterion (SBC) because of the increased number of neurons. Based on Hamadneh et al. [19], the SBC lowest value refers to the most optimum model. Due to MSE has a positive correlation to SBC, the fact that lower MSE will result in a lower value of SBC.

4. Regarding the computation time, RBFNN-2SATAIS has been faster compared with other models of RBFNN-2SAT. At NN > 20, the RBFNN-2SATNT possibility, as well as the RBFNN-2SATHT, which were trapped in the state of the trial and error, has increased, which has led the RBFNN-2SATNT to complete pre-mature convergence.

5. RBFNN-2SATGA, nonetheless, has slightly higher learning error because of ineffective and initial crossover. It took the RBFNN-2SATGA several iterations to be capable of producing the output weight, which is high-quality; mutation is the effective operator only during this time. However, when the suboptimal output weight has been a floating number, this has worsened the problem. In GA, novel generations were produced in GA by means of reproducing, whereas in AIS novel generations were generated via cloning and, thereby, the search agents' number in AIS was far from constant as the cloning operations increased it. The search agents in GA, however, were constant. Thus, the AIS clone and clones have moved to neighboring nodes. The search field in GA, however, includes all population.

## 8. Conclusion

A hybrid paradigm has been proposed in this work, i.e., the AIS algorithm incorporated with a radial basis function neural network (RBFNN-2SATAIS) in performing random 2SAT logic programming. The proposed model has been compared with no training method incorporated with radial basis function neural network (RBFNN-2SATNT), half training method incorporated with the radial basis function neural network (RBFNN-2SATHT), as well as a genetic algorithm incorporated with radial basis function neural network (RBFNN-2SATGA). Based on the results, there is a big difference in the performances of whole paradigms in varied four terms of the Root Mean Square Error (RMSE), Mean Square Error (MSE), Schwarz Bayesian Criterion (SBC), and process time (i.e., computation time in seconds). Moreover, based on the experimental results, the introduced paradigm has provided a lower SBC, a RMSE and MSE lower value error, and faster computation time compared with RBFNN-2SATNT, RBFNN-2SATHT, and RBFNN-2SATGA. Hence, RBFNN-2SATAIS was unequivocally found to be more efficient compare with RBFNN-2SATGA or any other method as RBFNN-2SATNT and RBFNN-2SATHT in certain aspects, including more efficiently reduced error, lower Schwarz Bayesian Criterion (SBC), and faster time of processing in executing 2SAT logic programming. For further studies, RBFNN2SATAIS can be utilized to solve traditional optimization methods like the travelling salesman, as well as the N-queen's problem.

## 9. Acknowledgment

The present research has been supported by the Research University Grant (RUI) (1001/PMATHS/8011131) provided by Universiti Sains Malaysia.

## References

1. Z. Li, G. He, M. Li, L. Ma, Q. Chen, J. Huang, J. Cao, S. Feng, H. Gao, and S. Wang, *RBF neural network based RFID indoor localization method using artificial immune system*, In 2018 Chinese Control And Decision Conference (CCDC) (2018, June) (pp. 2837-2842).
2. H. de Leon-Delgado, R. J. Praga-Alejo, D. S. Gonzalez-Gonzalez, and M. Cantú-Sifuentes, *Multivariate statistical inference in a radial basis function neural network*, Expert Systems with Applications, 93 (2018), pp. 313-321.
3. M. A. Mansor, M. S. M. Kasihmuddin, and S. Sathasivam, Artificial Immune System Paradigm in the Hopfield Network for 3-Satisfiability Problem, Pertanika Journal of Science & Technology, 25(4) (2017).
4. N. Hamadneh, *Logic Programming in Radial Basis Function Neural Networks*, Ph.D. diss., Universiti Sains Malaysia, 2013.
5. H. V. H. Ayala, and L. dos Santos Coelho, *Cascaded evolutionary algorithm for nonlinear system identification based on correlation functions and radial basis functions neural networks*, Mechanical Systems and Signal Processing, 68 (2016), pp. 378-393.
6. D. Dasgupta (Ed.), *Artificial immune systems and their applications*, Springer Science & Business Media, 2012.
7. L. N. Castro, L. N. De Castro, and J. Timmis, *Artificial immune systems: a new computational intelligence approach*. Springer Science & Business Media, 2002.
8. J. E. Hunt, and D. E. Cooke, *Learning using an artificial immune syste,*. Journal of network and computer applications, 19(2) (1996), pp. 189-212.
9. S. Valarmathy, and R. Ramani, *Evaluating the Efficiency of Radial Basis Function Classifier with Different Feature Selection for Identifying Dementia*, Journal of Computational and Theoretical Nanoscience, 16(2) (2019), pp. 627-632.
10. N. Hamadneh, S. Sathasivam, S. L. Tilahun, and O. H. Choon, *Learning logic programming in radial basis function network via genetic algorithm*, Journal of Applied Sciences(Faisalabad), 12(9) (2012), pp. 840-847.
11. S. Sathasivam, N. Hamadneh, & O. H. Choon, *Comparing neural networks: Hopfield network and RBF network*, Applied Mathematical Sciences, 5(69) (2011), pp. 3439-3452.
12. M. S. M. Kasihmuddin, M. A. Mansor, and S. Sathasivam, *Artificial Bee Colony in the Hopfield Network for Maximum k-Satisfiability Problem*, Journal of Informatics and Mathematical Sciences, 8(5) (2016), pp.317-334.
13. R. Miyashiro, and T. Matsui, *A polynomial-time algorithm to find an equitable home–away assignment*, Operations Research Letters, 33(3) (2005), pp. 235-241.
14. S. Even, A. Itai, and A. Shamir, *On the complexity of time table and multi-commodity flow problems*, In 16th Annual Symposium on Foundations of Computer Science (sfcs 1975) (1975, October), pp. 184-193. IEEE.
15. S. Mukherjee, and S. Roy, *Multi terminal net routing for island style FPGAs using nearly-2-SAT computation*, In VLSI Design and Test (VDAT), 2015 19th International Symposium on IEEE, 2015, pp. 1-6. IEEE.
16. J. Moody, and C. J. Darken, *Fast learning in networks of locally-tuned processing units. Neural computation*, 1(2) (1989), pp. 281-294.

17. A. Idri, A. Zakrani, and A. Zahi, *Design of radial basis function neural networks for software effort estimation*, IJCSI International Journal of Computer Science Issues, 7(4). journal of intelligent systems, 7(6) (2010), pp.513–519
18. N. Hamadneh, S. Sathasivam, *Solving Satisfiability Logic Programming Using Radial Basis Function Neural Networks*, Journal of Engineering and Applied Sciences, 1(4) (2017), pp. 1-7.
19. M. S. M. Kasihmuddin, M. A. Mansor, and S. Sathasivam, *Hybrid Genetic Algorithm in the Hopfield Network for Logic Satisfiability Problem*, Pertanika Journal of Science & Technology, 25(1) (2017).
20. W. A. T. W. Abdullah, *Logic programming on a neural network*, International journal of intelligent systems, 7(6) (1992), pp. 513-519.
21. L. N. De Castro, and F. J. Von Zuben, *Artificial immune systems: Part II–A survey of applications*, FEEC/Univ. Campinas, Campinas, Brazil, 2000.
22. A. Layeb, *A clonal selection algorithm based tabu search for satisfiability problems*. Journal of Advances in Information Technology, 3(2) (2012), pp. 138-146.
23. A. Layeb, H. Deneche and S. Meshoul, *A new artificial immune system for solving the maximum satisfiability problem,* in International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems, pp. 136-142, 2010, June.
24. M. A. Mansor, M. S. M. Kasihmuddin, and S. Sathasivam, Modified Artificial Immune System Algorithm with Elliot Hopfield Neural Network For 3-Satisfiability Programming, Journal of Informatics and Mathematical Sciences, 11(1) (2019), pp. 81-98.
25. L. N. De Castro, and F. J. Von Zuben, *Immune and neural network models: theoretical and empirical comparisons,* International Journal of Computational Intelligence and Applications, 1(03) (2001), pp. 239-257.
26. J. Timmis, and M. Neal, *A resource limited artificial immune system for data analysis*, In Research and Development in Intelligent Systems XVII (2001) (pp. 19-32). Springer, London.
27. N. Hamadneh, S. Sathasivam, S. L. Tilahun, and O. H. Choon, *Satisfiability of logic programming based on radial basis function neural networks*. In AIP Conference Proceedings (Vol. 1605, No. 1, pp. 547-550), AIP, (2014, July).
28. S. Sathasivam, *Upgrading logic programming in hopfield network*, Sains Malaysiana, 39(1) (2010), pp. 115–118.
29. C. Caleiro, F. Casal, and A. Mordido, *Generalized probabilistic satisfiability and applications to modelling attackers with side-channel capabilities*, Theoretical Computer Science, 2019.