International Journal of Computing and Digital Systems
ISSN (2210-142X)
Int. J. Com. Dig. Sys. 10, No.1 (May-2021)
http://dx.doi.org/10.12785/ijcds/100163

Hopfield type of Artificial Neural Network via Election Algorithm as Heuristic Search method for Random Boolean *k*Satisfiability

Hamza Abubakar[1,2*] **and Mohammed Lawal Danrimi**[3]

[1] *School of Mathematical Sciences, University of Sciences, Penang, Malaysia*
[2]*Department of Mathematics, Isa Kaita College of Education Dutsin-ma, Katsina, Nigeria*
[3]*Umaru Musa Yar'adua University, Katsina, Nigeria*

Received 25 Oct. 2020, Revised 26 Mar. 2021, Accepted 03 April, 2021, Published 02 May. 2021

Abstract: This study proposed a hybrid computational model by incorporating Election Algorithm (EA) as a heuristics search technique in a Hopfield type of artificial neural network (HNN). The main objective is to improve the learning phase of Hopfield type artificial neural network (HNN) for optimal Random Boolean *k*Satisfiability representation for higher-order logic. Many researchers in the area of artificial intelligence (AI), machine learning (ML), and artificial neural networks (ANNs) are motived by the multiple processing units that work together to learn, identify patterns and predict information which provides a powerful mechanism for optimizations/search problems and other decision-making problem. Election algorithm (EA) has been utilized due to the policy of extending the power and rule play by the political parties beyond its borders to seek endorsement from voters. This policy plays an important role in accelerating the learning process of Hopfield type of artificial neural network (HNN) for optimal random Boolean kSatisfiability representation. In this work, a different number of neurons (NN) has been manipulated invalidating the robustness and efficiency of EA in HNN for RANkSAT logical clauses. The proposed model has been compared with other existing model-based the global minima ratio, statistical error accumulations, and time complexity during the learning process. The simulated results generated have been presented in the form of graphs. Based on the result of this study, the proposed HNN-RAN-*k*SAT-EA agreed with the existing HNN-RAN*k*SAT-ACO model but outperformed HNN-RAN*k*SAT-ES in term of statistical measures used in this study.

Keywords: Artificial Neural Networks, Hopfield type of Artificial neural network, Random Boolean Satisfiability, Election algorithm.

1. Introduction

Artificial neural networks (ANN) proposed in the early 1980s as a model of the brain nervous system, with certain aspects that can learn patterns within a nonlinear, large, multi-model and multi-dimensional type of real-world application whose complexity makes them difficult to handle by exact or conventional approaches and in several applications to map and forecast complex energy systems[1]. The human brain is an extremely dynamic computer that can be used for nonlinear and concurrent computation[2]. It is identical to the brain in two respects that is the neural network (NN) acquires intelligence through learning from the external environment and the synaptic interconnection (synaptic weights) stores the information gained[3]. There are various types of artificial neural networks which served as computational models. They have been classified according to their structure and features of the learning process. Some of the typical architectures of an artificial neural network include, such as multilayer perceptron (MLP) [4], [5], Hopfield neural networks (HNN)[6], recurrent neural network (RNN)[7], extreme learning machine (ELM)[8], self-organizing map neural network (SOMNN) [9]and convolutional neural network (CNN)[10], AI and machine learning (ML)[11] have been successfully used in many real-life applications including detection problem [12], classification problem [13], prediction problem[14], traffic signal control problem [15], decision making problem [16]. Here, we briefly reviewed the structure of neural networks (ANN). For more insights and the recent developments in modelling & simulation of artificial neural networks (ANN), structural learning and or training process with implementation procedure can found in[17],[18].

Hopfield type of Artificial neural networks abbreviated as HNN is one of the many types of artificial neural

E-mail: zeeham4u2c@yahoo.com

network (ANN) that incorporate the concept of energy function known as Lyapunov energy function (LEF). The topological structure of the network (represented by the connection matrix) corresponds to the optimal configuration (described by the objective function) and transforms it into the evolution of dynamic structures of the network model. Stable state of the network satisfying the conditions in the Lyapunov energy function (LEF) continues to decrease during the operation of the network and finally reaches equilibrium at stable state. The divergence would not exist in the state of the system since the transforming function of the HNN is a minimal function. At present, to solve several issues, the use of artificial neural networks HNN in particular also points to a slow steady-state. Taking into consideration the equilibrium points of the system as a memory, the point of memory is to be reached by processing from the start of evolution into a stable region. If we take a stable point as a minimum of an energy function and consider the energy function as a fitness function, the training phase can be made into an optimization and search problem. The development of the Hopfield type of artificial neural network (HNN) is either a computational associative memory or the method of solving problems with optimization. It does not need to be solved, but its properly built relation weights and input can be accomplished by feedback neural networks. Initially, the computational functions of HNN were related to the pattern store and recovery of embedded memories[19]. Hopfield type of neural networks (HNN) has been used in the optimization of any function provided by the network parameters were set logically and appropriately and provide rapid computational capabilities [20]. The goal of this field is to represent several difficult combinatorial optimization problems in form of Satisfiability (SAT). It is the task of determining a Boolean formula's satisfiability by searching for the variable assignment if it exists which makes this formula true.

Boolean satisfiability (B-SAT) otherwise known as propositional satisfiability or Satisfiability abbreviated as SAT, is a decision or search problem in logic which determines whether there exists an interpretation to the given input variables that equate the output of a particular Boolean formula in Conjunctive Normal Form (CNF) to be evaluated to true. If such an interpretation/mapping exists, then the entire Boolean satisfiability formula will be classified as satisfiable (SAT) or (B-SAT). Otherwise, the entire Boolean formula is classified as unsatisfiable(UNSAT)[21]. Various optimization, search and decision problem can be transformed into a Boolean propositional formula which is closely related to several interesting NP-complete problems include Hitting Set problem [22], minimum circuit size problem[23], clique problem[24], graph colouring problem [25], independent set [26], timetabling problem[27]. All these problems are NP-complete problems that can be transformed and represented into Boolean Satisfiability (SAT). Many combinatorial optimizations fall into the category of optimization, sorting, decision or counting scheme for solving the satisfiability problem and is sub-optimal and partially heuristic in nature. The theoretical importance of the SAT was first to be considered as an NP-complete problem[28]. SAT is essential to several challenges in electronic design automation[29]. Boolean SAT is also at the fulcrum to many problems including decision problem [30], scheduling[31],error-correction[32] and security applications[33]. Many practical applications include model checking and combinational equivalence checking problem [34], automatic test pattern generation[35], planning in AI [36], automated theorem proving[37], software verification[38] and many more in Science, technology, engineering, communication, transportation and industrial applications. Efficient and effective solutions to NP-complete problems would help both types of research in academic and industry in various ways. There is always a trade-off between solution efficiency and effort for several optimization approaches, and particularly for modern metaheuristics, as the quality of solutions increases with increasing effort[39]**.**

The approach for Satisfiability representation via Hopfield type of artificial neural network (HNN) optimization capacity is not a straightforward process due to the convergence to local optimal solution of the Lyapunov energy function (LEF) in HNN. There are several attempts to provide solution Boolean Satisfiability (B-SAT) via conventional methods at a high computational cost. Recently, researchers in the field of machine learning (ML) and artificial intelligence (AI) are considering the HNN as a black box or symbolic structure, by the execution of logical learning in HNN which surfaced many versatile in HNN models. The primary work in [40] serves as the break rough for many researchers recently conducted in hybridizing Hopfield type of artificial neural network (HNN) with different types of logic programming yielding an extensive HNN modelling to reducing and or classify the logical inconsistency according to model setup. In layman terms, the logic program illustrates the symbolic knowledge applies in the “training” or “testing” of the HNN model. A work in [41] was proposed to foster the work proposed in [40] by implementing first-order logic in the neuro-symbolic integration model. Following the work of [40] and [41], several compelling logical rules were proposed based on Boolean propositional Satisfiability (SAT) in the Hopfield type of artificial neural network, this includes 2-Satisfiability (2-SAT logical rules) proposed in [42], maximum random kSatisfiability (MAX-R*k*SAT logical rule) proposed in[43], a random ksatisfibilty (RkSAT logical rule) proposed [44]. An Exact ksatisfibilty (Exact *k*SAT logical rules) proposed in [45], Agent-based modelling (ABM logical rule) was proposed in[46]. Hopfield neural network (HNN) has proposed for optimal representation to the various optimization problem.

Representing a network model within this framework implies "solving two key characteristics that allow it to be used to solve optimization problems. Its dynamics of activation and a related energy function that decreases as the network evolves dynamically. The energy function of an HNN has many local minima and is forced to decrease only if the network evolves according to its dynamics equations[47]. Consequently, the network probably will reach an equilibrium state that does not correspond to a problem solution. But on the same front, having many local minima is good for building content-addressable memories. The search for evolutionary metaheuristics algorithm strategies to move the network out of local minima and take it to a global minimum is an important task in this field. These limitations motivated researchers to propose different hybrid systems to increase the accuracy and stability of HNN for solving various optimization problems. Therefore, a rigorous training approach is required such as or robust metaheuristic techniques such as Election algorithm (EA), Ants Colony optimization algorithm (ACOA), firefly algorithm etc. to lower the complexities involve in search space and to accelerate the performance of HNN to avoid settling down at the local minimum energy (wrong solution).

The effectiveness of various metaheuristics algorithm (MA) used in the optimization of various mathematical and or engineering function has been proven by various researchers. The MA has the benefits of high potential for global optimization, rapid speed, broad flexibility and quick execution. Several metaheuristic approaches have been reported in the literature that works satisfactorily in enhancing the searching capacity of HNN toward finding an optimal solution to the variant of the Boolean satisfiability problem (SAT). This includes the hybridization of genetics algorithm (GA) and improved genetic algorithm (IGA) in the tuning of the parameters and structure of the artificial neural network(ANN) [48], work of [49] who used GA in modelling ANN to examine the proper stabilization of weak subgrade soil at high moisture contents. Particle Swarm Optimization (PSO) has been used in [50] to the modelling of Feed Axis in Machine Tools (FAMT) based Generalized Regression Neural Network (GRNN). Another powerful metaheuristic based on artificial bee colony (ABC) in ANN has been proposed in [51] for the classification of alcohols obtained by QCM sensors. differential evolution(DE) has been prosed [52] for the prediction of workload in the cloud using an artificial neural network (ANN).

Specifically, the Hopfield type of artificial neural network (HNN) embrace metaheuristics algorithm (MA) to reduce the logical inconsistency in interpreting logic clauses. The prototype optimization made use of the global and local convergence capacity of metaheuristics to confronting learning complexity within the HNN. The simulation implemented with a GA and other traditional exhaustive search methods with different Boolean satisfiability problems with metaheuristics been proposed. Modified Imperialistic Competitive Algorithm (MICA) for Boolean satisfiability (SAT) in Hopfield type of Neural Network (HNN) for Logic Mining was presented in [53], The performance of Hopfield type of artificial neural network has been enhanced by incorporating its learning with Election algorithm (EA) for Boolean random k satisfiability (RkSAT) in [54], The modified version of election algorithm (EA) for optimal random k satisfiability was proposed in[55], Hybrid ant colony optimization (HACO) was proposed for even-2 satisfiability (even2SAT) in Hopfield type of neural network by [56]. Metaheuristics Approach based on Colony selection (CS) for Maximum k Satisfiability (MAXkSAT) in Restricted Neural-Symbolic Integration n was proposed in [57]. Mean-Field Theory (MFT) in Hopfield type of artificial neural network (HNN) in carrying out 2 Satisfiability has been presented in [58] and another related study was developed in [59] in representing random satisfiability (RkSAT) logic programming in Hopfield type of neural network (HNN). Election algorithm as one of the novel metaheuristics is introduced in this study to supplement the in representing random satisfiability (RkSAT) logic programming in Hopfield type of neural network (HNN) to facilitate the search process of RAN*k*SAT for higher-order logic. HNN-RAN-*k*SAT-EA indicates the combination of the in representing random satisfiability (RkSAT) logic programming in Hopfield type of neural network (HNN) and Election algorithm in optimizing any given RAN*k*SAT problem. However, to the best of our knowledge, no effort combined the global optimization ability of Election algorithm (EA) to facilitate the training phase of HNN in carrying out random boolean satisfiability logic programming (RAN*k*SAT for higher order logical rule. Therefore, propose a new hybrid computational model by incorporating election algorithm (EA) to foster the learning phase of HNN in optimizing random *k*-satisfiability (HNN-RAN-*k*SAT-EA) in attaining better accuracy, sensitivity and robustness for higher-order networks. The contributions of the present study include the following:

1. To upgrade the RAN-*k*SAT logical rule to accommodate high order $(k \leq 3)$
2. To integrate newly proposed high order logical rule on Hopfield type of artificial neural network.
3. To incorporate Election algorithm in Hopfield type of artificial neural network.
4. The combination of two machine learning (ML) tools i.e. Election algorithm and Hopfield types of artificial neural network for optimal or near-optimal

representation to RAN-*k*SAT logical rule for high order logic.

5. To establish a comprehensive comparison of the HNN-RAN-*k*SAT-EA with existing models.

The proposed hybrid computational model will be proved an alternative method of doing computation in repressing various hard combinatorial optimizations problem. Our results explored that the new proposed hybrid computational model improves the learning phase of Hopfield types of artificial neural network by demonstrating good agreement with the performance of the existing work. The rest of this paper is organized as follows. In Section 2, Random Boolean *k*Satisfiability for higher-order logic been reported. Section 3 presented the mapping of Random Boolean *k*Satisfiability in Hopfield type neural network model. Section 4 cover Election algorithms and the proposed Election algorithm incorporated in Hopfield type of artificial network model for Random Boolean *k*Satisfiability representation in higher order. The model implementation and experimental setup have been presented in Section 5. Finally, Section 6 cover experimental results & discussion and conclusions of this exploration.

2. MATERIALS AND METHODS

A. Random Boolean kSatsifiability

Boolean Satisfiability (B-SAT or kSAT) is one of the central problems in the mathematical logic and theory of Complexity; they are canonical NP-complete problems and their approximate versions are also difficult and challenging. By fixing a distribution over clauses, then drawing i.i.d. clauses from this distribution, an instance of random satisfaction is created. The average difficulty of satisfaction concerns is often inspired by its contributions to physical device dysfunction models and cryptography which involve problems that are complicated on average. The random k Satisfiability (RANkSAT) of a Boolean formula is considered as a decision problem which decided a Boolean formula in Conjunctive Normal form (CNF) has a truth assignment satisfying random number literal in each clause or determine that no such label assignment exists; The Random Boolean *k*Satisfiability (RAN*k*SAT) problem is a variant of Boolean Satisfiability (B-SAT), where the input instance may be the same but the question is that clause, literal or both are generated at random[60]. Non-systematic Boolean Satisfiability logic (RAN*k*SAT) has been proven effective to represent simulated applications [61]. The formulation of RAN*k*SAT logical clauses is based on the properties listed as follows:

- A collection of logical variables $\forall S_i \in [s_1, s_2, s_3, \ldots, s_n]$, in logical clauses $\forall C_i \in [c_1, c_2, c_3, ..., c_n]$, consisting of s as a literal $or\ \neg S_i$ as a negation of literal.
- In RAN*k*SAT for $k \le 3$, random variables are selected from a set of n logical variables based on 50% chances of negating each logical variable in the random clause.
- Each literal S_i in each logical clauses C_i is joined by a disjunction symbol "$\vee$" and each logical clause C_i is connected by a conjunction "$\wedge$".
- Each literal s_i in RAN*k*SAT is represented by bipolar $x_i \in [1, -1]$ The general formulation $F_{RANkSAT}$ for is presented in (1) as follows.

$$F_{RANkSAT} = \overset{j}{\underset{i=0}{\wedge}} C_i^{(2)} \overset{d}{\underset{i=0}{\wedge}} C_i^{(1)} \ \forall k \le 2 \tag{1}$$

Equation (1) can be upgraded to accommodate higher-order neurons connection as follows

$$F_{RANkSAT} = \overset{t}{\underset{i=0}{\wedge}} C_i^{(3)} \overset{j}{\underset{i=0}{\wedge}} C_i^{(2)} \overset{d}{\underset{i=0}{\wedge}} C_i^{(1)} \ \forall k \le 3 \tag{2}$$

where $\forall t, j, d > 0$. The clause s $F_{RAN-2SAT}$ and $F_{RAN-3SAT}$ in Equation (1) and (2) can be defined as RAN*k*SAT $(\forall k \le 2)$ and RAN*k*SAT $(\forall k \ge 3)$ for $C_i^{(k)}$ clauses in Equation(3) and (4) respectively as follows.

$$C_i^{(k)} = \begin{cases} (\lambda_i \vee \mu_i), & \forall k = 2 \\ \tau_i, & \forall\ k = 1 \end{cases} \tag{3}$$

$$C_i^{(k)} = \begin{cases} (\lambda_i \vee \mu_i \vee \ell_i), & \forall k = 3 \\ (\lambda_i \vee \mu_i), & \forall k = 2 \\ \tau_i, & \forall\ k = 1 \end{cases} \tag{4}$$

where $\lambda_i \in [\lambda_i, \neg\lambda_i]$, $\tau_i \in [\tau_i, \neg\tau_i]$, $\mu_i \in [\mu_i, \neg\mu_i]$, $\ell_i \in [\ell_i, \neg\ell_i]$. These can represent literals and their negation in RAN*k*SAT logical clauses respectively. Specifically, $C_i^{(1)}$ denoted the first-order logic in Equation (3) and (4), $C_i^{(2)}$ denoted the second-order in Equations (3) and (4) and a third-order logical clause is denoted by $C_i^{(3)}$ in Equation (4).

In this work, we refer to $F_{RAN-kSAT}$ as a Random Boolean satisfiability written in Conjunctive Normal Form (CNF) whereby logical clauses $F_{RAN-kSAT}$ are chosen uniformly, independently among all $2^x \binom{d+j+t}{y}$ without replacement a logic clauses of length x. We refer to the optimal interpretation of $F_{RAN-kSAT} \to [1, -1]$ as a logical representation which can be expressed as 1 for (TRUE) and -1 otherwise. Theoretically, One interpretation of RANkSAT formulation considering $k \le 2$ is presented as follows.

$$F_{RANkSAT} = (\tau_1 \vee \neg\tau_2) \wedge (\neg\mu_1 \vee \mu_2) \wedge \neg\lambda_1 \quad (5)$$

Equation (5) can be upgraded to accommodate high order connection when $k \le 3$ as follow

$$F_{RANkSAT} = (\tau_1 \vee \neg\tau_2 \vee \tau_3) \wedge (\neg\mu_1 \vee \mu_2) \wedge \neg\lambda_1 \quad (6)$$

According to Equation (3), $F_{RANkSAT}$ comprises of the following,

$$C_i^{(3)} = (\tau_1 \vee \neg\tau_2) \quad (7)$$

$$C_2^{(2)} = (\neg\mu_1 \vee \mu_2) \quad (8)$$

and $$C_1^{(1)} = \neg\lambda_1 \quad (9)$$

Therefore, the output of Equation (3) is satisfied $F_{RANkSAT} = 1$ if

$$(\tau_1, \tau_2, \mu_1, \mu_2, \lambda_1) = (1,1,1,-1,1) \quad (10)$$

with 3 clauses are satisfied $\left(C_i^{(3)}, C_1^{(2)}, C_2^{(1)}\right)$. Equation (4), $F_{RANkSAT}$ comprises the following

$$C_i^{(3)} = (\tau_1 \vee \neg\tau_2 \vee \tau_3) \quad (11)$$

$$C_2^{(2)} = (\neg\mu_1 \vee \mu_2) \quad (12)$$

Therefore, the output in Equation (4) is satisfied $F_{RANkSAT} = 1$ if

$$(\tau_1, \tau_2, \tau_3, \mu_1, \mu_2, \lambda_1) = (1,1,1,1,-1,1) \quad (13)$$

with 3 clauses are satisfied $\left(C_i^{(3)}, C_1^{(2)}, C_2^{(1)}\right)$.

This research $F_{RANkSAT}$ will be embedded in HNN for RAN*k*-SAT in comparison with other learning algorithms. $F_{RANkSAT}$ will embrace the modified networks to lunch the true structure or behaviour of the data involved. Note that $F_{RANkSAT}$ is a symbolic form representation thus it is appropriate to be integrated with these networks as HNN is a non-symbolic platform.

B. *Computational Complexity of Boolean Satisfiability*

Computational Complexity is a field of mathematical logic in computational science that focuses on classifying and evaluating computational problems depending on the degree of difficulty and computational time consume in solving them, i.e. the number of resources needed to use any algorithm to find a solution to a problem[62].

Complexities such as non-convexity, nonlinearities, discontinuities, mixed existence of variables, various disciplines and broad dimensionality are involved in most real-world search and optimization problems, a mixture of which makes conventional known algorithms either inefficient, impractical or inapplicable. No known mathematically driven algorithms exist to find the optimal solution for all such issues in a short computational time.

Boolean Satisfiability problems known as SAT is one of the intensely studied areas in the computational science and most prominent NP-complete decision the first problem that was classified as NP problems by Cook in 1971[63]. The number of combinations for the SAT can be determined via the following formula as proposed in [64]

$$Combination = 2^{(NC*3)} \quad (14)$$

whereby NC is the number of clauses. The formula in Equation (9) is crafted based on the original formula [64]. In our study, 3 clauses are comprising of at least 4 literals for a $F_{RANkSAT}(k \le 2)$ and 6 literals for $F_{RANkSAT}(k \le 3)$ formula as follows.

$$Combination = 2^{(3*3)} = 512 \quad (15)$$

The complexity of Boolean Satisfiability has been displayed in Table 1. It shows how the search space is expanding exponentially with the number of clauses (NC). The size of the search space is the factorial of ***n***. Therefore, powerful search techniques are required to search for the solution to this type of problem. Metaheuristics algorithm (MA) are among the successful approaches based on evidence in optimizing different problems, including the Satisfiability problem (SAT)[65]- [66].

TABLE I. NUMBER OF CLAUSES (NC) VERSUS SEARCH SPACE (SS)

NC	Size SS
1	$2^{(1*3)} = 8$
2	$2^{(2*3)} = 64$
3	$2^{(3*3)} = 512$
4	$2^{(4*3)} = 4096$
5	$2^{(5*3)} = 32768$
6	$2^{(6*3)} = 262144$
7	$2^{(7*3)} = 16777216$

If the problem size is raised as the time complexity grows exponentially, the worst-case scenario occurs. Because as the number of clauses (NC) grows, the procedure requires a vast search space, the Satisfiability of the logical clause is systematically evaluated by integrating search techniques such as local search, exhaustive searching. As the number of correct interpretations in the Boolean Satisfiability problem increases in proportion to the number of clauses (NC), a local search or exhaustive searching techniques cannot handle it. Therefore, a rigorous training approach is required such as or robust metaheuristic techniques such as Election algorithm (EA), Ants Colony optimization algorithm(ACOA), firefly algorithm (FFA) etc. to lower the complexities involve in search space[46].

C. *Mathematical Representation of Random Boolean kSatisfiability in Hopfield type of Artificial Neural Network model*

The computational structure of the Hopfield type of artificial neural network (HNN) consists of a single layer recurrent neural network(RNN) that embodies the approach of storing information as the stable/equilibrium states of a dynamically evolving network configuration[67]. It uses an energy function known as Lyapunov Energy function (LEF) in terms of the synaptic weights matrix and output of the neurons and revealed how much a particular network is applied in the optimization of a particular problem in associative memory and combinatorial type of optimization or decision problem. The discrete version of the Hopfield neural network (DHNN) is used as associative memory whereby information is stored and later retrieved via association with input pattern, rather than by address programming[54].

Given an initial input pattern that is mapped to the neuron state in Equation (16), the HNN will converge to the equilibrium state corresponding to the minimum value known as minimum energy (Barra, 2018). Henceforth, the final state of the HNN corresponds to the solution of the combinatorial problem. The neurons in HNN are represented in a bipolar form whereby 1 is classified as true(-1 as false) obeying the dynamics $S_i \rightarrow \operatorname{sgn}(h_i)$ described by *Ising* variables found in the spin-glass problem of statistical mechanical (Sherrington, 2010) where h_i is defined as the local field function described as follows;

$$h_i(t) = \sum_{j=1, i\neq j}^{m} M_{ij}^{(2)} S_j(t) + M_i^{(1)} \tag{4}$$

The advantage of using bipolar values over binary values is the symmetry of the states of the network. If some pattern S_i in bipolar form is stable, its inverse is stable too whereas. The neurons $S_i = (s_1, s_2, s_3, \ldots, s_n)$ updated their status asynchronous as follows.

$$S_i(t+1) = \begin{cases} 1, & \mathit{if} \ \sum_{j}^{n} M_{ij} S_j(t) + \xi_i \geq 0 \\ -1, & \mathit{Otherwise} \end{cases} \tag{5}$$

where M_{jk} is the synaptic weight of HNN that established the connection mapping associated between j and k neurons, S_j is regarded as the unit condition k and ξ_i is described as the threshold function of neurons j. Some studies conducted to verify that the LEF of the HNN model always decreases monotonically to certain configuration equivalent to state include [68],[69], [68], [67]. Each time neuron linked J_{jk}, the value of the synaptic weight will be preserved as a stored pattern in HNN CAM in an interconnected matrix where $M^{(1)} = [M_{jk}^{(1)}]_{n\times n}$, $M^{(2)} = [M_{jk}^{(2)}]_{n\times n}$ and $M^{(3)} = [M_{ijk}^{(3)}]_{n\times n}$ or N-dimensional variable vectors as follows.

$$S_{ij} = (s_{1j}, s_{2j}, \ldots, s_{nj})^T, \forall i, j \in □ \tag{16}$$

The constraint of synaptic weight matrix $M^{(1)}$ and does not allow self-loop of neurons as follows

$$M_{ji}^{(2)} = M_{ij}^{(2)} = 0, \forall i, j \in □ \tag{17}$$

For third-order logic is given as

$$M_{jjj}^{(3)} = M_{kkk}^{(3)}, \ldots, = M_{iii}^{(3)} = 0, \forall i, j, k \in □ \tag{18}$$

symmetrical neuron synaptic weight matrix is given as

$$M_{ii}^{(2)} = M_{jj}^{(2)} = M_{kk}^{(2)}, \forall i, j \in □ \tag{19}$$

for third-order logic is given as,

$$M_{kji}^{(3)} = M_{kij}^{(3)} = M_{ijk}^{(3)} = M_{ikj}^{(3)} = M_{jki}^{(3)} = M_{jik}^{(3)}, \forall i, j \in □ \tag{20}$$

The LEF of the HNN model and the CAM provided high strength, scalable framework, error tolerance, fast memory retrieval and partial inputs pattern[70], [71].

To represent the HNN model for combinatorial optimization such as Boolean Satisfiability (B-SAT). it revolved around the synaptic strength of the system. HNN has been used as a logical rule that controls the behaviour of the device configuration. The random Boolean kSatisfiability(RAN-*k*SAT) can be embedded into HNN as a single model RAN-kSAT-HNN by simply assigning each connection to neurons S_i with the given cost function $\left(E_{F_{RANkSAT}}\right)$ which served as the fitness function or objectives function to be optimized. Furthermore, the fitness function $E_{F_{RANkSAT}}$ which controls the combinations of HNN and RAN-*k*SAT is given as

$$E_{F_{RANkSAT}} = \sum_{i=1}^{n} \prod_{j=1}^{m} M_{ijk} \tag{6}$$

where n and m designated as the number of clauses and the number variables in $F_{RANkSAT}$ respectively. The inconsistency of $F_{RANkSAT}$ in equation (6) is given as;

$$M_{ij} = \begin{cases} \frac{1}{2}\left(1 - S_\rho\right), \mathit{if} \ \neg\rho \\ \frac{1}{2}\left(1 + S_\rho\right), \mathit{otherwise} \end{cases} \tag{7}$$

the updating rule for $E_{F_{RANkSAT}}$ in HNN in Equation (4) and (5) can be upgraded to embraced third-order connection is defined in Equation (8) and (9) respectively as follows;

$$h_i(t) = \sum_{i=1, i\neq j, j\neq k}^{m} \sum_{i=1, i\neq j, j\neq k}^{m} M_{ijk}^{(3)} S_j(t) + \sum_{i=1, i\neq j, j\neq k}^{m} M_{ij}^{(2)} S_j(t) + M_i^{(1)} \tag{8}$$

In this case, two values for the output of each neuron are possibly presented as follows;

$$S_i(t+1) = \begin{cases} 1, \sum_{i=1, i\neq j,}^{m} \sum_{i=1, j\neq k}^{m} M_{ijk}^{(3)} S_j(t) + \sum_{i=1, j\neq k}^{m} M_{ij}^{(2)} S_j(t) + M_i^{(1)} \geq 0 \\ -1, \sum_{i=1, i\neq j,}^{m} \sum_{i=1, j\neq k}^{m} M_{ijk}^{(3)} S_j(t) + \sum_{i=1, j\neq k}^{m} M_{ij}^{(2)} S_j(t) + M_i^{(1)} < 0 \end{cases} \tag{9}$$

where $M_{ijk}^{(3)}$, $M_{ij}^{(2)}$ and $M_i^{(1)}$ are third, second and first order synaptic weights of the embedded $F_{RANkSAT}$ logic. Equation (8) and Equation (9) are important to ensure the neurons S_i will always converge to a stable state $E_{F_{RANkSAT}} \to 0$. The LEF in $H_{F_{RANkSAT}}$ has been utilized to ensures the energy dynamics for the network decrease monotonically. The quality of the retrieved S_i can be evaluated by employing the LEF as follows

$$H_{F_{RANkSAT}} = -\frac{1}{2}\sum_{i=1,i\neq j}^{m}\sum_{j=1,i\neq j}^{m} M_{ij}^{(2)} S_i S_j - \sum_{i=1,i\neq j}^{m} M_i^{(1)} S_j \, (\forall k \leq 2) \qquad (10)$$

The dynamics of energy is always decreasing till the system reaches its global minimum energy. Equation (10) is a monotonic decrease with the dynamics and can be generalized to include third-order connections as follows:

$$\begin{aligned} H_{F_{RANkSAT}} &= \ldots -\frac{1}{3}\sum_{i=1}^{m}\sum_{j\neq k}^{m}\sum_{k=1,i\neq k}^{m} M_{ijk}^{(3)} S_i S_j S_k \\ &\quad -\frac{1}{2}\sum_{i=1,i\neq j}^{m}\sum_{j=1}^{m} M_{ij}^{(2)} S_i S_j - \sum_{i=1,i\neq j}^{m} M_i^{(1)} S_j \end{aligned} \qquad (11)$$

The energy value derived from Equation (11) will be classified as minimum global energy abbreviated as Zm or local minimum energy abbreviated as Ym. When the induced neuron state reaches the Zm, the network will generate the correct solution. Since the energy of the network decreases finitely as the network states change, the energy gap between states approaches zero as the number of iterations rises to its maximum level. The functioning of the HNN modelling processing relies on the dynamic behaviour of the LEF in a network which always decreases from the initial states to the equilibrium states as it evolves. Equation (8) proves that the energy from $F_{RANkSAT}$ is often depicted reduces monotonically. The $H_{F_{RANkSAT}}$ shows energy values for the absolute final energy $H_{F_{RANkSAT}}^{\min}$ derived from $F_{RANkSAT}$ [40]. As the network approaches final energy, the changes in network energy approach zero. Hence the quality of the final neuron state can be properly classified according to the following condition.

$$\left| H_{RAN-kSAT} - H_{RAN-kSAT}^{\min} \right| \leq \xi \qquad (12)$$

where ξ is the pre-determined tolerance value. Note that, if the embedded $F_{RANkSAT}$ does not satisfy Equation (12), the final state has been trapped wrong pattern (local minimum solution).

The interest is to embed Random kSatisfiability in HNN as a proposed model via Election Algorithm (RAN-*k*SAT-HNN-EA) in the next section. The properties of RAN*k*SAT as a logical rule can be applied in governing the behaviour of discrete HNN. In optimization terms, Satisfiability (SAT) is a combinatorial type of optimization problem (COP). A calculus-based optimization approach can not easily solve this type of problem. Indeed, to search for optimal representation, it needs a versatile approach that can function in a complex manner. Hopfield type of artificial neural network (HNN) has proposed for optimal representation to various optimization problem as discussed in section 3 including random Boolean satisfiability problem. It is, however, associated with some limitations. One of the major limitations of the Hopfield type neural network (HNN) is the convergence to some local minimum energy of the network instead of finding the state that corresponds to the global minimum energy i.e desired solution. Therefore, a rigorous training approach is required such as or robust metaheuristic techniques such as Election algorithm (EA), Ants Colony optimization algorithm(ACOA), firefly algorithm etc. to lower the complexities involve in search space and to accelerate the performance of HNN to avoid settling down at the local minimum energy[46]. In this paper, the Election algorithm will be utilized in the learning phase of HNN.

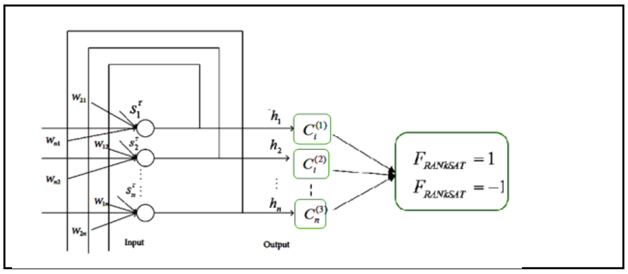

Figure 1. The architecture of the Hopfield neural network

The Hopfield type of artificial network (HNN) presented in Figure 1 is characterized by the presence of feedback corrections which means that the outputs are connected to the inputs. The Hopfield network is a cyclic neural network with feedback connections from output to input. After the signal is input, the state of the neuron will continue to change over time, and finally converge or periodically oscillate. No study combines a discrete version of HNN as a single computational model. Thus, the robustness of EA helps to improve the training process in HNN. The present study combines a discrete version of with a discrete version of HNN as a single computational model. Thus, the robustness of EA helps to improve the training process in HNN.

A. *Election Algorithm As Heuristic Search In the Hopfield Model*

Election Algorithm (EA) is a population-based iterative algorithm working with a range of solutions. The local search capabilities in EA has been converted into a partitioned search space. The optimization procedure is inspired by the process of elections conducted in human society [72]. Generally, the population of an individual in EA are divided into two parts, candidates and voters that make up the solution area of several electoral parties which later undertake series of operations such as initialization, eligibility stages, positive advertisement, negative advertisement and coalition that could lead to a better searching technique of the EA. The main purpose of EA is to pursue the candidates to converge to a global minimum solution (best solution) which believed to have shown vigorous mechanism in solving optimization tasks. EA is set to be different and advantageous compared to other metaheuristics because EA features to ease the performing neighbourhood movements in both continuous and discrete search space [73]. Optimization/search based on standard HNN has a high likelihood of becoming caught at suboptimal solution space as the numbers of neurons fired into the network increased [44], [64]-[74]. Metaheuristics algorithm such as EA has been utilized purposively in HNN to increase it the searching capability and to overcome the problem of premature convergence before reaching the global optimal, that would increase the number of satisfied clauses during the training phase of the network. EA utilized mechanism like positive campaign strategy, negative campaign, and a coalition to optimize the entire searching space due to its capacity to combined local search into a partitioned search space. Main steps of the procedure in HNN-RAN-*k*SAT-EA model considering $k \leq 3$ is presented from stage 1 to 5 as follows:

Stage 1: *Initialization*

The main goal of any search and optimization is to search for the best solution in terms of the variable's parameter of the problem. An array of vector parameter to be optimized is formed. An initial population of the size N_{POP} of individuals.

$$\tau_v = \left[\tau_i, \tau_2, \tau_3, ..., \tau_{N_{POP}}\right]^T \qquad (21)$$

Every solution is randomized within the variable boundaries range based on the following.

$$\tau_v = \lambda_v^{\min} + \tau_1 * \left(\lambda_v^{\max} - \lambda_v^{\min}\right) \qquad (22)$$

where $v \in \square$ and τ_v designates to the location of the *vth* voter in the N_{var} –dimensional optimization problem space and N_{POP} refers to the number of potential search agents. A uniformly distributed random is defined as $\tau_1 \in [0,1]$. This problem searches for the optimal RAN-*k*SAT clauses.

Step 2: *eligibility measurement*

Each individual's eligibility (fitness function) is measured according to the RAN-*k*SAT clauses in using as follows.

$$f_{\tau_v} = \sum_{i=1}^{t} C_i^{(3)} + \sum_{i=1}^{j} C_1^{(2)} + \sum_{i=1}^{d} C_i^{(1)} \qquad (23)$$

where f_{τ_v} denoted as eligibility of each person in search spaces τ_v, $C_i^{(k)}$ is the clause in $F_{RANkSAT}$ and $\forall t, j, d \in \square$ are the total number of $F_{RANkSAT}$ logical clauses.

Step 3: *Forming an initial population of individuals*

The EA uses a population of P_p *of* N_{pop} individuals search agent for the solution of the problems. Each solution represents candidate eligibility $\left(e_{\delta_C}\right)$, P_p represent search space. As part of the EA strategy, the population of individuals N_{pop} is divided into P_p parties and Each political parties comprises of a candidate $\left(\delta_c\right)$ and their supporters $\left(\tau_v\right)$ as search agents in the solution space. δ_c together τ_v form a P_P and divide τ_v among δ_c based on their eligibility $\left(e_v\right)$, in which the initial τ_v of a particular candidate is proportionate to e_{δ_C}. The number of individuals to serve as initial candidates δ_{C_i} was modelled according to the following

$$\delta_{C_i} = \alpha_r N_{pop} \qquad (24)$$

The initial number of supporters $\delta_{\tau_{v_j}}$ is calculated as follows.

$$\delta_{\tau_{v_j}} = N_{pop} - \delta_{C_i} \qquad (25)$$

Then, we randomly select $\delta_{\tau_{v_j}}$ of the supporters and give them to candidate δ_{C_i} to form political parties $\left(P_P\right)$ in the solution space.

Stage 4: *Campaign strategy:* this stage is modelled from step 1 to step 3 as follows

Step 1: Positive Campaign $\left(P_C\right)$

A voter position is selected in form of a variable of a party's candidate in the solution space to model a P_C. The purpose of sampling the random numbers is to pick the position of a voter to be replaced with a new voter. The selection rate procedure is $\lambda_\tau \in [0,1]$. The number of variables transferred to the elector by the applicant is defined in Equation (15) as follows.

$$\psi_\tau = \lambda_\tau S_C \qquad (26)$$

where the number of sampled variables to be substituted in the search space is denoted by ψ_τ and λ_τ represents the selection rate. The total number of candidates in the solution space vector variables outlined by S_C. Eligibility distance coefficient (e_d) has been utilized model the transfer of voters from one party to another represented as follows,

$$e_d = \frac{1}{dist\left(e_{\delta_{C_i}}, e_{\delta_{\tau_{vj}}}\right)+1} \tag{27}$$

where $e_{\delta_{C_i}}$ is used to define the eligibility of the candidate δ_{C_i} and $e_{\delta_{\tau_{vj}}}$ has been used to represent the eligibility of the voter $\delta_{\tau_{vi}}$ in the solution space. In EA, the advertising operator is applied, after selecting λ_τ and measure e_d, the vector values of the identified voter from the candidate and then multiplied with the coefficient e_d which will later be substituted with the identified voter. In other words, it τ_{iold} is the value for the voter selected before publicity progress, the updated value of the recognized voter a campaign was conducted as follows

$$\tau_{inew} = e_d * \tau_{iold} \tag{28}$$

We applied the model in Equation (28) to measure the influence of nearest voters by their associated candidate.

Step 2: Negative *Campaign* $\left(\tau_v{}^T\right)$

Contrast advertising is used in the execution of EA among various campaign strategies. The candidates are trying to fascinate supporters of other parties through their resistance campaign. This contributes to a resurgence and deterioration of success between the unpopular parties. EA algorithm uses negative campaign operation $\left(\tau_v{}^T\right)$ as a search mechanism as follows.

$$\tau_v{}^T = \begin{cases} \tau_{v^j}, r_j \le \psi \\ \tau_{v^j}, r_j > \psi, \end{cases} \quad \forall r_j \in [0, 1] \tag{29}$$

where ψ represents the negative campaign constant in EA. To pass through $\tau_v{}^T$ the layer. In EA, a population of solution space is randomly generated at the beginning of the optimization process which will be improved or changed completely by passing through the $\tau_v{}^T$ stage.

Step 3: Coalition strategy (C_L)

Confederates if they share the same ideas; EA, two or more parties may sometimes come together for a new party, having the same ideas and aims in space to find solutions. Some applicants are therefore going out of the advert with a new candidate called the "leader," the candidate who withdrawn from the election arena is called the "follower". EA uses coalition operation (C_L) to govern the optimal solution search in the search space. The coalition strategy can efficiently collect information regarding successful party merger through constructing trial vectors by using elements of established party candidates in the solution space by enhancing the solution search space.

$$\hat{\tau}_{v^i} = \tau_{v^1} + \lambda_\tau(\tau_{v^3} - \tau_{v^2}) \tag{30}$$

where $\hat{\tau}_{v^i}$ defined as a coalition parameter of political parties and $\lambda_\tau \in [0,1]$ served as a scaling factor and *i* is an index of current solution During the coalition process (C_L). In EA, a population of solution vectors is randomly created at the start. In the EA algorithm, each fresh solution achieved will compete with a united party in the search space. A party with the majority of the voters will be declared the winner.

Stage 5: *Stopping condition (Election Day)*

To update the population in the solution search space, three operators will be applied, these are positive and negative campaign as well as the coalition until the termination condition is fulfilled. A party candidate who wins the popular vote will be certified as the winner (optimal search)[72]. EA in searching for RAN*k*SAT logic program representation is expected to be an optimal one.

E. Experimental Setup for RAN-kSAT in HNN model

In this work, EA has been incorporated to enhance the learning capacity of HNN towards an optimal searching for RAN-*k*SAT logical clauses. The experimental results have been generated based on simulated datasets executed on RAN-*k*SAT logic considering the value of $k \le 2$ and $k \le 3$ in generating the program clause. The source code for EA in HNN for searching of RAN-*k*SAT logical clauses has been developed by authors based on C++ programming language and the graphs plotted using MATLAB 2018a. It was executed on a PC with Intel ® Celeron ® CPU T4800@ 4.2 GHz processor with 8GB RAM running on Windows 10.

F. Simulation Implementation Procedure

Implementation of Neuro-Heuristic searching method of RAN-*k*SAT in HANN. The program's main task is to find the best "model" that find the optimal occurrences of RAN-*k*SAT. Both logical variables and clauses were initially randomized. Simulations were executed by manipulating a different number of neurons complexity ranging from $10 \le NN \le 90$. The simulation has been conducted on RAN-*k*SAT as a logical clause in HNN according to the following steps;

i. Given a logic program:

$$F_2 \leftarrow F_2, \vee F_3,\ E_1 \leftarrow E_2, \leftarrow D$$

ii. Translate all the random-*k*SAT logical clauses into Boolean algebra form according to Equation (8):

$$F_{RAN3SAT} = (\neg F_1 \vee F_2 \vee F_3) \wedge (E_1 \vee \neg E_2) \wedge \neg D$$

$$F_{RAN-2SAT} = (\neg F_1 \vee F_2) \wedge (E_1 \vee \neg E_2) \wedge \neg D$$

iii. assign neurons to each logical variable in RAN-*k*SAT

iv. Randomize the state of the neurons and initialize all connection strengths zero.

$$M_{ijk}^{(3)} = M_{kji}^{(3)} = M_{ikj}^{(3)}, M_{ji}^{(2)} = M_{ij}^{(2)}, = M_{i}^{(1)} = M_{j}^{(1)} = M_{k}^{(1)} = 0$$

v. Derive the cost function, $E_{F_{RANkSAT}}$ for RAN*k*SAT using Equations (7) and (8).

vi. Equate the cost function in Equations (8) to energy dynamics in (10) to generate the values of the synaptic weight vector $M_{ijk}^{(3)}, M_{ji}^{(2)}, M_{i}^{(1)}, M_{j}^{(1)}$.

vii. Check clause satisfaction by applying EA, ES and ACO searching techniques that correspond to $E_{F_{RANkSAT}} = 0$. The satisfied assignment will be stored as CAM in HNN.

viii. Randomize the neurons configurations. Measure the respective local field $h_i(t)$ of the state space using Equation (9). it will be considered as a stable (converged) configuration if it stays the same after five cycles.

ix. Compute the corresponding final configuration of the network $H_{F_{RANkSAT}}$ by using the Lyapunov energy dynamics in Equation (10). Confirm if the final energy generated is a $H_{F_{RANkSAT}}^{\min}$ or $H_{F_{RANkSAT}}$ based on condition in Equation (11).

G. Statistical tests for model performance evaluation

The evaluation of performance is a key aspect of the design process of the HNN model. It is deemed that a sufficiently reliable estimate of accuracy and precision of predictions of a model is given by measurements made on the differences between the minimum energy reach and final energy attained, said "difference measures". Once the training process completed, the neural network calculated the values for the Global minimum ratio (Zm), Local minimum Ratio (Ym), Sum of square error (SSE), root means square error (RMSE) and the time complexity during training and or retrieval process for the network performance. The equation for these measures is presented as follows.

$$Zm = \frac{1}{ab}\sum_{i}^{t} H_{F_{RANkSAT}} \tag{31}$$

$$SSE = \sum_{i=1}^{d}\left(f_{NN} - h_d\right)^2 \tag{32}$$

$$RMSE = \sqrt{\sum_{i=1}^{n}\frac{f_{NN} - h_d}{n}} \tag{33}$$

where f_{NN} and h_d described the HNN the output and the target output values respectively, d defined as the number of the iterations in HNN.

3. SIMULATIONS RESULTS AND DISCUSSION

The simulations results presented here explored the performance of the proposed training approach using a different number of neurons from $4 \le NN \le 128$ and $10 \le NN \le 90$ in searching for RAN-2SAT and RAN-3SAT optimal representation respectively. Table 2 and Table 3 reported the global minimum ratio of RAN-2SAT and RAN-3SAT respectively. Figure 2 until Figure 4 displayed the performance of RAN-2SAT while Figure 5 until Figure 7 displayed the performance of RAN-3SAT in HNN.

The general trend of the model performance indicates a massive increase in errors and time consuming considering the complexity of the neuron fired to HNN in searching for both RAN-2SAT and RAN-3SAT logical clauses. The increasing trend in error accumulations reveals the complexity of the neuron states of RAN*k*SAT which proved to be an NP problem[28]. According to Table 2 and 3 based on the HNN searching during the learning, phase revealed that HNN-RAN-*k*SAT-EA and HNN-RAN-*k*SAT-ES were successfully arrived at $Zm = 1$ throughout the learning phase even when the number of neurons increased. However, HNN-RAN-*k*SAT-ES can only accommodate $4 \le NN \le 64$ and $10 \le NN \le 60$ in searching for optimal representation to RAN-2SAT and RAN-3SAT respectively. The SSE and RMSE in Figures 2 and 5 based on the HNN searching during the learning phase revealed that HNN-RAN-*k*SAT-EA and HNN-RAN-*k*SAT-ES were able to achieve $E_{F_{RANkSAT}} = 0$ with lower statistical errors accumulation than HNN-RAN-*k*SAT-ACO and HNN-RAN-*k*SAT-ES. This may be due to the multiple optimization layers possesses by EA which has a better screening stage in than other metaheuristics algorithm meaning that $E_{F_{RANkSAT}} = 0$ can converge in fewer iterations than HNN-RAN-*k*SAT-ACO and HNN-RAN-*k*SAT-ES. This explores the optimization capacity of EA in lowering the complexity of the network in reducing error accumulation by lowering the number of iterations in its optimization process.

TABLE II. ZM OF VARIOUS HNN MODEL FOR RAN-2SAT

NN	Global minimum ratio		
	EA	***ACO***	***ES***
4	1	1	1
8	1	1	1
16	1	1	1
32	1	0.9999	1
64	1	1	1
128	1	0.9891	-

TABLE III. ZM OF VARIOUS HNN MODEL FOR RAN-3SAT

NN	Global minimum ratio		
	EA	*ACO*	*ES*
10	1	1	1
20	1	1	1
30	1	1	1
40	1	0.9999	1
50	1	1	1
60	1	0.9997	1
70	1	1	-
80	1	1	-
90	1	0.9996	-

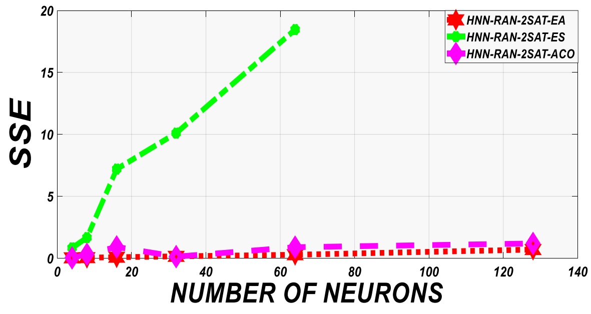

Figure 2. SSE of various HNN model for RAN-2SAT

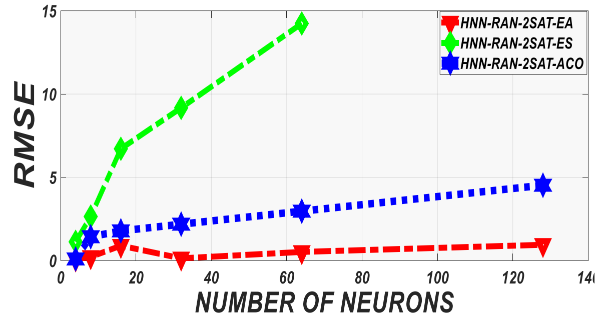

Figure 3. RMSE of various HNN model for RAN-2SAT

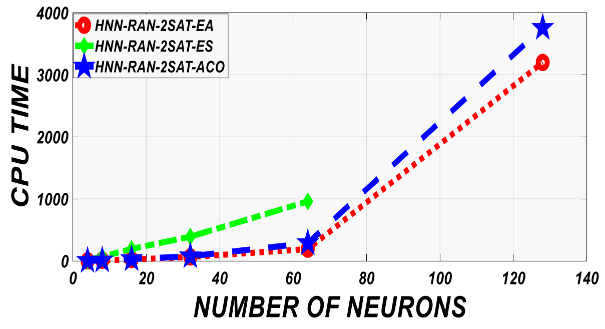

Figure 4. CPU TIME of various HNN model for RAN-2SAT

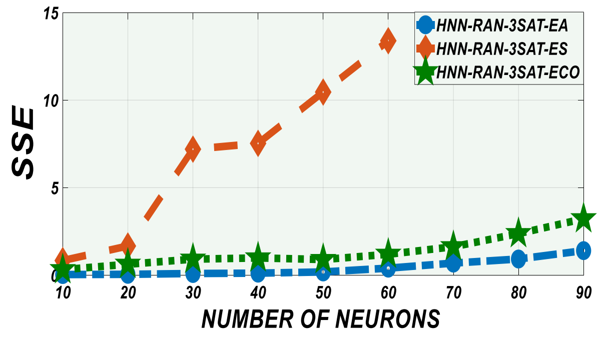

Figure 5. SSE of various HNN model for RAN-3SAT

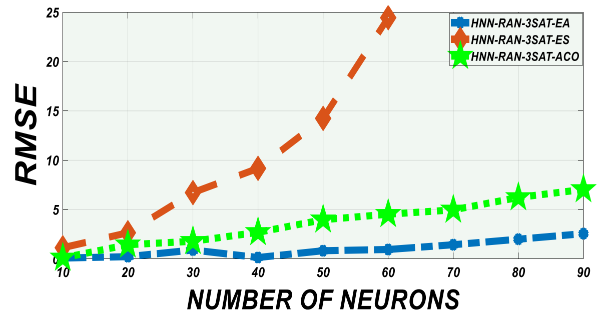

Figure 6. RMSE of various HNN model for RAN-3SAT

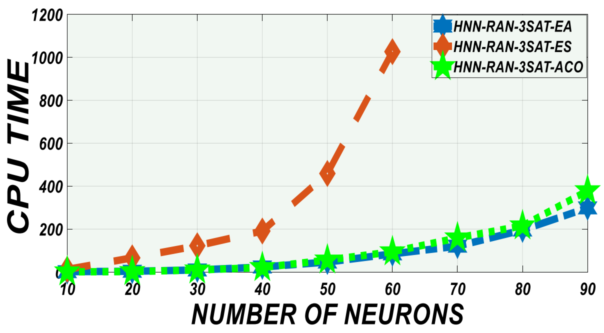

Figure 7. CPU TIME of various HNN model for RAN-3SAT

In Table 2 and 3, the performance of all HNN models in terms of the global minimum ratio achieved has been recorded. It can be noticed from the result displayed that the efficiency of the election algorithm (EA) in comparison with the Exhaustive search (EA) and Ants Colony Optimization (ACO) in enhancing the training capacity of Hopfield type of artificial neural network (HNN) have been explored for optimal searching of RAN*k*SAT logical leading to $E_{F_{RANkSAT}} = 0$. According to Table II-III,HNN-RAN-*k*-SAT-EA and ES-HNN-RAN-*k*SAT-ES have successfully retrieved all output accuracy that leads to $Zm = 1$ throughout the learning phase from $4 \leq NN \leq 128$ in RAN-2SAT and $10 \leq NN \leq 90$ in RAN-3SAT. HNN-RAN*k*SAT-ACO model some neural states trapped at a sub-optimal solution at $NN = 32, NN = 128$

in RAN-2SAT and $NN = 40, NN = 90$ in RAN-3SAT logical clauses still managed to achieve more than 97% success. Meanwhile, HNN-RAN-*k*SAT-ES could only withstand a maximum of $NN \le 60$ and $NN \le 64$ in RAN-3SAT and RAN-2SAT respectively, especially in the case of inconsistent output mapping $\neg F_{RANkSAT}$ as the neuron complexity becomes very high during the learning processing when the model crosses the execution time threshold. The main task of the metaheuristics such as EA to improve the flow of the learning phase of HNN by reducing the complexity of the neurons configuration so that the neurons can advance to the relaxation and restoration phase successfully.

The optimization capacity of HNN-RAN-*k*SAT-EA model to retrieve a more accurate final state that leads to $mR = 1$ (global minimum ratio) for both RAN-2SAT and RAN-3SAT logic. This indicates robustness and greater efficiency in neuro-searching incorporated by EA to enhance the HNN learning process for optimal RAN-*k*SAT logical representation. According to [64] if the global minimum ratio of the network is approaching one (1) at the end of the computational cycle, that means all solutions generated in the network have achieved global minimum energy(desired solution).

The Sum of square error (SSE) and root mean square error (RMSE) were portly in Figure 2 until Figure 3 and Figure 3 and 6 for RAN-2SAT and RAN-3SAT representation respectively. The general trend in performance based on SSE and RMSE metric. The errors accumulated during learning increase massively as the neurons out the weight $NN \le 40$ in RAN-2SAT and $NN \le 20$ in RAN-3SAT. The high accumulation of error was noticeable on HNN-RAN-*k*SAT-ES for both RAN-2SAT and RAN-3SAT. on the other hand, HNN-RAN-*k*SAT-EA accumulated lower error and achieve an accuracy of about 96% in both RAN-2SAT and RAN-3SAT logical representation. HNN-RAN*k*SAT-ECO displays good performance with low error accumulation at the initial optimization stage, as the number of neurons exceeds $NN \ge 20$ in RAN-3SAT and $NN \ge 40$ in RAN-2SAT the rapid increase in both SSE and RMSE errors were observed in the model. However, the performance of HNN-RAN-*k*SAT-ACO and HNN-RAN-*k*SAT-EA manifests a close margin. Generally, HNN-RAN-*k*SAT-ES has the worst performance by producing more than 45% searching failure.

The general trend of SSE and RMSE for HNN-RAN-*k*SAT optimization behaviour was reported to increase rapidly with neurons complexity. The proposed HNN-RAN-*k*SAT-EA was able to achieve $E_{F_{RANkSAT}} = 0$, with lower SSE and RMSE errors accumulation than HNN-RAN-*k*SAT-ACO and HNN-RAN-*k*SAT-ES. This is due to the multiple optimization layers possesses by EA in their optimization process that has a better screening stage in the solution space. This will enable the optimization process to converged to the desired solution in minimum iterations. According to error analysis in Figure 2,3,5 and 6, for both RAN-2SAT and RAN-3SAT logical clauses, it was reported that HNN-RAN-*k*SAT-EA recorded lower SSE and RMSE, compared to HNN-RAN-*k*SAT-ES and HNN-RAN-*k*SAT-ACO. This explored the robust capability of EA in lowering the sensitivity of the HNN towards the error iterations to the minimal level. Thus, the analysis of Zm, SSE, and RMSE suddenly stops $NN = 60$ for RAN-3SAT and $NN = 64$ for RAN-2SAT, this can be due to the ineffectiveness of the learning approach employed in HNN-RAN*k*SAT-ES that can withstand complexity. The solutions were struct at a sub-optimal solution (wrong pattern) as a result of too many oscillations by neurons. It is clear that HNN-RAN-*k*SAT-EA has an agreement with the existing HNN-RAN-*k*SAT-ACO but outperformed HNN-RAN*k*SAT-ES in term Zm, SSE, RMSE performance measures in both RAN-2SAT and RAN-3SAT logical representations.

Figure 4 and 7 displayed the behaviour of HNN-RAN-*k*SAT models in term of execution time during the implementation cycle. The proposed HNN-RAN-*k*SAT-EA can execute up to $NN = 90$ in 298.8 seconds which 80.01 seconds faster than HNN-RAN-3SAT-ACO and $NN = 128$ in 3198.7 seconds that is 558.3 seconds faster than HNN-RAN-2SAT-ACO. The conventional HNN-RAN*k*SAT-ES model could only withstand a maximum of $NN \le 60$ in 1026.53 seconds which is 979.93 seconds slower than HNN-RAN-3SAT-EA and 932.32 seconds slower than HNN-RAN-3SAT-ACO. Additionally, HNN-RAN-*k*SAT-ES model could only withstand a maximum of $NN = 64$ in 959.01 seconds which is 769.31 seconds slower than HNN-RAN-2SAT-EA and 672.01 seconds slower than HNN-RAN-2SAT-ACO. Looking at the CPU time behaviour in Figure 4 and 7, the HNN was stressful the optimization to the RAN*k*SAT logical clause which required additional time to achieve to achieve the desired result. HNN-RAN*k*SA-ES requires additional implementation time in searching from $30 \le NN \le 60$ in RAN-3SAT logic and $40 \le NN \le 60$ in RAN-2SAT logic. However, HNN-RAN-*k*SAT-EA and HNN-RAN-*k*SAT-ACO displayed closer execution time agreement from $10 \le NN \le 80$ for RAN-3SAT and $30 \le NN \le 128$ for RAN-2SAT logic. However, HNN-RAN-*k*SAT-EA was faster than HNN-RAN-*k*SAT-ACO at the initial and final states of the searching process. This is because further neurons are needed in the training process for HNN to move across the energy level to converge in optimal solutions. In other words, the accumulation of errors was less in HNN-RAN-kSAT-EA as the number of neurons increased which in turn reduced the CPU time. In this case, HNN-RAN-*k*SAT-ES required additional iterations to search for correct interpretation that leads to $E_{F_{RAN-kSAT}}=0$, which subsequently required additional CPU time.

The robustness of incorporating EA to facilitate the training process of HNN model can be seen in the perspective of the RAN-*k*SAT logical representation for both $k \leq 2$ and $k \leq 3$. The stochastic searching behaviour of EA diversifies the structure HNN during the learning phase for optimal RAN-*k*SAT representation. Thus, the structure of EA will indicate the diversification of the final neural states achieved by the EA-HNN-RAN*k*SAT. model. Hence, the dynamical swapping of the solutions occurs in EA-HNN-RAN*k*SAT, where the probability of attaining diversified $F_{RANkSAT}$ solutions is much higher. Hence, HNN-RAN*k*SAT-EA will generate more variation of $F_{RANkSAT}$ logical clauses that are feasible to achieve $E_{F_{RANkSAT}}=0$. On the other hand, the RAN*k*SAT logic will cause challenges in case of inconsistent assignment $\neg F_{RANkSAT}$ due to non-systematics behaviour displayed, this behaviour was observed in [74],[55],[59] and [54]. The hybridization of EA in the learning phase of HNN deals systematically with the higher learning complexity of in higher-order logic $F_{RAN-3SAT}$ as the number of neurons (NN) increased during the experiment simulation, the RAN-3SAT has successfully achieved closed 98% success as observed in Figure 5 and 6. The effectiveness of HNN-RAN-*k*SAT-EA in improving the non-fit neuron to achieve a global solution (100% success) is related to the robustness of the local and global search capacity employed by EA, which learning process in HNN. The local search capacity possesses in EA play a significant role at the early stage of the optimization process when the number of neurons (NN) is small. This explores better control parameters to enhance the learning process for optimal $F_{RANkSAT}$ logical representation. At the initial stage, the candidate selection requires an optimization operator which will accelerate the process of selecting the most eligible candidate to serve as a leader (solution).

Election algorithm (EA) possess multiple optimization layers which enable it to play around in diversifying the solution space to improve the non-fit solution in a particular region [72]. The first optimization layers of the Election algorithm (EA), which form the optimization among the candidate in a particular party. Following a positive advertisement stage is the negative advertisement stage which allows candidates from another party to arouse supporters from another party to their party to increase their popularity. The coalition layer plays a significant change in achieving the popular voters that correspond to the manifesto of the party representing the global solutions. This mechanism will form a collaborative candidate that shares similar fitness within a reasonable amount of time[73]. These features in the EA leads the hybrid model to lower the iterations an HNN required during the learning process by ensuring that a minimum error accumulated at the end of the experimentation. The systematic solution search space in EA facilitates the local search and global search process in achieving global solutions. The partition process of the search space allows the hybrid model to successfully search for the optimal solution in all described spaces. Generally, HNN-RAN-*k*SAT-EA model demonstrates faster computation time due to its campaign and coalition mechanism that systematically increase the chance of success of the united party in a reasonable amount of time.

4. Conclusion

In this work, a hybrid model was proposed EA algorithm has been incorporated with a Hopfield neural network in performing RAN*k*SAT representation as a new logical rule. From the results presented based on experimental simulations conducted, it can be conclusively proven that the proposed hybrid HNN-RAN*k*SAT-EA model is a robust heuristic technique that is successful in enhancing or pursuing desired assignment, even in the clauses of high complexity. This linked to the better optimization layers involve in the EA process that accelerates the learning process of HNN model in search of an optimal assignment for RAN-*k*SAT with greater eligibility. It was observed that the HNN-RAN-*k*SAT-EA model managed to complete the optimization process slightly faster than the existing model. However, all HNN models under study presented very good result in both RAN-2SAT and RAN-3SAT logical representation and compute the global solution leading to $E_{F_{RANkSAT}}=0$ within a feasible CPU timeframe. As a result, HNN-RAN-*k*SAT-EA exerted less computational pressure during the training process as compared to other models. This has been explored from the results reported in term of Zm, SSE RMSE and CPU time.

5. FUTURE DIRECTION

Our future research directions will focus on exploring another version of Boolean Satisfiability (B-SAT) or propositional satisfiability (SAT) problem such as Unrestricted satisfiability (SAT), Unambiguous-SAT, Exactly-1 3-satisfiability (Exact SAT), Not-all-equal 3-satisfiability, Linear SAT, MAJ-SAT, and other NP problems such as a reliability problem, Traveling Salesman problem, Knapsack, and Graph Coloring in the Hopfield type of artificial neural network (HNN). We will also explore other types of artificial neural networks such as Convolutional neural networks (CNN), Modular Neural Networks (MNN), Feedforward Neural networks (FNN), Radial basis function Neural networks (RBFNN), Kohonen Self Organizing Neural Network (KSONN), and many more. The learning and or training process will be enhanced to accelerate or speed the retrieval process using various type of metaheuristics approach such as Ant lion

Optimizer (ALO), Bat algorithm (BA), cat swarm optimization (CSO), Crow search algorithm (CSA), Differential Evolution (DE), Firefly algorithm (FFA), Genetic algorithm(GA), Dragonfly algorithm(DA), Particle swarm optimization(PSO) and other recent powerful algorithm be incorporated to enhance the computational phase of artificial neural network for optimal result. The research will further be extended to cater for reverse analysis (RA) which is involved in data mining techniques for real data sets.

References

[1] H. D. Majeed and R. M. Ibrahim, “Hybrid technique : text detection using a neural network and Boxes,” vol. 1, no. 1, 2021.

[2] A. Sharma, X. Liu, X. Yang, and D. Shi, “A patch-based convolutional neural network for remote sensing image classification,” *Neural Networks*, vol. 95, pp. 19–28, 2017.

[3] H. M. H. Mustafa, F. Ben Tourkia, and R. M. Ramadan, “An Overview on Evaluation of E-Learning/Training Response Time Considering Artificial Neural Networks Modeling,” *J. Educ. e-Learning Res.*, vol. 4, no. 2, pp. 46–62, 2017.

[4] M. S. Akhtar, A. Kumar, D. Ghosal, A. Ekbal, and P. Bhattacharyya, “A Multilayer perceptron based ensemble technique for fine-grained financial sentiment analysis,” 2017.

[5] H. C. Huang and S. K. Lin, “A Hybrid Metaheuristic Embedded System for Intelligent Vehicles Using Hypermutated Firefly Algorithm Optimized Radial Basis Function Neural Network,” *IEEE Trans. Ind. Informatics*, 2019.

[6] J. J. Hopfield, “Neural networks and physical systems with emergent collective computational abilities,” in *Feynman and Computation*, 2018.

[7] Y. Su, F. Tao, J. Jin, T. Wang, Q. Wang, and L. Wang, “Failure Prognosis of Complex Equipment With Multistream Deep Recurrent Neural Network,” *J. Comput. Inf. Sci. Eng.*, 2020.

[8] G. Bin Huang, Q. Y. Zhu, and C. K. Siew, “Extreme learning machine: Theory and applications,” *Neurocomputing*, 2006.

[9] T. V. Huynh, “Evaluation of Artificial Neural Network Architectures for Pattern Recognition on FPGA.,” *International Journal of Computing & Digital Systems*, 2017.

[10] S. Tausifa et. al., “Assessing the Efficacy of Logistic Regression, Multilayer Perceptron, and Convolutional Neural Network for Handwritten Digit Recognition,” *Int. J. Comput. Digit. Syst.*, 2020.

[11] T. J. Saleem and M. A. Chishti, “Assessing the efficacy of machine learning techniques for handwritten digit recognition,” *Int. J. Comput. Digit. Syst.*, 2020.

[12] O. Isa and R. Abdalhameed, “Automated Realtime Mask Availability Detection Using Neural Network,” 2020.

[13] S. Kaymak, A. Helwan, and D. Uzun, “Breast cancer image classification using artificial neural networks,” 2017.

[14] F. Bre, J. M. Gimenez, and V. D. Fachinotti, “Prediction of wind pressure coefficients on building surfaces using artificial neural networks,” *Energy Build.*, 2018.

[15] D. Zhao, Y. Dai, and Z. Zhang, “Computational intelligence in urban traffic signal control: A survey,” *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews*. 2012.

[16] A. Albu, R. E. Precup, and T. A. Teban, “Results and challenges of artificial neural networks used for decision-making and control in medical applications,” *Facta Univ. Ser. Mech. Eng.*, 2019.

[17] T. R. Cook, “Neural Networks,” in *Advanced Studies in Theoretical and Applied Econometrics*, 2020.

[18] P. M. Buscema, G. Massini, M. Breda, W. A. Lodwick, F. Newman, and M. Asadi-Zeydabadi, “Artificial neural networks,” in *Studies in Systems, Decision and Control*, 2018.

[19] J. Yang, L. Wang, Y. Wang, and T. Guo, “A novel memristive Hopfield neural network with application in associative memory,” *Neurocomputing*, 2017.

[20] K.-L. Du and M. N. S. Swamy, *Neural Networks and Statistical Learning*. 2019.

[21] V. N. Possani, A. Mishchenko, R. P. Ribas, and A. I. Reis, “Parallel Combinational Equivalence Checking,” *IEEE Trans. Comput. Des. Integr. Circuits Syst.*, vol. 39, no. 10, pp. 3081–3092, 2020.

[22] C. R. P. Tovar, E. Araujo, D. Carastan-Santos, D. C. Martins, and L. Rozante, “Finding attractors in biological models based on boolean dynamical systems using hitting set,” 2019.

[23] R. Santhanam, “Pseudorandomness and the Minimum Circuit Size Problem,” vol. 155, no. 155, pp. 1–29, 2019.

[24] T. Korhonen, J. Berg, and M. Järvisalo, “Enumerating potential maximal cliques via SAT and ASP,” 2019.

[25] E. Hebrard and G. Katsirelos, “Constraint and satisfiability reasoning for graph coloring,” *J. Artif. Intell. Res.*, 2020.

[26] Z. Ertem, E. Lykhovyd, Y. Wang, and S. Butenko, “The maximum independent union of cliques problem: complexity and exact approaches,” *J. Glob. Optim.*, 2020.

[27] G. P. Matos, L. M. Albino, R. L. Saldanha, and E. M. Morgado, “Solving periodic timetabling problems with SAT and machine learning,” *Public Transp.*, 2020.

[28] S. A. Cook, “The complexity of theorem-proving procedures,” 1971.

[29] X. Yin, B. Sedighi, M. Varga, M. Ercsey-Ravasz, Z. Toroczkai, and X. S. Hu, “Efficient analog circuits for boolean satisfiability,” *IEEE Trans. Very Large Scale Integr. Syst.*, 2018.

[30] M. Osama and A. Wijs, “Multiple Decision Making in Conflict-Driven Clause Learning,” *Proc. - Int. Conf. Tools with Artif. Intell. ICTAI*, vol. 2020-Novem, no. December, pp. 161–169, 2020.

[31] E. Demirović, N. Musliu, and F. Winter, “Modeling and solving staff scheduling with partial weighted maxSAT,” *Ann. Oper. Res.*, vol. 275, no. 1, pp. 79–99, 2019.

[32] A. Mahzoon, D. Grobe, and R. Drechsler, “Combining symbolic computer algebra and boolean satisfiability for automatic debugging and fixing of complex multipliers,” 2018.

[33] B. Olney and R. Karam, “Tunable FPGA bitstream obfuscation with boolean satisfiability attack countermeasure,” *ACM Trans. Des. Autom. Electron. Syst.*, 2020.

[34] A. Biere, “Bounded model checking,” *Front. Artif. Intell. Appl.*, 2009, doi: 10.3233/978-1-58603-929-5-457.

[35] S. Eggersglüß and R. Drechsler, *High quality test pattern generation and boolean satisfiability*. 2012.

[36] J. Rintanen, “Planning as satisfiability: Heuristics,” *Artif. Intell.*, vol. 193, pp. 45–86, 2012.

[37] Q. Zhou, J. Arulraj, S. Navathe, W. Harris, and D. Xu, “Automated verification of query equivalence using satisfiability modulo theories,” 2018.

[38] J. Chen and F. He, “Control flow-guided SMT solving for program verification,” 2018.

[39] N. Khattar, J. Sidhu, and J. Singh, “Toward energy-efficient cloud computing: a survey of dynamic power management and heuristics-based optimization techniques,” *J. Supercomput.*, 2019.

[40] W. A. T. W. Abdullah, “Logic programming on a neural network,” *Int. J. Intell. Syst.*, 1992.

[41] S. Sathasivam, “First order logic in neuro-symbolic integration,” *Far East J. Math. Sci.*, 2012.

[42] S. Alzaeemi, M. A. Mansor, M. S. Mohd Kasihmuddin, S. Sathasivam, and M. Mamat, “Radial basis function neural network for 2 atisfiability programming,” *Indones. J. Electr. Eng. Cmput. Sci.*, 2019.

[43] S. A. Alzaeemi, S. Sathasivam, M. Shareduwan, and M. Kasihmuddin, “MAXIMUM 2-SATISFIABILITY IN RADIAL BASIS FUNCTION,” vol. 16, no. November 2019, pp. 107–115, 2020.

[44] H. Abubakar, S. Yusuf, and S. A. Masanawa, "Exploring the Feasibility of Integrating Random k-Satisfiability in Hopfield Neural Network," 2020. [Online]. Available: www.ModernScientificPress.com/Journals/ijmms.aspx.

[45] H. Abubakar, S. A. Masanawa, S. Yusuf, and Y. Abdurrahman, "Agent Based Computational Modelling For Mapping Of Exact Ksatisfiability Representation In Hopfiedl Neural Network Model," *Int. J. Sci. tech. Res.*, vol. 9, no. 08, pp. 76–85, 2020.

[46] H. Abubakar, S. A. M, S. Yusuf, and Y. Abdurrahman, "Discrete Artificial Dragonflies Algorithm in Agent Based Modelling for Exact Boolean k Satisfiability Problem," vol. 35, no. 4, pp. 115–134, 2020.

[47] G. Joya, M. A. Atencia, and F. Sandoval, "Hopfield neural networks for optimization: Study of the different dynamics," *Neurocomputing*, 2002.

[48] F. H. F. Leung, H. K. Lam, S. H. Ling, and P. K. S. Tam, "Tuning of the structure and parameters of a neural network using an improved genetic algorithm," *IEEE Trans. Neural Networks*, 2003.

[49] S. Hanandeh, A. Ardah, and M. Abu-Farsakh, "Using artificial neural network and genetics algorithm to estimate the resilient modulus for stabilized subgrade and propose new empirical formula," *Transp. Geotech.*, 2020.

[50] G. Li, H. Ke, C. Li, and B. Li, "Thermal Error Modeling of Feed Axis in Machine Tools Using Particle Swarm Optimization-Based Generalized Regression Neural Network," *J. Comput. Inf. Sci. Eng.*, 2020.

[51] M. F. Adak, P. Lieberzeit, P. Jarujamrus, and N. Yumusak, "Classification of alcohols obtained by QCM sensors with different characteristics using ABC based neural network," *Eng. Sci. Technol. an Int. J.*, 2020.

[52] J. Kumar and A. K. Singh, "Workload prediction in cloud using artificial neural network and adaptive differential evolution," *Futur. Gener. Comput. Syst.*, 2018.

[53] N. E. Zamri, A. Alway, M. A. Mansor, M. S. M. Kasihmuddin, and S. Sathasivam, "Modified imperialistic competitive algorithm in hopfield neural network for boolean three satisfiability logic mining," *Pertanika J. Sci. Technol.*, 2020.

[54] S. Sathasivam, M. A. Mansor, M. S. M. Kasihmuddin, and H. Abubakar, "Election Algorithm for Random k Satisfiability in the Hopfield Neural Network," *Processes*, 2020.

[55] H. Abubakar, S. Rijal, M. Sabri, S. A. Masanawa, and S. Yusuf, "Modified election algorithm in hopfield neural network for optimal random k satisfiability representation," *Int. J. Simul. Multidisci. Des.Optim.*, vol. 16, no. 11, pp. 1–13, 2020.

[56] H. F. Sianipar, N. N. Y. Zaini, M. S. M. Kasihmuddin, M. A. Mansor, and S. Sathasivam, "Hybrid ant colony optimization for even-2 satisfiability programming in Hopfield neural network," 2020,,

[57] S. Sathasivam, M. Mamat, M. S. M. Kasihmuddin, and M. A. Mansor, "Metaheuristics approach for maximum k satisfiability in restricted neural symbolic integration," *Pertanika J. Sci. Technol.*, vol. 28, no. 2, pp. 545–564, 2020.

[58] S. Sathasivam, "Mean-Field Theory in Hopfield Neural Network for Doing 2 Satisfiability Logic Programming," no. August, pp. 27–39, 2020.

[59] H. Abubakar and S. Sathasivam, "Developing random satisfiability logic programming in Hopfield neural network," 2020.

[60] V. Feldman, W. Perkins, and S. Vempala, "On the complexity of random satisfiability problems with planted solutions," *SIAM J. Comput.*, 2018.

[61] M. Mézard, G. Parisi, and R. Zecchina, "Analytic and algorithmic solution of random satisfiability problems," *Science (80-.).*, 2002.

[62] M. L. Erana-Diaz, M. A. Cruz-Chavez, R. Rivera-Lopez, B. Martinez-Bahena, E. Y. Avila-Melgar, and M. Heriberto Cruz-Rosales, "Optimization for Risk Decision-Making through Simulated Annealing," *IEEE Access*, vol. 8, no. June, pp. 117063–117079, 2020.

[63] S. A. Cook, "The Complexity of Theorem-Proving Procedures Stephen A. Cook University of Toronto," *Proc. third Annu. ACM Symp. Theory Comput.*, pp. 151–158, 1971.

[64] S. Saratha, "Upgrading logic programming in hopfield network," *Sains Malaysiana*, 2010.

[65] P. Jonsson, V. Lagerkvist, G. Nordh, and B. Zanuttini, "Complexity of SAT problems, clone theory and the exponential time hypothesis," 2013.

[66] S. R. B. Bearden, Y. R. Pei, and M. Di Ventra, "Efficient solution of Boolean satisfiability problems with digital memcomputing," *Sci. Rep.*, 2020.

[67] M. E. Valle and F. Z. De Castro, "On the Dynamics of Hopfield Neural Networks on Unit Quaternions," *IEEE Trans. Neural Networks Learn. Syst.*, 2018.

[68] E. Cabrera and H. Sossa, "Generating exponentially stable states for a Hopfield Neural Network," *Neurocomputing*, 2018.

[69] R. Chaudhuri and I. Fiete, "Bipartite expander Hopfield networks as self-decoding high-capacity error-correcting codes," 2019.

[70] G. Gosti, V. Folli, M. Leonetti, and G. Ruocco, "Beyond the maximum storage capacity limit in Hopfield recurrent neural networks," *Entropy*, 2019.

[71] F. Z. de Castro and M. E. Valle, "A broad class of discrete-time hypercomplex-valued Hopfield neural networks," *Neural Networks*, 2020.

[72] H. Emami and F. Derakhshan, "Election algorithm: A new socio-politically inspired strategy," *AI Commun.*, 2015.

[73] H. Emami, "Chaotic election algorithm," *Comput. Informatics*, 2019.

[74] H. Abubakar, S. Abdu Masanawa, and S. Yusuf , "Neuro-Symbolic Integration of Hopfield Neural Network for Optimal Maximum Random kSatisfiability (Maxrksat) Representation," *J. Reliab. Stat. Stud.*, 2020.

Hamza Abubakar received both his B.Sc. (Mathematics) and MSc (Financial Mathematics) from the University of Abuja, Nigeria. He is currently pursuing a PhD degree in Mathematical Science at the University of Sciences Malaysia. Hamza joint the service of Isa Kaita College of Education, Dutsin-ma, Katsina, Nigeria in 2008 and promoted the rank Senior Lecturer. He is an active member of the Nigerian Mathematical Society, Mathematical Association of Nigeria, Science Teachers Association of Nigeria and International Association of Engineers (OR and AI). His research interest includes Financial Mathematics, neural network modelling and metaheuristics optimization.

Mohammed Lawal Danrimi received PhD at the Universiti Malaya, Malaysian. He is a Lecturer with Umaru Musa Yar'adua University, Katsina. His research interest focused on Accounting and financial modelling, research Methodology and Optimization problem in Finance.