Published in AJSE, Vol: 24, Issue:3
Received on 14th October 2025
Revised on 23rd May 2026
Published on 31th July 2026

AIUB JOURNAL OF SCIENCE AND ENGINEERING
ISSN: 1608 - 3679 (print) 2520 - 4890 (Online)
https://doi.org/10.53799/y7wvjq52

Optimizing Quadratic Satisfiability with Special Symbolic Logic in a Discrete Hopfield Neural Network Using Swarm Intelligence

Syed Anayet Karim, Sowmitra Das, Akib Jayed Islam*, Md Rabiol Amin, Mamun-Or-Rashid

Abstract— **Conventional second-order logic satisfiability with Discrete Hopfield Neural Network has major drawbacks in obtaining potential solutions. This causes problems in a network because suboptimal final neuron states will affect any optimization problems. To overcome such issues, special types of symbolic logic, such as Fuzzy logic, can be impactful, which has emerged as a valuable tool for various applications, including engineering system control and neural networks. In this article, the authors propose a novel hybrid model that combines fuzzy logic with a satisfiability-based discrete Hopfield neural network and a swarm-type metaheuristic algorithm. The proposed method ensembles Fuzzy logic with Quadratic Satisfiability to construct the bipolar structure, while the Artificial Bee Colony algorithm, a swarm-based optimization technique, is applied to optimize the solution. The hybrid model is benchmarked against the existing second-order satisfiability-based Discrete Hopfield Neural Networks and conventional Fuzzy logic-based Discrete Hopfield Neural Networks. Simulation results show that the proposed hybrid model outperforms existing models by 85%-95% in terms of training-testing error analysis, energy efficiency, neuron variation, and computational time.**

Index Terms— **Quadratic Satisfiability, Fuzzy logic, Artificial Bee Colony algorithm, Discrete Hopfield Neural Network**

Syed Anayet Karim is an Assistant Professor in the Department of Mathematics, Bangladesh Maritime University, Dhaka-1216, Bangladesh. E-mail: anayetkarim@portcity.edu.bd, s.anayet01@gmail.com,
ORCID ID: orcid.org/0000-0003-1659-9233

Sowmitra Das is with the Department of Computer Science and Engineering, Port City International University, Chattogram 4225, Bangladesh (e-mail: sowmitra@portcity.edu.bd).

*Akib Jayed Islam (corresponding author) is an Assistant Professor in the Department of Electrical and Electronic Engineering, Faculty of Science and Engineering, Port City International University, Chattogram 4225, Bangladesh. E-mail: akibjayed@portcity.edu.bd, akibjayedislam@gmail.com.
ORCID ID: orcid.org/0000-0003-3187-7035

Md. Rabiol Amin is with the Department of Computer Science and Engineering, CCN University of Science and Technology, Cumilla, Bangladesh (e-mail: rabiol.amin@gmail.com).

Mamun-Or-Rashid is with the Department of Mathematics, Premier University, Chattogram, Bangladesh (e-mail: mamun.math@puc.ac.bd).

I. INTRODUCTION

THE rise of intelligent systems has significantly reshaped the revolution in researchers groundwork and dimensions. These intelligent systems frequently use AI to improve decision-making and find important knowledge via the problem patterns. One of the most promising approaches in AI today is the neuro-symbolic model—a hybrid that blends the structured reasoning of symbolic logic with the learning power of artificial neural networks (ANNs) [1]. By guiding neural networks with formal rules and language, this transformed AI make it more transparent and understandable, addressing the "black box" nature of many traditional ANNs [2,3]. ANNs' basic formations, inherently mathematical and graphical structures, have proven to be highly effective in addressing a wide range of optimization, prediction, and forecasting challenges over the last two decades [4, 5, 6]. Despite their potential, integrating different ANN models into real-world intelligent systems still presents both exciting opportunities and ongoing challenges. Among the various ANN models, the Hopfield Neural Network is particularly notable for its efficacy as a feedback neural network [7]. This Hopfield network is uniformly functional in discrete form and can perform with high volume that is able to solve various combinatorial and mathematical problems [8],[9]. Discrete Hopfield Neural Network (DHNN) has become a foundational model for associative memory which can store and retrieve patterns through energy minimization principles [10], [11].

Now-a-days, logical rules can stand in symbolic form and the logical rules can be used to govern the connection of neurons [12]. In 1992, Wan Abdullah [13] introduced a vibrant concept to illustrate neuron management through the symbolic form that is propositional satisfiability (SAT) which embedded as a learning method for the DHNN. This concept clearly enables symbolic formation of logical rule. These neurons have similar property of conventional ANN where the aim of DHNN is to minimize the cost function that relates to the SAT logical rule [14]. This innovation opens up new approach for research within the field of ANNs. Having the black-box nature of DHNN, the exploration of the search space through neurons remains an area of ongoing investigation.

Afterwards, researchers have employed a variety of logical frameworks, including systematic logic, non-systematic logic, and hybrid combinations, to navigate this complex landscape. With a focus on logical variety, several researchers- Kashimuddin *et al*. [15], Mansor *et al*. [16], Sathasivam *et al*. [17], Alway *et al.* [18], and Chen *et al.* [19]

introduced different concepts of logical structures based different algorithms to illustrate the behaviour of the neurons through the network. Though these existing literatures emphasizes the systematic use of SAT concept in DHNN that enlarged the research scope for the researchers but the rigidity of logical structure sometimes created overfitting results for DHNN. Moreover, in several cases, for the higher number of neurons the attainment of global solutions is poor. Additionally, it is challenging to deal with real-world problems because the traditional SAT structure is limited to the binary classification of variables. A more flexible method that can deal with incomplete information and partial truth is essential in order to improve DHNN's ability to avoid local minima.

Another special type of symbolic logical approach that can be impactful for illustrating a neuron's behavior is Fuzzy logic. Generally, the word 'fuzzy' tends not to be adequate where fuzzy logic generates a pre-defined set of rules that evaluates risk and benefits to optimize a decision-making problem [20],[21]. Fuzzy logic utilizes a set of rules through fuzzification and defuzzification processes. One of the creative works by [22] introduced a new concept in logic and neural networks. This research intermingled Horn Satisfiability (HornSAT) with fuzzy logic in the DHNN to represent information in a more comprehensive way with effective logical rule. The combinatorial optimization problems in Hopfield networks can be resolved through the combination of HornSAT and fuzzy logic. Azizan *et al.* [23] extended this study in another way by involving the higher-order Satisfiability (SAT) concept. The authors introduced an enhanced approach to 3SAT that facilitates increased energy relaxation and circumvents local minima by integrating Hopfield networks with fuzzy logic. The precise melting fuzzy-NN concept further inspired the authors [24] to do more research in this DHNN domain. This work encompasses the complexity of DHNN that can be remedial by the inclusion of metaheuristics in the training phase of DHNN and the symbolic fuzzy-SAT structure, which accommodates a more stable final neuron state. Furthermore, Feng *et al.* [25] utilized 2nd order fuzzy-SAT concept with a crow search algorithm to strengthen the DHNN's training phase. This work revealed a dynamical clustering approach of fuzzy concepts that illustrates a generic behaviour of DHNN. Prior research has not thoroughly investigated the application of DHNN and fuzzy logic in structuring 2-SAT logic; instead, it has concentrated on fuzzifying the DHNN's input neurons while ignoring the effects on the knowledge-based data structure and quality.

In this proposed study, the performance of the swarm-type algorithm with DHNN will be evaluated by using simulated data for quadratic SAT, which it also combines the strengths of fuzzy logic with the optimization capabilities to find near-optimal solutions. This pioneering approach adeptly balances exploration and exploitation strategies, resulting in improved problem-solving efficacy.

The novelty of this proposed work lies in the formulation of a logical rule- Quadratic Satisfiability, which uniquely integrates systematic representations within a unified framework while enabling dynamic regulation of the distribution of positive/negative literals through second-order clauses. The dynamic mechanism is achieved by combining the fuzzy concept to adaptively regulate the distribution of the literals. Additionally, the incorporation of the Binary Artificial Bee Colony algorithm for optimizing logic generation further enhances the diversity and adaptability of neuron states in DHNN. This results in improved convergence, reduced overfitting, and increased solution diversity, which shows great potential in creating a vast solution space in a broader concept of data mining for extracting interpretable patterns from real-world datasets. To the best of our knowledge, this is one of the better approaches to systematically co-optimize both logic structure (quadratic and fuzzy) in SAT regulations, along with an effective swarm-intelligence type metaheuristics in modelling the DHNN. Here, the key contributions of this proposed research encompass:

- To introduce an effective search space paradigm by employing fuzzy logic principles within the context of Quadratic Satisfiability.
- To implement a swarm-based Artificial Bee Colony algorithm that exhibits a balanced exploration-exploitation strategy during the training phase of the Discrete Hopfield Neural Network.
- To evaluate the effectiveness of the proposed model, a thorough assessment was carried out using a variety of performance metrics. These metrics included Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), Mean Absolute Percentage Error (MAPE), Similarity Index, Energy Analysis, Total Neuron Variations, and Computational time.

This article is organized as follows: Section II offers an extensive examination of Quadratic Satisfiability, Discrete Hopfield Neural Networks, Fuzzy Logic, and the Artificial Bee Colony algorithm. Section III explores the experimental setup and the performance metrics utilized in the study. Section IV details the results obtained and discusses their significance. Lastly, Section V provides concluding remarks that encapsulate the main findings and suggest possible aspects for future research.

II. MATERIALS AND METHODS

A. Quadratic Satisfiability

Quadratic Satisfiability is the framework of the Boolean formula that is composed of 2 literals per clause that either result in TRUE or FALSE. In this context, the Quadratic Satisfiability is also known as Q2- Satisfiability (Q2-SAT). In this article, the systematic form of the SAT will be formulated. Thus, for any Quadratic order Boolean logic with the condition of the number of variables, the formation can be reduced to 2-SAT. The general properties of Quadratic logic or Boolean Q2-SAT are as follows:

1. The Quadratic SAT formula is composed of n variables, $a_1, a_2, ...a_n \in [1,-1]$ entranced each clause.

Here, $n=2$, This quadratic SAT will consist of 2 literals per clause.

2. A set of L clauses connected by AND $(\wedge)$ operator in a 2-SAT formula as follows: $\exists L : F = C_1 \wedge C_2 \wedge .... \wedge C_L$.
3. A set of $\alpha_{L,M}$ literals and each clause C_L, $\forall 1 \le L \le M, C_L = (\alpha_{(L,1)} \vee \alpha_{(L,2)})$ which consists of several literals connected by the OR $(\vee)$ operator.

The Quadratic or Q2-SAT formula is specified in a product of sums or Conjunctive Normal Form (CNF). A standard example of the 2-SAT formula is given in Equation (1).

$$P_{2-SAT}^{Q} = (A_1 \vee \neg A_2) \wedge (A_3 \vee A_4) \wedge (\neg A_5 \vee A_6) \tag{1}$$

Here, Equation (1) shows a Quadratic logic with 6 literals and 3 clauses. Each of the literals in Equation (1) takes the binary values -1 and 1, which represent a neuron in DHNN. This Equation (1) will be satisfiable and TRUE if the value of the literals represents as $(A_1, A_2, A_3, A_4, A_5, A_6) = (1,-1,1,1,-1,1)$. The key task for getting a satisfied interpretation of this logic by compute the respective synaptic weights according to the Boolean logical inconsistencies. The key justification for utilizing Quadratic Satisfiability in this study is that this systematic SAT makes it easy to calculate the cost function by which the logical inconsistency can be minimized, and also converts it into a minimization problem that aligns with the DHNN energy function.

B. Discrete Hopfield Neural Network

In recent years, research on Discrete Hopfield Neural Networks (DHNNs) has experienced substantial growth [26, 27]. The concept of feedback networks was popularized by John Hopfield, who introduced the notion of an associative memory system referred to as Content Addressable Memory (CAM) [28]. One of the primary advantages of DHNNs lies in their capacity to systematically store patterns while minimizing energy consumption. The architecture of DHNNs consists of a two-dimensional neural network characterized by interconnected neurons. These neurons typically undergo asynchronous updates until the network attains a stable equilibrium state, thereby effectively addressing optimization problems. In DHNNs, the neuron units are represented by either bipolar or binary values. Equation (2) delineates the general formulation of the updating rule for the i^{th} neuron within a DHNN.

$$S_i = \begin{cases} 1, \ if \sum_{j}^{n} W_{ab}^{(2)} S_b \ge U_p \\ -1, \ otherwise \end{cases} \tag{2}$$

Here, Equation (2) specifies the synaptic weight between units a and b as $W_{ab}^{(2)}$. The present state of unit b is represented by S_b, while U_p denotes the predefined threshold. Numerous studies have explained that configuring U_p to 0 guarantees a monotonic reduction in energy within the DHNN [29]. The synaptic weight W_{ab} between neurons a and b reflects the strength of the connection between these two neuronal units. Similarly, the neuron connections are attempted $W_{ab}^{(2)}$ as $W_{abc}^{(3)} = \left[W_{abc}^{(3)}\right]_{n\times n}$ with $\left[U_p\right]_{n\times 1} = \left[U_1, U_2, U_3, ..., U_n\right]^T$.

The precise calculation of the cost function in DHNN is crucial for alleviating logical inconsistencies, particularly the condition $C_P = 0$. Equations (3) and (4) provide the formulations for the cost function- C_P incorporating all potential logical combinations.

$$C_P = \frac{1}{4}\sum_{a=1}^{v}\left(\prod_{j=1}^{2} L_{ab}\right) + \frac{1}{2}\sum_{b=1}^{u}\left(\prod_{j=1}^{1} L_{ab}\right) \tag{3}$$

$$L_{ij} = \begin{cases} \frac{1}{2}(1 - S_{i_1}), if \neg i_1 \\ \frac{1}{2}(1 + S_{i_1}), otherwise \end{cases} \tag{4}$$

Where S_{i_1} is the neuron state where $i_1 \in \{-1,1\}$. Then the implementation of DHNN-Q2SAT will enter as the clauses rigid with 2 literals. During the testing phase, the local field calculation of DHNN-Q2SAT can be calculated as Equation (5).

$$h_{Q-2SAT} = \sum_{b=1, a\ne b}^{n} W_{ab} S_a S_b - W_a \tag{5}$$

The synaptic weight of Equation (5) is symmetric $W_{ab} = W_{ba}$ and zero diagonal $W_{aa} = W_{bb} = 0$. According to several research studies [30],[31], after the local field evaluation, the final neuron's state is calculated through the Lyapunov energy function, which is written in Equation (6).

$$H_{Q-2SAT} = -\frac{1}{2}\sum_{a=1}^{n}\sum_{b=1}^{n} W_{ab} S_a S_b - \sum_{a=1}^{n} W_a S_a \tag{6}$$

A notable characteristic of Equation (6) is its inherent monotonic decrease in energy, guaranteeing that the energy value associated with $Q-2SAT$ consistently declines. The parameter $H_{Q-2SAT}^{\min}$ represents the minimum energy state attained by the system [32]. Therefore, the quality of the final neuron state can be evaluated using Equation (7).

$$| H_{Q-2SAT} - H_{Q-2SAT}^{\min} | \le \lambda \tag{7}$$

Where $\lambda = 0.001$ is the tolerance value, which is determined by the user [33].

C. Fuzzy Logic

Fuzzy logic, a paradigm for approximate reasoning, has garnered increasing attention across diverse research fields

[34, 35]. This fuzzy logic system has been expanded by the concept of partial falsity, meaning that its assertions can take truth values that fall between those that are entirely true and those that are entirely false. Its foundation lies in fuzzy set theory, which serves as an extension of traditional set theory. Fuzzy logic, a type of multi-valued logic, extends beyond the binary classifications of traditional Boolean logic by allowing for intermediate values [24]. This flexibility proves advantageous in addressing complex problems that may not be easily resolved using Boolean logic. Zadeh [36] pioneered the development of fuzzy logic and fuzzy sets to address ambiguity and improve human reasoning capabilities.

Meanwhile, Authors [37] explained that fuzzy sets, as linear combinations of various fuzzy basis functions, are capable of approximating continuous linear functions. A fuzzy set can be described as: consider B as the universe of discourse and $b \in B$ then a fuzzy set F on B is a collection of ordered pairs:

$$F = \{b, \mu_F(b); b \in B\} \tag{8}$$

Where b is a crisp value and $\mu_F(b)$ is the membership value of y. A membership, or grade, is a function that quantifies the degree of membership for each b in the universe of discourse, and it is defined as:

$$\mu_F : B \to [0,1] \tag{9}$$

From Equation (9), $\mu_F(b)$ can take any real value between 0 and 1, where 1 and 0 denote full membership and no membership, respectively. Notably, different forms of membership can be defined, whereas the choice of membership function depends largely on the nature of the problem task and implementations. The fuzzy logic systems are composed of some basic components, which are fuzzification, rule evaluation, aggregation and defuzzification. With these components, the common fuzzy logic can be represented by Equation (10):

$$F = \{(b1, \mu_F(b1)), (b2, \mu_F(b2)), (b3, \mu_F(b3)), ...; b1, b2, b3... \in F\} \tag{10}$$

Mamdani [38] effectively implemented the principles of fuzzy logic systems in the domain of control theory, resulting in notable achievements. Moreover, the two most important stages in a fuzzy system are fuzzification and defuzzification. The fuzzification phase uses identities to adjust and capture the underlying ambiguity in the input data, while the membership function converts exact or sharp input values into fuzzy sets. Fuzzy output or fuzzy variables are mapped to clear, precise information that is simple to analyze or comprehend during the defuzzification phase. In order to generate an exact value based on the inference rule or results, the stage may also entail aggregating fuzzy output. In a nutshell, the fuzzy logic system, as depicted in Figure 1, includes a fuzzification stage that converts precise inputs into fuzzy sets. Subsequently, a rule-based process converts these fuzzy sets into their corresponding equivalents. Finally, the defuzzification stage outputs a crisp evaluation derived from the fuzzy sets.

D.Artificial Bee Colony (ABC) algorithm

This study presents the Artificial Bee Colony (ABC) algorithm, a swarm optimization method recognized for its capacity to identify near-optimal solutions. Modeled after the foraging behavior of honey bees, the ABC algorithm begins with a randomly initialized population of bees distributed throughout the search space. The algorithm is structured into three optimization layers: employed bees, onlooker bees, and scout bees, which collectively navigate the search space in pursuit of global solutions [39]. Prior work by Sidik *et al.* [40] has shown the efficacy of the ABC algorithm in optimizing the logical phase of weighted satisfiability representation within DHNN. In this research, the ABC algorithm will be utilized during the training phase of the DHNN to enhance the optimization of solutions. During training, the fittest bee is determined as the one exhibiting the highest fitness value, with the fitness function for the ABC algorithm articulated as follows:

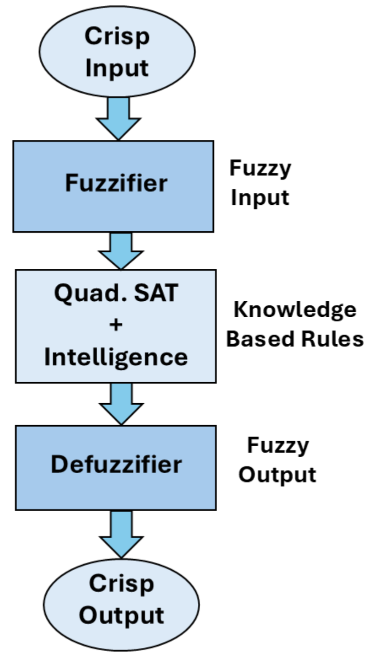

Fig. 1 Process flow of Fuzzy Logic System

$$f_{ABC} = \sum_{i=0}^{m} C_{(k)}^{i} \tag{11}$$

Where $C_{(k)}^{i}$ are the order of the clauses and this is arranged as

$$C_{(k)}^{i} = \begin{cases} 1, Satisfied \\ 0, Otherwise \end{cases} \tag{12}$$

Meanwhile, $C_{(k)}^{i} = 1$ means the clause arrangements have consistent interpretations. Since ABC is not required to have complex fitness functions so the WA method benefits the fitness evaluation function. Inspired by the binary ABC proposed by [39], the total structure of the ABC Algorithm for *k*SAT in DHNN is depicted through different stages.

Stage 1: Initialization and Fitness Evaluation

The ABC algorithm was initialized with a population comprising 50 employed bees, 50 onlooker bees, and 1 scout bee. Each bee was represented by a bipolar string consisting of 1s and -1s. The fitness of each employed bee was assessed using Equations (11) and (12), which were predicated on the number of satisfied clauses in the target logic. The initial food source was randomly generated within the specified parameter bounds. The overall population consisted of SN bees, characterized by D optimization parameters. Equation (13) delineates the initialization process for the random food source $x_{i,j}$.

$$x^{i,j} = x_{\min}^{j} + rand(0,1)(x_{\max}^{j} - x_{\min}^{j}) \tag{13}$$

Stage 2: Employed bee stage

In this stage, the employed bee will identify a new food position for $V_{employed}^{ij}$. The location of food is given as follows in Equation (14).

$$V_{employed}^{ij} = y^{ij} \vee (\phi^{ij} \otimes (y^{ij} \wedge y^{kj})) \tag{14}$$

Where y^{ij} is the initial stage for food source, y^{kj} food source that was observed and φ^{ij} defined via Equation (15).

$$\varphi^{ij} = \begin{cases} 1, rand(0,1) < 0.5 \\ -1, rand(0,1) \geq 0.5 \end{cases} \tag{15}$$

Algorithm 1: The Pseudo-code for the Artificial Bee Colony algorithm

Input: Set Population size, Each bee size, Generation, trial.
Output: The final neuron state
Generate the Population size consisting $S_i \in \{S_1, S_2, ...S_{SN}\}$;
Initialize $trial_i = 0$ for each bee;
while $g \leq Generation$ **Do**
Calculate the fitness of each bee and evaluate them by using Equation (11)

[Employed Bee Phase]

for $i \in \{1,2,3,...M_{Pop}\}$ **Do**
Produce new food source by using Equation (14)
Evaluate the fitness V^{ij}
If $V^{ij} < x^{i,j}$
Replace $x^{i,j}$ with V^{ij} in the next generation;
$trial_i = 0$
else $trial_i = trial_i + 1$
end if
end for
Compute the probability by using Equation (15);

[Onlooker Bee Phase]

Set $t = 0, i = 1$;
while $t \leq SN$ do
If $rand < P^i$
Produce a new food source V^{ij} by using Equation (16)
Evaluate the fitness of V^{ij}
end if
end while

[Scout bee Phase]

for $l = 0, i = 1$
If $f_L = 0$
Record the best solution
else $l = l + 1$;
end if
end for
Return Final neuron state
END

Stage 3: Onlooker Bee Stage

Upon returning to the hive, employed bees communicate information regarding the discovered food sources through a symbolic dance. This information is then utilized by onlooker bees, who employ the roulette wheel selection (RWS) method to make informed choices about which food sources to explore.

$$P_{onlooker}^{i} = \frac{f_{V_{employed}^{ij}}}{\sum_{i=1}^{SN} f_{V_{employed}^{ij}}} \tag{16}$$

Where $\sum_{i=1}^{SN} f_{V_{employed}^{ij}}$ portrays desired fitness and SN represents the total number of bee group.

Onlooker bees are entrusted with the task of identifying food sources by employing Equation (16). Following this phase, the fitness of each onlooker bee will be evaluated using Equation (11). The generation of food sources will persist until the number of trials reaches the predefined threshold.

Stage 4: Scout Bee

If the quality of the solution fails to meet the threshold fitness (i.e., $(f_L = 0)$), the food source will be classified as excessive [41]. In such cases, the scout bee will abandon the current search space. The role of the scout bee is crucial in helping the algorithm circumvent local minima. Should the food source be identified as $f_{V_{onlooker}^{ij}} = f_{NC}$, the optimal bipolar instance will be outputted. Subsequently, the pseudo-code for the ABC algorithm is presented in Algorithm 1.

E. Fuzzy Logic System and Artificial Bee Colony Algorithm with Quadratic Satisfiability Logic

This article discusses the methodology of utilizing neuron expressions to calculate various fitness functions. Initially, we incorporate fuzzy logic into the system, presenting the framework of fuzzy sets and functions within a logical structure. In the subsequent phase, the metaheuristic Artificial Bee Colony (ABC) algorithm is integrated to achieve optimal convergence towards the best solution. This section elucidates how fuzzy logic is integrated with the ABC algorithm in the context of Quadratic Satisfiability, specifically in the framework of DHNN-Q2SATFuzzyABC.

The proposed methodology encompasses two distinct phases: (a) the Fuzzy Logic System phase and (b) the ABC algorithm phase. Initially, the fuzzy logic system is employed to compute the output (Z_i) based on two inputs (A_1 and A_2). For clarification, the inputs are represented by two literals, with the output designated as Z_1. This procedure is iteratively applied across all clauses. The output utilizes two membership functions. In the first phase, each input A_1 is randomly assigned a value, after which fuzzy logic is applied to assess the initial output, Z_i. The fundamental step in the fuzzy logic system is fuzzification, during which each literal is allocated to its respective membership function, as detailed in Equation (17).

$$P_i = \{(A_i, L_t(A_i)) \mid A_i \in X\}, \text{Where } 0 \le L_t(A_i) \le 1 \tag{17}$$

Subsequently, fuzzy rules are applied to the initial clause block, Z_i, utilizing the rule $Z_1 = A_1 \vee A_2$. By permitting $L_t(A_i)$ to accept values within the interval [0, 1], Zadeh's operators are employed. The logical operator AND is characterized by the function $\max\{L_t(A_i), L_t(A_j)\}$, while the logical operator OR is defined by the function $\min\{L_t(A_i), L_t(A_j)\}$.

The defuzzification process subsequently transforms the fuzzy values into a precise output. The alpha-cut method adjusts neuron clauses through non-fuzzification until the desired neuron state is achieved [27]. The enhanced alpha-cut formula is represented by the following estimation model:

$$\text{If } Z_i < \alpha \text{ then } Z_i' = 0,$$

$$\text{Otherwise } Z_i > \alpha \text{ then } Z_i' = 1 \tag{18}$$

The second phase of the training process entails the implementation of the ABC algorithm. In this stage, the food sources for the bees (f_{ABC}) are derived from the solutions generated in the prior fuzzy logic system phase. The fitness function is computed according to Equation (19).

$$f_{ABC} = \sum_{i=1}^{l} Z_i^{(2)} + \sum_{i=1}^{m} Z_i^{(1)} \tag{19}$$

$$\text{Where,} \quad Z_i^k = \begin{cases} 1, \textit{Satisfied} \\ 0, \textit{otherwise} \end{cases} \tag{20}$$

Consequently, the aggregate of all clauses Z_i^k within the context of this framework signifies the optimal fitness, which can be articulated through Equation (21).

$$\max[f_{ABC}] \tag{21}$$

Here, the ABC process will be repeatedly executed until the maximum fitness is achieved. Note that the ABC can introduce additional computational overhead due to population-based fitness evaluations involving 101 bees, each fitness evaluation is computationally inexpensive because it is based on clause satisfaction counting using logical operations and simple summations. Furthermore, the fuzzy logic stage generates structured initial solutions prior to the ABC optimization phase, reducing search randomness and improving the quality of candidate solutions. As a result, the additional overhead associated with the swarm algorithm is partially offset by a reduction in redundant evaluations and more efficient search behavior. In the next, the pseudo code for FuzzyABC is written through Algorithm 2 where it is explored with two different stages.

Algorithm 2: The Two-stage Pseudocode for FuzzyABC model:

First stage

[Initialization]

Initialize the inputs A_i, output Z_i, membership function $L_t(A_i)$.

[Fuzzification]

Assign the literals to it membership function which aligns with $0 \le L_t(A_i) \le 1$

[Fuzzy rules]

Apply rule lists using the first clause arrangements, such as $Z_1 = \neg A_1 \vee A_2$.

[Defuzzification]

Implement the Defuzzification step and estimation model.

END

Second stage

[Initialization]

Initialize random populations for the instances Z_i.

[Fitness evaluation]

Evaluate the fitness of the population of the instances Z_i.

[Employed Bee Phase]

Produce the new food source, evaluate the best food source and save as $S_i^/$.

[Onlooker Bee Phase]

Randomly select several food sources.

Generate new solutions and save them as $B_i^/$.

[Scout Bee Phase]

Check a solution x_i from $B_i^/$

if $x_i < \beta_i$ **then**

Update x_i

End if

Update $x_i = x_i^{//}$

RETURN Best Solution

End

III. EXPERIMENTAL SETUP AND PERFORMANCE METRICS

To maintain objectivity in the experimental process, the DEV C++ software was employed to implement fuzzy logic and the ABC algorithm within the DHNN-Q2SAT framework. To ensure a fair and reproducible comparison, identical stopping criteria and computational constraints are imposed across all models. Convergence is determined using the energy tolerance condition through Equation (7), and each simulation is terminated if the total CPU time exceeds 24 hours [18]. These criteria are applied uniformly to DHNN–Q2SAT, DHNN–Q2SATFuzzy, and DHNN–Q2SATFuzzyABC, ensuring that performance differences arise from algorithmic behavior rather than unequal termination conditions. Although the ABC introduced additional population-based fitness evaluations, each evaluation is computationally inexpensive because it is based only on clause satisfaction. Furthermore, the fuzzy logic stage generates structured initial solutions that reduce search randomness and improve the efficiency of the model. Meanwhile, The performance of the proposed DHNNQ2SAT-FuzzyABC model will be compared against two other models: DHNN-Q2SAT and DHNN-Q2SAT-Fuzzy. The DHNN models of this experiment shaped 2 literals per clause as the general formation of Q2-SAT by using simulated datasets. Furthermore, we choose one of the popular activation function named Hyperbolic Tangent Activation Function (HTAF) because of its stability and, it offers a standard method for acting an activation function in the DHNN. Due to the effective reduction of statistical errors, the final energy implementation requires a value of 0.001 [15],[18],[30]. Notably, this section delineates the training and testing phases, both of which are crucial to the analysis. The specific objectives for each phase are geared towards attaining the optimal solution. The training phase is primarily concerned with minimizing the cost function, as articulated in Equation (3). In contrast, the testing phase seeks to produce global minima solutions within a specified tolerance range. The evaluation metrics applied during this phase encompass testing error analysis, energy analysis, similarity analysis, and computational time. Tables 1 through Table 3 present a detailed overview of the parameters utilized in this study.

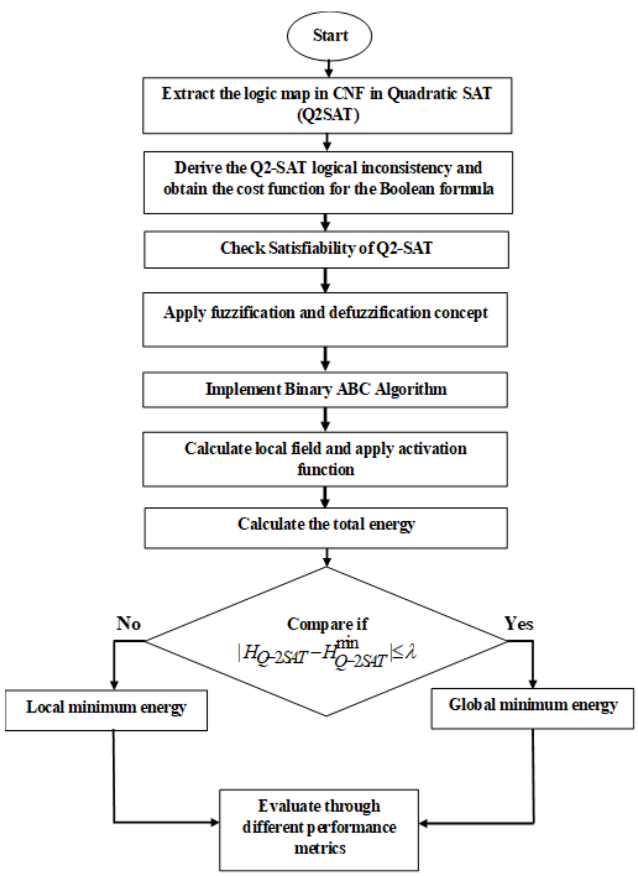

Fig. 2 A detail of flowchart of implementation of DHNN-Q2SATFuzzyABC

TABLE I
RELATED PARAMETERS FOR DHNN-Q2SAT

Parameter	Parameter Value
Number of Neurons (NN)	$8 \le NN \le 80$
Total Neuron Combinations	10 [18]
CPU time	24 hours [18]
Activation Function	HTAF [17]
Tolerance measurement	0.001 [17]
Synaptic weight method	Wan Abdullah (WA) [13]

TABLE II
RELATED PARAMETERS FOR DHNN- Q2SATFUZZY [25]

Parameter	Parameter Value
Fuzzy membership neuron (L_t)	[0,1]
Input Parameter (A_i)	[0,1]
Output Parameter (Z_i)	[0,1]
Number of inputs	2

In this experimental analysis, a range of performance metrics was utilized to assess the efficacy of the DHNN models. The error analysis metrics for each DHNN-Q2SAT model during the training phase comprised the MAE, RMSE, and MAPE, as specified in Equations (22) through (24).

TABLE III
RELATED PARAMETERS FOR DHNN-Q2 Q2SATFUZZYABC

Parameter	Parameter Value
Total Neuron Combination	10 [18]
Number of Trials	10
Employed bees	50 [15]
Onlooker bees	50 [15]
Scout bees	1 [15]
Tolerance measurement	0.001 [17]
Activation function	HTAF [17]

$$RMSE_{training} = \sum_{i=1}^{n} \sqrt{\frac{1}{n}(F_{\max} - F_A)^2} \tag{22}$$

$$MAE_{train} = \sum_{i=1}^{n} \frac{1}{n} | F_{\max} - F_A | \tag{23}$$

$$MAPE_{train} = \sum_{i=1}^{n} \frac{100}{n} \frac{| F_{\max} - F_A |}{| F_A |} \tag{24}$$

In this context, $F_{\max}$ signifies the highest fitness level attained by the proposed model, whereas F_A indicates the fitness value achieved by the network. Furthermore, the variable n represents the total number of iterations conducted throughout the training phase.

During the testing phase, the error analysis metrics—including the RMSE, MAE, and MAPE—for each model are expressed in Equations (25) to (27).

$$RMSE_{test} = \sum_{i=1}^{n} \sqrt{\frac{1}{C.T}(MO_i - O_i)^2} \tag{25}$$

$$MAE_{test} = \sum_{i=1}^{n} \frac{1}{C.T} | MO_i - O_i | \tag{26}$$

$$MAPE_{train} = \sum_{i=1}^{n} \frac{100}{C.T} \frac{| MO_i - O_i |}{| O_i |} \tag{27}$$

In this context, MO_i denotes the maximum optimal solution attained by the proposed model, while O_i represents the optimal solution achieved by the network. The variable $C.T$ signifies the product of the number of combinations and the number of trials. To assess the diversity of the Q2SAT solutions produced by the proposed models, the final neuron state, S_i, can be compared with the ideal neuron state, S_i^{ideal} [31]. The overall relationship between the benchmark state and the final neuron state is expressed through Equation

(28). The similarity metric utilized for the similarity index analysis is the Jaccard Index, as outlined in Equation (29). Furthermore, an energy analysis is conducted to evaluate the robustness of the proposed network. The global minimum (Z_m) is assessed during the testing phase to confirm the accuracy of the final neuron states retrieved by the model, as indicated in Equation (30). Finally, Equation (31) defines computational time as the total duration required to complete the simulation.

$$V_{S_i^{\max} S_i} = \left\{ (S_i^{\max}, S_i), i = 1, 2, \dots n \right\} \tag{28}$$

$$\text{Jaccard Index, } JI = \frac{l}{l+m+n} \tag{29}$$

$$\text{And } Z_{global} = \frac{1}{C.T} \sum_{i=1}^{n} Z_i \text{, then} \tag{30}$$

Computational Time = Training phase time + Testing phase time (31)

Here, l consider the value 1(for both $S_i^{\max}, S_i$) in $V_{S_i^{\max} S_i}$

m consider the value 1 for $S_i^{\max}$ and -1 for S_i in $V_{S_i^{\max} S_i}$;

n consider the value -1 for $S_i^{\max}$ and 1 for S_i in $V_{S_i^{\max} S_i}$.

Additionally, C.T is the product of the number of combinations and trials and Z_i is the number of global minima solutions.

IV RESULT AND DISCUSSIONS

This section provides a comprehensive analysis of the results derived from three distinct models: DHNN-Q2SAT, DHNN-Q2SATFuzzy, and DHNN-Q2SATFuzzyABC. The DHNN-Q2SAT model serves as the baseline, operating within a fundamental random framework without any modifications to the training phase. In contrast, DHNN-Q2SATFuzzy integrates fuzzy logic, while DHNN-Q2SATFuzzyABC incorporates the Artificial Bee Colony algorithm alongside fuzzy logic. Notably, within the proposed DHNN-Q2SATFuzzyABC framework, the ABC algorithm is utilized as the training method during the training phase. To assess the performance of our proposed model, this study constrains the number of neurons (*NN*) to a range of $8 \le NN \le 80$. Furthermore, a thorough analysis encompassing training error, testing error, energy evaluation, similarity assessment, and computational time was conducted for all models. To present a clear view of the results of the models' tabular results are also presented through Table IV, Table V and Table VI.

A. Training Analysis

Figures 3 to Figure 5 show the results of RMSE, MAE, and MAPE during the training phase for the three models: DHNN-Q2SAT, DHNN-Q2SATFuzzy, and DHNNQ2SAT Q2SATFuzzyABC, respectively.

In the total simulation ($8 \le NN \le 80$) of the training phase, the value for $RMSE_{train}, MAE_{train}, MAPE_{train}$ is different from each model. From Figure 3 to Figure 5, it is visible that DHNN-Q2SATFuzzyABC outperforms other existing models by securing shallow error values. Though in the initial stage (when the NN is small), the training errors for all models are not very high, but when *NN* increases, the errors for DHNN-Q2SAT and DHNN-Q2SATFuzzy increase rapidly. This happened due to the lack of an efficient training algorithm. The training algorithm for the existing models only employs a trial-and-error dynamical process, which can lead to the accumulation of errors as the number of iterations increases. This trial-and-error mechanism lacks guided exploration and becomes increasingly susceptible to stagnation in local minima as the number of neurons increases. Consequently, the baseline models repeatedly revisits suboptimal neuron states and requires a larger number of ineffective iterations before reaching convergence. In contrast, the proposed DHNN–Q2SATFuzzyABC model improves both initialization and search behavior, thereby reducing the total number of ineffective iterations required to obtain stable neuron states. Point to noting that a fuzzy logic system is more likely to widen the search during the training and use fuzzification for precise interpretation. Then the proposed fuzzy model exploits a systematic approach by exploring the defuzzification procedure [24]. After completing Training Error Analysis, it is necessary to determine how the models can be used to manage the synaptic weights, which is highly related to obtaining final neuron states, and this will be checked through Testing Error Analysis in the next subsection.

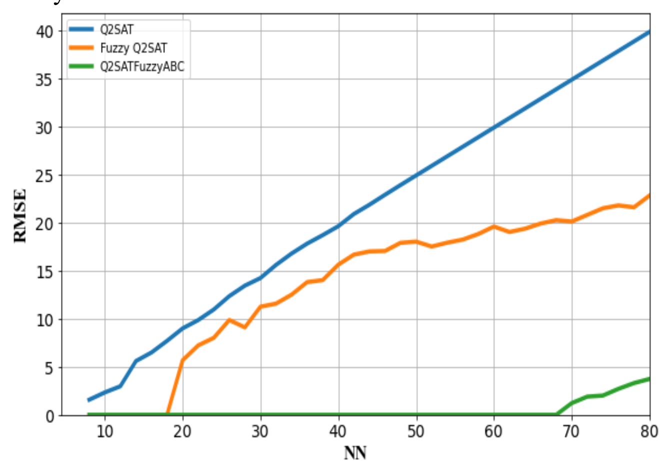

Fig. 3 Training RMSE values for all DHNN models

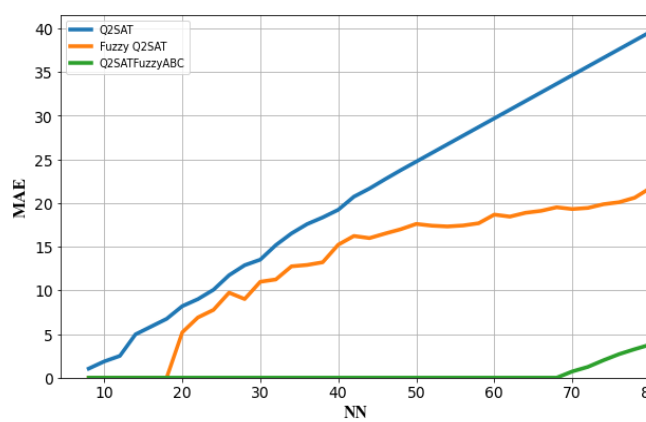

Fig. 4 Training MAE values for all DHNN models

B. Testing Error Analysis

This subsection will examine the behavior of the DHNN concerning the management of synaptic weights. The efficacy of the testing phase within a model depends on its capacity to achieve optimal synaptic weights, which facilitates the retrieval of final neuron states and the generation of global minimum solutions. Figures 6, 7, and 8 present the results for $RMSE_{test}, MAE_{test}, MAPE_{test}$ respectively.

From the figures, the testing error of the DHNN-Q2SAT model is higher compared to other models. This showed that the model cannot attain satisfied interpretation throughout the simulations and that the model is overfitted as well due to the unsatisfied clauses. In this consequence, a logical optimizer such as fuzzy logic is useful since this type of special logic can widen the search space, which is beneficial for overcoming the overfitting solutions. Given the limitations of the training environment, the proposed model ensures a fixed number of training iterations. In this concern, the inclusion of a dynamic ABC algorithm in the proposed model (DHNN-Q2SATFuzzyABC), which has an exploration and exploitation mechanism, ensures almost zero error in the training phase. Thus, the proposed DHNN-Q2SATFuzzyABC model successfully retrieves accurate final neuron states and achieves superior global minima solutions in comparison to the other models.

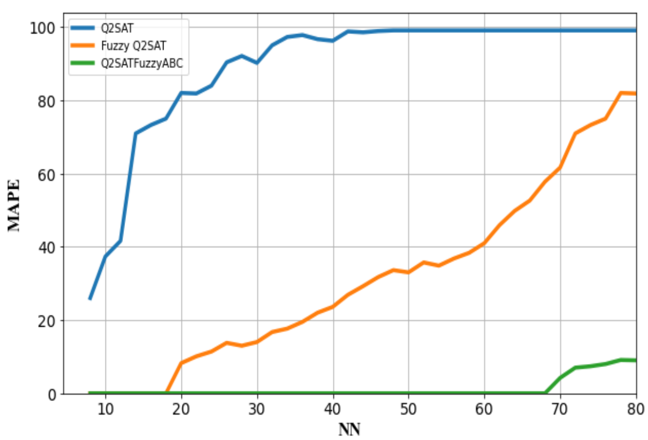

Fig. 5 Training MAPE values for all DHNN models

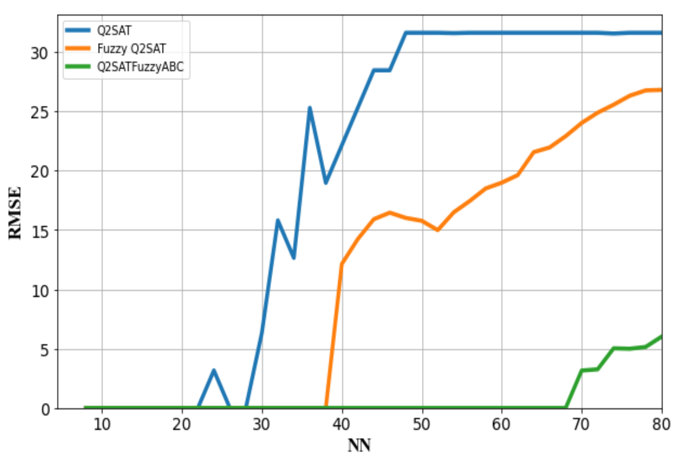

Fig. 6 Testing RMSE values for all DHNN models

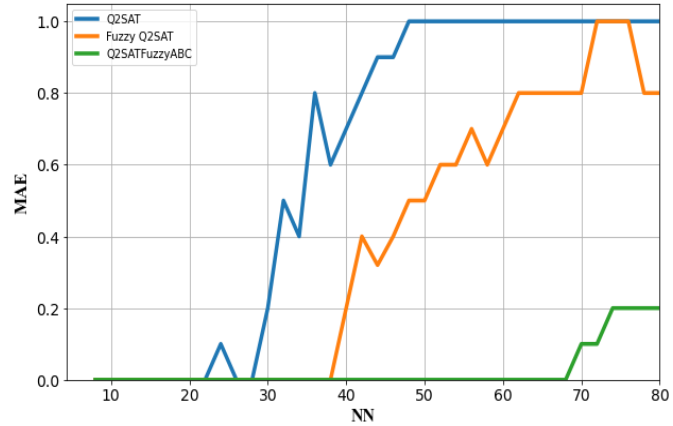

Fig. 7 Testing MAE values for all DHNN models

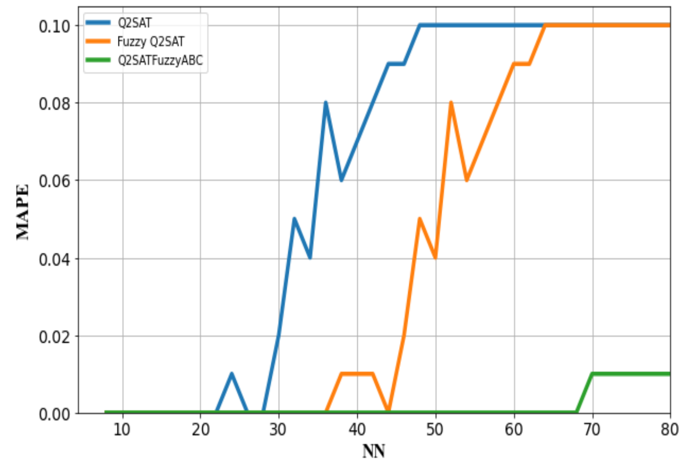

Fig. 8 Testing MAPE values for all DHNN models

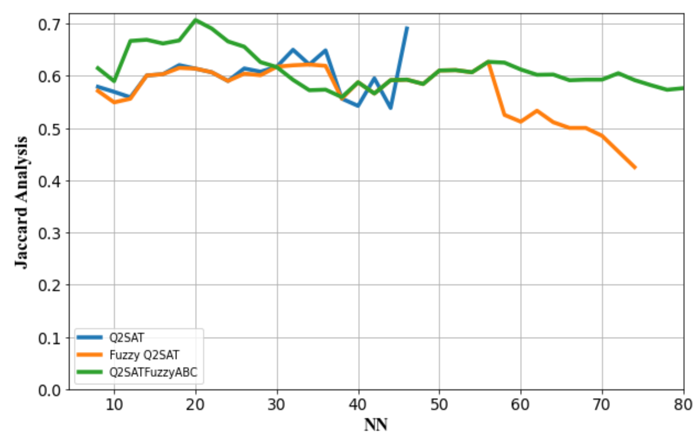

Fig. 9 Jaccard Similarity Analysis for all DHNN models

C. Similarity Index Analysis

The similarity index (SI) serves as a decisive tool for evaluating both the resemblance between final neuron states and a benchmark state, as well as the stability of network dynamics under varying neuron counts. Using the Jaccard Index (JI) as the primary similarity metric, Figure 9 illustrates a distinct separation in performance trends across the three models. The proposed DHNN-Q2SATFuzzyABC maintains a consistently high JI, ranging approximately from 0.60 to 0.68, even as *NN* increases, demonstrating strong resilience to network scaling. In contrast, both DHNN-Q2SAT and DHNN-Q2SATFuzzy exhibit larger oscillations in JI values, with declines becoming more distinct beyond NN ≈ 60 for DHNN-Q2SATFuzzy and beyond $NN \approx 40$ for DHNN-Q2SAT. These fluctuations reflect instability in converging to optimal neuron states and increased susceptibility to overfitting in higher-dimensional search spaces.

Importantly, the flatness of the Q2SATFuzzyABC curve across most of the NN range implies that its hybrid fuzzy-ABC mechanism preserves a robust exploration–exploitation balance, allowing it to maintain similarity to the benchmark state while accommodating neuron variability. The drop-offs seen in the other two models suggest that without the swarm optimization component, their search processes degrade as problem dimensionality increases, leading to reduced similarity and less reliable retrieval of optimal solutions. Thus, the consistently balanced SI values for Q2SATFuzzyABC provide strong evidence of its superiority in producing stable, generalizable neuron configurations over a broad range of network sizes. In the SI analysis, another key subdomain is to find the neuron variations of the model. In this regard, how the neuron varies will discuss in Figure 10.

Meanwhile, Figure 10, shows the total number of neuron variations *(TNV)*. The *TNV*, which are adapted to quantify the dynamical behavior of the DHNN models.

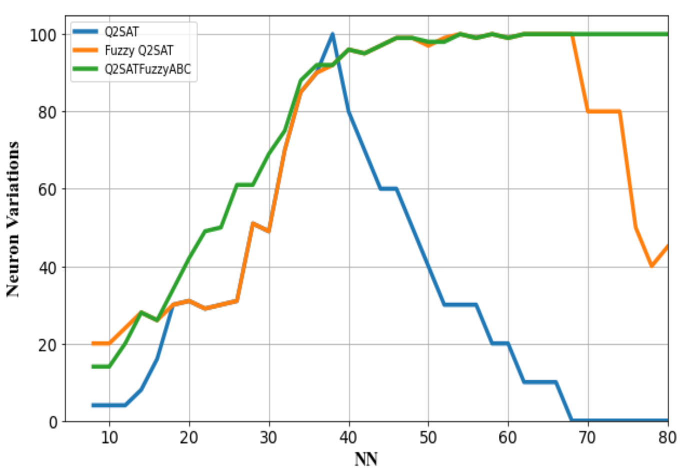

Fig. 10 Total Neuron Variations for all DHNN models

This is very significant for illustrating how the different models can recall different solutions in each trial. By taking a direct composition of neuron variation for $NN = 60, 70, 80$, the proposed model shows higher variability of the final solutions. On the other side, the DHNN-Q2SAT model cannot generate any value at $NN \geq 70$. This is due to the ineffective training algorithm where there is no optimization operator to improve the solution space [9]. Overall, our proposed model outperformed DHNN-Q2SAT and DHNN-Q2SATFuzzy in terms of solution quality which certifies that our model generates less overfit. After that, these models need to verify how these are performed in achieving global minima solutions. In this regard, Energy Analysis of the models will be conducted in the next.

D. Energy Analysis

In this subsection, the energy profile of the DHNN models will be further discussed. In this analysis, the energy function of DHNN acts as an indicator of whether the solutions produced are optimal or not. An important side for energy analysis is determining the extent to which the models can produce the maximum global minimum energy throughout the simulations.

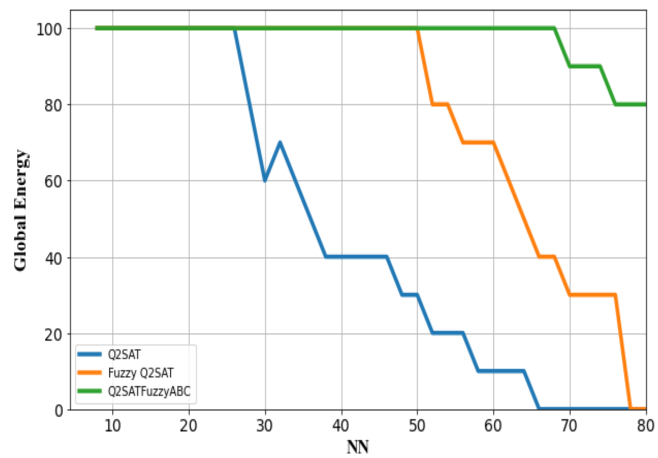

Fig. 11 Global Energy for all DHNN models

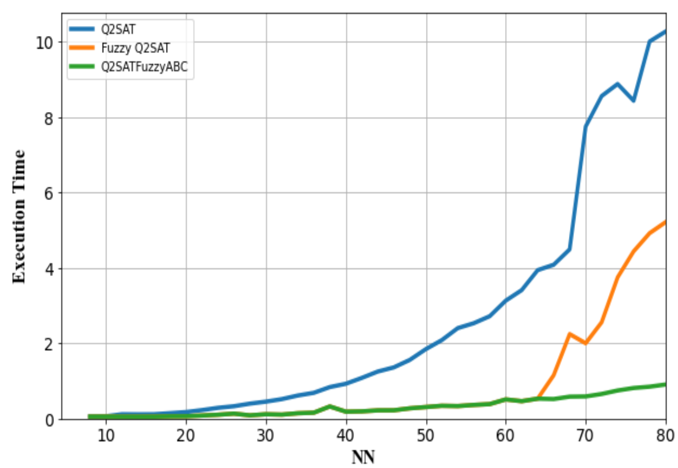

Fig. 12 Computational Time for all DHNN models

As shown in Figure 10, the DHNN-Q2SATFuzzyABC model maintains the highest incidence of global minimum solutions throughout the evaluation process. Unlike the Q2SAT and Fuzzy Q2SAT models, which exhibit a progressive drop in global energy achievement as NN increase, the proposed hybrid model sustains optimal energy states over a substantially wider range. This stability is attributed to two synergistic mechanisms: (i) fuzzy logic, which enhances the precision of literal and clause handling, reducing premature convergence to suboptimal minima, and (ii) the swarm-based Artificial Bee Colony (ABC) algorithm, which broadens the search space exploration and helps the network escape local minima.

A closer examination of the Q2SAT and Fuzzy Q2SAT curves reveals that their performance degradation is not linear but exhibits abrupt declines at specific NN thresholds, suggesting sensitivity to network scaling. In contrast, the

DHNN-Q2SATFuzzyABC curve remains nearly flat until very high *NN* values, indicating strong scalability and robustness.

From a computational perspective, while energy optimality is the primary focus, the time-to-solution metric also plays a critical role. Our experiments show that the proposed model, despite its additional fuzzy and ABC components, achieves competitive convergence times due to faster stabilization of neuronal states—this is a direct consequence of reduced oscillatory transitions in the energy landscape. Therefore, the DHNN-Q2SATFuzzyABC model not only achieves higher optimality rates but also sustains them without significant computational overhead, making it superior for large-scale problem instances.

E. Computational Time

Finally, we will discuss how the models can perform their simulations through computations. The computational time, called CPU period, can be described as how long a simulation can be completed through different *NN*. Figure 12 depicts that DHNN-Q2SATFuzzyABC surpasses its contemporary DHNN-Q2SATFuzzy and DHNN-Q2SAT models. The inclusion of the ABC algorithm along with fuzzy logic makes the model dynamic which resulting less computational time from other models. It needs to clarify that the proposed DHNN–Q2SATFuzzyABC model does not reduce the computational cost of each iteration. On the contrary, the integration of the ABC algorithm increases per-iteration computational cost because multiple candidate solutions are evaluated in each generation. However, as emphasized by Yang et al. (2012) [45], the efficiency of a metaheuristic algorithm should be assessed based on the total computational effort, rather than solely on the complexity of a single iteration. A computationally more expensive iteration may still result in lower overall runtime if it significantly reduces the number of iterations required to reach convergence.

The reduction in CPU time observed in our model arises from two different mechanisms. First, the fuzzy logic component transforms binary logical variables into graded membership values, which smooths the search space and generates higher-quality initial neuron states. Second, the ABC algorithm provides a structured exploration–exploitation process through employed, onlooker, and scout bees, enabling the optimization to escape stagnation regions and avoid redundant search cycles. In contrast, the baseline models rely on a trial-and-error dynamical update mechanism that becomes increasingly susceptible to local minima as the number of neurons grows, resulting in a larger number of ineffective iterations. From the above evaluation metrics, it can be stated that the proposed DHNN-Q2SATFuzzyABC outperformed all other existing models throughout the performance analysis.

In an overall point of view in the result and discuss section, we can affirm that the proposed DHNN-Q2SATFuzzyABC framework demonstrates several compelling advantages. It maintains stability across a wide range of neuron sizes, converges more rapidly, and shows a reduced tendency toward overfitting compared with baseline models. Its hybrid architecture—combining fuzzy logic for adaptive, rule-based reasoning with ABC optimization for effective search-space navigation - achieves a well-calibrated balance between interpretability and robustness. Nevertheless, its effectiveness is somewhat sensitive to parameter choices, and the performance gains are less noticeable in highly sparse Q2SAT clause–literal configurations. Moreover, the current design is tailored specifically to quadratic satisfiability; extending it to higher-order SAT problems would demand substantial adjustments to the underlying logic representation. All together, these findings suggest that the framework is most advantageous for Q2SAT clausal instances with balanced clause densities, where the search space is sufficiently intricate to leverage the full potential of the hybrid approach.

F. Statistical Significance

To evaluate whether the observed differences in performance metrics were statistically significant, paired-sample t-tests were conducted for all model pairs on the Training phase of this model, since the training phase results indicate the total statistical scenario of the models.

TABLE VIII PAIRED T-TEST RESULTS FOR TRAINING PHASE AND COMPARISONS BETWEEN THE MODELS.

Comparison	RMSE			MAE			MAPE		
	t-Stat	***p-value (two-tailed)**	**Significance**	**t-Stat**	***p-value (two-tailed)**	**Significance**	**t-Stat**	***p-value (two-tailed)**	**Significance**
E1	9.7799	5.61×10^{-12}	**HS	9.110	7.06×10^{-11}	**HS	9.110	7.76×10^{-11}	**HS
E2	11.3900	8.59×10^{-13}	**HS	11.435	4.54×10^{-13}	**HS	11.435	6.03×10^{-11}	**HS
E3	11.7712	6.71×10^{-13}	**HS	11.353	6.88×10^{-13}	**HS	11.353	9.92×10^{-13}	**HS

Note: E1: Q2SAT vs Fuzzy Q2SAT; E2: Fuzzy Q2SAT vs Q2SAT-FuzzyABC and E3: Q2SAT vs Q2SAT-FuzzyABC
*indicates $p < 0.001$. Significance determined at $\alpha = 0.05$.
**HS-Highly significant

TABLE IV COMPARATIVE PERFORMANCE ANALYSIS OF THREE DIFFERENT MODELS WITH DIFFERENT METRICS IN THE TRAINING PHASE. THE LOWER VALUE INDICATES BETTER RESULT.

NN	Q2SAT			Fuzzy-Q2SAT			Q2SAT-FuzzyABC		
	RMSE	MAE	MAPE	RMSE	MAE	MAPE	RMSE	MAE	MAPE
8	1.568	1.040	26.000	0.000	0.000	0.000	0.000	0.000	0.000
10	2.324	1.867	37.333	0.000	0.000	0.000	0.000	0.000	0.000
12	2.967	2.495	41.590	0.000	0.000	0.000	0.000	0.000	0.000
14	5.585	4.963	70.901	0.000	0.000	0.000	0.000	0.000	0.000
16	6.476	5.855	73.187	0.000	0.000	0.000	0.000	0.000	0.000
18	7.689	6.745	74.949	0.000	0.000	0.000	0.000	0.000	0.000
20	8.982	8.195	81.945	0.000	5.195	8.232	0.000	0.000	0.000
22	9.843	8.998	81.796	5.671	6.898	10.091	0.000	0.000	0.000
24	10.941	10.072	83.935	7.212	7.768	11.403	0.000	0.000	0.000
26	12.336	11.739	90.297	8.000	9.739	13.777	0.000	0.000	0.000
28	13.423	12.885	92.033	9.852	9.011	12.982	0.000	0.000	0.000
30	14.213	13.522	90.148	9.085	10.989	14.000	0.000	0.000	0.000
32	15.578	15.190	94.939	11.234	11.255	16.713	0.000	0.000	0.000
34	16.762	16.528	97.224	11.555	12.756	17.658	0.000	0.000	0.000
36	17.793	17.593	97.739	12.487	12.914	19.493	0.000	0.000	0.000
38	18.672	18.357	96.618	13.791	13.222	22.000	0.000	0.000	0.000
40	19.599	19.235	96.173	14.000	15.235	23.592	0.000	0.000	0.000
42	20.866	20.733	98.728	15.599	16.232	26.879	0.000	0.000	0.000
44	21.832	21.666	98.484	16.647	15.988	29.232	0.000	0.000	0.000
46	22.866	22.732	98.835	16.980	16.511	31.719	0.000	0.000	0.000
48	23.881	23.762	99.010	17.026	17.000	33.612	0.000	0.000	0.000
50	24.876	24.752	99.010	17.875	17.613	33.000	0.000	0.000	0.000
52	25.871	25.743	99.010	18.000	17.412	35.712	0.000	0.000	0.000
54	26.866	26.733	99.010	17.489	17.323	34.832	0.000	0.000	0.000
56	27.861	27.723	99.010	17.888	17.434	36.783	0.000	0.000	0.000
58	28.856	28.713	99.010	18.210	17.692	38.338	0.000	0.000	0.000
60	29.851	29.703	99.010	18.794	18.679	40.967	0.000	0.000	0.000
62	30.846	30.693	99.010	19.565	18.451	45.872	0.000	0.000	0.000
64	31.841	31.683	99.010	19.000	18.888	49.765	0.000	0.000	0.000
66	32.836	32.673	99.010	19.343	19.111	52.591	0.000	0.000	0.000
68	33.831	33.663	99.010	19.875	19.519	57.721	0.000	0.000	0.000
70	34.826	34.653	99.010	20.232	19.317	61.590	1.217	0.717	4.212
72	35.821	35.644	99.010	20.088	19.444	70.901	1.888	1.238	7.002
74	36.816	36.634	99.010	20.777	19.871	73.187	2.002	2.000	7.387
76	37.811	37.624	99.010	21.464	20.098	74.949	2.698	2.698	8.000
78	38.806	38.614	99.010	21.765	20.614	81.945	3.297	3.266	9.101
80	39.801	39.604	99.010	21.569	21.719	81.796	3.727	3.791	9.007

TABLE V COMPARATIVE PERFORMANCE ANALYSIS OF THREE DIFFERENT MODELS WITH DIFFERENT METRICS IN THE TESTING PHASE. THE LOWER VALUE INDICATES BETTER RESULT.

NN	Q2SAT			Fuzzy-Q2SAT			Q2SAT-FuzzyABC		
	RMSE	MAE	MAPE	RMSE	MAE	MAPE	RMSE	MAE	MAPE
8	0.000	0	0	0.000	0.000	0.000	0.000	0.000	0.000
10	0.000	0	0	0.000	0.000	0.000	0.000	0.000	0.000
12	0.000	0	0	0.000	0.000	0.000	0.000	0.000	0.000
14	0.000	0	0	0.000	0.000	0.000	0.000	0.000	0.000
16	0.000	0	0	0.000	0.000	0.000	0.000	0.000	0.000
18	0.000	0	0	0.000	0.000	0.000	0.000	0.000	0.000
20	0.000	0	0	0.000	0.000	0.000	0.000	0.000	0.000
22	0.000	0	0	0.000	0.000	0.000	0.000	0.000	0.000
24	3.162	0.1	0.01	0.000	0.000	0.000	0.000	0.000	0.000
26	0.000	0	0	0.000	0.000	0.000	0.000	0.000	0.000
28	0.000	0	0	0.000	0.000	0.000	0.000	0.000	0.000
30	6.325	0.2	0.02	0.000	0.000	0.000	0.000	0.000	0.000
32	15.811	0.5	0.05	0.000	0.000	0.000	0.000	0.000	0.000
34	12.649	0.4	0.04	0.000	0.000	0.000	0.000	0.000	0.000
36	25.298	0.8	0.08	0.000	0.000	0.000	0.000	0.000	0.000
38	18.974	0.6	0.06	0.000	0.000	0.010	0.000	0.000	0.000
40	22.136	0.7	0.07	12.136	0.200	0.010	0.000	0.000	0.000
42	25.298	0.8	0.08	14.232	0.400	0.010	0.000	0.000	0.000
44	28.460	0.9	0.09	15.908	0.320	0.000	0.000	0.000	0.000
46	28.460	0.9	0.09	16.460	0.400	0.020	0.000	0.000	0.000
48	31.623	1	0.1	16.018	0.500	0.050	0.000	0.000	0.000
50	31.623	1	0.1	15.776	0.500	0.040	0.000	0.000	0.000
52	31.623	1	0.1	14.988	0.600	0.080	0.000	0.000	0.000

Continue with Table V

54	31.591	1	0.1	16.490	0.600	0.060	0.000	0.000	0.000
56	31.623	1	0.1	17.445	0.700	0.070	0.000	0.000	0.000
58	31.623	1	0.1	18.502	0.600	0.080	0.000	0.000	0.000
60	31.623	1	0.1	18.988	0.700	0.090	0.000	0.000	0.000
62	31.623	1	0.1	19.623	0.800	0.090	0.000	0.000	0.000
64	31.623	1	0.1	21.562	0.800	0.100	0.000	0.000	0.000
66	31.623	1	0.1	21.959	0.800	0.100	0.000	0.000	0.000
68	31.623	1	0.1	22.909	0.800	0.100	0.000	0.000	0.000
70	31.623	1	0.1	24.021	0.800	0.100	3.154	0.100	0.010
72	31.623	1	0.1	24.888	1.000	0.100	3.259	0.100	0.010
74	31.560	1	0.1	25.560	1.000	0.100	5.028	0.200	0.010
76	31.623	1	0.1	26.312	1.000	0.100	4.991	0.200	0.010
78	31.623	1	0.1	26.766	0.800	0.100	5.143	0.200	0.010
80	31.623	1	0.1	26.810	0.800	0.100	6.011	0.200	0.010

TABLE VI COMPARATIVE PERFORMANCE ANALYSIS OF THREE DIFFERENT MODELS WITH DIFFERENT METRICS.

NN	**Q2SAT**			**Fuzzy-Q2SAT**			**Q2SAT-FuzzyABC**		
	NGS	JI	TNV	NGS	JI	TNV	NGS	JI	TNV
8	100	0.579	4	100	0.571	14	100	0.614	20
10	100	0.569	4	100	0.549	14	100	0.590	20
12	100	0.559	4	100	0.556	20	100	0.667	24
14	100	0.600	8	100	0.600	28	100	0.669	28
16	100	0.603	16	100	0.603	26	100	0.662	26
18	100	0.620	30	100	0.615	34	100	0.668	30
20	100	0.614	31	100	0.614	42	100	0.707	31
22	100	0.607	29	100	0.607	49	100	0.690	29
24	100	0.590	30	100	0.590	50	100	0.666	30
26	100	0.614	31	100	0.604	61	100	0.656	31
28	80	0.608	51	100	0.601	61	100	0.626	51
30	60	0.617	49	100	0.617	69	100	0.617	49
32	70	0.650	70	100	0.620	75	100	0.592	70
34	60	0.621	85	100	0.621	88	100	0.572	85
36	50	0.649	90	100	0.619	92	100	0.573	90
38	40	0.556	100	100	0.556	92	100	0.559	92
40	40	0.542	80	100	0.588	96	100	0.588	96
42	40	0.595	70	100	0.566	95	100	0.566	95
44	40	0.538	60	100	0.592	97	100	0.592	97
46	40	0.690	60	100	0.593	99	100	0.593	99
48	30	-	50	100	0.584	99	100	0.584	99
50	30	-	40	100	0.610	98	100	0.610	97
52	20	-	30	80	0.611	98	100	0.611	99
54	20	-	30	80	0.607	100	100	0.607	100
56	20	-	30	70	0.627	99	100	0.627	99
58	10	-	20	70	0.525	100	100	0.625	100
60	10	-	20	70	0.512	99	100	0.612	99
62	10	-	10	60	0.533	100	100	0.602	100
64	10	-	10	50	0.511	100	100	0.602	100
66	0	-	10	40	0.500	100	100	0.591	100
68	0	-	0	40	0.500	100	100	0.592	100
70	0	-	0	30	0.485	80	90	0.593	100
72	0	-	0	30	0.455	80	90	0.605	100
74	0	-	0	30	0.425	80	90	0.592	100
76	0	-	0	30	-	50	80	0.582	100
78	0	-	0	0	-	40	80	0.573	100
80	0	-	0	0	-	45	80	0.576	100

The paired t-test results make it clear that the differences in RMSE, MAE, and MAPE between the methods are not just random fluctuations—these are 'highly statistically significant' at the 0.1% level ($p < 0.001$). When comparing Q2SAT with Fuzzy Q2SAT, the results show that Fuzzy Q2SAT consistently delivers better performance across all three metrics, with very high t-values and extremely small p-values. The gap becomes even more striking when Fuzzy Q2SAT is compared to Q2SAT-FuzzyABC, where the t-values climb even higher, confirming that Q2SAT-FuzzyABC offers a substantial improvement. The final comparison, Q2SAT versus Q2SAT-FuzzyABC, produces the strongest results of all, leaving little doubt about the superiority of Q2SAT-FuzzyABC. In short, across every metric—RMSE, MAE, and MAPE—Q2SAT-FuzzyABC stands out as the best-performing model, with improvements that are both large in magnitude and backed by strong statistical evidence

V. Conclusion and future work

In this proposed work, we developed a model that combines DHNN, Fuzzy logic, and a metaheuristic algorithm to effectively interpret the symbolic representation of ANN via Quadratic Satisfiability, and this article reveals a different hybrid concept in DHNN with the composition of fuzzy logic. The inclusion of swarm-type ABC algorithm in the training phase of this network creates a new dimension of systematic fuzzy logic and DHNN study. By creating a variety of network states according to the membership values, this proposed work demonstrates how the incorporation of fuzzy logic increased the optimization process's flexibility. From the result and discussions section, the efficiency of three different models is analysed through training-testing error analysis, energy analysis, neuron variation, and finally the CPU time. The alpha-cut strategy was employed to use fuzzification and defuzzification techniques in order to enhance the strategies of global minimum solutions and minimizing the computational effort involved in creating optimal results. According to the training and testing error view, our proposed DHNN-Q2SATFuzzyABC acquires a very lower value compared to other existing baseline models. Furthermore, this DHNN-Q2SATFuzzyABC model achieved higher global minimum solutions, created more neuron variations, and showed standard consistency in similarity index analysis. In future, the expansion of this study can be improved by the inclusion of higher-order non-systematic logic with fuzzy concepts [42]. Notably, the architecture of the DHNN-Q2SATFuzzyABC can be useful for the logic mining study, which will be very potential for solving real-life problems [43]. Furthermore, fuzzy logic can be implemented on different logical comparisons such as RAN*k*SAT or any other higher order systematic SAT or different types of metaheuristics can be implemented and analyzed through different datasets to understand the real scenario of the model's efficiency in the real-world applications [44].

References

[1] E. Bas, E. Egrioglu, and T. Cansu, "Robust training of median dendritic artificial neural networks for time series forecasting," *Expert Systems With Applications*, vol. 238, p. 122080, Oct. 2023, doi: 10.1016/j.eswa.2023.122080.

[2] H. R. Maier *et al.*, "Exploding the myths: An introduction to artificial neural networks for prediction and forecasting," *Environmental Modelling & Software*, vol. 167, p. 105776, Jul. 2023, doi: 10.1016/j.envsoft.2023.105776.

[3] A. Shah *et al.*, "A comprehensive study on skin cancer detection using artificial neural network (ANN) and convolutional neural network (CNN)," *Clinical eHealth*, vol. 6, pp. 76–84, Aug. 2023, doi: 10.1016/j.ceh.2023.08.002.

[4] V. V. Bukhtoyarov, V. S. Tynchenko, V. A. Nelyub, I. S. Masich, A. S. Borodulin, and A. P. Gantimurov, "A Study on a Probabilistic Method for Designing Artificial Neural Networks for the Formation of Intelligent Technology Assemblies with High Variability," *Electronics*, vol. 12, no. 1, p. 215, Jan. 2023, doi: 10.3390/electronics12010215.

[5] A. Alhussen, M. A. Haq, A. A. Khan, R. K. Mahendran, and S. Kadry, "XAI-RACapsNet: Relevance aware capsule network-based breast cancer detection using mammography images via explainability O-net ROI segmentation," *Expert Systems With Applications*, p. 125461, Sep. 2024, doi: 10.1016/j.eswa.2024.125461.

[6] A. A. Khan, R. K. Madendran, U. Thirunavukkarasu, and M. Faheem, "D2PAM: Epileptic seizures prediction using adversarial deep dual patch attention mechanism," *CAAI Transactions on Intelligence Technology*, vol. 8, no. 3, pp. 755–769, Jul. 2023, doi: 10.1049/cit2.12261.

[7] C. Li, Y. Yang, X. Yang, X. Zi, and F. Xiao, "A tristable locally active memristor and its application in Hopfield neural network," *Nonlinear Dynamics*, vol. 108, no. 2, pp. 1697–1717, Feb. 2022, doi: 10.1007/s11071-022-07268-y.

[8] S. Abdeen, M. S. M. Kasihmuddin, N. E. Zamri, G. Manoharam, Mohd. A. Mansor, and N. Alshehri, "S-Type Random k Satisfiability Logic in Discrete Hopfield Neural Network Using Probability Distribution: Performance Optimization and Analysis," *Mathematics*, vol. 11, no. 4, p. 984, Feb. 2023, doi: 10.3390/math11040984.

[9] X. Jiang *et al.*, "J-type random 2,3 satisfiability: a higher-order logical rule in discrete hopfield neural network," *Evolutionary Intelligence*, vol. 17, no. 5–6, pp. 3317–3336, Apr. 2024, doi: 10.1007/s12065-024-00936-5.

[10] C. Chen, F. Min, F. Hu, J. Cai, and Y. Zhang, "Analog/digital circuit simplification for Hopfield neural network," *Chaos Solitons & Fractals*, vol. 173, p. 113727, Jun. 2023, doi: 10.1016/j.chaos.2023.113727.

[11] Q. Deng, C. Wang, and H. Lin, "Memristive Hopfield neural network dynamics with heterogeneous activation functions and its application," *Chaos Solitons & Fractals*, vol. 178, p. 114387, Dec. 2023, doi: 10.1016/j.chaos.2023.114387.

[12] Y. Gao *et al.*, "Binary Ant colony Optimization algorithm in learning random satisfiability logic for discrete Hopfield neural network," *Applied Soft Computing*, p. 112192, Sep. 2024, doi: 10.1016/j.asoc.2024.112192.

[13] W. A. T. W. Abdullah, "Logic programming on a neural network," *International Journal of Intelligent Systems*, vol. 7, no. 6, pp. 513–519, Aug. 1992, doi: 10.1002/int.4550070604.

[14] S. A. Karim, M. S. M. Kasihmuddin, Mohd. A. Mansor, S. Z. M. Jamaludin, N. E. Zamri, and M. R. Amin, "A socio-inspired hybrid election algorithm for random k satisfiability in discrete Hopfield neural network," *AIP Conference Proceedings*, vol. 3080, p. 040008, Jan. 2024, doi: 10.1063/5.0194531.

[15] M. S. M. Kasihmuddin, Mohd. A. Mansor, and S. Sathasivam, "Robust artificial bee colony in the Hopfield network for 2-satisfiability problem," *Pertanika Journal of Science & Technology*, vol. 25, no. 2, pp. 453–468, 2017.

[16] Mohd. 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*, vol. 25, no. 4, pp. 1173–1188, 2017, [Online]. Available: http://pertanika2.upm.edu.my/resources/files/Pertanika%20PAPERS/JST%20Vol.%2025%20(4)%20Oct.%202017/10%20JST-0640-2016-2ndProof.pdf

[17] S. Sathasivam, Mohd. A. Mansor, A. I. M. Ismail, S. Z. M. Jamaludin, M. S. M. Kasihmuddin, and M. Mamat, "Novel Random k Satisfiability for in Hopfield Neural Network," *Sains Malaysiana*, vol. 49, no. 11, pp. 2847–2857, Nov. 2020, doi: 10.17576/jsm-2020-4911-23.

[18] A. Alway, N. E. Zamri, S. A. Karim, Mohd. A. Mansor, M. S. M. Kasihmuddin, and M. M. Bazuhair, "Major 2 Satisfiability logic in discrete Hopfield neural network," *International Journal of Computer*

Mathematics, vol. 99, no. 5, pp. 924–948, Jun. 2021, doi: 10.1080/00207160.2021.1939870.
[19] J. Chen *et al.*, "PRO2SAT: Systematic Probabilistic Satisfiability logic in Discrete Hopfield Neural Network," *Advances in Engineering Software*, vol. 175, p. 103355, Nov. 2022, doi: 10.1016/j.advengsoft.2022.103355.
[20] J. Serrano-Guerrero, F. P. Romero, and J. A. Olivas, "Fuzzy logic applied to opinion mining: A review," *Knowledge-Based Systems*, vol. 222, p. 107018, Apr. 2021, doi: 10.1016/j.knosys.2021.107018.
[21] H. Agrawal *et al.*, "A Fuzzy-Genetic-Based integration of renewable energy sources and E-Vehicles," *Energies*, vol. 15, no. 9, p. 3300, Apr. 2022, doi: 10.3390/en15093300.
[22] F. L. Azizan, S. Sathasivam, M. K. M. Ali, and S. A. S. Alzaeemi, "Solving HornSAT Fuzzy Logic Neuro-symbolic Integration," in *Studies in systems, decision and control*, 2022, pp. 49–64. doi: 10.1007/978-3-031-04028-3_5.
[23] F. L. Azizan, S. Sathasivam, and M. K. M. Ali, "3SAT and Fuzzy-HornSAT in Hopfield Neural Network," in *Studies in systems, decision and control*, 2022, pp. 65–79. doi: 10.1007/978-3-031-04028-3_6.
[24] F. L. Azizan, S. Sathasivam, M. K. M. Ali, N. Roslan, and C. Feng, "Hybridised network of fuzzy logic and a genetic algorithm in solving 3-Satisfiability Hopfield neural networks," *Axioms*, vol. 12, no. 3, p. 250, Mar. 2023, doi: 10.3390/axioms12030250.
[25] C. Feng, S. Sathasivam, N. Roslan, and M. Velavan, "2-SAT discrete Hopfield neural networks optimization via Crow search and fuzzy dynamical clustering approach," *AIMS Mathematics*, vol. 9, no. 4, pp. 9232–9266, Jan. 2024, doi: 10.3934/math.2024450.
[26] J. Shen and W. Li, "Discrete Hopfield neural networks for evaluating service quality of public transit," *International Journal of Multimedia and Ubiquitous Engineering*, vol. 9, no. 2, pp. 331–340, Feb. 2014, doi: 10.14257/ijmue.2014.9.2.33.
[27] B. Bao, H. Tang, H. Bao, Z. Hua, Q. Xu, and M. Chen, "Simplified discrete Two-Neuron Hopfield neural network and FPGA implementation," *IEEE Transactions on Industrial Electronics*, pp. 1–11, Jan. 2024, doi: 10.1109/tie.2024.3451052.
[28] G. R. Murthy, "Toward optimal synthesis of Discrete-Time Hopfield Neural Network," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 34, no. 11, pp. 9549–9554, Mar. 2022, doi: 10.1109/tnnls.2022.3156107.
[29] Mohd. A. Mansor, M. S. M. Kasihmuddin, and S. Sathasivam, "Grey Wolf Optimization algorithm with Discrete Hopfield Neural Network for 3 Satisfiability analysis," *Journal of Physics Conference Series*, vol. 1821, no. 1, p. 012038, Mar. 2021, doi: 10.1088/1742-6596/1821/1/012038.
[30] S. A. Karim *et al.*, "Random Satisfiability: a Higher-Order logical approach in discrete Hopfield neural network," *IEEE Access*, vol. 9, pp. 50831–50845, Jan. 2021, doi: 10.1109/access.2021.3068998.
[31] M. S. M. Kasihmuddin, Mohd. A. Mansor, M. F. M. Basir, and S. Sathasivam, "Discrete mutation Hopfield neural network in propositional satisfiability," *Mathematics*, vol. 7, no. 11, p. 1133, Nov. 2019, doi: 10.3390/math7111133.
[32] M. S. M. Kasihmuddin, S. Sathasivam, and Mohd. A. Mansor, "Hybrid genetic algorithm in the Hopfield network for maximum 2-satisfiability problem," *AIP Conference Proceedings*, vol. 1870, p. 050001, Jan. 2017, doi: 10.1063/1.4995911.
[33] S. Sathasivam, "Upgrading logic programming in Hopfield Network," *Sains Malaysiana*, vol. 39, no. 1, pp. 115–118, Feb. 2010, [Online]. Available: https://www.ukm.my/jsm/pdf_files/SM-PDF-39-1-2010/17.pdf
[34] M. Hudec, "Fuzzy set and fuzzy logic theory in brief," in *Springer eBooks*, 2016, pp. 1–32. doi: 10.1007/978-3-319-42518-4_1.
[35] L. Donzé and A. Meier, "Applying fuzzy logic and fuzzy methods to marketing," in *IGI Global eBooks*, 2012, pp. 1–14. doi: 10.4018/978-1-4666-0095-9.ch001.
[36] L. A. Zadeh, "Outline of a new approach to the analysis of complex systems and decision processes," *IEEE Transactions on Systems Man and Cybernetics*, vol. SMC-3, no. 1, pp. 28–44, Jan. 1973, doi: 10.1109/tsmc.1973.5408575.
[37] M. J. Alshayeb, A. A. Bindajam, J. Mallick, and A. A. A. Shohan, "Integrating fuzzy logic and land use dynamics to delineate urban heat island formation zones and assess land surface temperature changes over three decades: a case study of Asir region, Saudi Arabia," *Stochastic Environmental Research and Risk Assessment*, Jun. 2025, doi: 10.1007/s00477-025-03036-7.
[38] E. H. Mamdani, "Application of fuzzy algorithms for control of simple dynamic plant," *Proceedings of the Institution of Electrical Engineers*, vol. 121, no. 12, p. 1585, Jan. 1974, doi: 10.1049/piee.1974.0328.
[39] G. Yavuz, B. Durmuş, and D. Aydın, "Artificial Bee Colony Algorithm with Distant Savants for constrained optimization," *Applied Soft Computing*, vol. 116, p. 108343, Dec. 2021, doi: 10.1016/j.asoc.2021.108343.
[40] S. S. M. Sidik, N. E. Zamri, M. S. M. Kasihmuddin, H. A. Wahab, Y. Guo, and Mohd. A. Mansor, "Non-Systematic weighted satisfiability in discrete Hopfield neural network using binary artificial bee colony optimization," *Mathematics*, vol. 10, no. 7, p. 1129, Apr. 2022, doi: 10.3390/math10071129.
[41] H. Gao, Y. Shi, C.-M. Pun, and S. Kwong, "An Improved Artificial Bee Colony Algorithm With its Application," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 4, pp. 1853–1865, Jul. 2018, doi: 10.1109/tii.2018.2857198.
[42] S. A. Karim, M. S. M. Kasihmuddin, S. Sathasivam, Mohd. A. Mansor, S. Z. M. Jamaludin, and M. R. Amin, "A novel Multi-Objective Hybrid Election Algorithm for Higher-Order random Satisfiability in discrete Hopfield neural network," *Mathematics*, vol. 10, no. 12, p. 1963, Jun. 2022, doi: 10.3390/math10121963.
[43] Karim SA, Kasihmuddin MS, Das S, Zamri NE, Islam AJ, Alway A, Chowdhury DK. An explainable logic mining framework with multi-objective metaheuristic algorithm for knowledge extraction in discrete Hopfield neural network. AIMS Mathematics. 2025 Jan 1;10(12):29342-79.
[44] Khan AA, Mahendran RK, Perumal K, Faheem M. Dual-3DM 3 AD: mixed transformer based semantic segmentation and triplet pre-processing for early multi-class Alzheimer's diagnosis. IEEE Transactions on Neural Systems and Rehabilitation Engineering. 2024 Jan 23;32:696-707.
[45] Yang XS. Metaheuristic optimization: algorithm analysis and open problems. International symposium on experimental algorithms 2011 May 5 , 21-32.

Syed Anayet Karim achieved his PhD from the School of Mathematical Sciences, Universiti Sains Malaysia. He completed his B.Sc. (Hons.) and M.S. in Mathematics from the Department of Mathematics, University of Chittagong, Bangladesh. Currently, he is an Associate Professor in the Department of Natural Science at Port City International University, Chattogram, Bangladesh. He also serves as an Additional Director of the Institutional Quality Assurance Cell (IQAC) at Port City International University. Moreover, he is a distinguished reviewer for prominent international journals published by Elsevier, Wiley, Springer, and others. His current research interests include metaheuristics, artificial neural networks, logic mining, and data mining.

Sowmitra Das is an Assistant Professor in the Department of Computer Science and Engineering at Port City International University. He also serves as a Member of the Institutional Quality Assurance Cell (IQAC) at Port City International University. He completed his M.S. in Computer Science from Bangladesh Agricultural University, and earned his B.Sc. in Computer Science and Engineering from University of Chittagong. His research interests include meta-heuristics, ANN, NLP, LLM, CV, ML, and DL. He has published several research papers in international conferences and journals and is actively involved in AI-based research, thesis supervision, and ACM ICPC coaching.

Akib Jayed Islam received his B.Sc. in Electrical and Electronic Engineering from American International University-Bangladesh (AIUB). He also completed an Erasmus Mundus Joint Master's degree in Computational Color and Spectral Imaging (COSI) from the Norwegian University of Science and Technology (NTNU), Norway, and Jean Monnet University (UJM), France. He is currently an Assistant Professor in the Department of Electrical and Electronic Engineering at Port City International University, Chattogram, Bangladesh. He serves as a reviewer for Elsevier, Scopus-indexed, and IEEE publications. His research interests include image and signal processing, bio-telemetry, medical imaging, wearable electronics, energy harvesting, optics, and photonics.

Md Rabiol Amin is a Lecturer of Computer Science and Engineering at CCN University of Science and Technology, Cumilla. Prior to joining CCNUST, he received his Bachelor's in CSE from Port City International University, Chattogram, Bangladesh in 2018. His research interests include natural language processing, computer vision, LLM, and artificial neural network. He also serves his duties as an advisor to the Programming Club and mentors ICPC teams at CCNUST.

Mamun-Or-Rashid is working as an Assistant Professor in the Department of Mathematics at Premier University. He is also the doctoral (PhD) applicant in the Department of Mathematics, University of Chittagong. He achieved his M.Phil degree from the University of Chittagong at Jamal Nazrul Islam Research Centre for Mathematical and Physical Sciences (JNIRCMPS). He completed his B. Sc (Hons) and MS(Thesis) in Mathematics from the Department of Mathematics, University of Chittagong. His current research Interests are Nonlinear optimization techniques, Multi-objective optimization, Numerical Methods, Mathematical modeling, optimal control theory and Reverse Logistic process etc.