I.J. Intelligent Systems and Applications, **2016, 10, 44-50**
Published Online October 2016 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijisa.2016.10.05

Accelerating Activation Function for 3-Satisfiability Logic Programming

Mohd Asyraf Mansor
School of Mathematical Sciences, Universiti Sains Malaysia, 11800, Pulau Pinang, Malaysia
E-mail: asyrafalvez@live.com

Saratha Sathasivam
School of Mathematical Sciences, Universiti Sains Malaysia, 11800, Pulau Pinang, Malaysia
E-mail: saratha@usm.com

***Abstract*—**This paper presents the technique for accelerating 3-Satisfiability (3-SAT) logic programming in Hopfield neural network. The core impetus for this work is to integrate activation function for doing 3-SAT logic programming in Hopfield neural network as a single hybrid network. In logic programming, the activation function can be used as a dynamic post optimization paradigm to transform the activation level of a unit (neuron) into an output signal. In this paper, we proposed Hyperbolic tangent activation function and Elliot symmetric activation function. Next, we compare the performance of proposed activation functions with a conventional function, namely McCulloch-Pitts function. In this study, we evaluate the performances between these functions through computer simulations. Microsoft Visual C++ 2013 was used as a platform for training, validating and testing of the network. We restrict our analysis to 3-Satisfiability (3-SAT) clauses. Moreover, evaluations are made between these activation functions to see the robustness via aspects of global solutions, global Hamming distance, and CPU time.

***Index Terms*—**3-Satisfiability, Hyperbolic tangent activation function, Elliot symmetric activation function, McCulloch-Pitts function, Logic programming, Hopfield neural network.

I. Introduction

Artificial intelligence is one of the most eminent and staple fields in mathematics, physics, and computer science [2]. In fact, the emergence of artificial intelligence attracted a prolific amount of research in combinatorial optimization problems [3, 8]. In addition, the popularity of artificial intelligence is due to the ability to solve many computational, classification, and pattern recognition problems via learning based algorithm [7, 21]. Generally, the integration of neural network, logic programming, and satisfiability problem are vital in enhancing the artificial intelligence field. As we all know, there are numerous neural networks, from the modest to intricate, similarly to the evolution human biological nervous system [5]. The popularity of artificial neural network is due to the collective behavior of the network, inspired by the biological human brain. Thus, the network can be hybridized with some other computational methods in order to solve sophisticated problems and applications.

One of the well-known neural networks is Hopfield neural network. Specifically, the Hopfield network is a recurrent neural network introduced by John Hopfield [3]. It is renowned for the remarkable power in learning, memory and storage [1, 11, 18]. The Hopfield model is a conventional model for content addressable memory (CAM) [1, 8]. It is an important feature of Hopfield which motivated by the manner of biological brain works. In addition, Hopfield network comprised of a set of N interconnected neurons which all neurons are connected to all others in both directions [6, 13]. Then, the Hopfield neural network can be demarcated as a simple recurrent network which can work as an efficient associative memory, and it can store definite memories [1, 2]. One of the popular work is done by Little who proposed a little Hopfield network based on the energy minimization scheme. Moving on, Sathasivam introduced a powerful Hopfield model based on reverse analysis method in solving Horn-satisfiability problem [14, 22]. The work on Hopfield network is still ongoing for the promising future work. Moreover, the Hopfield model is a branch of neural network that been applied in various mathematical problems such as Travelling Salesperson problem (TSP), scheduling, pattern recognition, and satisfiability problem.

Boolean satisfiability is a contemporary computational problem. Given a problem determining whether a problem instance is satisfiable or not, one can hunt for a satisfying assignment in time linear in the number of variables. This has been motivated by recent work of Aiman & Asrar [4] who proposed the genetic algorithm approach to solve the satisfiability problem. For instance, we limit our investigation until 3-Satisfiability problem. Generally, 3-SAT involves a very massive search space since it is considered as NP-hard problem [17]. In this paper, we will use the 3-SAT clauses as a problem in logic programming. On the other words, logic programming can be treated as a problem in combinatorial optimization standpoint [10]. Therefore, it can be carried out in a neural network to obtain desired solutions.

Logic Programming is an auspicious field that can be applied in solving various constraint or combinatorial optimization problems [5, 10]. In this paper, we considered logic programming in Hopfield network. We will translate the 3-SAT clauses into a logic program. The process involved the 3-SAT in Conjunctive Normal form (CNF) with different number of instances. Then, we compute the respective weights and energy values by using Wan Abdullah's method. The conventional model is Wan Abdullah's logic programming based on McCulloch-Pitts function [7]. Wan Abdullah proposed a paradigm of doing the logic program on the Hopfield network [1, 14]. We extended the work related with the neural-symbolic integration by implementing the Hopfield network and activation function in doing logic program.

In order to accelerate the algorithm, we introduced as post optimization technique in order to minimize the logical inconsistency. The common post optimization method is by implementing activation function. As an outcome, we proposed the Hyperbolic tangent activation function and Elliot symmetric activation function as the catalyst of doing logic programming in Hopfield network for 3-SAT clauses. The ability for the activation function to squash and organize the input will enhance the global output obtained after each iteration. The performance analysis of the activation functions will determine the robust paradigm. The best activation function can be used to solve a various combinatorial optimization problem.

This paper is organized as follows. Part II introduces the fundamental concept of 3-Satisfiability and logic programming. In section III, the Hopfield neural network is discussed briefly. Besides, section IV covers conceptual discussion of activation functions. In section V, theory implementation of the activation functions is been discussed. Finally, section VI and VII enclose the experimental results and conclusion of this research.

II. 3-Satisfiability Problem

Strictly speaking, satisfiability (SAT) can be defined as the task of searching a truth assignments that creates an arbitrary Boolean expression true [4, 7]. Basically, the satisfiability problem can be treated as the combinatorial optimization problem according to logic programming perspective. In other words, satisfiability problem refers to decision making problems based on particular constraints. On the contrary, k-SAT problem is deliberated as a NP problem or non-deterministic problem.

The three core components of k-SAT are summarized as follows:

1. A set of m variables, $x_1, x_2, ..., x_m$
2. A set of literals. A literal refers to the variable or a negation of a variable. The number of literals for each clauses depend of the number of variables.
3. A set of distinct clauses: $C_1, C_2 ..... C_n$. Each clause consists of only literals combined by just logical operator OR. Each clause must consist of k variables.

In this paper, we only focus on 3-Satisfiability (3-SAT) clauses.

A. 3-Satisfiability Problem

In this paper, we emphasize a paradigmatic NP-complete problem namely 3-Satsifiability (3-SAT). Generally speaking, 3-SAT can be defined as a formula in conjunctive normal form where each clause is limited to at most or strictly three literals [16]. The problem is an example of non-deterministic problem [15]. In our analysis, the following 3-SAT logic program which consists of 3 clauses and 3 literals will be used. For instance:

$$P = \left(A \vee B \vee \bar{C}\right) \wedge \left(\bar{A} \vee \bar{B} \vee C\right) \wedge \left(\bar{A} \vee B \vee D\right) \quad (1)$$

We represent the above 3 CNF (Conjunctive Normal Form) formula with P. Thus, the formula can be in any combination as the number of atoms can be varied except for the literals that are strictly equal to 3 [19]. Hence, it is vital for combinatorial optimization problem. The higher number of literal in each clause will increase the possibilities or chances for a clause to be satisfied [17].

The general formula of 3-SAT for conjunctive normal form (CNF):

$$P = \bigwedge_{i=1}^{n} Z_i \quad (2)$$

So, the value of k denotes the number of satisfiability [21]. In our case, k-SAT is 3-SAT.

$$Z_i = \bigwedge_{j=1}^{k} (x_{ij}, y_{ij}, z_{ij}), k > 3 \quad (3)$$

B. 3-SAT Logic Programming

Logic programming is the integration of mathematical logic and neural network concept [21]. Logic is deals with true and false while in the logic programming, a set of clauses that formed by atoms are represented to find the truth values of the atoms in the clauses [7, 14]. We use neurons to store the truth value of atoms to write a cost function for minimization when all the clauses are satisfied [6]. The reason we implement logic programming is as follows:

i) Logic program is writable and readable.
ii) Well-founded semantic for further simplification.
iii) Flexible and easy to implement with the other algorithm.
iv) Can be treated as a problem in combinatorial optimization.

So, we can set the Boolean formula according to the clauses that we want. Hence, we can easily integrate the

3-SAT logic programming with Hopfield neural network.

Based on Sathasivam's method [1, 14], the following algorithms summarize on 3-SAT logic programming in Hopfield network.

i) Given a logic program, translate and transform all the 3-SAT clauses in the logic program into basic Boolean algebraic form.
ii) Recognize a neuron to each ground neuron according to the 3-SAT principles.
iii) All connections strengths are initialized to zero. Hence, it assumed the connection with A, B and C is zero value. The other connection strengths are computed by implementing the cost function and Hopfield energy function.
iv) Derive a cost function that is incorporated with the negation of all clauses. For example, $X=\frac{1}{2}(1+S_X)$ and $\overline{X}=\frac{1}{2}(1-S_X)$. $S_X=1$ (True) and $S_X=-1$(False). Multiplication represents CNF and addition represents DNF.
v) Comparing the cost function with energy by obtaining the values of connecting strengths. The values of the connection strengths or weights are vital in logic programming.
vi) Let the neural network to progress until minimum energy is reached. The neural states then provide a solution interpretation for the logic program.
vii) Next, verify whether the solution obtained is a global solution or not.
viii) Compute the global minima ratio, Hamming distance and CPU time.

III. Discrete Hopfield Network

Discrete Hopfield neural network is staple in numerous aspects in artificial intelligence and logic. As an expended form of Hopfield network illustration, discrete Hopfield network is the simplest form [1]. Generally, Hopfield network is extensively been used to solve combinatorial optimization problem and content addressable memory. Beside, Hopfield network has few remarkable features.

One of the features is distributed representation [18]. Hence, memory can be stored as a pattern and overlay upon one another different memories over the equivalent set of processing elements [2, 14]. Order than that, it is distributed and asynchronous control. Every single processing agent creates their own decisions according to their local circumstances [4, 5]. The Hopfield neural network is an attractor neural network. Thus, the units in Hopfield nets are called binary threshold unit [1], which strictly take binary values such as 1 and -1. The fundamental delineations for unit I's activation, a_i are given:

$$a_i=\begin{cases}1 & if \sum_j w_{ij}S_j>\xi_i \\ -1 & Otherwise\end{cases} \quad (4)$$

Where w_{ij} is the connection strength from unit j to i.

We focus on Hopfield energy minimization function in solving combinatorial optimization or satisfiability problem [6]. Hence, an energy function for the discrete Hopfield networks for second order Hopfield networks is formulated as follows:

$$h_i=\sum_j w_{ij}^{(2)}S_j+w_i^{(1)} \quad (5)$$

Since the Lyapunov function is decreasing monotonically with the dynamics. The two-connection model can be generalized to include higher order connections. The "local field" equation is simplified into

$$h_i=\left(\ldots+\sum_j\sum_k w_{ijk}x_jx_k+\sum_j w_{ij}x_j+w_i\right)_i \quad (6)$$

Where "….." indicates still higher orders, and an energy minimization function can be written as follows:

$$E=..-\frac{1}{3}\sum_i\sum_j\sum_k w_{ijk}x_ix_jx_k-\frac{1}{2}\sum_i\sum_j w_{ij}x_ix_j-\sum_i w_ix_i \quad (7)$$

provided that Jijk = J[ijk] for i, j, k distinct, with […] denoting permutations in cyclic order, and Jijk = 0 for any i, j, k equal, and that similar symmetry requirements are satisfied for higher order connections.

The updating rule maintains as in (8).

$$S_i(t+1)=\mathrm{sgn}\left[h_i(t)\right] \quad (8)$$

The network is set up in an initial state, released, and the dynamics will take the network to the closest fixed point. The job of the learning rule of a Hopfield network is to find some weight matrix which stores the required patterns as fixed points of the network dynamics so that patterns can be recalled from noisy or incomplete initial inputs.

The relaxation technique in Hopfield network is an essential method for using contextual information to minimize local uncertainty and achieve global consistency in many applications [2]. Haykin highlighted the solution of an optimization problem for Hopfield neural network can be obtained systematically after the network undergoing relaxation process [5]. Since we are treating the 3-SAT problem as an optimization problem, Sathasivam's relaxation technique can be employed in order to guarantee the network relaxed to the equilibrium state [14].

Precisely, this technique will control certain parameter in the energy function. The function of the relaxation rate in the network dynamics is to regulate the rapidity of the relaxation so that solutions with superior quality can be achieved. According to Sathasivam [1, 14], the network will undergo the relaxation process in order to increase the convergence of the solutions. Thus, the input of the neuron is reorganized via the following equation:

$$\frac{dh_i^{new}}{dt} = R\frac{dh_i}{dt} \tag{9}$$

R represents the relaxation rate and h_i denotes to the local field as in equation (6).

IV. Post Optimization Technique: Activation Function

In order to accelerate the process of doing logic programming in Hopfield network for 3-SAT, we required to implement a post optimization paradigm. So, in this paper, we introduced the activation functions as the catalyst to power up the training process.

According to [14], the solutions of McCulloch-Pitts function will stuck easily in local minima of the energy and it is time consuming to achieve global solutions. Hence, it doesn't reduce the computation burden for training the network.

In our analysis, we integrate the Hyperbolic tangent activation function and Elliot symmetric activation function as catalyst for 3-SAT logic programming in Hopfield network. We combine the advantages of each activation function and Hopfield neural network as a single hybrid network. Other than that, activation functions will be tested as a medium to check the network complexity.

Strictly speaking, the activation function are commonly used to transform the activation level of a unit (neuron) into an output signal in neural network [8]. In layman's term, it is also known as transfer function or squashing function due to the capability to squash the permissible amplitude range of output signal to some finite value [9].

Therefore, the desired output can be generated systematically when doing the logic programming in Hopfield network. Theoretically, it will lead to produce global solutions. In addition, it will assist the Hopfield's content addressable memories to recall the correct global states.

In our study, we punch in the value obtained from the local field equation into the value of x for each of our activation. Hence, the Hopfield local field value is vital before squashing it into the activation function as the input [8, 22]. Then, the formula of generating the output can be summarized as follows:

$$y(x) = \begin{cases} -1, g(x_i) < 0 \\ +1, g(x_i) \geq 0 \end{cases} \tag{10}$$

Hence, if output exceeds the threshold value, the actual state is 1 else the value of the actual state is -1. The updated neuron states via activation functions are crucial in hunting the optimal state which may lead to global solutions. We will implement the following formula after performing the computation in equation (6), (7) and (8).

A. McCulloch-Pitts Function/Linear Activation Function

The McCulloch-Pitts model of a neuron is simple yet has considerable computing prospective. The McCulloch-Pitts function is also has precise mathematical definition and easy to implement. However, this model is so simplistic and tends to generate local minima outputs [9, 14]. Many researcher urged the alternatives for this function as it is very primitive and leads to slower convergence.

$$g(x) = x \tag{11}$$

Another drawback is the linear activation function will only produce positive numbers over the whole real number. Thus, the output values are unbounded and not monotonic. Hence, the outputs obtained will diverges from the global solutions.

B. Hyperbolic Tangent Activation Function

Hyperbolic tangent activation function is proven as the most powerful activation function in neural network. Hence, we want to investigate whether the robustness of this function will apply to the 3-SAT logic programming. Generally, this function can be defined as the ratio between the hyperbolic sine and the cosine functions expanded as the ratio of half-difference and half-sum of two exponential functions in the points x and –x [8, 14].

The main feature of Hyperbolic tangent activation function is the broader output space than the linear activation function. The output range is -1 to 1, similar to the conventional sigmoid function. The scaled output or bounded properties will assist the network to produce good outputs. Hence, the outputs will converge to the global solutions. The hyperbolic tangent activation function can be visualized as follows:

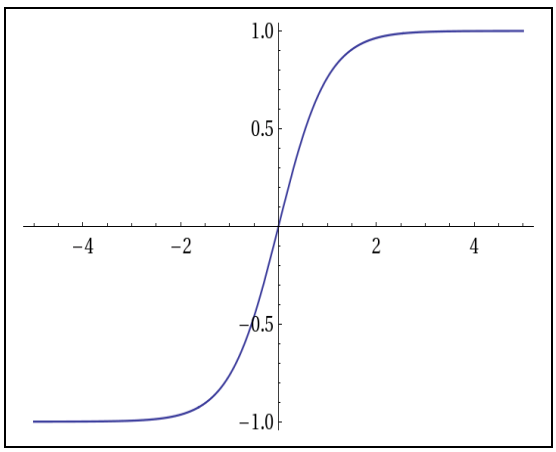

Fig.1. Hyperbolic tangent activation function.

The Hyperbolic tangent activation function is written as follows:

$$g(x) = \tanh(x) = \frac{\sinh(x)}{\cosh(x)} = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \quad (12)$$

The main feature of Hyperbolic tangent activation function is the broader output space than the linear activation function. The output range is -1 to 1, similar to the conventional sigmoid function. The scaled output or bounded properties will assist the network to produce good outputs. Hence, the outputs will converge to the global solutions.

C. Elliot Symmetric Activation Function

According to Sibi *et al.* [9], the Elliot symmetric activation function is a class of transfer function with higher speed approximation. This activation function has an ability to compute on modest computing hardware as it does not involve any exponential or trigonometric functions. The drawback is that it only flattens out for large inputs, so a small number of outputs produced will be trapped into the local minima. As a result, this might result in more training iterations, or require more neurons to achieve the same accuracy.

$$g(x) = \frac{x}{1 + |x|} \quad (13)$$

The output range for Elliot symmetric activation function is -1 to 1. The performance analysis of Elliot symmetric activation function will be compared with the other counterparts as mentioned earlier.

V. Implementation

The implementation consists of methodical procedures. First of all, we define the number of neurons. Next, program clauses are generated randomly based on 3SAT or 1-SAT formula. Thirdly, the initial states are being initialized for the neurons in the clauses.

Then, initialize the connection weights to zero including energy parameters plus with other related parameters for clauses together. The network evolves or relaxes until minimum energy is reached.

During the energy relaxation phase, the neuron state is updated by using activation function. After the network relaxed to an equilibrium state, test the final state obtained for the relaxed neuron whether it is a stable state. If the states remain unchanged for five time steps, then consider it as stable state. Subsequently, compute the corresponding final energy for the stable state. If the difference between the final energy and the global minimum energy is within the tolerance value, then consider the solution as global solution.

Otherwise, we need to return to beginning procedure. Then, we compute the ratio of global solutions and the hamming distance between stable state and global solution. Global solution is the interpretation used in the model for the given logic program or better known as global minima ratio. Global minima ratios are computed via the following formula:

$$Global\ Minima\ Ratio = \frac{Global\ solutions}{Number\ of\ iterations} \quad (14)$$

In addition, we run the relaxation for 1000 iterations and 100 combinations to minimize the statistical error. The selected termination criterion is 0.001. Furthermore, all these values are obtained by trial and error technique, where we tried several values as tolerance values and selected the value which yields better performance than other values. Finally, we compared the experimental results such as global minima ratio, Hamming distance and computation time between the McCulloch-Pitts function, Elliot symmetric activation function and Hyperbolic activation function.

VI. Results and Discussion

For getting more clear result and demonstrate the performance of doing logic programming in Hopfield network for 3-Satisfiability clauses, we integrate various activation functions namely McCulloch-Pitts function, Elliot symmetric activation function and Hyperbolic tangent activation function. The Microsoft Visual C++ 2013 is used as a platform of simulating, training and testing the network.

A. CPU Time

In this paper, the CPU time can be defined as the total time taken for which our network was used for training and generating the maximum satisfied clauses [7, 14]. Hence, the CPU time can be seen as the indicator of the performance of the post optimization paradigms that used in this study.

Table 1. CPU time for various activation functions

Number of Neurons	CPU Time (second)		
	Elliot Symmetric	*McCulloch-Pitts*	*Hyperbolic Tangent*
10	4	10	1
20	25	77	18
30	80	469	56
40	222	9800	156
50	561	84356	360
60	1230	-	974
70	1888	-	1489
80	5764	-	5002

From the table 1, it had shown the CPU time was increased as the network complexity also increased. When we integrate the Hyperbolic tangent activation function in logic programming, this problem can be minimized. So, the Hyperbolic tangent activation function had accelerated the training or computation

process in order to obtain global solutions. Hence, the CPU time is faster than the other counterparts due to less computation burden. Hence, less energy is needed to achieve the convergence. From table 1, it can be seen that McCulloch-Pitts function took more time to generate the global solutions for 3-SAT logic programming. The Elliot symmetric activation function recorded good CPU time but still slower than Hyperbolic tangent activation function. Faster computation time is vital in logic programming, as it is an indicator that we can satisfy the 3-Satisfiability problem within the stipulated time-scale.

B. The global minima ratio

Table 2. The global minima ratio in various methods

Number of Neurons	Global Minima Ratio		
	Elliot Symmetric	*McCulloch-Pitts*	*HyperboliTangent*
10	0.9981	0.9895	0.9999
20	0.9902	0.9811	0.9945
30	0.9843	0.9748	0.9898
40	0.9796	0.9674	0.9812
50	0.9710	0.9566	0.9741
60	0.9656	-	0.9703
70	0.9603	-	0.9658
80	0.9585	-	0.9594

Table 2 depicts the global minima ratio obtained from the computer simulation for different types of activation functions. In our analysis, the global minima ratio obtained for the Hyperbolic tangent activation function in 3-SAT logic programming is better than the Elliot symmetric function and McCulloch-Pitts function. According to the global minima ratio achieved for the Hyperbolic tangent activation function, it can be seen that the value is much closer to 1 compared with the Elliot symmetric activation function and McCulloch-Pitts function although we enriched the network complexity by increasing the number of neurons (NN). This is because the Hyperbolic tangent activation function enhanced the energy relaxation process in order to get better solutions. Thus, the global solutions can be obtained perfectly without being stuck in local minima or sub-optimal solutions. This is due to fewer barriers for the neurons to achieve the global solutions. For the 3-satisfiability problem, the McCulloch-Pitts function unable to sustain more neurons due to the complexity. The solutions seem to oscillate and stuck much easier in local minima. For 3-SAT, the McCulloch-Pitts functions only endured and sustained until 50 neurons. The implementation of Elliot symmetric activation function can help to sustain more neurons but the solutions obtained were slightly closer to the Hyperbolic tangent activation function for both satisfiability problem. However, Hyperbolic tangent activation function performs better in generating better global solutions. For instance, the effective updating ability of the proposed model is a key to perform well in 3-satisfiability logic programming.

C. The global Hamming distance

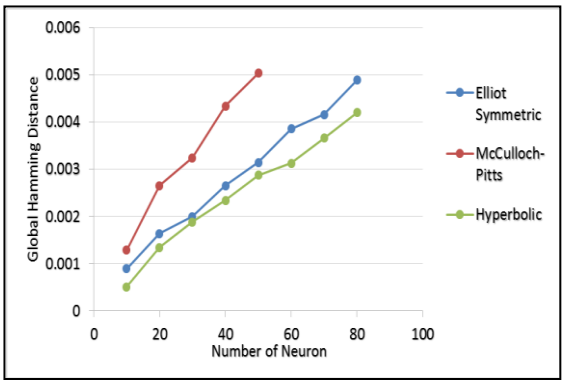

Fig.2. The global Hamming distance in various methods

As shown in Figure 2, the performance of different activation functions in doing logic programming for 3-SAT can be evaluated by analyzing the value of global Hamming distance obtained. The global Hamming distance was initially considered for the sake of errors detection and rectification [1]. Hence, the Hamming distance is measured between the final state and global state of the neurons upon relaxation during training process. From Figure 1, we observed that the Hamming distance for Hyperbolic tangent activation function is much closer to 0 compared to the Elliot symmetric activation function and McCulloch-Pitts function. In our research, we limit the maximum number of neurons until 80. After this value, the complexity of the network will be increased massively. If more than it, memory deterioration will occurs. As the complexity of the network increased, the ability to sustain a huge number of neuron is the main advantages of Hyperbolic tangent activation function. Hence, the Hyperbolic tangent activation function outperformed the Elliot symmetric activation function and McCulloch-Pitts function in term of global hamming distance. Hence, the training process will be accelerated.

VII. Conclusion

We have presented various activation functions to check the robustness in doing 3-SAT logic program. From the theory and experimental results, the Hyperbolic activation function outperformed Elliot symmetric activation function and the McCulloch-Pitts function. The performance analysis is supported by the very good agreement of global minima ratio, global Hamming distance and CPU time obtained. It was proven by computer simulations that the Hyperbolic tangent activation function has a better performance, provides good solutions and achieves an acceptable stability compared to the Elliot symmetric activation function and McCulloch-Pitts function.

References

[1] S. Sathasivam, P. F. Ng, and N. Hamadneh, Developing agent based modelling for reverse analysis method, Journal of Applied Sciences, Engineering and Technology

6 (2013) 4281-4288.
[2] R. Rojas, *Neural Networks: A Systematic Introduction*, Berlin: Springer, 1996.
[3] J. J. Hopfield and D. W. Tank, Neural computation of decisions in optimization problems, Biological Cybernatics 52 (1985) 141-151.
[4] U. Aiman and N. Asrar, Genetic algorithm based solution to SAT-3 problem, Journal of Computer Sciences and Applications 3 (2015) 33-39.
[5] S. Haykin, *Neural Networks: A Comprehensive Foundation*, New York: Macmillan College Publishing, 1999.
[6] G. Pinkas, and R. Dechter, Improving energy connectionist energy minimization, Journal of Artificial Intelligence Research 3 (1995) 223-237.
[7] W. A. T. Wan Abdullah, The logic of neural networks. Physics Letters A 176 (1993) 202-206.
[8] K. Bekir and A. O. Vehbi, Performance analysis of various activation functions in generalized MLP architectures of neural network. International Journal of Artificial Intelligence and Expert Systems 1(4) (2010) 111-122.
[9] P. Sibi, S. A. Jones and P. Siddarth, Analysis of different activation functions using back propagation neural networks. Journal of Theoretical and Applied Information Technology 47 (2013) 1264-1268.
[10] R. A. Kowalski, *Logic for problem solving*, New York: Elsevier Science Publishing Co, 1979.
[11] Y. L. Xou and T. C. Bao, Novel global stability criteria for high-order Hopfield-type neural networks with time-varying delays, Journal of Mathematical Analysis and Applications 330 (2007) 144-15.
[12] N. Siddique and H. Adeli, *Computational Intelligence Synergies of Fuzzy Logic, Neural Network and Evolutionary Computing*, United Kingdom: John Wiley and Sons, 2013.
[13] B. Sebastian, H. Pascal and H. Steffen, Connectionist model generation: A first-order approach, Neurocomputing 71(13) (2008) 2420-2432.
[14] S. Sathasivam, Upgrading Logic Programming in Hopfield Network, Sains Malaysiana, 39 (2010) 115-118.
[15] B. Tobias and K. Walter, An improved deterministic local search algorithm for 3-SAT, Theoretical Computer Science 329 (2004) 303-313.
[16] D. Vilhelm, J. Peter and W. Magnus, Counting models for 2SAT and 3SAT formulae, Theoretical Computer Science, 332 (2005) 265-291.
[17] L. J. James, A neural network approach to the 3-Satisfiability problem, Journal of Parallel and Distributed Computing 6(2) (1989) 435-449.
[18] H. Asgari, Y. S. Kavian, and A. Mahani, A Systolic Architecture for Hopfield Neural Networks. Procedia Technology, 17 (2014) 736-741.
[19] P. Steven, SAT problems with chains of dependent variables, Discrete Applied Mathematics 130(2) (2003) 329-22.
[20] K. Ernst, B. Tatiana and C. W. Donald, *Neural Networks and Micromechanics*, New York: Springer Science & Business Media, 2009.
[21] C. Rene and L. Daniel, Mathematical Logic: Propositional Calculus, Boolean Algebras, Predicate Calculus, United Kingdom: Oxford University Press, 2000.
[22] M. Kaushik. Comparative analysis of exhaustive search algorithm with ARPS algorithm for motion estimation. International Journal of Applied Information Systems 1(6) (2012) 16-19.

Authors' Profiles

Mohd Asyraf Mansor was born in Sarawak, Malaysia in 1990. He obtained his MSc (2014) and BSc(Ed) (2013) from Universiti Sains Malaysia. He is currently pursuing Ph.D degree at School of Mathematical Science, Universiti Sains Malaysia.

Saratha Sathasivam is working as Associate Professor at School of Mathematical Sciences, Universiti Sains Malaysia. She received her MSc and BSc(Ed) from Universiti Sains Malaysia. She received her Ph.D at Universiti Malaya, Malaysia. Her current research interest are neural networks, agent based modeling and constrained optimization problem.

How to cite this paper: Mohd Asyraf Mansor, Saratha Sathasivam, "Accelerating Activation Function for 3-Satisfiability Logic Programming", International Journal of Intelligent Systems and Applications (IJISA), Vol.8, No.10, pp.44-50, 2016. DOI: 10.5815/ijisa.2016.10.05

