ISSN 2278-3091
Volume 9, No.3, May - June 2020
International Journal of Advanced Trends in Computer Science and Engineering
Available Online at http://www.warse.org/IJATCSE/static/pdf/file/ijatcse136932020.pdf
https://doi.org/10.30534/ijatcse/2020/136932020

Accelerating Hyperbolic Tangent Activation Function in Neuro Symbolic Integration using Agent Based Modelling

Shehab Abdulhabib Alzaeemi[1], Saratha Sathasivam[2*]
[1] School of Mathematical Sciences, Universiti Sains Malaysia, 11800 USM, Penang Malaysia. Email: shehab_alzaeemi@yahoo.com
[2,*]School of Mathematical Sciences, Universiti Sains Malaysia, 11800 USM, Penang Malaysia. Email: saratha@usm.my

ABSTRACT

Artificial Neural Network are roughly based on our brains neural network not in the way that they are made up of a biochemical mixture, but in the way that multiple nodes or neurons are interconnected and signals can pass through these nodes. It is incredibly exciting to study the mathematical relations of deep neural networks and ways of improving them. Activation functions introduce nonlinear properties to neural network. By using a nonlinear activation, the mapping of the input to the output is nonlinear. The whole idea of activation functions is to roughly model the way neurons communicate with each other. In this paper, we shall discuss on the differences of various activation functions, focusing on Hyperbolic Tangent Activation Function, McCulloch-Pitts Activation Function, Zeng Martinez Activation Function and Bipolar Activation Function. Performance comparisons of these activations functions in Hopfield neural network to do logical programming will be carried out. The outcome of the simulation of the Agent-Based modelling by using Netlogo program in this paper is that the Hyperbolic Tangent Activation Function outperform the other activation functions to raise the capacity of Hopfield network to do logic programming.

Key words: Agent-Based Modeling, Activation Functions, Neural Network, Logic Programming.

1. INTRODUCTION

Artificial Neural Network (ANN), or commonly known as a connectionist system, are system ambiguously exhilarated by the biological neural network. Today's computers possess an immense computing and information processing capabilities, in which case has been presumed that the computer has advanced more in its abilities compared to a human brain [1]. It plays a vital role in our daily life, manufacturing sites as well as scientific research. Some ANNs are recognized as basic functional units, where it has the ability to create building blocks for a more sophisticated network [2]. Evolution has settled on a specific kind of structure for our brain that can represent layers of abstraction in an ordered manner. When this rule is modeled on computers, similar results are obtained, even though the substrate is different from each other. The basic idea of how a neural network works is where a vectorized input is fed into the network, whereby a series of matrix operations are performed on this input data layer by layer. In the simplest case, the input is multiplied on each layer by the synaptic weights and added to a bias value.

Then the activation function is implemented to the system/results and the output of the current layer is then passed on to the next layer to carry out the same task, repeated this process until it reaches the last layer. The final output from this is merely a prediction, hence finding the difference with the expected output will provide an error. Activation functions introduced non-linear attributes to the neural networks. Neural networks are considered to be Universal Function Approximators, which means that the network has the capability to compute any type of function. This paper shall contribute in determining which of the four activation functions performs the best. The activation functions identified are Hyperbolic Tangent Activation Function, McCulloch-Pitts Activation Function, Zeng Martinez Activation Function and Bipolar Activation Function.

2. RESEARCH METHOD

2.1 Hopfield Neural Network (HNN)

HNN is one of the conventional neural networks discovered by John Hopfield. Hopfield Neural Network is a reoccurring neural network with very high capacity in memory, storage, and learning [3]. HNN takes possession of synaptic connection patterns in which there is a Lyapunov energy function for the activities dynamic. The state of the neuron in the system tends to a final stage when the state is minima to the Lyapunov energy function. An example of the discrete HNN construction with three neurons [4] is represented in Figure 1 below.

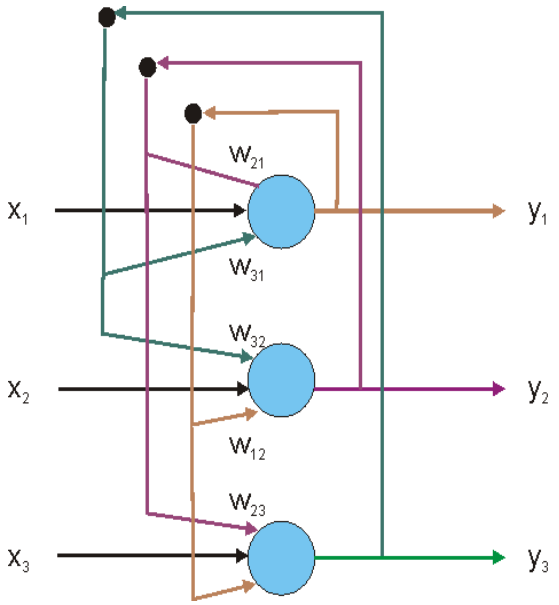

Figure 1: Hopfield Neural Network

HNN is enhanced by combining multiple interconnected neurons, which are generally known as bipolar threshold neurons[4] as the following threshold function:

$$S_i = \begin{cases} 1 & if \ \ h_i(t) > \theta_i \\ -1 & Otherwise \end{cases} \quad (1)$$

where S_i is the status or outputs of the i^{th} unit, $h_i(t)$ is the local field of HNN, and θ_i is threshold of unit i. The following equation shows the local field of HNN:

$$h_i(t) = \sum_j J_{ijk}^{(3)} S_j S_k + \sum_j w_{ij}^{(2)} S_j + w_i^{(1)} \quad (2)$$

Lyapunov or energy equation will be used to investigate the final state of the neurons in HNN by using following equation [5]:

$$E = -\frac{1}{3}\sum_i \sum_j \sum_k w_{ijk}^3 S_i S_j S_k - \frac{1}{2}\sum_i \sum_j w_{ij}^{(2)} S_i S_j - \sum_i w_i^{(1)} S_i \quad (3)$$

2.2 Logic Programming in Hopfield Network

The logic programming basically comprises of a set of programming clauses and a commencing goal statement will activate the logic program [7]. The clauses possess only one positive literal or neuron in a Conjunctive Normal Form (CNF). In a Hopfield Network, logic program can be managed as a combinatorial optimization problem [8]. A desired solution can be obtained given a logic program in a neural network. A set of Horn clauses in the form $A \leftarrow B_1, B_2, \ldots, B_n$, where the arrow is read as "if" and the commas are interpreted as "and", are provided with the goal of searching the interpretation sets [9]. A logic program in whole is much easier to understand, verify and change compared with neural network.

A direct method was proposed by Wan Abdullah method [10] on the steps of how a logical programming can be conducted in a Hopfield neural network.

1. With a given logic program, all clauses are translated into a normal Boolean algebraic configuration.
2. From each ground neuron, a neuron or literal is selected.
3. All connection strengths are initiated to zero.
4. A cost function is established, which is connected with all the clauses' negation, in a way that the logic value of neuron X is depicted by $\frac{1}{2}(1 + S_x)$, where S_x is the neuron corresponding to X. S_x value is 1 if X is ture, and -1 is X is false. $\frac{1}{2}(1 - S_x)$ exhibits the negation if neuron X does not occur.
5. Connection strengths' values are attained by comparing the Lyapunov energy function with cost function.
6. Till the minimum energy is achieved, the neural networks will evolve and the solution will be scrutinized whether a global solution is attained.

2.3 Activation Functions

In the neural networks which inspired biologically, the activation function is customarily a concept describing the potential firing rate in the cell. In its simplest form, this activation function is binary or bipolar that is, unless the neuron is firing or not [11]. There are several significant activation functions used in Hopfield Neural Network. Research and engineering widely uses the most common option of activation functions as transfer functions. Its universality is due to several reasons, mainly its limits in the unit period, rapid computability of the function and its derivative and several adjustable mathematical properties in the field of rounding theory.

Their net inputs are the most vital units in the structure of the neural network, using a scalar-to-scalar function known as transfer function or threshold function, hence creating a result known as the unit's activation [12]. This paper shall discuss on four activation functions, which are Hyperbolic Tangent Function, McCulloch-Pitts Function, the Zeng Martinez Function and Bipolar Function.

A. Hyperbolic Tangent Activation Function

Hyperbolic Tangent Activation Function, or commonly known as TanH, is used as an alternative to sigmoid functions [13]. The derivative of this activation function will be included in the calculation for error effects on weights when

there is a back propagation. The outputs produce by this function are [-1, +1], and this is an interminable function. Figure 2 shows the Hyperbolic Tangent Function.

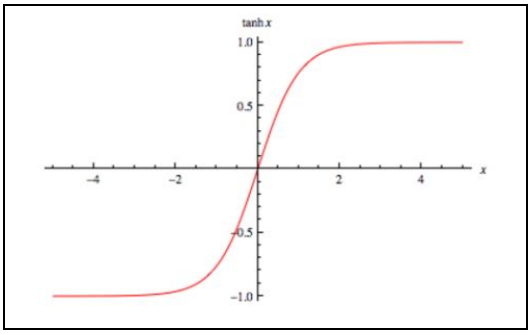

Figure 2: Hyperbolic Tangent Function

It is principally the ratio of hyperbolic functions as defined in the following:

$$\tanh(x)=\frac{\sinh(x)}{\cosh(x)}=\frac{e^{x}-e^{-x}}{e^{x}+e^{-x}} \quad (4)$$

A study on the application of Hyperbolic Tangent Activation Function was conducted by Zamanlooy and Mirhassani [14] on efficient approximation scheme for VLSI Implementation. The estimation is purely based on a mathematically analysis, with the design parameter being the maximum allowable parameter. The TanH function resulted in a depletion of delay, area and power. This activation function is a non-liner function, therefore there is a high chance of stacking multiple layers. However, Hyperbolic Tangent Activation function has a difficulty with a dissipating gradient.

B. McCulloch-Pitts Activation Function

McCulloch and Pitts [15] first introduced a neuron's mathematical model (M-P) model in the year 1943. It was from then where numerous artificial neural networks were invented from this model [16]. An M-P neuron consist of a component with n inputs and only one output as shown in figure 3, and is given by the following function:

$$g(x)=f\left(\sum_{i=1}^{N}x_i W_i\right) \quad (5)$$

where W_i is the weight factor and X_i is the input vector. There were a few limitations in the original M-P model. Due to that, some additional attributes were added, in which case, permitted them to learn. The perceptron essentially is an M-P model where the inputs are directed firstly into some "preprocessors", or known as association units. These units then recognize certain specific characteristics in the inputs. In other words, it plays a role of a pattern recognition device. Hence, the association units proportionate to the feature. Linear Threshold Function as shown in figure 4.

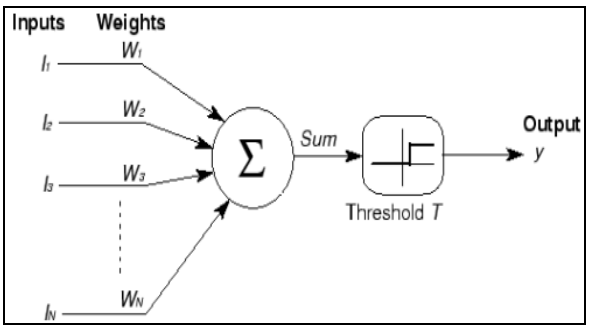

Figure 3: McCulloch-Pitts Activation Function

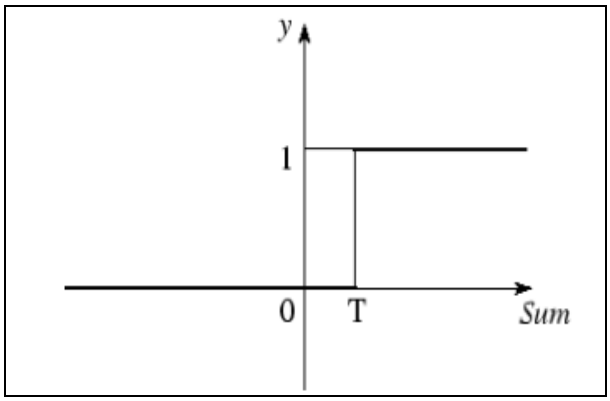

Figure 4: Linear Threshold Function

Takefuji and Lee [17] demonstrates a hysteresis binary McCulloch-Pitts Activation Function in order to overpower a complex neural dynamics' oscillatory pattern. It is utilized for organizing a time-multiplex crossbar switches, so that it will portray the hysteresis effect. The system's quality solution does not degenerate in relation to the size of the problem.

The McCulloch-Pitts Activation Function is straightforward, but it does have a significant computing prospective and an accurate mathematical definition. In spite of that, it only generates a binary output due to its simplicity.

C. Zeng Martinez Activation Function

Zeng Martinez Activation Function in HNN is defined as the sigmoid function as shown in the equation (8). Nonetheless, this activation function places further much focus on insignificant noise disturbances rather than on cost-related signals and network-encoded constraints.

$$V_{x_i}=\begin{cases}\dfrac{0.5\left(1+\tanh\left(\dfrac{U_{x_i}+x_0}{u_0}\right)\right)}{1+\tanh\left(\dfrac{x_0}{u_0}\right)} & ,\left(U_{x_i}<0\right)\\[2ex] \dfrac{\tanh\left(\dfrac{x_0}{u_0}\right)+0.5\left(1+\tanh\left(\dfrac{U_{x_i}-x_0}{u_0}\right)\right)}{1+\tanh\left(\dfrac{x_0}{u_0}\right)} & ,\left(U_{x_i}\geq 0\right)\end{cases} \quad (6)$$

where U_{x_i} is initial states, V_{x_i} is activation function, x_0 symbolizes the threshold for V_{x_i}, and u_0 is measurement the steepness of Zeng Martinez Activation Function. Basically, the input of the neurons is updated by following equation:

$$S_i(t+1) = \mathrm{sgn}[g(h(t))] \quad (7)$$

whereas $h_i(t)$ is local field of HNN. Zeng Martinez Activation Function is in the style of g(h) as followed:

$$g(h_i) = \begin{cases} \dfrac{0.5\left(1+\tanh\left(\dfrac{h_i + x_0}{u_0}\right)\right)}{1+\tanh\left(\dfrac{x_0}{u_0}\right)} & ,(h_i < 0) \\ \dfrac{\tanh\left(\dfrac{x_0}{u_0}\right) + 0.5\left(1+\tanh\left(\dfrac{h_i - x_0}{u_0}\right)\right)}{1+\tanh\left(\dfrac{x_0}{u_0}\right)} & ,(h_i \geq 0) \end{cases} \quad (8)$$

The neuron state updating rules are integrated with Zeng Martinez Activation Function that is given in equation (11). This feature will withstand noise and act well when the network is becoming larger [21, 22].

$$S_i(t+1) = \begin{cases} 1 \ for \ \ g(h_i) \geq 0 \\ -1 \ for \ \ g(h_i) < 0 \end{cases} \quad (9)$$

D. Bipolar Sigmoid Activation Function

Bipolar Sigmoid Function is popular activation function as it is easily differentiable. These details are essentially essential to backpropagation training algorithms [19]. The function of Bipolar Sigmoid activation function is given as follows:

$$g(x) = \frac{1-e^{-x}}{1+e^{-x}} \quad (10)$$

Bipolar Sigmoid function, compared to a Binary Sigmoid function, gives a finer convergence speed. The function shows a similar pattern, but with a value range of {-1, 1}. Figure 5. show the Bipolar Sigmoid Function.

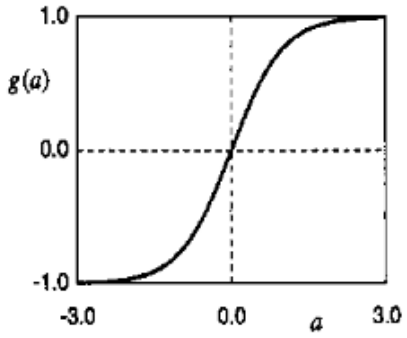

Figure 5: Bipolar Sigmoid Function

The graph portrays the monotonicity as well as the capability to differentiate the function straight forward. In a research conducted on unemployment rate with the highest education completed [20], it utilized a combination of Levenberg-Marquardt algorithm with bipolar sigmoid function. Two stages of data processing were conducted, whereby the first stage was recognition of pattern, and the following is by prediction.

The advantages of the Bipolar Sigmoid Function is that it "squashes" the output values within the range of [-1, +1]. Similar to the Binary Sigmoid Function, it however has a small gradient value.

2.4 Comparison between Activation Functions

The performances of the recommended methods, Hyperbolic Tangent Activation Function, McCulloch-Pitts Activation Function, Binary Activation Function and Bipolar Activation Function, are analyzed by using the emphasis on its theoretical features, which will lead towards the hypothesis, and experimental features, where this shall uphold the results. In Table 1. show a comparison between the proposed Activation Functions.

Table 1: Comparison between the proposed Activation Functions

Function Type	**Hyperbolic Tangent Function**	**McCulloch-Pitts Function**	**Zeng Martinez Function**	**Bipolar Sigmoid Function**
Type	Hyperbolic Tangent Function is similar to sigmoid function	The McCulloch-Pitts Function portrays a sign function	Zeng and Martinez function is known as an "S"-shaped function [21]	Bipolar Sigmoid Function is a "squashing" type function.
Output Range	[-1, +1]	$[-\infty, +\infty]$	[0, +1]	[-1 , +1]
Bounded Function	Yes	No	Yes	Yes
Applications	The TanH function resulted in a depletion of delay, area and power in the efficient approximation scheme of VLSI implementation	M-P model is utilized for organizing a time-multiplex crossbar switches, so that it will portray the hysteresis effect.	Zeng and Martinez Activation function streamlines the equations of the models and exhibits a similarity pattern immanent in the shapes of all the hysteresis curves	Bipolar sigmoid function has the capability to differentiate the function straight forward
Advantages	The derivative of this activation function will be included in the calculation for error effects on weights when there is a back propagation	McCulloch-Pitts Activation Function is straightforward, but it does have a significant computing prospective and an accurate mathematical definition	Zeng Martinez Activation Function doesn't stimulate all the neurons at the same time. Because the output of some neurons is zero so only, wherefore some neurons are stimulated in the network, which makes the network sparse, efficient, and easy for computation.	Bipolar Function is popular as it is easily differentiable
Limitations	Tangent Activation function has a difficulty with a dissipating gradient.	McCulloch-Pitts model only generates a binary output due to its simplicity	The gradients of this function is too small, therefore it has the ability to vanish. This will then lead to the network restricting itself from enhancing further [22]	Bipolar sigmoid function has a small gradient value, therefore network will refuse to enhance the learning process

From right here, this study will be developed Agent-Based Modeling (ABM) by utilizing NETLOGO as the platform to show how Hopfield Neural Network doing Logic Programming (LP) with different activation functions.

3. AGENT BASED MODELLING

NETLOGO softwere was edited and developed by "Uri Wilensky" who is the director of the "Center for Connected Learning and Computer-Based Modelling" as Agent-based modeling at Northwestern University [23]. Agent-based modelling (ABM) is a methodology widely utilized in a wide area of social sciences [24]. This requires the development of a computer model consisting of "agents" representing both a social entity and a "world," NETLOGO being an agent-based language for programming and an essential framework for modeling. NETLOGO was planned to be a "low threshold and no ceiling" in the spirit of logical programming. NETLOGO teaches programming concepts in tortoises, patches, connections and observers using agents [25]. NETLOGO was designed for a variety of publics, particularly: educational children and field experts without programming history to model-related phenomena. NETLOGO has been used in many academic publications [26]. When connecting agents to mobile devices, they build networks, charts and aggregates to allow users to gain a greater understanding of system performance. In fact, the runs are exactly cross-platform reproducible. An agent-based simulation is used to model the interaction of HNN to do LP by utilized different activation functions such as the Tangent Activation Function,

McCulloch-Pitts Activation Function, Zeng Martinez Activation Function and Bipolar Activation Function. Agent-Based Modeling uses NETLOGO as a platform. A flow chart shows how ABM for comparing the activation functions is created in the following figure 6 and the figure 7 shows a screenshot of the diagram model.

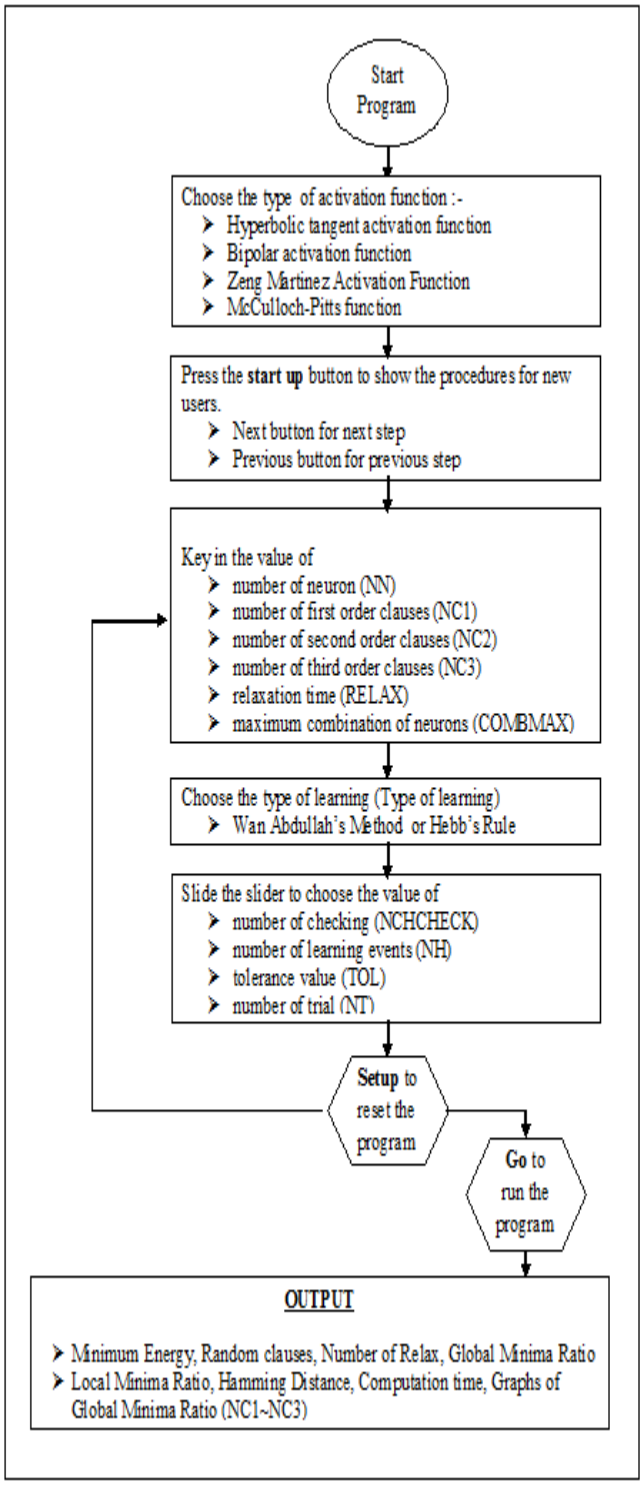

Figure 6: Flowchart of Agent-Based Modeling of HNN with Activation Functions [27]

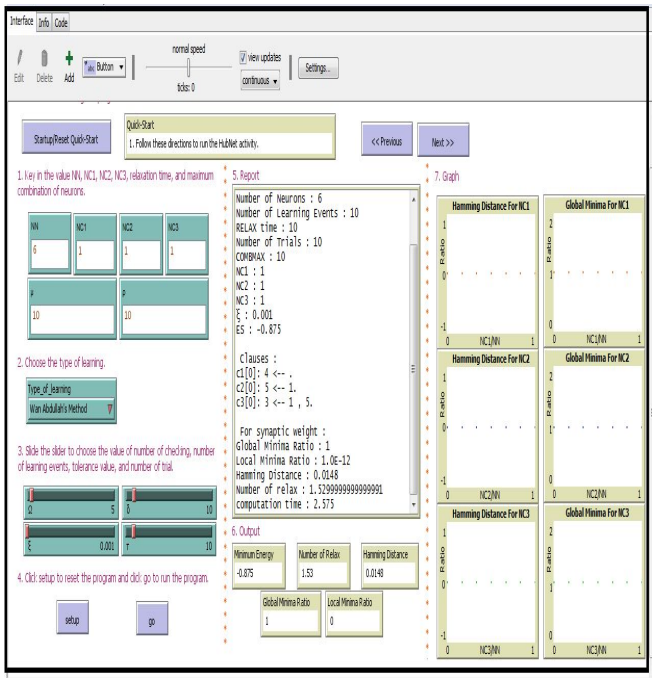

Figure 7: The screenshot of the NETLOGO software for doing Agent-Based Modeling

Figure 7 above is explained as follows:

Input Data:

STAGE 1: Entering the Value of Each Parameters

1. Press the startup / Reset Quick-Start button for the new user. Users will click the next button to take the next step and the last step.

2. In this step enter in the button the value of *NN*, NC1, NC2, NC3, and setups the value of Relax μ, value of COMBMAX ρ.

3. In this step should to choose type of Activation Function and learning.

4. In the step should to slides the slider to settle number of trial τ, number of learning events δ while the maximum of tolerance ξ is 0.001.

5. After adjusting each value, press the setup button in the program to adjust these values and set.

6. Finally, press the button of "go" to run the programming which in turn will be taken place to generating randomlly programmming clauses.

Output Data:

STAGE 2: Training

7. Initialising initial states of neuron in each clause. Based on the design of HNN that derived from the behaviors of neurons movement in the magnetic domain, all neuron will communicate with each other in a complete bonded form as all of them tries to reach active appropriate state and this means minimal function energy. This case is called activation by using the activation function. This state of neurons rotates until allowing every other to keep rotating. Thus, let the network evolve to the minima capacity when the minima energy equals the energy prerequisite to achieve the global minima.

8. The neurons relaxed if the state of neurons remainders unchanged after five running then the system in the network will classify this is the final state of the neurons as the stable state.
9. Goes to compute the final energy of the stable state.
10. Accept the final solution as a global solution within the variation between the final energy and the minima energy within the tolerance value, or go back to step1.
11. Finally, computing the ratio of global solutions and the program of the system will be printing out all the information of each running as shown in **Figure 7**.

4. RESULTS AND DISCUSSION

4.1 Experimental

Computer simulations have been developed using software NETLOGO version 6.1.1, which is a feature of a state-of - the-art device and buttons that minimize the length of the program. The global minimal ratio, hamming distance, and calculation time were experimentally achieved with computer simulations for the four activation functions. We examined all levels of the provisions including NC_1, NC_2 and NC_3 ($1 \leq NC_i \leq 15$) for different number of neurons ($9 \leq NN \leq 135$) for each of these results all value of the parameters are chosen by try and error technique with different setup as shown in the Table 2. The outputs of the system as Ratio of the Global Minimum (*zM*), Hamming Distance (*HD*), and *CPU time* will be helping the researcher in endorsing the performance of all of the activation functions to do logic program in HNN.

Table 2: List of Parameters

Parameter	Parameter Value
Ω	5
δ	100
ξ	0.001
τ	100
μ	100
ρ	100

4.2 Discussion

A. Global minima ratios (zM):

zM is the ratio between global total minima energy and total simultaneous numbers is defined by Kasihmuddin *et al* and Alzaeemi *et a l*[28, 29]. When the final energy is within the limit, it is known as global minimum energy. The ratio of global minimum equation is defined as:

$$zM = \frac{1}{tc}\sum_{i=1}^{n} N_P \qquad (11)$$

The local minimum ratio equation of is defined as [30]:

$$L_m = 1 - zM \qquad (12)$$

where *c* is the combination of the neuron, *t* is the trial, and N_P is the global minimum energy number of the propose system. The higher accuracy of HNN model has higher value of zM.

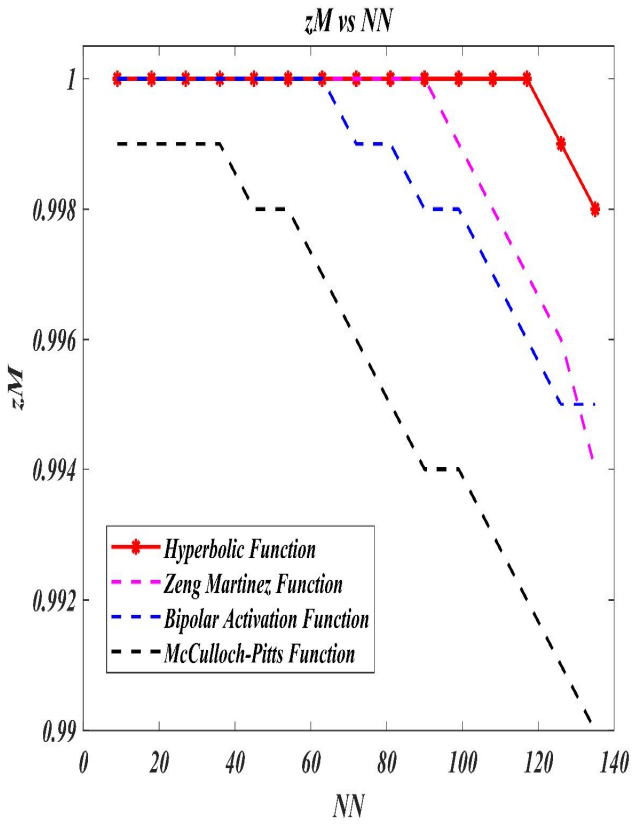

Figure 8: Global minimum ratio for all activation functions

In this paper we implemented the Tangent Function, McCulloch-Pitts Function, Zeng Martinez Function and Bipolar Function to stimulate the performance to do logical program in HNN. Figure 8 show the results for *zM* obtained for both activation functions for doing LP in HNN based on Wan Abdullah's method for the different number of literals per clauses with the different number of neurons and with values of setups as number of learning events δ, number of trial τ, Relax μ, and COMBMAX ρ. We can observe that the global solutions ratio *zM* for Hyperbolic Function is equal to or equal to 1 compared to Zeng Martinez Function, Bipolar Function, and McCulloch-Pitts Function, even though by increasing the number of neurons (NN) led to increase the network complexity with different value of setups as number of learning events δ, number of trial τ, Relax μ, and COMBMAX ρ in Figure 8 still *zM* of Tangent Activation Function equal to or closer to 1. This indicates that the output by Tangent Activation Function in HNN is almost all solutions are "correct". It can be seen that when the network

becomes complex or larger, the hyperbolic tangent activation function still can accommodate more neurons and produce a good output. Almost all the solutions by using the Tangent Activation Function are global minimum solutions.

B. Hamming Distance (HD):

Hamming Distance has defined as the dimension between the global solution and the stable state of the neurons [31].

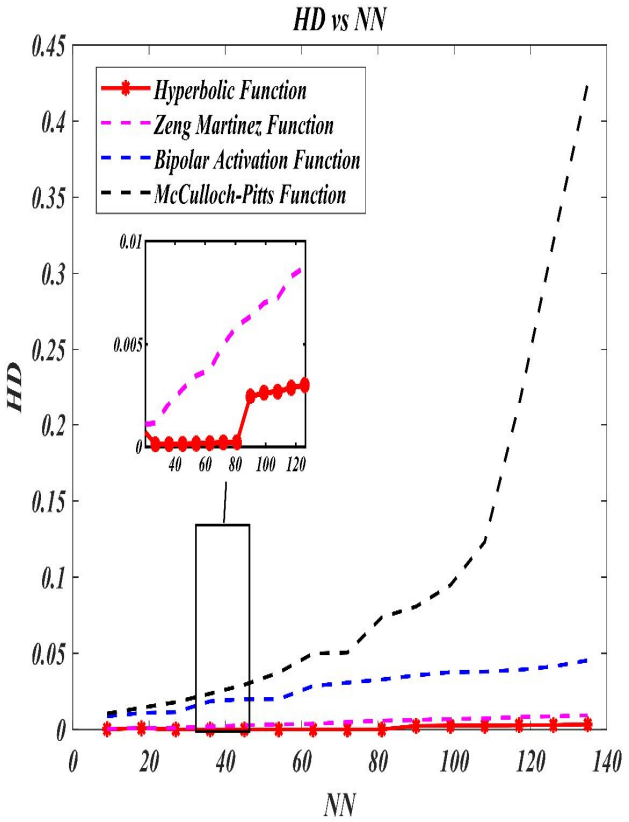

Figure 9: *HD* for all activation functions

Figure 9 show the *HD* for doing LP in HNN by using different activation functions as Hyperbolic Function, McCulloch-Pitts Function, Zeng Martinez Function and Bipolar Function in various literals numbers in clauses with the different neurons number and values of setups parameters as number of learning events δ, number of trial τ, Relax μ, and COMBMAX ρ. It can be observed that the HD for Hyperbolic Function is equaled to 0 or near to 0 compared to Zeng Martinez Function, Bipolar Function, and McCulloch-Pitts Function despite of increased the complexity of the network by increase the neurons number (NN) as shown in both Figure 9. As the network complexity increased, Hyperbolic Function outperformed Zeng Martinez Function, Bipolar Function, and McCulloch-Pitts Function in terms of HD. Hence, the training process in HNN by Hyperbolic Function will be accelerated better than Zeng Martinez Function, Bipolar Function, and McCulloch-Pitts Function, respectively.

C.*CPU time*:

By this study, we define the *CPU time* is the total time that taken for the network was utilized for generate maxima satisfied clauses in the logic programming by using different activation function that used in this paper [32, 33]. *CPU time* is given by follow equation [34]:

$$CPU\_Time = Learning\_Time + Retrieval\_Time \quad (12)$$

As defined by Sathasivam & Abdullah [35], the best model HNN hat has the least *CPU time* during learning phase and the retrieval phase. Consequently, the best model of HNN with the different activation functions will have the shortest *CPU time* in this study.

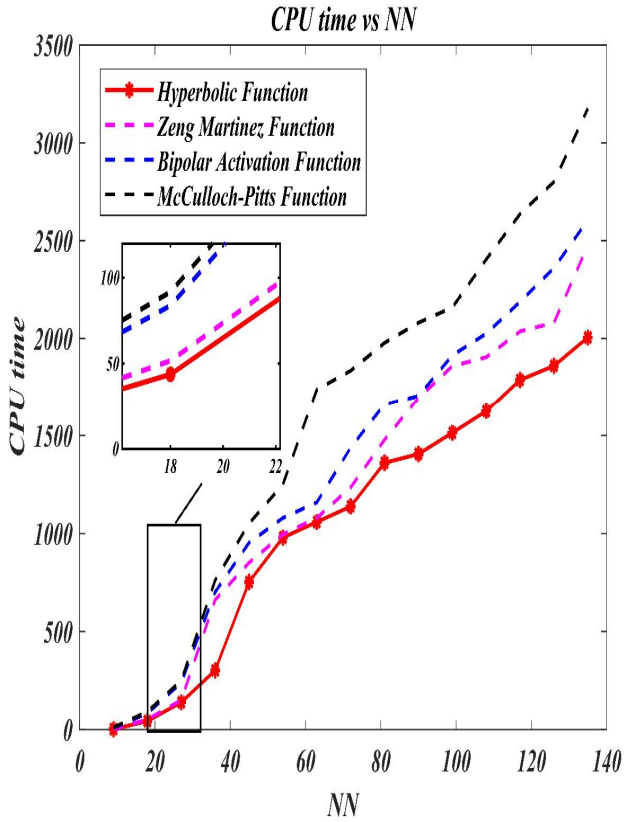

Figure 10: *CPU time* for all activation functions

Figure 10 shows the *CPU time* to do LP in HNN by using different activation functions as Hyperbolic Function, McCulloch-Pitts Function, Zeng Martinez Function and Bipolar Function in various literals number in clauses with different neurons number and with values of setups as number of learning events δ, number of trial τ, Relax μ, and COMBMAX ρ. Figure 10 showed that the *CPU time* of Hyperbolic Function outperforms the *CPU time* of Zeng Martinez Function, Bipolar Function, and McCulloch-Pitts Function, respectively. Hyperbolic Function is preferred compared to other Activation Function especially when the network gets longer because by using Hyperbolic Function the

network manages to find quickly the global minima due to the derivatives of Hyperbolic Function are longer than Zeng Martinez Function, Bipolar Activation Function, and McCulloch-Pitts Function. In other words, the cost function is minimized faster when Hyperbolic Activation Function is utilized.

Finally, from the results above, the study can conclude that Hyperbolic Function is better than Zeng Martinez Function, Bipolar Function, and McCulloch-Pitts Function to do LP in HNN, respectively. The estimated output or restricted properties of Tangent Activation Function will support the network to provide good outputs. Due to this, the output will be reached global solutions faster.

5. CONCLUSION

In this paper had developed agent-based modeling (ABM) to model the acceleration ability of the four various activation functions: Hyperbolic Function, McCulloch-Pitts Function, Zeng Martinez Function and Bipolar Function for doing logic programming (LP) in Hopfield Neural Network (HNN) by using NETLOGO as the platform. ABM presents a natural explanation of the system to be ready to integrate/link of activation function to do logic programming (LP) in Hopfield Neural Network (HNN). From the experimental results obtained for ratio of global minima, hamming distances and CPU time, Hyperbolic Function outperformed Zeng Martinez Function, Bipolar Function, and McCulloch-Pitts Function in doing LP in HNN. The setups parameters for the number of learning events δ, the number of trial τ, Relax μ, COMBMAX ρ were used for the simulation. The results obtained verified our proposed theory that Hyperbolic Activation Function outperformed Zeng Martinez Activation Function, Bipolar Activation Function, and McCulloch-Pitts Activation Function. Also, it can have observed that Zeng Martinez Activation Function outperformed Bipolar Activation Function, and McCulloch-Pitts Activation Function, and Bipolar Activation Function outperformed McCulloch-Pitts Activation Function.

ACKNOWLEDGEMENT

This research is supported by Research University Grant (1001/ PMATHS/8011131) from Universiti Sains Malaysia.

REFERENCES

1. A. Al-Qerem, & A. Alahmad, (2019). ***Human Body Poses Recognition Using Neural Networks with Data Augmentation***. International Journal of Advanced Trends in Computer Science and Engineering, 8 (5), 2117-1220. https://doi.org/10.30534/ijatcse/2019/40852019
2. B. K. Durga, V. Rajesh, (2019). ***In Illumination Variations Facial Emotion Recognisation by Using Local Ternary Patterns.*** International Journal of Advanced Trends in Computer Science and Engineering, 8 (5), 2591-2596. https://doi.org/10.30534/ijatcse/2019/110852019
3. J.J. Hopfield, (1982). **Neural networks and physical systems with emergent collective computational abilities**. *Proceedings of the national academy of sciences*, *79*(8), 2554-2558.
4. S. Sathasivam, N.P. Fen, & N. Hamadneh, (2013). **Developing agent based modelling for reverse analysis method**. *Journal of Applied Sciences, Engineering and Technology*, *6*(22), 4281-4288.
5. M.S.M. Kasihmuddin, M. A. Mansor, & S. Sathasivam, (2017). **Hybrid Genetic Algorithm in the Hopfield Network for Logic Satisfiability Problem**. *Pertanika Journal of Science & Technology*, *25*(1). https://doi.org/10.1063/1.4995911
6. G. Pinkas, (1991). **Energy minimization and the satisfiability of propositional logic**. In *Connectionist Models* (pp. 23-31). Morgan Kaufmann.
7. M., Mansor, S. Z. Mohd Jamaludin, M. S. Mohd Kasihmuddin, S. A. Alzaeemi, M. F. Md Basir, & S. Sathasivam, (2020). **Systematic Boolean Satisfiability Programming in Radial Basis Function Neural Network**. *Processes*, *8*(2), 214.
8. S. Sathasivam, M. Mansour, M. S. M. Kasihmuddin, & H. Abubakar, (2020). **Election Algorithm for Random k Satisfiability in the Hopfield Neural Network**. *Processes*, *8*(5), 568.
9. S. Sathasivam, & W.A.T.W. Abdullah, 2010. **The Satisfiabilty Aspect of Logic on Little Hopfield Network**. *SSJ*, ***1***(2), p.2.
10. W.A.T. Wan Abdullah, ***Logic Programming on a Neural Network***, Int.J. Intelligent Sys, 7, (1992), pp 513-519. https://doi.org/10.1002/int.4550070604
11. B. Karlik, and A.V. Olgac, 2011**. Performance analysis of various activation functions in generalized MLP architectures of neural networks**. *International Journal of Artificial Intelligence and Expert Systems*, *1*(4), 111-122.
12. R. P. Lippmann, "**An introduction to computing with neural nets**". IEEE Acoustics, Speech and Signal Processing, **4**(2):4–22, 1987
13. G. Cybenko, 1989. **Approximation by superpositions of a sigmoidal function**. *Mathematics of control, signals and systems*, ***2***(4), pp.303-314.
14. S. Gomar, M. Mirhassani, & M. Ahmadi, (2016, November). **Precise digital implementations of hyperbolic tanh and sigmoid function**. In *2016 50th Asilomar Conference on Signals, Systems and Computers* (pp. 1586-1589). IEEE.
15. W. S. McCulloch and W. Pitts, (1990). **A logical calculus of the ideas immanent in nervous activity**. *Bulletin of mathematical biology*, *52*(1-2), 99-115.

16. K. Saggie-Wexler, A. Keinan, & E. Ruppin, (2006). **Neural processing of counting in evolved spiking and McCulloch-Pitts agents**. *Artificial Life*, *12*(1), 1-16. https://doi.org/10.1162/106454606775186428
17. G. Xia, Z. Tang, Y. Li, & J. Wang, (2005). **A binary Hopfield neural network with hysteresis for large crossbar packet-switches**. *Neurocomputing*, *67*, 417-425.
18. L. M. Zhang, (2015, October). **Genetic deep neural networks using different activation functions for financial data mining**. In *2015 IEEE International Conference on Big Data (Big Data)* (pp. 2849-2851). IEEE.
19. S. Jeyanthi, & M. Subadra, (2014, May). **Implementation of single neuron using various activation functions with FPGA**. In *2014 IEEE International Conference on Advanced Communications, Control and Computing Technologies* (pp. 1126-1131). IEEE. https://doi.org/10.1109/ICACCCT.2014.7019273
20. V. S.,Bawa, & V. Kumar, (2019). **Linearized sigmoidal activation: A novel activation function with tractable non-linear characteristics to boost representation capability**. *Expert Systems with Applications*, *120*, 346-356.
21. X. Zeng, and T. R. Martinez, "***A new activation function in the Hopfield Network for Solving Optimization Problems.***" Artificial Neural Nets and Genetic Algorithms. Springer, Vienna, 1999, pp. 73-77.
22. X. Zeng, and T. R. Martinez, "**A New Relaxation Procedure in the Hopfield Network for Solving Optimization Problems**." *Neural Processing Letters* 10.3, 1999, pp. 211-222.
23. M. Lippe, M. Bithell, N. Gotts, D. Natalini, P. Barbrook-Johnson, C. Giupponi, ... & M. Schlüter, (2019). **Using agent-based modelling to simulate social-ecological systems across scales**. *GeoInformatica*, *23*(2), 269-298. https://doi.org/10.1007/s10707-018-00337-8
24. Y. Dijkxhoorn, C. Plaisier, T. Verwaart, C. V. Wagenberg, & R. Ruben, (2019). **Trusted sorghum: simulating interactions in the sorghum value chain in Kenya using games and agent-based modelling**. *Journal of Development Effectiveness*, *11*(2), 146-164.
25. C. Delcea, L. A. Cotfas, L. Craciun, & A. G. Molanescu, (2020). **An agent-based modeling approach to collaborative classrooms evacuation process**. *Safety science*, *121*, 414-429.
26. A. Alzaeemi, and S. Sathasivam, "**Hopfield neural network in agent based modeling**". *MOJ App Bio Biomech*, 2(6), 2018, pp. 334-341. https://doi.org/10.15406/mojabb.2018.02.00089
27. S. A. Alzaeemi, M. A. Mansor, M. S. M. Kasihmuddin, & S. Sathasivam, (2019, December). **Comparing the logic programming between Hopfield neural network and radial basis function neural network**. In *AIP Conference Proceedings* (Vol. 2184, No. 1, p. 060044). AIP Publishing LLC.
28. M. S. M. Kasihmuddin, M. A. Mansor, S. A. Alzaeemi, M. F. M. Basir, & S. Sathasivam, (2019, November). **Quality Solution of Logic Programming in Hopfield Neural Network**. In *Journal of Physics: Conference Series* (Vol. 1366, No. 1, p. 012094). IOP Publishing.
29. S. A. Alzaeemi, S. Sathasivam, & S. A. Adebayo, (2017) "**Analysis of Performance of Various Activation Functions for doing the logic programming in Hopfield Network**.". *Journal of Computational Bioinformatics and in Silico Modeling*, 6, 2, pp. 911-921.
30. M. S. M. Kasihmuddin, M. A. Mansor, & S. Sathasivam, (2018). **Discrete Hopfield Neural Network in Restricted Maximum k-Satisfiability Logic Programming**. *Sains Malaysiana*, *47*(6), 1327-1335.
31. S. Sathasivam, M. Mamat, M. Mansor, & M. S. M. Kasihmuddin, (2020). **Hybrid Discrete Hopfield Neural Network based Modified Clonal Selection Algorithm for VLSI Circuit Verification**. *Pertanika Journal of Science & Technology*, *28*(1).
32. S. Sathasivam, "**Upgrading logic programming in Hopfield network**." *Sains Malaysiana* 39.1, 2010, pp.115-118.
33. S. A. Alzaeemi, M. Mansor, M. S. M. Kasihmuddin, & S. Sathasivam, (2019, December). **2 satisfiability logic programming in radial basis function neural networks**. In *AIP Conference Proceedings* (Vol. 2184, No. 1, p. 060045). AIP Publishing LLC.
34. S. Sathasivam, & W. A. T. W. Abdullah, "**Logic mining in neural network: reverse analysis method**." *Computing* 91.2, 2011, pp. 119-133. https://doi.org/10.1007/s00607-010-0117-9