Applied Mathematical Sciences, Vol. 5, 2011, no. 78, 3853 - 3860

Boltzmann Machine and New Activation Function Comparison

Saratha Sathasivam

School of Mathematical Sciences, Universiti Sains Malaysia
11800 USM, Pulau Pinang, Malaysia
saratha@cs.usm.my

Abstract. Boltzmann machine and new activation function are examined for its ability to accelerate the performance of doing logic programming in Hopfield neural network. Boltzmann machine has a higher capacity than the new activation function. This learning rule also suffers significantly less capacity loss as the network gets larger and more complex. Comparisons are made between these rules to see which rule is better or outperformed other rule in the aspects of computation time, memory and complexity.

Keywords: Boltzmann machine, new activation function, capacity, complexity

1. INTRODUCTION

Neural network is a parallel processing network which generated with simulating the image intuitive thinking of human, on the basis of the research of biological neural network, according to the features of biological neurons and neural network and by simplifying, summarizing and refining. It uses the idea of non-linear mapping, the method of parallel processing and the structure of the neural network itself to express the associated knowledge of input and output.

The Hopfield neural network is a simple recurrent network which can work as an efficient associative memory, and it can store certain memories in a manner rather similar to the brain. Wan Abdullah [6, 7] proposed a method of doing logic program on a Hopfield network.

Optimization of logical inconsistency is carried out by the network after the connection strengths are defined from the logic program; the network relaxes to neural states which are models (i.e. viable logical interpretations) for the corresponding logic program. Type of learning implemented in this network is known as Wan Abdullah's learning. The connection weights are determined by comparing the cost function with energy function of the network.

In this paper, we will analyze the usage of new activation function and Boltzmann machine in enhancing the performance of doing logic programming in Hopfield network. Part of this work has been reported in [3, 4, 5].

2. LOGIC PROGRAMMING IN HOPFIELD MODEL

In order to keep this paper self-contained we briefly review the Little-Hopfield model [2]. The Hopfield model is a standard model for associative memory. The Little dynamics is asynchronous, with each neuron updating their state deterministically. The system consists of *N* formal neurons, each of which is described by an Ising variable $S_i(t), (i = 1, 2, ...N)$. Neurons then are bipolar, $S_i \in \{-1, 1\}$, obeying the dynamics $S_i \to \text{sgn}(h_i)$, where the field, $h_i = \sum_j J_{ij}^{(2)} V_j + J_i^{(1)}$, *i* and *j* running over all neurons *N*, $J_{ij}^{(2)}$ is the synaptic strength from neuron *j* to neuron *i,* and $-J_i$ is the threshold of neuron *i*.

Restricting the connections to be symmetric and zero-diagonal, $J_{ij}^{(2)} = J_{ji}^{(2)}, J_{ii}^{(2)} = 0$, allows one to write a Lyapunov or energy function,

$$E = -\frac{1}{2}\sum_i \sum_j J_{ij}^{(2)} S_i S_j - \sum_i J_i^{(1)} S_i \tag{1}$$

which monotone decreases with the dynamics.

The two-connection model can be generalized to include higher order connections. This modifies the "field" to be

$$h_i = ... + \sum_j \sum_k J_{ijk}^{(3)} S_j S_k + \sum_j J_{ij}^{(2)} S_j + J_i^{(1)} \tag{2}$$

where "….." denotes still higher orders, and an energy function can be written as follows:

$$E = ... - \frac{1}{3}\sum_i \sum_j \sum_k J_{ijk}^{(3)} S_i S_j S_k - \frac{1}{2}\sum_i \sum_j J_{ij}^{(2)} S_i S_j - \sum_i J_i^{(1)} S_i \tag{3}$$

provided that $J_{ijk}^{(3)} = J_{[ijk]}^{(3)}$ for *i, j, k* distinct, with […] denoting permutations in cyclic order, and $J_{ijk}^{(3)} = 0$ for any *i, j, k* equal, and that similar symmetry requirements are satisfied for higher order connections. The updating rule maintains

$$S_i(t+1) = \text{sgn}[h_i(t)] \tag{4}$$

In the simple propositional case, logic clauses take the form $A_1, A_2, ..., A_n \leftarrow B_1, B_2, ..., B_m$. which says that (A_1 or A_2 or …. or A_n) if (B_1 and B_2 and … and B_n); they are program clauses if $n = 1$ and $m \geq 0$: we can have

rules e.g. $A \leftarrow B, C.$ saying $A \vee \neg(B \wedge C) \equiv A \vee \bar{B} \vee \bar{C}$, and assertions e.g. $D \leftarrow .$ saying that D is true.

A logic program consists of a set of program clauses and is activated by an initial goal statement. In Conjunctive Normal Form (CNF), the clauses contain one positive literal. Basically, logic programming in Hopfield model can be treated as a problem in combinatorial optimization. Therefore it can be carried out in a neural network to obtain the desired solution. Our objective is to find a set of interpretation (i.e., truth values for the atoms in the clauses which satisfy the clauses (which yields all the clauses true). In other words, we want to find 'models'.

The following algorithm shows how a logic program can be done in a Hopfield network based on Wan Abdullah's method:

i) Given a logic program, translate all the clauses in the logic program into basic Boolean algebraic form.

ii) Identify a neuron to each ground neuron.

iii) Initialize all connections strengths to zero.

iv) Derive a cost function that is associated with the negation of all the clauses, such that $\frac{1}{2}(1+S_x)$ represents the logical value of a neuron X, where S_x is the neuron corresponding to X. The value of S_x is define in such a way that it carries the values of 1 if X is true and -1 if X is false. Negation (neuron X does not occur) is represented by $\frac{1}{2}(1-S_x)$; a conjunction logical connective is represented by multiplication whereas a disjunction connective is represented by addition.

v) Obtain the values of connection strengths by comparing the cost function with the energy, H.

vi) Let the neural networks evolve until minimum energy is reached. Checked whether the solution obtained is a global solution.

The applied methodology may be summarized in the following way: given an optimization problem, find the cost function that describes it, design a Hopfield network whose energy function must reach (one of) its minima at the same point in configuration space as the cost function, so that the stable configurations of the network correspond to solutions of the problem [8]. We do not provide a detail review regarding neural network logic programming in this paper, but instead refer the interested reader to Wan Abdullah [6].

3. BOLTZMANN MACHINE

Boltzmann Machines are neural networks whose behavior can be described statistically in terms of simple interactions between the units consist in that network [1]. Boltzmann machine is classified as a stochastic neural network which consists of one layer of visible units (neurons) and one layer of hidden units

(neurons). The visible neurons provide an interface between the network and its environment. Meanwhile, the cconnections between neurons are bidirectional and it is a symmetric network. This shows that information can flows in both directions during the training and also during the usage of the network and the weights are the same in both directions. During the training phase, the visible neurons are usually clamped whereas the hidden neurons will always operate freely.

In order to implement the Boltzmann machine in a parallel networks, a simple modification of Hopfield's updating rule will be required. The Boltzmann machine operates by picking a hidden unit at random, let say unit i, and spinning the state of neuron i from S_i to $-S_i$ at temperature T (during the annealing cycle) with probability:

$$p_i = \frac{1}{1+e^{(-\frac{\Delta E_i}{T})}} \tag{5}$$

where ΔE_i, is given by:

$$E_i = \sum_{j=i}^{n} u_j w_{ij} - \theta_i \tag{6}$$

The temperature parameter T is important for the method of simulated annealing. If the spinning procedure is applied continually to the units, the units will change state and it will ensure the relative probability of two global states is determined exclusively by their energy difference when the system reached the thermal equilibrium and follows a Boltzmann distribution:

$$\frac{P_\alpha}{P_\beta} = e^{-\frac{(E_\alpha - E_\beta)}{T}} \tag{7}$$

where P_α is the probability of being in the α-th global state, and E_α is the energy of that state.

The time required to reach equilibrium may be long at low temperatures although there is a strong bias in favour of states with low energy. On the other hand, at higher temperatures the bias is not so favourable, but equilibrium is reached faster. The ideal way to reach equilibrium or to find a stable configuration that is suited to the problem at hand at a given temperature, Boltzmann learning proceeds by first operating the network at high temperature, and gradually lowering it until it reaches thermal equilibrium at a series of temperatures, as agreed by the simulated annealing procedure. Several different groups have been studied on the idea of using simulated annealing to find low energy states in parallel networks. Simulated annealing can be very efficiently in order to remove noise from images and recognized limits on the permissible speed of the annealing schedule. Hinton and Sejnovski [1] solved some problems that outbreak other relaxation techniques, in particular the problem of learning the weights by using the binary stochastic elements.

4. IMPLEMENTATION OF BOLTZMANN MACHINE IN HOPFIELD NETWORK

The following Pseudo code shows how Boltzmann machine has been applied in Hopfield network.

(i)Generate minimum energy, *ES* via the following equation:

$$ES = \frac{1}{8}(1 - S_A)(1 + S_B)(1 + S_C) + \frac{1}{4}(1 - S_D)(1 + S_B) + \frac{1}{2}(1 - S_C)$$

(ii) Calculate energy function, *E* via the following equation:

$$E = -\frac{1}{3}\sum_i\sum_j\sum_k J^{(3)}_{[ijk]} S_i S_j S_k - \frac{1}{2}\sum_i\sum_j J^{(2)}_{[ij]} S_i S_j - \sum_i J^{(1)}_{[i]} S_i$$

for $i, j, k = 1, 2, \ldots, NN$

(iii) Set $\Delta E_i = |ES - E|$ for $i = 1, 2, NN$

(iv) Define $p(i) = \frac{1}{1+e^{\left(-\frac{\Delta E_i}{T}\right)}}$ for $i = 1, 2, \ldots, NN$ where T=temperature

(v) if $|p(i)| \leq 0.5$

z = 1 (Global minimum)

(vi) if $|p(i)| > 0.5$

y = 1 (Local minimum)

(vii) end

5. NEW ACTIVATION FUNCTION

The activation function in the Hopfield network is the sigmoid function. However this activation function puts too much emphasis on minor noise perturbation instead of the signals related to the cost and the constraints encoded in the network.

Zeng and Martinez [9] proposed a new activation function as followed:

$$V_{X_i} = \frac{0.5(1+\tanh(\frac{U_{X_i}+x_o}{u_o}))}{1+\tanh(\frac{x_0}{u_0})} (U_{X_i} < 0)$$
$$V_{X_i} = \frac{\tanh(\frac{x_0}{u_0})+0.5(1+\tanh(\frac{U_{X_i}-x_o}{u_o}))}{1+\tanh(\frac{x_0}{u_0})} (U_{X_i} \geq 0) \quad (8)$$

where the parameters are defined as followed: V_{X_i}= activation function, U_{X_i} =initial states, x_o represents the threshold for V_{X_i} to become steep, and u_0 measures the steepness of the activation function. This function can tolerate with noise and do perform well when the network gets larger.

6. THEORY IMPLEMENTATION

We generate random program clauses. Then, we initial states are initialize randomly for the neurons in the clauses. Next, we let the network relax until minimum energy is reached. We test the final state obtained whether it is a stable state where the states remain unchanged for more than five time steps, then we consider it as stable state.

Later, we calculate the corresponding final energy for the stable state. If the difference between the final energy and the global minimum energy is within a tolerance value (determine by the user), then we consider the solution as global solution. We measure the global solution ratio between the Boltzmann machine and new activation function.

We run the relaxation for 100 trials and 1000 combinations of neurons so as to reduce statistical error. The selected tolerance value is 0.0001. All these values were obtained by trial and error.

7. RESULTS AND DISCUSSION

We simulated the network for the both methods. The following figures shows the global minima ratio (Number of global minima solutions/Number of runs) obtained by using both methods. From the graphs obtained, we observed that the ratio of global solutions is consistently 1 for all the cases, although we increased the network complexity by increasing the number of neurons (NN) and number of literals per clause (NC1, NC2, NC3) for the new activation function compare with the sigmoid function. It can be observed that when the network gets larger or more complex, the Boltzmann machine seems to perform better and continuously compared with the activation function. This is due to the simulated

annealing procedure carried out by the Boltzmann machine where the neurons are forced to jump the energy barriers to relax into global solutions. By using this method the neurons are able to relax to global minima values rather than stuck in local minima values. While using activation function function, the neurons get stuck and unable to jump the energy barrier to relax in global states.

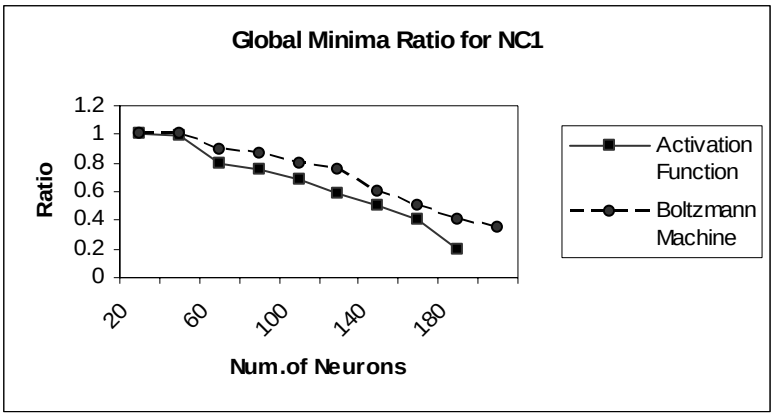

Figure 1: Global Minima Ratio for NC1

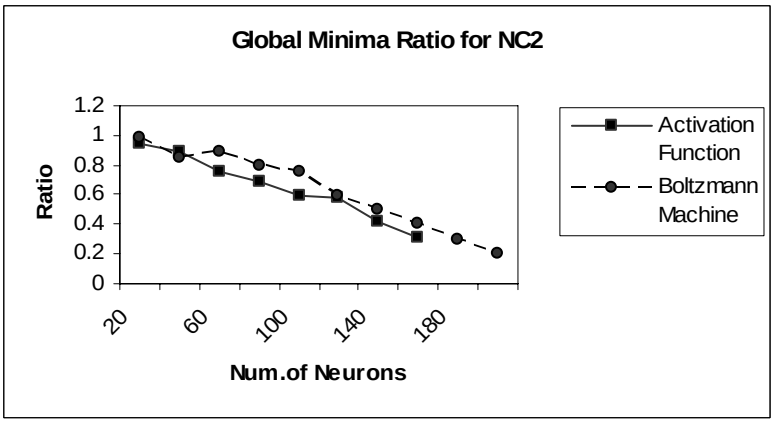

Figure 2: Global Minima Ratio for NC2

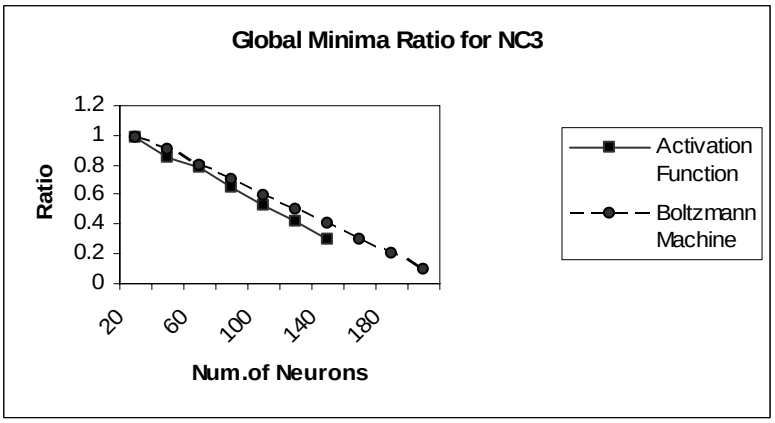

Figure 3: Global Minima Ratio for NC3

8. CONCLUSION

We have discussed the implementation of Boltzmann machine and new activation function in upgrading the performance of doing logic programming in Hopfield network. Computer simulations that had been carried out verified the validity of the Boltzmann machine performance compare to new activation function. This completes our illustration of computer simulation to test the validity and strength of the proposed method of doing logic programming in Hopfield network.

ACKNOWLEDGEMENT

This research is partly financed by KPI grant (1002/ PMATHS/ARSP13000) from Universiti Sains Malaysia.

REFERENCES

[1] D.H. Ackley, G.E.Hilton and T.J. Sejnovski, A learning algorithm for Boltzmann machines, Cognitive Science, 62 (1985), 147-169.
[2] J.J. Hopfield, Neural Networks and Physical Systems with Emergent Collective Computational Abilities. *Proceedings. Natl. Acad. Sci. USA.* , 79(1982), 2554-2558.
[3] S. Sathasivam, Accelerating Neuro-symbolic integration by using Boltzmann machine, Far East Journal of Mathematical Sciences, Vol. 43, Num .2, (2010), 241-251.
[4] S. Sathasivam, Upgrading Logic Programming in Hopfield Network, Vol.39, Num.1, (2010), 115-118.
[5] S. Sathasivam & W.A.T. Wan Abdullah, Logic Mining in Neural Network, Computing, Volume 91, Issue 2 ,(2011), 119-133.
[6] W.A.T. Wan Abdullah, Neural Network logic, Proc of the workshop held in Elba International Physics Center, Italy, (1991), 135-142.
[7] W.A.T. Wan Abdullah, Logic Programming on a Neural Network, Int .J. Intelligent Sys*,* 7, (1992), 513-519.
[8] P.H. Wen, A Review of Hopfield Neural Networks for Solving Mathematical Programming Problems, European Journal of Operational research, Vol.198, Num.3, (2008), 675-688.
[9] X. Zeng and R. Martinez, A new activation function in the Hopfield Network for Solving Optimization Problems, In Fourth International Conference on Artificial Neural Networks and Genetic Algorithms, (1999), 1-5.

Received: May, 2011