DATA MINING IN NETWORK WITHOUT HIGHER ORDER CONNECTIONS

Saratha Sathasivam
School of Mathematical Sciences, Universiti Sains Malaysia, 11800 USM, Penang, Malaysia.
604-6532428
saratha@cs.usm.my

ABSTRACT

Higher order neural networks (HONN) have been shown to have impressive computational, storage and learning capabilities. However, networks with higher order connections need more computational time, higher storage requirements and have higher complexity, so we look at the possibility of networks without higher order connections. In this paper, we explore the constraints and effects of insisting on having neural networks without these higher order connections in learning three-atom clauses.

Keywords: Higher order neural networks, Hebbian learning, logical clauses, learning, storage requirement.

1. INTRODUCTION

Recurrent single field neural networks are essentially dynamical systems that feed back signals to themselves. Popularized by John Hopfield, these models possess a rich class of dynamics characterized by the existence of several stable states each with its own basin of attraction. The Little-Hopfield neural network (Hopfield 1982; Little 1974) minimizes a Lyapunov function, also known as the energy function due to obvious similarities with a physical spin network. Thus, it is useful as a content addressable memory or an analog computer for solving combinatorial-type optimization problems because it always evolves in the direction that leads to lower network energy. This implies that if a combinatorial optimization problem can be formulated as minimizing the network energy, then the network can be used to find optimal (or suboptimal) solution by letting the network evolve freely.

Gadi Pinkas (Pinkas 1991) and Wan Abdullah (Wan Abdullah 1992) defined a bi-directional mapping between propositional logic formulas and energy functions of symmetric neural networks. Both methods are applicable in finding whether the solutions obtained are models for a corresponding logic program.

Previously Wan Abdullah has shown on see how Hebbian learning in an environment with some underlying logical rules governing events is equivalent to hardwiring the network with these rules (Wan Abdullah 1991). Then, by looking at this process backwards, we can show how synaptic changes from learning correspond to the formation of logical rules (Sathasivam 2006). As logical clauses which contain at least three atoms are required (from which clauses with more atoms can be built) for meaningful knowledge bases, neural networks with higher order connections (Abott & Arian 1987; Gardner 1987; Wan Abdullah 1987) are needed. In this paper, we explore the constraints and effects of insisting on having neural networks without these higher order connections in learning three-atom clauses.

This paper is organized as follows. In section 2, we give the outline of doing logic programming in Hopfield model and in section 3; Hebbian learning of higher-order logical clauses is described. In section 4, we explore the effects of requiring the network to have at most second-order interactions. Finally concluding remarks regarding this work occupy the last section.

2. LOGIC PROGRAMMING IN HOPFIELD MODEL

In order to keep this paper self-contained we briefly review the Hopfield model. The Hopfield model is a standard model for associative memory. The Hopfield 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 $S_i(t), (i = 1, 2, ...N)$ (Sathasivam 2008). Neurons then are bipolar, $S_i \in \{-1, 1\}$, obeying the dynamics $S_i \rightarrow \text{sgn}(h_i)$, 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 (Sathasivam 2008),

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

which decreases monotonically with the dynamics.

The two-connection model can be generalized to include higher order connections.

This modifies the "field" into

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

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 logic programming, a set of Horn clauses which are logic clauses of the form $A \leftarrow B_1, B_2, .., B_N$ where the arrow may be read "if" and the commas "and", is given and the aim is to find the set(s) 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' corresponding to the given logic program.

In principle logic programming can be seen as a problem in combinatorial optimization, which may therefore be carried out on a neural network. This is done by using the neurons to store the truth values of the atoms and writing a cost function which is minimized when all the clauses are satisfied.

As an example, consider the following logic program, $A \leftarrow B, C$, whose three clauses translate as $A \vee \neg(B \wedge C) = A \vee \neg B \vee \neg C$. The underlying task of the program is to look for interpretations of the atoms, in this case *A, B* and *C* which are model for the given logic program. This can be seen as a combinatorial optimization problem where the "inconsistency",

$$E_p = \frac{1}{2}(1 - S_A)\frac{1}{2}(1 + S_B)\frac{1}{2}(1 + S_c) \tag{5}$$

where S_A, etc. represent the truth values (*true* as 1) of *A*, etc., is chosen as the cost function to be minimized. We can observe that the minima value for E_p is *0*, and has otherwise value proportional to

the number of unsatisfied clauses. The cost function (5), when programmed onto a third order neural network yields:

Table 1: Synaptic strengths for $A \leftarrow B, C$ using Wan Abdullah's method

Synaptic Strengths	Clause $A \leftarrow B,C$
$J^{(3)}_{[ABC]}$	1/16
$J^{(2)}_{[AB]}$	1/8
$J^{(2)}_{[AC]}$	1/8
$J^{(2)}_{[BC]}$	-1/8
$J^{(1)}_{[A]}$	1/8
$J^{(1)}_{[B]}$	-1/8
$J^{(1)}_{[C]}$	-1/8

We addressed this method of doing logic programming in neural network as *Wan Abdullah's method*.

3. HEBBIAN LEARNING OF LOGICAL CLAUSES

The Hebbian learning for a two-neuron synaptic connection can be written as

$$\Delta J^{(n)}_{ij} = \lambda(2V_i - 1)(2V_j - 1) \tag{6}$$

(or, for bipolar neurons, $\Delta J_{ij} = \lambda S_i S_j$), where λ is a learning rate. For connections of higher orders, we can generalize this to

$$\Delta J^{(n)}_{ij....m} = \lambda(2V_i - 1)(2V_j - 1)....(2V_n - 1) \tag{7}$$

This gives the changes in synaptic strengths depending on the activities of the neurons. In an environment where selective events occurred, Hebbian learning will reflect the occurrences of the events. So, if the frequency of the events is deducted by some underlying logical rule, logic should be entrenched in the synaptic weights.

Wan Abdullah (Wan Abdullah 1992) has shown that the synaptic strengths obtained using Wan Abdullah's method is similar with Hebbian learning provided that equation (8) is true.

$$\lambda^{(n)} = \frac{1}{(n-1)!} \tag{8}$$

We do not provide a detail review regarding Hebbian learning of logical clauses in this paper, but instead refer the interested reader to Wan Abdullah's paper (Wan Abdullah 1991; Wan Abdullah 1992).

For meaningful knowledge bases, clauses with at least three atoms are needed [12]. As can be observed, this then requires higher-order neural connections. The complexity of the network, and thus processing time, is increased. In the upcoming section, we explore the effects of insisting on having neural networks without these higher order connections in learning three-atom clauses.

4. LOGICAL CONTENT IN NETWORK WITHOUT HIGHER-ORDER CONNECTIONS

Higher order neural networks (HONN) have been shown to have impressive computational, storage and learning capabilities (Maxwell et.al 1986). Besides that, it has also been shown that HONN are effective for associative memory, pattern recognition and combinatorial optimization problems (Cooper

1995). As increasing the order of the network, the information quantity of the network also will increase. Obviously, by extending the HONN to even higher orders (more than 3 spins), will yield further improvement; at the expense of increasing network complexity and computation (Yanali & Sawada 1990).

However, networks with higher order connections need more computational time, higher storage requirements and have higher complexity, so we look at the possibility of networks without higher order connections. In this section, we examine logical clauses formed by Hebbian learning in a network without higher order connections . We carried this out by ignoring the third order connections by setting the contributions of third order connection strengths as zero.

Firstly, from general considerations, it can be predicted that removing higher order connections will deteriorate the knowledge content of neural networks, due to decrease of memory capacity as given by Lee (Lee & Maxwell 1987):

$$\text{Memory capacity} = \frac{\binom{m}{k}}{2\log m} \tag{9}$$

where, m is the dimension of input vectors, corresponding to the maximum number of literals per clause and k is the order of the connections. Furthermore, the associative ability of the network also decreased as given by Yatsuki (Yatsuki & Miyajima 1997):

$$Associative\_Ability \alpha \frac{P}{\binom{m}{k}} \tag{10}$$

where P is the number of memory patterns. As the number of patterns becomes large, weights begin to look like a random variable. The term in the weight sum, which correspond to a particular pattern is greatly over weighted by all the other weights and associative memory become poor.

We also expect that the ignoring of higher order connections will result in appearances of spuriously learnt clauses (Wan Abdullah 1997), which learnt the clauses partially or less. Let us look at an example. Here, we will see on how third order clause $A \vee \neg B \vee \neg C$, is learnt by the network without higher order connections. As expected, ignoring the higher order connections resulted in appearance of spuriously learnt clauses as shown by Sathasivam (Sathasivam 2008).

$$(\neg B \vee \neg C) \wedge (A \vee \neg B) \wedge (B) \tag{11}$$

By carrying out analysis using truth table (Table 2), we observed that the set of interpretations for the underlying rule ($A \vee \neg B \vee \neg C$) is dissimilar with the set of interpretations for the spuriously learnt clauses (11).

Table 2: Truth table for underlying rule and the spuriously learnt clauses

A	B	C	$A \vee \neg B \vee \neg C$	$(\neg B \vee \neg C) \wedge (A \vee \neg B) \wedge (B)$
-1	-1	-1	1	-1
-1	-1	1	1	-1
-1	1	-1	1	-1
-1	1	1	-1	-1
1	-1	-1	1	-1
1	-1	1	1	-1
1	1	-1	1	1
1	1	1	1	-1

5. COMMENTS AND CONCLUSION

In this paper we examined logical clauses formed by Hebbian learning in a network without higher order connections. We can conclude that logical clauses formed by Hebbian learning in a network without higher order connection is not good as logical clauses formed by Hebbian learning in a network with higher order connection. Furthermore, the computational time required for both cases are similar.

ACKNOWLEDGEMENT

This research was supported by RU grant.

REFERENCES

1. Abott, L.F. & Arian, Y. (1987). Storage capacity of generalized networks. *Phys. Rev. A*. 36: 5091-5094.

2. Cooper, B.S. (1995). Higher Order Neural Networks-Can theory helps us optimize? *Proceedings of the 6th Australian Conference on Neural Networks (ACNN 95)*: 29-32.

3. Gardner, E. (1987). Multiconnected neural network models. *Phys. A: Maths. Gen*. 20: 3453-3464.

4. J.J. Hopfield. (1982). Neural Networks and Physical Systems with Emergent Collective Computational abilities, *Proc. Natl. Acad. Sci* (USA). 79: 2554-2558.

5. Lee, G.C. & Maxwell, T. (1987). Learning, Invariance and Generalization in Higher Order Neural Networks. *Applied Optics*. 26(23): 4973-4978.

6.Little,W.A. (1974) . *Math. Biosci*. 19: 101-120.

7. Maxwell, T., Lee, Y.C & Chen, H.H. (1986). Transformation Invariance Using High Order Correlations in Neural Net Architectures. *IEEE Trans. Syst. Man. Cybern*. 627: 2364-2368.

8. Pinkas, G. (1991). Energy minimization and the satisfiability of propositional calculus. *Neural Computation*. 3: 282-291.

9. Saratha Sathasivam. & Wan Abdullah, W.A.T. (2008). Flatness of the Energy Landscapes of Horn Clauses. *MATEMATIKA*. 23(2): 147-156.

10. Saratha Sathasivam. (2008). Energy Landscapes for Hopfield Network Programmed with Program Clauses. *In 4th IASTED International Conference in Advances Computer Science and Technology, Langkawi* (Malaysia): 179-182.

11. Saratha Sathasivam. (2006). *Logic Mining in Neural Networks*. PhD Thesis: University of Malaya, Malaysia.

12. Wan Abdullah, W.A.T. (1997). Logic in Hopfield Network without Higher order connections. *In Int. Conf. on Neural Information Processing and Intelligent Information Systems* (New Zealand): 5-8.

13. W.A.T. Wan Abdullah. (1992). Logic programming on a neural network. *Int. J. Intelligent Sys*. 7: 513-519.

14. Wan Abdullah, W. A. T. (1991). Neural Network logic. In O. Benhar et al. (Eds.), *Neural Networks: From Biology to High Energy Physics*. Pisa: ETS Editrice: 135-142.

15. Wan Abdullah, W.A.T. (1987). Dendritic trees and nonquadratic combinatorial optimization. *Malaysian J. Sci*. 9: 105-109.

16.Yanali, H. & Sawada, Y. (1990). On some properties on sequence-association type model neural networks. *IEIECE Trans.D-II,* J73-DII . 8: 1192-1197.

17. Yatsuki, S. & Miyajima, H. (1997). Associative ability of higher order neural networks. *Proceedings ICNN'97*. 2: 1299-1304.