Spiking Neurons with Boltzmann-like Properties to Learn Xor

Christian R. Huyck

Middlesex University - Dept of Computer Science
London - UK

Abstract. A computational model of biological neurons is used to learn the exclusive or relation. The neural model is a fatiguing leaky integrate and fire model, and spontaneous firing emerges from the hypo-fatigue. This spontaneous firing enables the system to move from direct stimulation from the environment. A form of Hebbian learning, post-synaptic weighted compensatory learning, is used to support firing in interior neurons. A range of numbers of interior neurons performs the task at above 99% precision.

1 Introduction

There are many computational models of biological neurons, and of more complex biological neural systems composed of neurons and their connections. However, it is difficult to build neural models, which adhere to biological constraints, that perform complex computational tasks.

One biological constraint is that learning is Hebbian in nature [1]. Typically with neural systems, learning is merely the change of connection strengths, which are biologically synaptic strengths; with Hebbian learning, the strengths are changed based solely on the properties of the pre and and post-synaptic neurons. This is typically the firing behaviour of these neurons, and the prototypical rule increases the strength when the neurons co-fire (see sections 2.2 and 4).

One biological requirement, from Hebbian learning, is that neurons need to fire to positively influence neural circuits. However, in many computational models only neurons that are directly linked to sensors fire, and in mammalian brains most neurons are not directly linked to sensors. Yet, biological neurons fire without external input [2]; they fire spontaneously.

The Boltzmann model [3] does fire spontaneously, without being directly linked to sensors. Other models can also be modified to randomly spike [4]. The author has used a fatiguing leaky integrate and fire (FLIF) neural model for quite some time, and has recently extended the fatigue model so that the neurons spontaneously fire ([5] and see section 2.1); the initial model reflected biological neuron firing behaviour relatively accurately, and the extension improved the fit to the biological data.

The emergence of spontaneous firing without input from the model provides a mechanism for a neural system, based on these FLIF neurons, to make use of neurons that are not directly stimulated. This in turn has created a system that can learn the exclusive or (xor) function.

Xor is a function with two binary inputs, see table 1. If either is true, the result is true, but if both are true or both are false, the output is false. It is difficult to learn xor in a net where the input neurons are directly connected to the output neurons, where both are directly stimulated by the environment. It is difficult because the synaptic weights, following a Hebbian rule, will reflect the co-firing behaviour of the neurons, and these behaviours are identical between any input/output pair.

The remainder of this paper describes reasonably biologically accurate computational system that solves the xor problem. It then links this system to other existing work, including Boltzmann machines, describes some short-comings, and considers how this work might be extended.

2 Model

Broadly speaking, the components of the model are the neural model, the learning algorithm, and the topology (the way the neurons are connected). The system can be found at http://www.cwa.mdx.ac.uk/undone.

2.1 FLIF Model

The FLIF neural model is a point model in the family of integrate and fire [6] neural models. An integrate and fire model is described by equation 1.

$$\theta < A_j = \sum_{i \in V_i} w_{ij} \tag{1}$$

The neuron integrates activity sent from other neurons that fired in the last cycle (V_i) weighted by the synaptic strength w_{ij}. It fires if the activity surpasses a threshold θ.

The model is discrete and runs in cycles that roughly correspond to 10 ms of time. It is leaky as described by equation 2, so if it does not fire in one cycle, it retains some of the activation for the next cycle. Without input, activation decays from step $t-1$ to step t being divided by a constant $D > 1$.

$$A_j^t = A_j^{t-1}/D \tag{2}$$

The neuron also fatigues each step it fires. Fatigue is increased by a constant F_c each step a neuron fires. The neuron's fatigue is added to the threshold so that neurons that frequently fire require more activation to fire.

When a neuron does not fire, its fatigue is reduced. In older models this was reduced by a constant F_r in each step that the neuron did not fire, but the fatigue value of a neuron never went below zero. The modified version allows fatigue to be negative. When a neuron is hypo-fatigued, it will fire when fatigue is negative enough ($-F > \theta$).

In this model, if the neuron fired and fatigue was less than -.25, the fatigue value was halved. Otherwise, it was increased by F_c as usual. When fatigue was below -.25, it was reduced exponentionally as described in equation 3.

$$F_i^{t+1} = F_i^t - (-4)^{(3-F_i^t)} \tag{3}$$

This leaves four parameters to describe the neural model. Threshold *theta* is 2.2; decay D is 1.12; fatigue increase F_c is 0.45; and fatigue recovery F_r is 0.01. In past simulations, these were free parameters for simulation, but these values have been selected to fit the firing behaviour to biological neurons [5]. The particular neurons modelled were rat somatosensory neurons under a widely varying direct current injection regime. Similarly, the fatigue rules, equation 3, is quite complex. However, its inclusion lead to a closer fit to the biological firing behaviour [7]. Fit to neural spiking behaviour is over 90% with an average difference of less than two cycles (17 ms.).

All neurons in the system, are of this form. They all will spike spontaneously if hypo-fatigued.

2.2 Compensatory Learning

The learning mechanism is another component of the model. While all evidence points to biological modification of synaptic weights being Hebbian, this leaves a vast range of possible rules. In the simulations described below, a compensatory learning rule has been used. In addition to the firing behaviour of the two neurons a synapse connects, a compensatory rule takes into account the total weight of the synapses in these neurons, forcing the total weight toward a target total in conjunction with the firing behaviour. The author has a used a compensatory rule based on the total of the pre-synaptic neuron's synapses in earlier work to learn hierarchical categories [8]. The simulations described below make use of compensatory rule bsed on the post-synaptic neuron's total synaptic weight. This rule increases weights to neurons that fire infrequently, typically due to spontaneous firing.

Hebbian rules are typically a combination of two rules, one for when the neurons co-fire, and one for when they do not. Equation 4 is applied when the neurons co-fire. Equation 5 is applied when the pre-synaptic neuron fires, and the post-synaptic neuron does not. When the pre-synaptic neuron does not fire, the weights do not change.

In equations 4 and 5, R is the learning rate, which is 0.01 in the simulations below. W_B is the target post-synaptic weight and W_j is the incoming synaptic weight to the post-synaptic neuron. In these simulations, W_B was set to 4.

$$\Delta_+ w_{ij} = (1 - w_{ij}) * R * 10^{(W_B - W_j)} \tag{4}$$

$$\Delta_- w_{ij} = w_{ij} * -R * 10^{(W_j - W_B)} \tag{5}$$

Neurons may also be inhibitory. A parallel learning rule applies to inhibitory neurons so that that the inhibitory synaptic weight is more negative for neurons that do not co-fire. In these simulations, for excitatory neurons, synapses have a weight between 0 and 1, and inhibitory neurons, a weight between -1 and 0.

2.3 Topology

The final component of the model is the topology, or how the neurons are connected. Firstly, each neuron is either inhibitory or excitatory, following Dale's principle [9]. The network is divided into three subnets as shown in figure 1.

The Input subnet has no internal connections, as it acts as a proxy for environmental stimulus. All its neurons are excitatory.

Within the Gas and Output subnets, connections are random, but there are no self-connections. 50% of the neurons are inhibitory, chosen randomly. For both inhibitory and excitatory neurons there are 20 connections.

There are connections between nets, and these only come from the excitatory neurons. Each neuron has 10 connections, randomly selected, to neurons in the destination net.

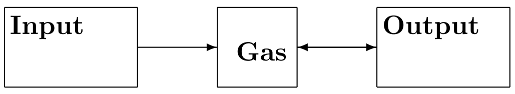

Fig. 1: Gross Topology of xor simulations. Boxes represent subnets, and arrows connections between subnets.

The Input net has 600 neurons, and the Ouput net has 400; this represents 200 per category (see section 3). Different runs have different numbers of neurons for the Gas subnet, but 800 is an example.

3 Experiment

The experiment was done by training the overall net, turning learning off, and then testing the net. Training and testing were both done in sets of 20 simulations cycles termed epochs.

In a training epoch, the appropriate inputs were externally stimulated. These inputs (and outputs) were just instances of the xor problem, see table 1. The Input net was broken into three patterns, E energy, A, and B. Each consisted of 200 neurons that belonged only to that pattern. Similarly, the Output net was broken into two patterns of 200 neurons each, Yes and No. An alternative presentation mechanism would have been four input patterns, A, $notA$, B and $notB$. It can be argued that the input pattern used is more natural as E could be considered a go signal. It is more difficult than the four input mechanism because the network is essentially storing four patterns, one with two elements, two with three, and one with four; it then retrieves the output based only on the input.

All of the neurons in the input pattern are externally stimulated for 10 cycles, and then the net is allowed to run for 10 cycles without input. That is one epoch, and then the next pattern is presented in the next epoch. Each stimulated neuron is given $(2 + r) * \theta$ units of activation, where r is a random number between 0

Input 1	Input 2	Input 3	Ouput
E			No
E	A		Yes
E	B		Yes
E	A	B	No

Table 1: Xor Input Table.

and 1. Due to fatigue and inhibition, some externally stimulated neurons may not fire in a given step. Note that neurons in the Gas net receive no external stimulation.

Training proceeds for 1000 epochs. Then, learning is switched off, and the network is run for 100 epochs with no external activation of the Output net. During a testing epoch, the number of neurons firing in the Yes and No patterns in the Output net are summed. Yes is the winner if it has more neurons firing, and No is the winner otherwise. The answer is thus easily determined correct or not.

This was all done for a given network, but networks are quite random so, it was run over 100 networks per test condition. In table 2, different numbers of neurons in the Gas subnet are shown.

Neurons in Gas	400	800	1200	1600	2000	800L	800SL
% Correct	98.31	99.77	99.86	99.88	99.95	98.94	93.71

Table 2: Performance Perecentage with Varying Size of Gas Subnet.

Another issue is the persistence of learning. It is not entirely clear if neurons are always learning at the same rate, but continued learning is an issue, in particular in the form of the stability plasticity dilemma [10]. With this in mind, two other simulations were run. These are both with 800 neurons in the Gas subnet, and are displayed in table 2; the 800L entry shows performance with learning remaining on during testing, and the 800SL line shows 2000 cycles with no input, just spontaneous activation, with learning on constantly. These do show degraded performance, but they are relatively stable.

4 Background

Modelling the brain has a long history, going back to at least 1907. Neurons have been modelled, learning has been modelled, and topology is a key issue.

Computational models of neurons can be broadly grouped into compartmental models and point models. Compartmental models break the neuron into 3D parts, and then use conductance to determine the electrical properties of the neuron dynamically. An early example of these is the Hodgkin-Huxley model [11]. Almost without exception, these models are a more accurate reflection of the biology than are point models, but they are expensive to simulate.

Point models are widely used, with integrate and fire models [6] being an early and simple model. This is also used in the Hopfield model [12]. A widely used extension includes leak [13], leading onto the FLIF model described above.

Boltzmann machines are another type of point model [3]. They fire, but they fire on a regular basis without input. Increasing input increases their firing rate.

Hebbian learning is ill defined, but in all cases it is a local rule based on the behaviour of adjacent neurons. Biologically, learning appears to be Hebbian, with the spike timed dependent plasticity having solid biological support [14]. It is possible to separate independent and principal components with Hebbian rules [15]. Anti-Hebbian rules force co-firing neurons apart to decorelate their behaviour. This is still a local rule.

Another issue is topology. It is widely known that the brain is loosely connected. However, well-connected topologies have had extensive use (e.g. [12]) as these have statistical mechanical properties that enable powerful analytical results.

5 Conclusion

The author is encouraged by these simulations as it shows learning into multiple layers using biologically accurate mechanisms. However, there are problems.

These results are a form of associative memory with the inputs associated with the outputs via the intermediate layers. However, this misses the ability of cell assemblies to persist, acting as both long and short-term memory.

Similarly, one might argue that linking biological reality with machine learning is unnecessary. However, from both perspectives it is important. From the neuropsychological perspective, it is important to understand how neurons might learn functions. From the machine learning perspective, learning across multiple layers may provide a mechanism to escape from the brittleness of current solutions, and learn a wide range of information.

Another concern is that there are a really large number of neurons used to learn this relatively simple task. Future work with anti-Hebbian learning and decorrelation will explore this.

References

[1] D. Hebb. *The Organization of Behavior: A Neuropsychological Theory.* J. Wiley & Sons, 1949.

[2] M. Abeles, H. Bergman, E. Margalit, and E. Vaddia. Spatiotemporal firing patterns in the frontal cortex of behaving monkeys. *Journal of Neurophysiology*, 70(4):1629–1638, 1993.

[3] D. Ackley, G. Hinton, and T. Sejnowski. A learning algorithm for boltzmann machines. *Cognitive Science*, 9:147–169, 1985.

[4] C. Huyck and R. Bowles. Spontaneous neural firing in biological and artificial neural systems. *Journal of Cognitive Systems*, 6:1:31–40, 2004.

[5] C. Huyck. Parameter values for FLIF neurons. In *Complexity, Informatics and Cybernetics: IMCIC 2011*, 2011.

[6] W. McCulloch and W. Pitts. A logical calculus of ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, 5:115–133, 1943.

[7] C. Huyck and A. Parvizi. Parameter values and fatigue mechanisms for flif neurons. *Journal of Systemics, Cybernetics and Informatics*, accepted.

[8] C. Huyck. Creating hierarchical categories using cell assemblies. *Connection Science*, 19:1:1–24, 2007.

[9] J. Eccles. Chemical transmission and dale's principle. *Prog. Brain Research*, 68:3–13, 1986.

[10] G. Carpenter and S. Grossberg. The art of adaptive pattern recognition by a self-organizing neural network. *IEEE Computer*, 21:77–88, 1988.

[11] A. Hodgkin and A. Huxley. A quantitative description of membrane current and its application to conduction and excitation in nerve. *Journal of Physiology*, 117:500–544, 1952.

[12] J. Hopfield. Neural nets and physical systems with emergent collective computational abilities. *PNAS*, 79:2554–2558, 1982.

[13] D. Amit. *Modelling Brain Function: The world of attractor neural networks.* Cambridge University Press, 1989.

[14] G. Bi and M. Poo. Synaptic modifications in cultured hippocampal neurons: dependence on spike timing, synaptic strength, and postsynaptic cell type. *Journal of Neuroscience*, 18:24:10464–10472, 1998.

[15] C. Fyfe. *Hebbian Learning and Negative Feedback Networks.* Springer, 2005.