University of Sydney

Exploration of Computation Neuronal Network for Solving a Foraging Task

Author
William A. Talbot
Supervisor
Dr. Cliff Kerr

May, 2019

Abstract

Introduction

Neuroscience and the Rise of Neural Networks

The development of modern artificial intelligence has its roots in the field of neuroscience since the inception of neural modelling in the 1943 with neuroscientist Warren McCulloch's and mathematician Walter Pitts' seminal paper on the "logical calculus" of "nervous activity" [1]. Theories from neuroscience laid the foundations for successes in the following decades such as the first neural network with a real-world application, MADELEINE, in 1959. The unsuccessful theory of "Perceptron" networks, ultimately proven limited in Minsky and Papert's 1969 book [2], was developed around the time and coupled with a cultural fear of robotics and AI stemming from science fiction, halted funding and progress. The mid 1980s saw a resurgence of the computational neuroscience field. Lacking digital means, several neural networks were simulated in analogue electrical circuits such as the parallel collective analog circuits of Hopfield and Tank's tackling of the travelling salesman problem [3] that improved on the previous two-state neuron approach. Their electrical circuits were biologically inspired, with standard electrical components such as amplifiers, capacitor and resistors used to model the neuronal dynamics. The problem, which aims to find a shortest-path circuit between points is well studied and is known to have exponential time complexity and be np-complete, however their circuit was able to produce excellent solutions in short time spans, rivalled only by the Lin-Kernighan Monte Carlo approach.

The core application of neural networks was realised at the time. It was noted in Hopfield and Tank's that "A person ... quickly finds a very good path" and that therefore, conceptually at least, it should be "an easy problem". The question raised is obvious - why is it that the human brain is able to quickly solve computationally difficult problems with relative ease, and is this replicable? The advances in robotics, and perceptual, pattern-intensive and data-dependent problems predicted [3] are evident and ongoing today. Noticeably in the field of neural networks after the 1980s, the balance between the biological principles of neuroscience and the mathematical constructs underlying the networks has shifted in favour of the latter. Today many such mathematical network models exist, such as feed-forward, recurrent, bi-directional recurrent, biological, spiking, convolutional, max-pooling convolutional, deep belief, self-delimiting and time delay neural networks [4]. Object classification is an example of a widely developed subset of neural net applications, however even today the best performances of these networks are sub-human and sometimes significantly worse. One recent convolutional neural network developed by Liang and Hu [5] trained on 50000 images of the 10-class CIFAR-10 database achieved accuracies of less than 93% while another, trained on 60000 images of the 100-class CIFAR-100 database has a best accuracy of only 68.25%. Not only is the accuracy of current artificial neural networks inferior to the human brain but also the size of the training set required to teach a neural network is disproportionately large comparatively. The potential for networks that require much smaller training sets is another motivator for investigating biologically realistic neural networks.

While research in the computational neuroscience certainly exists, it is much smaller than the artificial intelligence field and research into applied biological or spiking neural networks is even smaller. This is potentially a result of the lower observed performance of spiking neural networks for practical applications in comparison to the best traditional neural networks [4]. The combination of advances in the understanding of neuronal dynamics, powerful computational tools and demonstrations of computational viability [6] has inspired some researchers around the world to return to modelling the human brain and investigate biological neural networks [7][8][9][10][11]. There are currently several well-known large-scale projects currently being developed. The most famous and longest-lasting is the Human Brain Project [12] with its mission to build a novel and unified infrastructure for computational neuroscience. Another is "BioSpawn" [13], a dynamic brain simulation consisting of 2.5 million neurons and 8 billion connections comprised of visual input, motor output and memory models. It is based on the NEURON simulation environment, a computational neuroscience tool that has been developed at Yale University for over 35 years. It is worth noting that the field of computational neuroscience is not limited to just the digital domain, but includes research done on real biological neuron networks, such as Frega et al.'s three-dimensional hippocampal network with embedded micro-transducer arrays which achieved a density of 80000 cells per cubic millimetre and an average of 600 synaptic connections per neuron [14].

The complex systems lab at the University of Sydney has also made significant developments in the field's digital side, with sensorimotor cortex models able to demonstrate reinforcement learning through spike time dependent plasticity (STDP) in a virtual arm [15][16]. Sanda et al.'s team at the University of California have similarly demonstrated effective STDP reinforcement learning on a toy game where a sprite attempts to forage for food on a two-dimensional grid as efficiently as possible [17], similar in computational difficulty to the travelling salesman problem. This report will demonstrate an implementation of this network and investigate the effects of modifications to the network. This implementation will be done through the use of NetPyNE, a tool for creating and running large-scale network simulations in NEURON [18].

Understanding Neurons and Spiking Neural Networks

Understanding of the biochemistry and resultant dynamics of neurons, both alone and within neuronal populations, has accelerated in the past century. Models of neurons are used extensively within the software implemented and thus this report will provide an introduction to neuronal dynamics that underpin these models (sourced primarily from the *Neuronal Dynamics* textbook [19]). The simple neuron consists of a central "soma" that acts as the non-linear processing unit of the cell, "dendrites", into which electric signals enter the cell, and the "axon" which is the output device of the cell. The dendrites are connected to the axons of other neurons using "synapses", of which there are many different types that enables neurons to have categorically different electrical characteristics depending on their purpose. In the human brain, neurons belong to populations of like neurons and often exist in layers, such as the cortex, approximately 3 millimetres deep on the outer surface of the cerebellum, where much sensory, motor and association processes occur.

The basic unit of signal transmission is the "action potential", a sharp voltage spike with a magnitude of approximately 100mV and typical duration of 1-2 milliseconds. When many spikes arrive at a soma from its dendrites, they accumulate and when the potential of the cell is high enough the neuron will produce an action potential of its own which travels along its axon to dendrites of other neurons. It is these through these signals that communication occurs between neurons and sophisticated behaviour is able to emerge. Gradually models have been developed to model the electrical characteristics of various neurons and the most prevalent is the Integrate-and-Fire models. The simplest is the "Leaky" Integrate-and-Fire model, where input is linearly integrated - the cell's potential $u(t)$ (also called membrane potential) is described by a linear differential equation which decays with some time constant τ and resets to 0 if it surpasses some threshold value ϑ. This mathematical model can be simplified further by treating the action potential spikes as Dirac-delta pulses.

$$\tau \frac{du}{dt} = -[u(t) - u_r] + RI(t) \tag{1}$$

$$\text{if } u(t) > \vartheta \text{ then } \lim_{\delta \leftarrow 0; \delta > 0} u(t+\delta) = u_r \tag{2}$$

This model is of course an extremely simplified model of a neuron, and so far more sophisticated models, non-linear "Generalised" Integrate-and-Fire models, have been developed based on the empirical biophysical properties of neurons. This dates back to at least the 1950s with the Hodgkin-Huxley model of a giant squid's neuron that led to Hodgkin and Huxley's Nobel prize in 1963. The neuron membrane has hundreds of types of ion channels and this rich biophysics leads to much nuance in neuron electrical behaviour. This includes hyper-polarisation of the cell and refractory periods of suppressed firing, spike-frequency adaptation, sub-threshold effects and post-inhibitory rebound. Exponential and quadratic Integrate-and-Fire models with adaptation and stochastic variables are some examples of non-linear models that have been reasonably fit to experimental data. When these models are used for their appropriate class of simulated neurons, we are able to generate networks that exhibit biologically realistic behaviour. It is important to introduce at this point the concept of inhibition in neuronal networks. Inhibitory neurons, unlike excitatory neurons which have been discussed so far, produce action potentials that decrease the membrane potential of the receiving neuron rather than increase it, meaning that the receiving neuron must receive more excitatory input in order to overcome its threshold and fire compared to what it normally would. Inhibitory neurons maintain balance in a network, preventing it from becoming over-stimulated or "epileptic". It is with populations of excitatory and inhibitory neurons that biological, spiking neural networks can be created.

The final major component of the neural network model is how it is able to be modified over time. The approach taken is reinforcement learning through spike time dependent plasticity (STDP). STDP is simply the increasing of the strength of a connection if the input, or presynaptic spike to a neuron is followed by a postsynaptic spike, also called a "pre before post" event [17]. In unsupervised STDP, whenever this occurs the connection between the respective neurons is increased, and conversely if a "post before pre" event occurs the connection is weakened. This report will utilise a form of reinforcement learning STDP mechanism, whereby the strengthening or weakening of a connection that experiences a "pre before post" event is delayed and the selection of reward or punishment is chosen based on a heuristic.

Methods

The Foraging Problem and Network Structure

The details of the foraging problem are almost identical to that of Sanda et al.'s implementation [17], in which "epochs" of 300 milliseconds with a 0.5 millisecond timestep divided up the simulation time. The virtual environment was composed of a 49 by 49 occupancy grid containing some density of randomly scattered food. At the end of each epoch, the virtual sprite moved one square in one of the eight compass directions and if any food occupied that square, it was "gathered" and another food item was placed randomly on the map to maintain constant density. Each epoch lasted enough time for the neural network to receive input stimuli, process this stimuli and produce an output. The neural network itself contained three layers of Izhikevich regular spiking pyramidal cell neurons [20][21]; an 7 by 7 input layer (I), a 28 by 28 middle layer (M) and 3 by 3 output layer (O). The input layer which provided the only stimulus to the network is related to the 7 by 7 area around the virtual sprite, such that at each epoch, only the neurons in this layer that corresponded to a food-occupied grid location were stimulated. Thus, the "middle" neuron of this layer corresponding to the virtual sprite was never stimulated, 8 neurons of the layer corresponded to the inner circle around the sprite, 16 corresponded to one move further out and 24 corresponded to the outermost circle of occupancy in the visual field of the sprite. Each of these input layer neurons had uniform weight excitatory connections to 9 randomly selected middle layer neurons, and each of these middle layer neurons are connected to every output layer neuron with weights initially drawn from a normal distribution. The output layer neurons correspond to a compass direction with the middle neuron unused. The highest spiking neuron of the output neurons was used as the direction, and pseudo-random numbers were generated to break ties. Additionally to prevent movement cycles in the learning process, a 1% chance was introduced for a random direction to be chosen irrespective of activity in the output layer.

Implementing STDP

Rewarded spike time dependent plasticity (STDP) was implemented between the middle layer neurons and output neurons such that at the end of each epoch, the sprite moved and if food was gathered then the synaptic connections which had experienced "pre before post" activity were strengthened. If food was not gathered, then these connections were weakened, however since not gathering food is much more likely than gathering food, this punishment was set to be -0.1 times that of the reward case. The model for spike time dependent plasticity was taken from a model of reinforcement learning for a two-joint virtual arm [15] [22], as was the algorithmic structure of fixed frequency update calls within the NetPyNE simulation. However STDP alone is insufficient to train a network, and so synaptic balancing mechanisms, random variability and inhibition were used in order to obtain above-chance level results.

Synaptic Balancing Mechanisms

In total four synaptic balancing mechanisms were used to obtain above-chance gathering rate performance. Without any of them the network tended to some extreme of synaptic activity that did not allow for learning to occur.

1. Hetero-synaptic Balancing

In both experimental studies and detailed biological models it has been shown that the sum of input synapse weights to any cell must be held approximately constant in order to prevent run-away spiking dynamics [17]. Therefore a simple rule was implemented such that whenever a single input synapse's weight was adjusted by an STDP reward or punishment, all other input synapses were adjusted to compensate and thus constant synaptic input to each cell was maintained. Due to the implementation of frequency targeting, this constant synaptic input is in fact a very slowly changing variable that depends on the firing frequencies of the output neurons.

2. Frequency Targeting

It was found that the spiking frequency of the output neurons was immensely important on decision making [17]. When spiking frequency of the output layer was low (< 1 Hz), the number of epochs with zero spikes were high as excitability was insufficient to reach the spiking threshold. In order to avoid these lengths of silent output epochs, it was desired that the overall spiking frequency of the output layer was high, however increasing the output layer excitability too high resulted in epochs where all the output cells spiked, and some multiple times, thus increasing the number of non-zero ties. Hence a frequency of 1.6 Hz was chosen as a default output firing rate as a good balancing point between silent epochs and high-spiking spike-tied epochs. In order to achieve this desired firing frequency, the excitatory weight sum was gradually adjusted using a slow-moving variable that represented the target synaptic weight W_T of the cell. If the particular output cell was firing under the desired frequency, this variable was increased and if it was over-firing this variable was decreased by Δ_W as per equation 3.

$$W_T(n{+}1) = \begin{cases} W_T(n) \cdot (1 + \Delta_W) & f_{spike} < f_{target} \\ W_T(n) \cdot (1 - \Delta_W) & f_{spike} > f_{target} \end{cases} \tag{3}$$

3. Output Balancing

Previous work by the authors of the paper upon which this project is based [23] revealed that balancing the weights of output synapses helped prevent a several neurons controlling the whole network and others from dying off entirely. This was implemented by dividing the STDP reward by a normalising ratio R_n, equal to the current synpatic output sum divided by the initial synaptic output sum for each cell (equation 4).

$$R_n = \frac{\sum W_{curr}}{\sum W_{init}} \tag{4}$$

As a result already strengthened neurons were less able to increase in weight compared to weaker neurons, giving the later an opportunity to recover and an advantage over the already dominating neurons.

4. Soft Thresholding

The final synaptic balancing is a method called "soft thresholding" which was in-built into the STDP model used. This mechanism is similar to output balancing in that it adds another multiplicative factor to the synaptic reward or punishment that depends not on any other synaptic connection weights but on a chosen maximum weight. As the synaptic weight increases towards this value, this ratio S for positive rewards approaches 0 and 1 for punishments as per equation 5. As the weight decreases towards zero, the inverse of this relation is true, and so no weight can ever reach 0 or some maximum weight (set to 10 times the average starting excitatory weight) by the STDP mechanism.

$$S = \begin{cases} (1 - \frac{W}{W_{max}}) & \text{if rewarding} \\ \frac{W}{W_{max}} & \text{if punishing} \end{cases} \tag{5}$$

Random Variability

Noise and random variability is an omnipresent component of almost any neural network and allows random outputs to occur for a given input, which may or may not be rewarded. If this is able to occur in some way several times then it is likely that the synapse's strength will surpass its peers and if "correct" is much more likely to strengthen further, while if not, is likely to fall in strength back to average or lower. Random variability was introduced into the system in several ways. The initial weights of the excitatory connections were drawn from the absolute value of a normal distribution with 0.5% variance. Additionally the planar coordinates of each layer (x and z coordinates) were drawn from a uniformly random distribution within a fixed square and the layers separated along the y axis by 100 µm. The action potentials traveled between connected neurons with a propagation velocity of 100 µm/ms and thus a random geometrical arrangement enabled variability in the arrival of action potential signals.

Inhibition

A simple inhibition model was used in this simulation based on the model employed by Sanda et al. [17] whereby each excitatory connection between the middle and output layers was paired with an inhibitory connection with roughly the same net weight which was essential in preventing over-stimulation of the output layer. It was essential that the weight of the inhibitory connections was carefully tuned to be under the net final excitatory weight since over-inhibition led to silent epochs while under-inhibition achieved epochs with many non-zero ties. NetPyNE is very capable of creating populations of inhibitory neurons and implementing a middle layer with mixed excitatory and inhibitory neurons would be of interest for further study as a more biologically realistic neuronal network model.

Results

Discussion

Conclusion

Acronyms

CIFAR Canadian Institute For Advanced Research

MADELEINE 1959 seminal neural network

NetPyNE "Networks using Python and NEURON" library

NEURON Yale-developed neuron simulation environment

STDP Spike Time Dependent Plasticity

References

[1] W. S. McCulloch and W. Pitts, “A logical calculus of the ideas immanent in nervous activity,” *The bulletin of mathematical biophysics*, vol. 5, no. 4, pp. 115–133, 1943.

[2] M. Minsky and S. Papert, “Perceptron: An introduction to computational geometry,” *The MIT Press, Cambridge, expanded edition*, vol. 19, no. 88, p. 2, 1969.

[3] J. J. Hopfield and D. W. Tank, “Neural computation of decisions in optimization problems,” *Biological cybernetics*, vol. 52, no. 3, pp. 141–152, 1985.

[4] J. Schmidhuber, “Deep learning in neural networks: An overview,” *Neural networks*, vol. 61, pp. 85–117, 2015.

[5] M. Liang and X. Hu, “Recurrent convolutional neural network for object recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3367–3375.

[6] F. Zenke and W. Gerstner, “Limits to high-speed simulations of spiking neural networks using general-purpose computers,” *Frontiers in neuroinformatics*, vol. 8, p. 76, 2014.

[7] F. G. Ashby and S. Helie, “A tutorial on computational cognitive neuroscience: Modeling the neurodynamics of cognition,” *Journal of Mathematical Psychology*, vol. 55, no. 4, pp. 273–289, 2011.

[8] F. G. Ashby, S. Ell, V. Valentin, and M. Casale, “Frost: A distributed neurocomputational model of working memory maintenance,” *Journal of Cognitive Neuroscience*, vol. 17, no. 11, pp. 1728–1743, 2005.

[9] M. J. Frank, “Dynamic dopamine modulation in the basal ganglia: A neurocomputational account of cognitive deficits in medicated and nonmedicated parkinsonism,” *Journal of cognitive neuroscience*, vol. 17, no. 1, pp. 51–72, 2005.

[10] M. Hartley, N. Taylor, and J. Taylor, “Understanding spike-time-dependent plasticity: A biologically motivated computational model,” *Neurocomputing*, vol. 69, no. 16-18, pp. 2005–2016, 2006.

[11] J. Leveille, M. Versace, and S. Grossberg, “Running as fast as it can: How spiking dynamics form object groupings in the laminar circuits of visual cortex,” *Journal of Computational Neuroscience*, vol. 28, no. 2, pp. 323–346, 2010.

[12] H. Markram, E. Muller, S. Ramaswamy, M. W. Reimann, M. Abdellah, C. A. Sanchez, A. Ailamaki, L. Alonso-Nanclares, N. Antille, S. Arsever, *et al.*, “Reconstruction and simulation of neocortical microcircuitry,” *Cell*, vol. 163, no. 2, pp. 456–492, 2015.

[13] C. Eliasmith, J. Gosmann, and X. Choo, “Biospaun: A large-scale behaving brain model with complex neurons,” *arXiv preprint arXiv:1602.05220*, 2016.

[14] M. Frega, M. Tedesco, P. Massobrio, M. Pesce, and S. Martinoia, “Network dynamics of 3d engineered neuronal cultures: A new experimental model for in-vitro electrophysiology,” *Scientific reports*, vol. 4, p. 5489, 2014.

[15] S. A. Neymotin, G. L. Chadderdon, C. C. Kerr, J. T. Francis, and W. W. Lytton, “Reinforcement learning of two-joint virtual arm reaching in a computer model of sensorimotor cortex,” *Neural computation*, vol. 25, no. 12, pp. 3263–3293, 2013.

[16] S. Dura-Bernal, S. A. Neymotin, C. C. Kerr, S. Sivagnanam, A. Majumdar, J. T. Francis, and W. W. Lytton, “Evolutionary algorithm optimization of biological learning parameters in a biomimetic neuroprosthesis,” *IBM journal of research and development*, vol. 61, no. 2/3, pp. 6–1, 2017.

[17] P. Sanda, S. Skorheim, and M. Bazhenov, “Multi-layer network utilizing rewarded spike time dependent plasticity to learn a foraging task,” *PLoS computational biology*, vol. 13, no. 9, e1005705, 2017.

[18] S. Dura-Bernal, B. Suter, P. Gleeson, M. Cantarelli, A. Quintana, F. Rodriguez, D. J. Kedziora, G. L. Chadderdon, C. C. Kerr, S. A. Neymotin, *et al.*, “Netpyne: A tool for data-driven multiscale modeling of brain circuits,” *bioRxiv*, p. 461 137, 2018.

[19] W. Gerstner, W. M. Kistler, R. Naud, and L. Paninski, *Neuronal dynamics: From single neurons to networks and models of cognition*. Cambridge University Press, 2014.

[20] E. M. Izhikevich, *Dynamical systems in neuroscience*. MIT press, 2007.

[21] E. M. Izhikevich and G. M. Edelman, “Large-scale model of mammalian thalamocortical systems,” *Proceedings of the national academy of sciences*, vol. 105, no. 9, pp. 3593–3598, 2008.

[22] G. L. Chadderdon, S. A. Neymotin, C. C. Kerr, and W. W. Lytton, “Reinforcement learning of targeted movement in a spiking neuronal model of motor cortex,” *PloS one*, vol. 7, no. 10, e47251, 2012.

[23] S. Skorheim, P. Lonjers, and M. Bazhenov, “A spiking network model of decision making employing rewarded stdp,” *PloS one*, vol. 9, no. 3, e90821, 2014.

Contribution and Reflection