Spiking Neural Networks: An Algorithmic Perspective (Extended Abstract)*

Nancy Lynch
MIT
lynch@csail.mit.edu

Cameron Musco
MIT
cnmusco@mit.edu

Merav Parter
MIT
parter@mit.edu

June 12, 2017

Abstract

We initiate the study neural networks from the perspective of distributed algorithms. Our ultimate aim is to abstract real neural networks in a way that, while not capturing all interesting features, preserves high-level behavior and allows us to make biologically relevant conclusions.

Towards this goal, we consider the implementation of various algorithmic primitives in a simple yet biologically plausible model of *stochastic spiking neural networks*. Our model captures the spiking behavior observed in real neural networks along with the widely accepted notion that spike responses and neural computation in general are inherently stochastic.

We first show how this stochastic behavior can be leveraged to solve a basic *symmetry-breaking task* in which we are given neurons with identical firing rates and want to select a distinguished one. In computational neuroscience, this is known as the winner-take-all (WTA) problem, and it is believed to serve as a basic building block in many tasks, e.g., learning, pattern recognition and clustering. Our main contribution is an explicit construction of an efficient WTA circuit, a proof of correctness and runtime bounds, and a concrete application to the problem of selecting a neuron of maximum firing rate from a group.

We next consider the use of stochastic behavior in the somewhat orthogonal task of similarity testing. We present a neural network which, given two n-length patterns of firing neurons, is able to distinguish whether the patterns are equal or ϵ-far from being equal. Randomization allows us to solve this task with a very compact network, using just $\tilde{O}(\sqrt{n}/\epsilon)$ auxiliary neurons. At the heart of our solution is the design of a t-round neural *random access memory* (or neuro-RAM) mechanism that can be implemented with $O(n/t)$ auxiliary neurons. We show that this tradeoff between runtime and network size, which can also be achieved using deterministic threshold gates, is nearly optimal. This demonstrates a neural primitive in which stochastic behavior does not seem to give any computational advantages, contrasting with the key role of randomness in solving the WTA and similarity testing problems.

*This extended abstract partially overviews work contained in [LMP17a], which can be accessed at https://arxiv.org/abs/1610.02084 and [LMP17b], which can be accessed at https://arxiv.org/abs/1706.01382.

1 Introduction

Neural networks are studied in a number of academic communities from a wide range of perspectives. Significant work in computational neuroscience focuses on developing somewhat realistic mathematical models for these networks and generally studying their capacity to process information [Izh04, Tra09]. On the more theoretical side, a variety of artificial network models such as perceptron and sigmoidal networks, Hopfield networks, and Boltzmann machines have been developed. These models are tractable to theoretical analysis and studied in the context of their computational power, and applications to general function approximation, classification, and memory storage [HSW89, MSS91, SS95, Maa97]. In practical machine learning, biological fidelity and often theoretical tractability are put aside, and researchers study how neural-like networks and learning rules can be used to efficiently represent and learn complex concepts [Hay09, LBH15].

1.1 Our Agenda

In contrast to the common approach in computational neuroscience and machine learning, we focus not on general computation ability or broad learning tasks, but on specific algorithmic implementation and analysis. We define a model of neural computation and a number of simple algorithmic problems that seem to be important building blocks for higher level processing and learning tasks. We then design neural networks in our model that solve these problems, rigorously analyzing the complexity of our solutions in terms of asymptotic runtime and network size bounds. We hope that this new paradigm will provide new insights about computational tradeoffs, the power of randomness, and the role of noise in biological systems.

While focusing on somewhat different questions, our line of work is inspired by (1) work on the computational power of spiking neural networks, most notably by Maass et al. [Maa97, Maa99, Maa00] and (2) the work of Les Valiant [Val00a, Val00b, Val05], who defined the *neuroidal model* of computation and investigated implementations of basic learning modules within this model.

2 Computational Model

We work with biologically inspired *spiking neural networks* (SNNs) [Maa96, Maa97, GK02, Izh04, HJM13], in which neurons fire in discrete pulses, in response to a sufficiently high membrane potential. This potential is induced by spikes from neighboring neurons, which can have either an excitatory or inhibitory effect (increasing or decreasing the potential). As observed in biological networks, and departing from many artificial network models, neurons are either strictly inhibitory (all outgoing edge weights are negative) or excitatory.

In this work we consider *static* networks and ignore synaptic plasticity and possibly dynamic synapse weights. This is natural as the problems considered are not primarily focused on learning or recall, where synaptic plasticity seems to play a major role. An interesting direction for future work is extending our model to include synaptic plasticity and tackling important learning problems within this model, possibly employing the algorithmic primitives developed here as subroutines.

Our model is *stochastic* – each neuron functions as a probabilistic threshold unit, spiking with probability given by applying a sigmoid function to the membrane potential. While a rich literature focuses on deterministic circuits [MP69, HT+86] we employ a stochastic model as it is widely accepted that neural computation is inherently noisy [AS94, SN94, FSW08], and that while this can lead to a number of challenges, it also affords significant computational advantages [Maa14]. Of course, our noise model is a significant simplification of noise in biological systems. For example, we model noisy behavior at each neuron, rather than at each synapse, which seems to be the dominant

source of noise in real systems [AS94]. As always, extending the theoretical results for our simplified model to more biologically accurate models is an interesting direction for future research.

2.1 Network Structure

We now give a formal mathematical definition of our computational model. A *Spiking Neural Network* (SNN) $\mathrm{N} = \langle \mathrm{X}, \mathrm{Y}, \mathrm{Z}, w, b \rangle$ consists of n input neurons $\mathrm{X} = \{x_1, \ldots, x_n\}$, m output neurons $\mathrm{Y} = \{y_1, \ldots, y_m\}$, and ℓ auxiliary neurons $\mathrm{Z} = \{z_1, ..., z_\ell\}$. The directed, weighted synaptic connections between X, Y, and Z are described by the weight function $w : [\mathrm{X} \cup \mathrm{Y} \cup \mathrm{Z}] \times [\mathrm{X} \cup \mathrm{Y} \cup \mathrm{Z}] \to \mathbb{R}$ (a weight of 0 indictions no connection). For any neuron v, $b(v) \in \mathbb{R}_{\geq 0}$ is the activation bias – roughly, v's membrane potential must reach $b(v)$ for a spike to occur with good probability.

The synapse weight function is restricted in a few notable ways. The in-degree of every input x_i is zero. That is, $w(u, x) = 0$ for all $u \in X \cup Y \cup Z$ and $x \in X$. This restriction bares in mind that the input layer might in fact be the output layer of another circuit and so incoming connections are avoided to allow for the composition of sub-circuits in modular designs. Note however that feedback to the inputs is an important feature of neural computation, and can still be implemented by connecting them to a set of intermediate neurons which replicate the input behavior and receive feedback from other neurons. For example, this strategy is used in our WTA circuit construction.

We additionally restrict each neuron to be either inhibitory or excitatory: if v is inhibitory, then $w(v, u) \leq 0$ for every u, and if v is excitatory, $w(v, u) \geq 0$ for every u. All inputs and outputs are excitatory.

2.2 Network Dynamics

An SNN[1] evolves in discrete, synchronous *rounds* as a Markov chain. The firing probability of every neuron in round t depends on the firing status of its neighbors in the preceding round $t - 1$. This probabilistic firing is modeled using a standard sigmoid function, with details given below.

For each neuron u, and each time $t \geq 0$, let $u^t = 1$ if u fires (i.e., generates a spike) in time t. Let u^0 denote the initial firing state of the neuron. For simplicity we typically consider static inputs, so $x_j^0 = 1$ for firing inputs and $x_j^0 = 0$ for non-firing inputs. The values y_j^0 and z_i^0 are arbitrary. For each non-input neuron u and every $t \geq 1$, let $pot(u, t)$ denote the membrane potential at round t and $p(u, t)$ denote the corresponding firing probability ($\Pr[u^t = 1]$). These values are calculated as:

$$pot(u,t) = \sum_{v \in \mathrm{X} \cup \mathrm{Y} \cup \mathrm{Z}} w_{v,u} \cdot v^{t-1} - b(u) \text{ and } p(u,t) = \frac{1}{1 + e^{-pot(u,t)/\lambda}} \tag{1}$$

where $\lambda > 0$ is a *temperature parameter*, which determines the steepness of the sigmoid. It is easy to see that λ does not affect the computational power of the network. A network can be made to work with any λ simply by scaling the synapse weights and biases appropriately.

We will use X^t, Y^t and generally A^t for some set of neurons A to denote the collective firing status of these neurons at time t. As the input is typically fixed, we often just write X to refer to the n-bit string corresponding to the input firing pattern.

In prior work [LMP17a], we considered a very similar network model with the distinction that we enforced a synchronous 'round-robin' firing pattern where the inputs X, the auxiliary neurons Z, and the outputs Y fire in succession. This was needed for some of the highly tuned algorithms and lower bounds in that work, however is not required for the results discussed here.

[1] Throughout this work, we use 'SNN' not as a general term, but to refer to a network defined under our model.

2.3 Computational Problems in our Model

We primarily focus on network design tasks. In the simplest form, we are given a (possibly multi-valued) target function $f : \{0,1\}^n \to \{0,1\}^m$ and seek to design a small SNN (using few auxiliary neurons) that given fixed input X converges quickly to the output $Y = f(X)$ (or any $Y \in f(X)$ if f is multi-valued).

In particular, we say a network converges in t rounds if for $t' \geq t$, for any input X, $Y^t = f(X)$ with high probability. Here 'with high probability' or w.h.p. denotes with probability at least $1 - 1/n^c$ for some constant c. Our main interest in network design is in characterizing the *tradeoff* between our two complexity measures: the number of auxiliary neurons and the convergence time.

In addition to the simple fixed input setting, we will present some preliminary results in which X is a collection of probabilistic firing input neurons, where the i^{th} neuron x_i fires in each round t with probability p_i. In this case, we design networks whose output converges to some function of the vector of probabilities $P = [p_1, ..., p_n]$.

3 Winner-Take-All Networks

We first discuss the winner-take-all problem, which is important for many neural learning and pattern recognition tasks [KU87, Now89, LIKB99, IK01, KK94, GL09]. The simplest setting is the binary input case: given $X \in \{0,1\}^n$ with at least one firing neuron, we want our network to converge to output $Y \in \{0,1\}^n$ which has a single firing neuron, corresponding to a firing input. In this way, we break symmetry and select of a 'winning' input. We present a very simple SNN which solves WTA using just two auxiliary inhibitor neurons, which we show are required.

Theorem 1. *There exists an SNN with two inhibitory neurons, that given fixed input $X \in \{0,1\}^n$, converges with probability $1 - 1/n^c$ to a single firing output corresponding to a firing input in $O(\log^2 n)$ rounds. In contrast, any SNN with just one inhibitor needs $\Omega(n^c)$ rounds for convergence.*

A full proof of Theorem 1 is given in [LMP17a]. As discussed, this work uses a slightly different model of SNN dynamics, however the proof goes through for the simpler model presented here. Here we sketch the two inhibitor construction. We use the well-studied idea of lateral inhibition: firing inputs excite the inhibitors, which indirectly lets them suppress each others firing and 'compete' to become the winning input.

For simplicity, we assume that $\lambda = 1/(c_1 \log n)$ for large c_1, which ensures via (1) that if $pot(u,t) \geq .5$, $u^t = 1$ w.h.p.

We label our inhibitors z_s – the stability inhibitor which ensures that the WTA state stabilizes once it is reached, and z_c – the convergence inhibitor, which drives competition and convergence to a single winning input.

We place an excitatory connection from every output to both inhibitors with weight $\mathrm{w}^{\mathrm{out}} = 1$ and set the biases $b(z_s) = .5$, $b(z_c) = 1.5$. By (1), z_s fires w.h.p. in round t whenever at least one output fires in round $t-1$; z_c fires w.h.p. whenever at least two outputs fire.

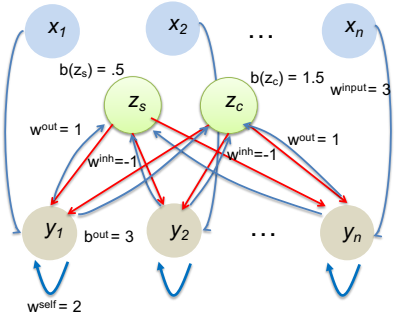

Figure 1: Two Inhibitor WTA

We place an inhibitory connection from z_s and z_c to every output with weight $\mathrm{w}^{\mathrm{inh}} = -1$, an excitatory connection from x_i to y_i with weight $\mathrm{w}^{\mathrm{input}} = 3$, and an excitatory self-loop on each output with weight $\mathrm{w}^{\mathrm{self}} = 2$. Finally, we set the output biases to $\mathrm{b}^{\mathrm{out}} = 3$.

The above parameters ensure that only outputs corresponding to firing inputs ever fire w.h.p. If we have not yet reached WTA and both z_s and z_c fire in round t, any output that fired in round t (and so has an active self-loop) will have $pot(y_i, t+1) = 0$ and fire with probability $1/2$ in round $t+1$ by (1). Thus, in $O(\log n)$ rounds w.h.p. at most a single firing output remains. If exactly one output fires in two consecutive rounds, t and $t+1$, then just z_s fires in round $t+1$ w.h.p. Its inhibition is large enough to prevent any other output from beginning to fire w.h.p. but small enough that the winner will continue to fire in round $t+2$ and beyond, ensuring convergence.

Two consecutive rounds in which exactly one output fires will occur with constant probability in each execution of the tournament described above, giving our bound of $O(\log^2 n)$ rounds to solve WTA w.h.p. Note that our solution critically leverages the stochastic nature of our neurons – by introducing inhibition that causes the output neurons to fire with probability $1/2$, these neurons randomly compete with each other, breaking the symmetry of the original input.

Non-Binary WTA As an example application of our binary WTA circuit, we outline how it can be used to solve WTA in the non-binary setting: each input has firing rate $p_i \in [0, 1]$ and we want a single output firing that corresponds to an input with a relatively high firing rate. We consider, as a preliminary step, the case where inputs fire at discrete rates: $p_i = 2^{-j}$ for some integer $j \in [0, k]$ and $k = O(1)$. Our goal is to design a circuit that given randomly firing inputs converges to a output Y in which w.h.p. the only firing neuron corresponds to an input neuron of maximum rate.

Theorem 2. *There exists an SNN with $O(n \log n)$ auxiliary neurons that, for any vector $P \in \{1, 2^{-1}, ..., 2^{-k}\}^n$, given inputs firing independently according to P, converges w.h.p. within $O(\log^2 n)$ rounds to a single firing output corresponding to an input with maximal firing rate.*

We omit details due to space constraints. At a high level, for every input x_i, we use a chain of $k' = \Theta(\log n)$ excitatory neurons $H_i = [x_i \to h_{i,1} \to \ldots \to h_{i,k'}]$, which we call a *history module*. The number of firing neurons in the module equals the number of times that x_i fired in the last $\Theta(\log n)$ rounds, and thus serves as an estimate of p_i.

Next, for each firing rate 2^{-j}, we have a designated binary WTA module N_j that is responsible for selecting a winner among all inputs with firing rate of *at least* 2^{-j}. A neuron will only compete in N_j if the number of activated neurons in H_i is large enough – i.e., it has a high enough rate. We finally connect the outputs of these $k+1$ WTA modules to a simple circuit using $O(k)$ neurons which ensures that the nonzero output corresponding to the highest firing rate is selected.

4 Similarity Testing and a Neuro-RAM

We conclude by briefly discussing similarity testing, a basic building block for pattern recognition.

4.1 A Randomized Algorithm for Similarity Testing

Given $X_1, X_2 \in \{0, 1\}^n$ we seek to test if $X_1 = X_2$ or if the hamming distance between the vectors is $\geq \epsilon n$ for some parameter ϵ. It is not hard to see that this can be accomplished by selecting $\Theta(\frac{\log n}{\epsilon})$ indices at random and checking if X_1 and X_2 match at these indices. Of course if $X_1 = X_2$, they will always match. If the hamming distance is $\geq \epsilon n$, w.h.p. they will not match at least once.

To implement this idea in an SNN we need to 1) generate random indices, 2) given an index, 'read' the corresponding position of the inputs. The first task can be accomplished using the stochastic behavior of our neurons – any neuron with potential 0 spikes with probability $1/2$ in each around, and so a set of $\log n$ of these neurons can be used to generate a random index.

The second task is more difficult – it amounts to implementing a neural random access memory module – a *neuro-RAM*, which given $X \in \{0,1\}^n$ and index $Y \in \{0,1\}^{\log n}$, outputs the bit $X(Y)$ (i.e., the bit at the position of X given by interpreting Y as the binary encoding of an integer.)

4.2 An Efficient Neuro-RAM Construction

It is easy to build a Nueron-RAM using $O(n)$ neurons (that simulate AND/OR gates), however with this many neurons we can also directly compare the inputs in the similarity testing problem. We show that this can be significantly improved (proof in [LMP17b]):

Theorem 3. *For any $t \leq \sqrt{n}$, there is an SNN implementing a neuro-RAM with $O(n/t)$ auxiliary neurons that converges in t rounds w.h.p. For $t = \sqrt{n}$, the circuit uses $O(\sqrt{n})$ auxiliary neurons.*

The above, used in our randomized approach, implies the existence of an $O\left(\frac{\sqrt{n}\log n}{\epsilon}\right)$ neuron SNN for similarity testing. More generally, it shows that a compressed representation (e.g. and index) can be used to access a much larger datastore (e.g. X), using a very compact circuit. While binary coding is not very 'neural' we can imagine similar ideas extending to more natural coding schemes [AZGMS14] used for example for memory retrieval, scent recognition, or other tasks.

Notably, the construction for Theorem 3 *does not use the random nature of SNNs*. It can be implemented with simple linear threshold circuits. Surprisingly, employing VC-dimension-based arguments [Koi96], we still show that it is nearly optimal in the SNN model:

Theorem 4. *Any SNN neuro-RAM converging in t rounds w.h.p. must use $\Omega\left(\frac{n}{t\log^2 n}\right)$ neurons.*

Thus, for the neuro-RAM problem, SNNs are no more powerful than deterministic threshold circuits, up to an $\Omega(\log^2 n)$ factor. This contrasts with the key role of randomness in solving WTA and similarity testing. It also separates our spiking networks with a sigmoid probability function from less biologically-plausible but well-studied deterministic sigmoidal networks whose gates output real values, which can solve the RAM problem in one round using $O(\sqrt{n})$ gates [Koi96].

References

[AS94] Christina Allen and Charles F Stevens. An evaluation of causes for unreliability of synaptic transmission. *Proceedings of the National Academy of Sciences*, 91(22):10380–10383, 1994.

[AZGMS14] Zeyuan Allen-Zhu, Rati Gelashvili, Silvio Micali, and Nir Shavit. Sparse sign-consistent johnson–lindenstrauss matrices: Compression with neuroscience-based constraints. *Proceedings of the National Academy of Sciences*, 111(47):16872–16876, 2014.

[FSW08] A Aldo Faisal, Luc P J Selen, and Daniel M Wolpert. Noise in the nervous system. *Nature Reviews Neuroscience*, 9(4):292–303, 2008.

[GK02] Wulfram Gerstner and Werner M Kistler. *Spiking neuron models: Single neurons, populations, plasticity*. Cambridge University Press, 2002.

[GL09] Ankur Gupta and Lyle N Long. Hebbian learning with winner take all for spiking neural networks. In *2009 International Joint Conference on Neural Networks*, pages 1054–1060, 2009.

[Hay09] Simon S Haykin. *Neural networks and learning machines*, volume 3. Pearson, 2009.

[HJM13] Stefan Habenschuss, Zeno Jonke, and Wolfgang Maass. Stochastic computations in cortical microcircuit models. *PLoS Computational Biology*, 9(11), 2013.

[HSW89] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366, 1989.

[HT+86] John J Hopfield, David W Tank, et al. Computing with neural circuits- a model. *Science*, 233(4764):625–633, 1986.

[IK01] Laurent Itti and Christof Koch. Computational modelling of visual attention. *Nature Reviews Neuroscience*, 2(3):194–203, 2001.

[Izh04] Eugene M Izhikevich. Which model to use for cortical spiking neurons? *IEEE Transactions on Neural Networks*, 15(5):1063–1070, 2004.

[KK94] Samuel Kaski and Teuvo Kohonen. Winner-take-all networks for physiological models of competitive learning. *Neural Networks*, 7(6-7):973–984, 1994.

[Koi96] Pascal Koiran. VC dimension in circuit complexity. In *Proceedings of the 11th Annual IEEE Conference on Computational Complexity (CCC)*, pages 81–85, 1996.

[KU87] Christof Koch and Shimon Ullman. Shifts in selective visual attention: towards the underlying neural circuitry. In *Matters of Intelligence*, pages 115–141. Springer, 1987.

[LBH15] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 2015.

[LIKB99] Dale K Lee, Laurent Itti, Christof Koch, and Jochen Braun. Attention activates winner-take-all competition among visual filters. *Nature Neuroscience*, 2(4):375–381, 1999.

[LMP17a] Nancy Lynch, Cameron Musco, and Merav Parter. Computational tradeoffs in biological neural networks: Self-stabilizing winner-take-all networks. In *Proceedings of the 8th Conference on Innovations in Theoretical Computer Science (ITCS)*, 2017.

[LMP17b] Nancy Lynch, Cameron Musco, and Merav Parter. Neuro-RAM unit with applications to similarity testing and compression in spiking neural networks. *arXiv:1706.01382*, 2017.

[Maa96] Wolfgang Maass. On the computational power of noisy spiking neurons. *Advances in Neural Information Processing Systems 9 (NIPS)*, pages 211–217, 1996.

[Maa97] Wolfgang Maass. Networks of spiking neurons: the third generation of neural network models. *Neural Networks*, 10(9):1659–1671, 1997.

[Maa99] Wolfgang Maass. Neural computation with winner-take-all as the only nonlinear operation. In *Advances in Neural Information Processing Systems 12 (NIPS)*, pages 293–299, 1999.

[Maa00] Wolfgang Maass. On the computational power of winner-take-all. *Neural Computation*, 2000.

[Maa14] Wolfgang Maass. Noise as a resource for computation and learning in networks of spiking neurons. *Proceedings of the IEEE*, 102(5):860–880, 2014.

[MP69] Marvin Minsky and Seymour Papert. Perceptrons. 1969.

[MSS91] Wolfgang Maass, Georg Schnitger, and Eduardo D Sontag. On the computational power of sigmoid versus boolean threshold circuits. In *Proceedings of the 32nd Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 767–776, 1991.

[Now89] Steven J Nowlan. Maximum likelihood competitive learning. In *Advances in Neural Information Processing Systems 2 (NIPS)*, pages 574–582, 1989.

[SN94] Michael N Shadlen and William T Newsome. Noise, neural codes and cortical organization. *Current Opinion in Neurobiology*, 4(4):569–579, 1994.

[SS95] Hava T Siegelmann and Eduardo D Sontag. On the computational power of neural nets. *Journal of Computer and System Sciences*, 50(1):132–150, 1995.

[Tra09] Thomas Trappenberg. *Fundamentals of computational neuroscience*. OUP Oxford, 2009.

[Val00a] Leslie G Valiant. *Circuits of the Mind*. Oxford University Press, 2000.

[Val00b] Leslie G Valiant. A neuroidal architecture for cognitive computation. *Journal of the ACM (JACM)*, 47(5):854–882, 2000.

[Val05] Leslie G Valiant. Memorization and association on a realistic neural model. *Neural Computation*, 17(3):527–555, 2005.