Bridging the Semantic Gap: Emulating Biological Neuronal Behaviors with Simple Digital Neurons

Andrew Nere, Atif Hashmi, Mikko Lipasti, and Giulio Tononi
nere@wisc.edu, ahashmi@wisc.edu, mikko@engr.wisc.edu, gtononi@wisc.edu
University of Wisconsin - Madison

Abstract

The advent of non von Neumann computational models, specifically neuromorphic architectures, has engendered a new class of challenges for computer architects. On the one hand, each neuron-like computational element must consume minimal power and area to enable scaling up to biological scales of billions of neurons; this rules out direct support for complex and expensive features like floating point and transcendental functions. On the other hand, to fully benefit from cortical properties and operations, neuromorphic architectures must support complex non-linear neuronal behaviors. This semantic gap between the simple and power-efficient processing elements and complex neuronal behaviors has rekindled a RISC vs. CISC*-like debate within the neuromorphic hardware design community.*

In this paper, we address the aforementioned semantic gap for a recently-described digital neuromorphic architecture that constitutes simple Linear-Leak Integrate-and-Fire (LLIF) spiking neurons as processing primitives. We show that despite the simplicity of LLIF primitives, a broad class of complex neuronal behaviors can be emulated by composing assemblies of such primitives with low area and power overheads. Furthermore, we demonstrate that for the LLIF primitives without built-in mechanisms for synaptic plasticity, two well-known neural learning rules–spike timing dependent plasticity and Hebbian learning–can be emulated via assemblies of LLIF primitives. By bridging the semantic gap for one such system we enable neuromorphic system developers, in general, to keep their hardware design simple and power-efficient and at the same time enjoy the benefits of complex neuronal behaviors essential for robust and accurate cortical simulation.

1 Introduction

For the better part of the past decade, computer architects have leveraged the increased transistor count provided by Moore's Law toward chip multiprocessor (CMP) designs. Scaling up the number of cores has improved processor performance through parallelism, while scaling down the supply voltage has ensured these CMPs maintain a reasonable power budget. However, a recent study by Esmaeilzadeh et al. shows that, as supply voltage scaling diminishes within future technology generations, power constraints will prevent future designs from simply scaling up the number of concurrently operating cores [12]. This has motivated computer architects to re-investigate microarchitectural decisions of the von Neumann model and explore application accelerators and novel computational paradigms.

In this vein, *neuromorphic architectures*–systems that utilize modeled neurons as basic processing units and modeled synapses (i.e. the connections between neurons) as the functional memory store–present a promising novel computational paradigm. First, neurons and synapses capture both the computation and memory storage locally, thereby avoiding the "von Neumann bottleneck", or the performance limitations of separating the data from processing elements. This allows neuromorphic architectures to dissipate a significantly less amount of power as compared to contemporary CMPs with cache hierarchies and power-hungry DRAMs. Second, these neuromorphic architectures can be designed to mimic the event-driven communication behavior of biological neurons, hence providing further power savings, since their neuron-like processing elements only perform a computation and burn power when their inputs change [4, 18, 36]. Third, these neuromorphic architectures can leverage massive amount of parallelism at a finer granularity than conventional CMPs and GPGPUs [34], allowing the neuromorphic architectures to operate at a much slower frequency than conventional von Neumann machines and still match their computational capabilities for many tasks [4, 36]. Finally, these neuromorphic architectures utilize spontaneous neural activity paired with automatic abstraction capability to achieve high degrees of fault tolerance [15, 37].

The advent of neuromorphic architectures has presented the computer architecture community with an interesting challenge of its own. To build systems with billions of neurons at acceptable energy costs, each neuron must be simple and consume very little power. This seems to rule out direct support for complex neuronal behaviors, which typically require floating point representation and transcendental functions. At the same time, robust realization of complex cortical capabilities appears to require complex non-linear neuronal behaviors that are most naturally implemented using these expensive hardware primitives. As a result, there exists a *semantic gap* between these neuromorphic architectures with simple, yet energy efficient processing primitives and the complex neuronal behaviors these architectures must support for accurate cortical modeling and simulation. A few decades ago, computer architects resolved a similar problem in the context of supporting high-level language semantics on a reduced instruction set computing (RISC) architecture; a similar approach can be applied within the domain of neuromorphic architectures to bridge the *semantic gap* identified in this paper.

Finding the ideal neuromorphic primitive by itself is a very interesting research problem for computer architects; however, in this paper, we do not address this broader problem. Rather, we pick an existing neuromorphic architecture (IBM's Neurosynaptic Core) and demonstrate that a wide variety of complex neuronal behaviors can be emulated via the simple processing primitive that this architecture provides. While this architecture demonstrates a novel, low-power, and highly-parallel computing substrate, the processing primitives provided by the simple hardware neurons do not directly support many of the complex behaviors of biological neurons. Although many artificial neural network (ANN) algorithms can be directly deployed on the Neurosynaptic Core [4, 36], a growing number of neural networks rely on complex neuronal behaviors not readily supported by the IBM's hardware neurons.

Considering IBM's Neurosynaptic Core as a case study, we address the *neuromorphic semantic gap* and make the following novel contributions:

- Arguing from first principles, we build a case for simple neural building blocks that eschew support for floating point and transcendental functions, and argue that the IBM Neurosynaptic Core is an attractive substrate for deploying complex neural networks.
- We describe several neural networks, including our own model of the mammalian visual system, which require complex neuronal behaviors not directly supported by the Neurosynaptic Core hardware.
- We provide implementation abstractions necessary to realize several complex neuronal mechanisms built using the simple primitives provided by the Neurosynaptic Core and demonstrate their functional equivalence.
- We propose novel schemes for emulating various neural learning rules via simple spiking neurons.
- We analyze the power and area overheads of our proposed assemblies emulating complex neuronal behaviors and learning mechanisms.

2 Neuromorphic Architectures

In recent years, a number of neuromorphic architectures have been proposed, designed, and fabricated [1, 2, 4]. Despite the fact that the primitives provided by these substrates, their implementations (analog, digital, or mixed circuits), and application scope may vary significantly, the majority of neuromorphic architectures share some common properties. These include modeling the basic processing elements after biological neurons, designing the functional memory store after biological synapses, and leveraging massive amounts of fine-grained parallelism inspired by the parallel processing of biological brains.

2.1 Leaky-Integrate-and-Fire Neurons

Throughout decades of neural network research, different neural models have emerged, varying extensively in their biological realism and computational efficiency [19]. These models span from the simplest implementation of a memoryless perceptron [33], to the highly complicated Hodgkin-Huxley model which emulates biological details such as neuron compartments, ion channels, and neural conductances [17]. While application specific perceptron-based neural networks have been quite successful at a number of tasks, such a limited model may miss many of the features that make the mammalian brain a powerful computing device [9]. Conversely, Hodgkin-Huxley neuron models may be an attractive alternative for understanding biological phenomena in detail, but their computational cost renders them infeasible for practical applications [13]. In this context, Leaky-Integrate-and-Fire (LIF) neurons have emerged as an approximate model of the biological neurons that is attractive to both hardware designers and neural system modelers.

Figure 1 illustrates the structural and functional aspects of a LIF spiking neuron. A LIF neuron has multiple inputs (the dendrites), a simple processing unit (the soma), and an output (the axon). The soma maintains the neuron's *membrane potential*, which is a state element that stores the temporal history of the input activations of the neuron. This soma may also include other modeled parameters such as the synaptic weights corresponding to each of the dendrites, a *reset* and a *resting* membrane potential values, a firing *threshold*, a *leak*[1] of the membrane potential, and other stochastic elements and transcendental operations [21]. Figure 1 also highlights the role of each of these parameters during the LIF neuron's computation. At each time step, the soma integrates its inputs by evaluating the dot-product of the neuron's synaptic weights (W_0, W_1, and W_2) and the activations on the dendrites (D_0, D_1, and D_2). This value is then added to the membrane potential of the LIF neuron.

[1]Leak refers to the way a biological neuron's membrane potential drifts towards a resting state, and should not be confused with the leakage power or leakage current of neuromorphic hardware.

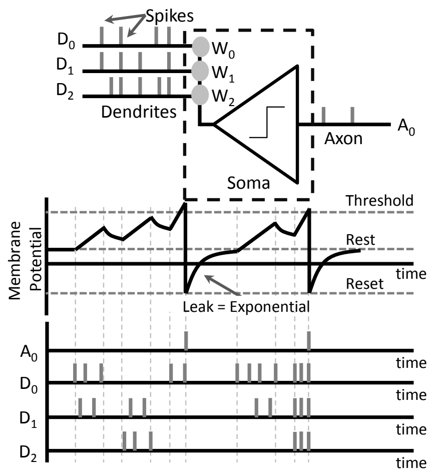


Figure 1. *LIF neuron–structure and operation.*

Next, the updated membrane potential is compared with the *threshold* value, and if the membrane potential exceeds the threshold, the neuron generates a *spike*[2] that propagates via the axon to the dendrites of other neurons. The membrane potential of the neuron is then set to a pre-specified *reset* potential. If the neuron does not generate a spike, a *leak* value or a transcendental leak function is used to update the membrane potential, which models the tendency of biological neurons to drift towards a pre-specified *resting* potential. All of the aforementioned modeled behaviors allows the LIF neurons to incorporate the concept of time into their operational model.

2.2 Design of a Neuromorphic Substrate

Given the rich and varied history of neuromorphic hardware implementations, it may seem impossibly difficult to select any specific substrate as the context for the work described in this paper. However, one can construct a reasonable argument from first principles that an approach very much in line with the Neurosynaptic Core recently described by IBM makes for a compelling neuromorphic design [4, 36]. The salient differentiating characteristics of this approach include reliance on conventional CMOS digital logic and memory, provision for binary synapses only, and fixed point arithmetic with no transcendental functions. Here, we argue that each of these choices is sensible and attractive, given the objectives of areal efficiency and extremely low power consumption at biological time scales.

First of all, avoiding exotic device, fabrication, or design technologies dramatically improves the odds of success, since the design team is able to leverage existing tools and design flows, ongoing improvements in digital CMOS technology, and existing design expertise carried over from prior projects. Second, density (areal efficiency) and low power operation can benefit from decades of experience in optimizing digital designs with these objectives in mind. Third, prior theoretical work shows that neurons with binary synapses are fundamentally no less powerful than real-valued synapses [35], enabling a space-efficient SRAM crossbar memory that uses an all-to-all bit vector representation for synaptic connectivity. These design choices are clear and easy to justify in the context of this work.

The final choice–reliance on fixed-point arithmetic with no support for transcendental functions–is a bit harder to justify. Areal efficiency and power consumption constraints would seem to preclude more complex and power-hungry arithmetic hardware, since it is quite difficult to realize the inherent nonlinearities of biological components without highly complicated digital circuits [27]. However, there is no clear evidence from the neuroscientific literature that a lack of complex neuronal behaviors will (or will not) compromise the computational capability of the neuromorphic hardware. The decision to employ linear or piecewise linear approximations for modeling of behaviors that are clearly more complex in vivo, is an obviously risky one. On the other hand, there is no solid evidence either way, and the results presented in this paper constitute prima facie evidence that complex system-level behaviors can be faithfully replicated even while compromising the behaviors of individual neurons. Hence, the initial evidence appears to support this design simplification as well.

2.3 The Neurosynaptic Core

IBM has recently developed two Neurosynaptic Core designs which utilize a digital implementation of simple Linear Leaky-Integrate-and-Fire (LLIF) neurons as the basic computational element. Rather than modeling the leak mechanism using complex transcendental functions, a LLIF processing element simply subtracts the leak value from its membrane potential at each time step.

The first Neurosynaptic Core design incorporates a number of configurable components, including 256 LLIF processing elements, 1024 axons, and a 1024x256 SRAM crossbar memory for synapses, as shown in Figure 2. This chip does not incorporate online learning capabilities. To distinguish these hardware-implemented LLIF processing elements from the general concept of LIF neurons, we will refer to them as Neurosynaptic Core Neurons, or NCNs. Each of the 1024 axons is responsible for propagating the input spikes to the system. In terms of these digital neurons, one can think of a spike to simply mean that the axon is held to a value of ‘1’, while a non-spike means the axon is held to ‘0’. These axons can be configured to route the spikes to off-core elements or to other NCN’s residing on the same core, hence allowing recurrent connectivity. The SRAM crossbar is a configurable set of binary synapses between the incoming axons (horizontal lines) and the digital neurons (vertical lines), as shown in Figure 2. A set bit (shown as a circle in the figure) indicates that a particular

[2]Spike refers to the biological mechanism of communication among neurons, and should not be confused with a hardware glitch.

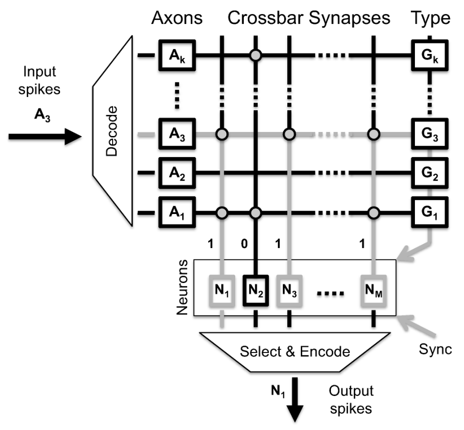


Figure 2. *IBM's Neurosynaptic Core [4].*

neuron must integrate the spikes on the corresponding axon, while an unset bit (no circle) means that this particular neuron should ignore that axon's behavior.

Because the Neurosynaptic Core is an entirely digital implementation, its elements are updated at discrete time steps. Furthermore, since ultra-low operating power is a primary design goal, the digital neurons evaluate at a biological clock rate of 1kHz. At each time step t, an input spike to the system is decoded and is routed to the appropriate axon j. Therefore, each axon j is held at a binary activity level $A_j(t)$.

When the chip is initially configured, each axon j is assigned an axon type G_j which can be parameterized to one of three values (0, 1, 2). Likewise, each NCN neuron i has three parameterizable synaptic weight multipliers (S_i^0, S_i^1, S_i^2) which correspond to each of the three axon types. In this way, each NCN in the system can be configured to have both excitatory and inhibitory connections of different strengths, but the overall crossbar can be a dense 1024x256 bit SRAM. This binary connectivity, between axon j and NCN i, is defined as W_{ij}.

Each NCN also has a parameterized linear leak term L, which models the tendency of neurons to drift to a resting potential. Finally, the present state of the neuron, its membrane potential, is captured by the variable $V_i(t)$. Taken all together, at each time step, the membrane potential of NCN i is updated by:

$$V_i(t+1) = V_i(t) + L_i + \sum_{j=1}^{K} A_j W_{ji} S_i^{G_j} \tag{1}$$

Here, K = 1024, the total number of axon inputs of the Neurosynaptic Core. When $V_i(t)$ exceeds a parameterizable firing threshold θ, the neuron produces a spike output, and its membrane potential V_i is reset to 0.

Figure 2 also demonstrates the operation of the Neurosynaptic Core for a single time step. At the time step t, an input spike on the axon 3 is presented to the system i.e. A_3 is set to 1. As a results, the decode logic pulls axon 3 to a value of '1' (as indicated by the gray horizontal line), which causes a readout of line 3 from the synapse SRAM. In the figure, we see that the SRAM has been configured so that neurons N_1, N_2, and N_M synapse with axon 3, and thus, receive values of '1' from the SRAM, while neuron N_2 receives a value of '0'. The neurons N_1, N_2, and N_M also receive axon 3's type, as defined by G_3, so they can increment their membrane potential with the appropriate multipliers $S_1^{G_3}$, $S_2^{G_3}$, and $S_M^{G_3}$ respectively. We note here that $S_1^{G_3}$, $S_2^{G_3}$, and $S_M^{G_3}$ may all be configured to different values (e.g. 100, -2, and 15). After the neurons have integrated all the spikes in this time step, neuron N_1 has crossed the firing threshold θ and has produced an output spike that will become an input spike to the chip at time step $t+1$.

IBM has also implemented a second Neurosynaptic Core design which includes stochastic online learning [36]. Many of the parameters described above are the same, though the alternative design includes only 256 input axons, and the synaptic connections are captured by a 256x256 SRAM crossbar. This alternative design allows each of the bit cells to be written during run time as well, hence emulating learning by altering synaptic connectivity.

For the rest of this paper, we assume a neuromorphic substrate in line with the 1024x256 non-learning Neurosynaptic Core [4] for several reasons. First, depending on the neural network model of interest, the all-to-all synaptic plasticity provided by the learning Neurosynaptic Core may be significantly underutilized. Here, one must consider that to support all-to-all online learning, every neuron and axon must incur additional area and power overheads to support transposable SRAMs, spike-time counters, and linear feedback shift registers [36]. Second, the learning Neurosynaptic Core implements four learning schemes (Hebbian, anti-Hebbian, STDP, and anti-STDP) [36] and in Section 4.2 we show that using composable NCN assemblies, the non-learning Neurosynaptic Core can emulate these learning schemes with reasonable area and power overhead. Furthermore, using the composable NCN assembly approach, many more aspects of the STDP and Hebbian learning schemes can be parameterized and novel learning mechanisms can be developed. It should also be noted that even though the learning core may not benefit from the NCN assemblies for learning proposed in this paper, it can benefit from the NCN assemblies emulating complex neuronal behaviors. Third, the non-learning Neurosynaptic core design provides 4x as many input axons as compared to the learning design. As Section 4.1 will show, these additional inputs are used for sparse random spikes that may be necessary to emulate complex neuronal behaviors.

3 Why Emulate Complex Behaviors?

Several classes of artificial neural networks (ANN) have already been deployed on the IBM's non-learning Neurosynaptic Core. Some of the examples include a Restricted Boltzmann Machine for handwritten digit recognition [4],

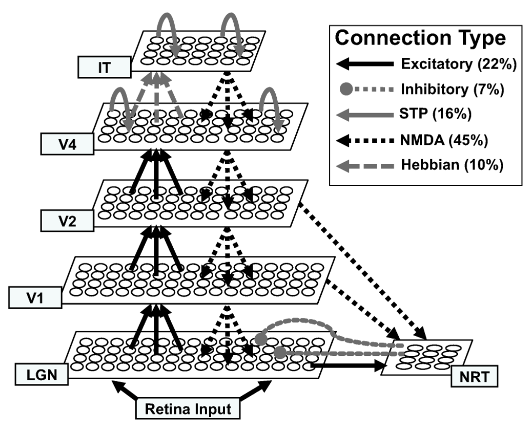


Figure 3. *In-house Visual-system Spiking Neural Network (VSNN) model.*

a Hopfield Neural Network performing an auto-associative memory task [36], a feedforward neural network performing a simple autonomous driving task [4], and an artificial neural network Pong player [4]. In each of these examples, the software implementation of the artificial neural network does not rely on any complex neuronal behavior. Thus, the neural networks corresponding to each of these applications can directly be deployed on the Neurosynaptic Core by mapping the neuron parameters to the available LLIF processing primitives. It should be noted that for the non-learning Neurosynaptic Core, the ANN-based applications are trained prior to deployment on the core.

In the recent years, recurrent neural network (RNN) and attractor-based networks have demonstrably outperformed traditional feedforward artificial neural networks on a number of tasks [22, 26, 29]. Studies also suggest that these RNN and attractor-based networks rely on complex neuronal behaviors like prolonged signalling effects to extract long term contextual dependencies in the data and perform robust time series predictions [3, 11, 32]. These prolonged effects and metastable dynamics have also been attributed to the robust and powerful processing capabilities of the mammalian brain [14]. While it is clear that many ANN applications can be deployed on the Neurosynaptic Core, it is not obvious if neural networks relying on such complex neuronal behaviors can be deployed on the Neurosynaptic Core in a straightforward manner.

To study the compatibility between the aforementioned complex neuronal behaviors and the IBM's Neurosynaptic Core, we use two biologically inspired models proposed by Liu et al. [23] and Richert et al. [31] and an in-house hierarchical network that models various regions of the mammalian visual cortex. Liu et al. have developed a dynamical memory network capable of storing and recalling memory states which relies on synapses modulated by short term plasticity (STP) mechanisms [25] and N-methyl D-aspartate (NMDA) receptors [10]. Similarly, Richert et al. have used spiking neurons with synapses modulated STP and NMDA receptors to implement a large scale model of several regions in the mammalian visual cortex.

A block level diagram of the in-house hierarchical Visual-system Spiking Neural Network (VSNN) is shown in Figure 3. Inspired by the organization of the mammalian visual cortex, the VSNN models regions including the Lateral Geniculate Nucleus (LGN), the Thalamic Reticular Nuclues (NRT), the Primary Visual Cortex (V1), the Secondary Visual Cortices (V2, V4), and the Inferior Temporal Cortex (IT) [6]. This model was designed to perform vision related tasks such as invariant object recognition, short-term memory retention, motion detection and object tracking, and pattern completion. While full details of VSNN model are out of the scope of this paper, we note that the VSNN leverages complex neuronal behaviors. These include synapses modulated by short term plasticity, synapses that exhibit prolonged signaling effects, and voltage-dependent[3] connections associated with NMDA receptors [7], and a Hebbian-like learning rule [20]. As Figure 3 shows, many of the connections in the VSNN rely on these complex neuronal behaviors and only 25% of the neurons in the VSNN can be directly implemented with the simple digital LLIF neurons provided by the Neurosynaptic Core hardware.

4 Composing NCN Assemblies

In this section, we demonstrate how assemblies of NCNs can be designed that accurately emulate complex neuronal behaviors and learning mechanisms described in Section 3.

4.1 Complex Neuronal Behaviors

In Section 3 we have highlighted three complex neuronal behaviors utilized by RNN and attractor based network models including the models proposed by Liu et al., Richert et al., and the VSNN model. These complex behaviors include STP modulated synapses, synapses modulated by prolonged signalling effects, and voltage-dependent synapses.

4.1.1 Short Term Plasticity (STP) Modulated Synapses

A necessary characteristic of recurrent attractor networks is their ability to quickly enter stable states based on context and sensory input. Such stable states must also dissolve just as quickly, allowing new inputs to drive the system. Biological systems achieve these behaviors via synapses that are modulated by STP. STP means that the connection strength between two neurons is temporarily modulated, either in the positive direction (potentiation) or the negative (depression). We note that this synaptic behavior is quite different from *long term plasticity*, which refers to the long lasting synaptic changes associated with learning.

[3]Voltage-dependent refers to a biological phenomena observed in the neurons present in the mammalian brain, and must not be confused with supply voltage of the neuromorphic hardware

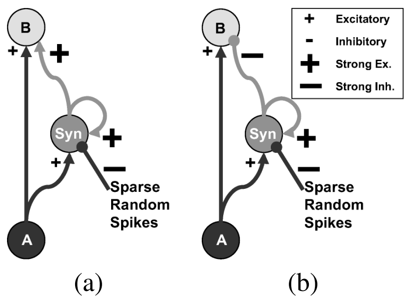


Figure 4. *NCN assembly emulating short term potentiating effects (a) and short term depressing effects (b).*

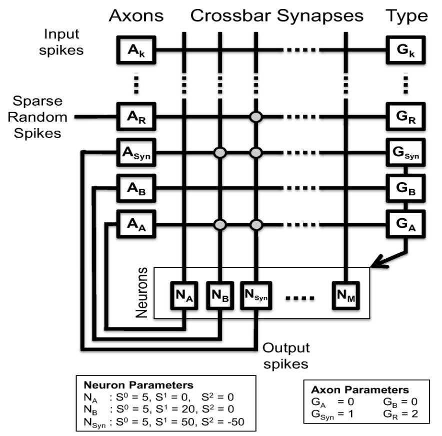


Figure 5. *Short term potentiation NCN assembly deployed on the Neurosynaptic Core.*

First, we consider short term potentiation, a synaptic effect that results in a temporary strengthening of a connection between neurons a presynaptic (source) and postsynaptic (target) neurons. Figure 4(a) shows the equivalent NCN assembly that emulates the functionality of short term potentiating synapses. One additional NCN utilizes a recurrent connection to ensure the short term potentiating effect lasts beyond a single spiking event. Initially the *Syn* NCN is not active, and the presynaptic neuron signals to the postsynaptic with a weak excitatory connection. If the presynaptic neuron continues a high level of spiking activity, the *Syn* NCN activates, and the postsynaptic neuron gets an extra excitatory boost. In biology, once the presynaptic neuron stops firing for a period of time, the synaptic strength returns to normal values (i.e. before short term potentiation took place) [24]. To achieve this behavior, the *Syn* NCN must also receive some sparse random inhibitory spikes to counter the strong excitatory self-connection (see Figure 4(a)). Since the baseline Neurosynaptic Core has 1024 input axons, these random spikes can be provided via external connections. Given these ingredients, the NCN assembly shows short term potentiation effects when the presynaptic neuron spike rate is high, but eventually returns to normal synaptic strength once the presynaptic neuron rests.

Figure 5 illustrates how the aforementioned short term potentiation assembly maps onto the IBM's Neurosynaptic Core. In this figure, N_A corresponds to the presynaptic neuron *A*, N_B corresponds to the postsynaptic neuron *B*, and N_{Syn} corresponds to the *Syn* neuron (see Figure 4(a)). N_A projects its axon back onto the same core and synapses with neurons N_B and N_{Syn}. The axon of N_{Syn} routs back to the same core and synapses with itself and the postsynaptic neuron N_B. Finally, an external connection projects onto axon A_R that synapses with the neuron N_{Syn} to provide sparse random inhibitory spikes. The figure also shows several of the neuron and axon parameters needed to implement the short term potentiating NCN assembly. The rest of the assemblies described in this section can also be deployed onto the Neurosynaptic Core in a similar manner.

Figure 6 validates the emulation of short term potentiation using an NCN assembly. The presynaptic neuron (leftmost panel) is shown firing at 100 Hz. The middle panel shows the response of a biologically realistic postsynaptic neuron model that includes built-in short term potentiating synapses. The right most panel shows how the NCN assembly achieves a functionally equivalent behavior.

Short term depression is a synaptic effect in biological neural networks that results in a temporary weakening of the connection between a presynaptic and postsynaptic neurons. Figure 4(b) shows the equivalent NCN assembly emulating the functionality of a short term depressing connection. This assembly also requires an additional intermediate NCN with a self-connection, labeled *Syn* in the figure and is equivalent to the one used to emulate short term potentiation, except the connection between the *Syn* NCN and postsynaptic neuron is inhibitory. Initially, when the *Syn* NCN is inactive, the postsynaptic neuron *B* follows the activity of the presynaptic neuron *A*. If the presynaptic neuron fires at a high spiking rate, the *Syn* NCN accumulates spikes and becomes activated. A strong, positive self-connection ensures that the *Syn* NCN stays active for an extended period of time. In turn, the *Syn* NCN projects inhibition to the postsynaptic neuron, mimicking the effect of a depressing synapse. The *Syn* NCN also requires random inhibitory input spikes to ensure that once the presynaptic neuron has stopped firing, it will eventually inactivate, and the strength of the synapse will return to normal. Like the short term potentiation NCN assembly, these inhibitory spikes can come from axons routing external connections.

Figure 7 validates the emulation of short term depression using an equivalent NCN assembly. In this figure, the presynaptic neuron (leftmost panel) fires at 100 Hz. The middle panel shows the response of a biologically realistic neuron model with short term depression, and the right panel shows the response of the NCN equivalent.

4.1.2 NMDA Modulated Synapses

Synapses modulated by the NMDA receptors provide neurons with a mechanism that allows integration of temporally correlated activity across spatially distributed neurons [5].

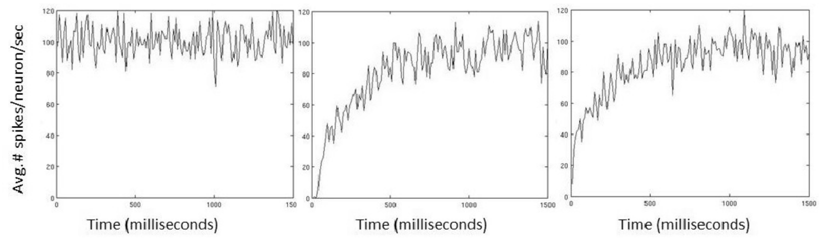


Figure 6. *Left: Average firing rate of the presynaptic neuron. Middle: Average firing rate response of a postsynaptic neuron model with short term potentiating synapses. Right: Average firing rate response of a postsynaptic NCN using the short term potentiation NCN assembly from Figure 4(a).*

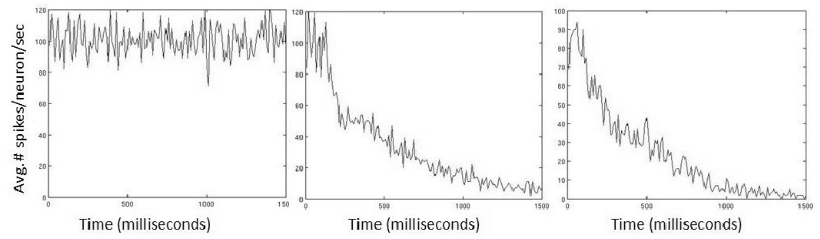


Figure 7. *Left: Average firing rate of the presynaptic neuron. Middle: Average firing rate response of a postsynaptic neuron model with short term depressing synapses. Right: Average firing rate response of a postsynaptic NCN using the short term depression emulating assembly from Figure 4(b).*

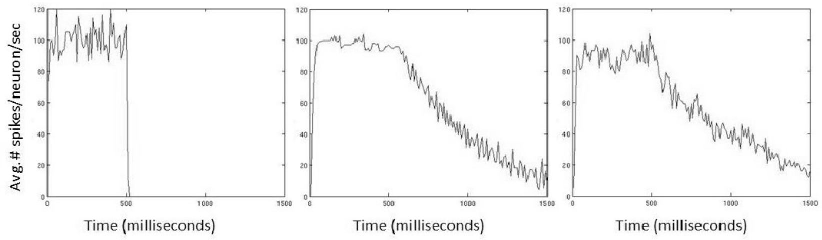


Figure 8. *Left: Average firing rate of the presynaptic neuron. Middle: Average firing rate response of a postsynaptic neuron model with NMDA-like prolonged effect. Right: Average firing rate response of a postsynaptic NCN using the NMDA emulating assembly from Figure 9(a).*

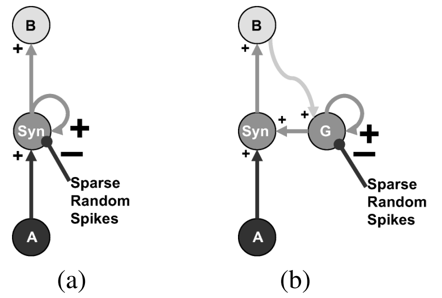


Figure 9. *NCN assembly emulating NMDA-like prolonged effects (a) and voltage-dependent synapses (b).*

For many of the connections in biological nervous systems, a spike from a presynaptic (i.e. source) neuron effects the postsynaptic (i.e. target) neuron for a brief time, typically less than a few milliseconds. As described in Section 2, the Neurosynaptic Core captures the short timescale effects of these spikes, as each spike lasts for a single time step of the digital hardware (1 ms). However, biological studies have shown that for synapses modulated by NMDA receptors, a spike from a presynaptic neuron effects the postsynaptic neuron for 10's to 100's of milliseconds [5].

Figure 9(a) shows how the NMDA-like prolonged signalling effects can be modeled using NCNs. These prolonged effects are important, since they provide spiking neurons with a mechanism to integrate temporally correlated activity across spatially distributed neurons. In this figure, an intermediate NCN (*Syn*) is inserted between the presynaptic and the postsynaptic neuron. Again, this NCN has a strong excitatory self-connection which ensures that it can maintain activity for an extended period of time, even after the presynaptic neuron has stopped firing. Thus, the *Syn* NCN continues to modulate the membrane potential of the postsynaptic neuron, emulating a prolonged effect. Similar to the short term plasticity NCN assemblies, the *Syn* NCN requires random inhibitory input to ensure the effect eventually decays over time. Figure 8 compares the behavior of a biologically realistic neuron with modeled NMDA capabilities and the equivalent NCN assembly. In the left panel, a presynaptic neuron is activated for 500 milliseconds. In the middle panel, we see the response of a biologically realistic postsynaptic neuron with modeled NMDA. Finally, the right panel shows the response of the equivalent NCN assembly. Comparing the middle and the rightmost graph, we can clearly see that the NMDA-like behavior of the biologically realistic neuron model is emulated quite accurately by the equivalent assembly, requiring the overhead of one NCN.

4.1.3 Voltage-Dependent Synapses

Connections modulated by NMDA receptors also exhibit another important biological property: voltage-dependent synapses. Stimulus at a voltage-dependent synapse affects the membrane potential of a postsynaptic neuron only if the postsynaptic neuron has already been depolarized (i.e. the membrane potential has risen above its resting level) [7]. Simply, a neuron must receive stimuli at voltage-independent synapses before the stimuli at the voltage-dependent synapses can have an effect. As a result, the effects of voltage-dependent synapses are mostly considered modulatory rather than driving. In practice, voltage-dependent synapses are quite important in hierarchically organized networks with top-down connections, such as the VSNN presented in this work. In this way, top-down signaling can be both very general and diverging in structure, and voltage-dependent connections ensure that this top-down signaling is consistent with feedforward spiking evidence.

Figure 9(b) shows how voltage-dependent synapses can be modeled using NCNs. Initially, both the synapse gate (*G*) and the synapse (*Syn*) NCNs are inactive. Even if the presynaptic neuron exhibits a high firing rate, the postsynaptic neuron is unaffected. However, if the postsynaptic neuron fires once, indicating that it has received additional evidence on another synapse, the synapse gate *G* becomes active, and stays active using a strong self-connection. When the presynaptic neuron fires again, spikes are passed through the *Syn* NCN to the postsynaptic neuron. Again, sparse random inhibitory inputs from external axons are required to ensure that the this behavior decays over time once the assembly has been inactive.

4.2 Online Learning Assemblies

Beyond the complex neuronal mechanisms described above, biological neural networks and many state-of-the-art models, such as our VSNN, Richert's model, and Liu's model, require long term synaptic plasticity, or *learning*. Spike time dependent plasticity (STDP) and Hebbian learning have been identified as two key learning rules supported by biological neurons [8, 20]. STDP adjusts the synapse strength by considering the relative timing between the presynaptic and postsynaptic spikes. Hebbian learning, on the other hand, strengthens a synapse if both the presynaptic and postsynaptic neurons are spiking at the same time. A number of studies demonstrate that learning rules inspired by STDP and Hebbian learning are essential for pattern classification and memory formation in any biological neural network [8, 20, 36]. We emulate these two types of neural plasticity on non-plastic neuromorphic hardware.

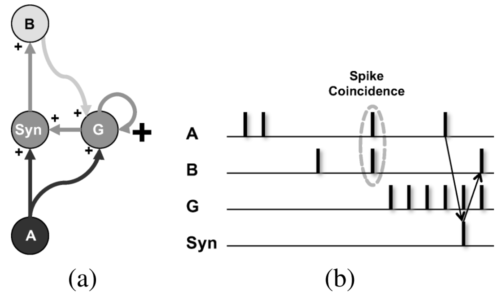


Figure 10. *(a) NCN assembly emulating Hebbian learning. (b) Spike-time plot of emulated Hebbian learning.*

4.2.1 Hebbian Learning

In the simplest form of Hebbian learning, the co-occurrence of spiking activity of both the presynaptic and postsynaptic neurons strengthens the synapse between them. With non-plastic digital neurons, this type of behavior can be realized using two additional NCNs. In Figure 10(a), we see a presynaptic neuron *A*, a postsynaptic neuron *B*, a synaptic gating NCN *G*, and the synapse NCN *Syn*. Figure 10(b) demonstrates how Hebbian plasticity is approximated with this neural assembly. Initially, the synaptic gating neuron *G* is silent, and the spikes of neurons *A* and *B* are completely independent. However, if neurons *A* and *B* fire at the same time, the synaptic gating neuron *G* acts as a coincidence detector. Afterward, the synaptic gating neuron will fire at 1kHz (i.e. the Neurosynaptic Core time step rate) due to its strong recurrent connection, and it will allow spikes from the presynaptic neuron to pass through the synapse NCN *Syn*. When *A* fires again, we see the spike propagate through the synapse NCN *Syn* to the postsynaptic neuron *B*.

4.2.2 STDP Learning

Emulating STDP learning requires a more complicated NCN assembly, as shown in Figure 11, since the relative spike times of the presynaptic and postsynaptic neurons dictate whether a synapse shows long term potentiation or long term depression. When either the presynaptic (*A*) or postsynaptic (*B*) neurons fire, an additional NCN (A_{buff}, B_{buff}) is used as a *history buffer* which remembers that the neuron has fired in the recent past. Since STDP typically operates on the order of tens of milliseconds, an inhibitory NCN (A_{inh}, B_{inh}) ensures that the history buffer is valid for a finite window of time. These inhibitory neurons accumulate spikes from the history buffer for a parameterizable period of time, and once this window is reached, the history buffer is cleared through a strong inhibitory connection.

Two additional NCNs are required for detecting spiking events that cause plasticity, one for synaptic potentiation (*P*), and one for depression (*D*). As can be seen in Figure 11, potentiation is invoked if the postsynaptic neuron *B* fires while the presynaptic neuron history buffer is indicating that *A* has fired in the recent past. In the opposite direction, NCN *D* will fire if the postsynaptic history buffer

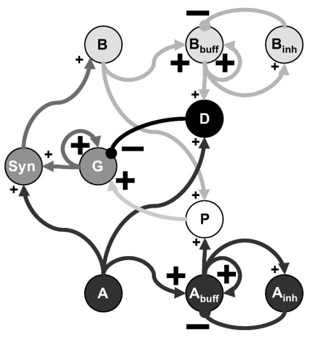


Figure 11. *NCN assembly emulating STDP learning.*

indicates spikes in the recent past, and the presynaptic neuron *A* fires in the current time step. Inhibitory connections (not shown in the figure for simplicity) from both plasticity detector NCNs (*D* and *P*) project to the history buffer neurons to make the learning rule a bit simpler.

Finally, two additional NCNs act as the synapse between the presynaptic and postsynaptic neurons. The first is the synaptic gate *G*, which is enabled or disabled by the plasticity detector neurons. A recurrent connection ensures that the gate is enabled indefinitely if potentiation has taken place. Finally, once the gate has been activated by potentiation, spikes may pass from the presynaptic neuron, through the synapse NCN (*Syn*), to the postsynaptic neuron.

In Figure 12, we show how this NCN assembly achieves STDP-like plasticity by observing the spiking behavior of each of the NCNs. In Figure 12(a), the synapse between *A* and *B* has not been yet potentiated, as indicated by *G*'s (the gating NCN) silence. Presynaptic neuron *A* fires, and its history buffer A_{buff} shows a train of spikes afterward. Later, the postsynaptic neuron *B* spikes. The potentiation detector (*P*) spikes for the conjunction of *B* and A_{buff}, and one cycle later activates *G* (gating NCN). Some time later, *A* spikes again, the spike propagates through the synapse NCN (*Syn*) one cycle later, and finally causes a spike on the postsynaptic neuron. So long as synaptic depression doesn't occur, the synapse is potentiated indefinitely, and every spike on *A* will cause a spike on *B* two time steps later.

Figure 12(b) demonstrates the opposite effect: long term depression. Here, we assume the synapse is initially potentiated, as indicated by *G*'s continuous spiking. Postsynaptic neuron *B* fires first, and this spike is remembered by B_{buff}. Shortly later, the presynaptic neuron fires, and the depression detector *D* spikes. NCN *D* projects strong inhibition to the synaptic gate *G*, disabling it. Later, when the presynaptic neuron spikes, we can see the synapse NCN *Syn* and the postsynaptic neuron *B* are completely unaffected. So long as a synaptic potentiating event does not occur, the synapse is depressed indefinitely.

4.2.3 Extensions to Learning Assemblies

So far we have demonstrated how NCN assemblies can emulate both Hebbian and STDP learning on a single synapse,

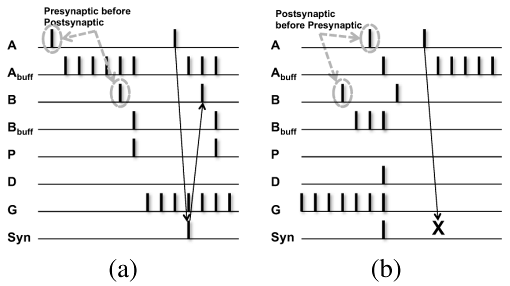


Figure 12. *(a) Spike-time plot of emulated STDP potentiation. (b) Spike-time plot of emulated STDP depression.*

where the *synaptic weight* between the presynaptic and postsynaptic neuron can take two values: zero (in the case of long term depression), or a single parameterized value determined at chip configuration time (in the case of long term potentiation). However, depending on the learning task at hand, it may be beneficial to have synaptic weights capable of a broader range of values. In the context of non-learning neuromorphic hardware, such behavior is possible, albeit at a high overhead. In Figure 13, we see a chain of plastic synapses between the presynaptic neuron *A* and postsynaptic neuron *B*. This chain of synapses is generalized to work in conjunction with either the Hebbian or STDP assemblies described above (although we note that the figure shows only the synaptic chain assembly, and not the learning rule assembly, for simplicity). To utilize this chain of synapses for Hebbian learning, each of the synaptic gating NCNs (*Gs*) must receive inputs, and detect coincident spikes, from neurons *A* and *B*. To utilize the chain for STDP learning (Figure 11), each of the synaptic gating NCNs must receive connections from the plasticity detectors *P* and *D*.

Regardless of the underlying learning rule, when potentiation occurs, the synaptic gating NCN (*G1*) is the first to activate, allowing spikes to propagate through the synapse NCN labeled *Syn1*. An excitatory connection from *G1* projects to *G2* so that the next potentiating plastic event will activate *G2*. This type of chaining can be continued indefinitely, so synapses could potentially take many different values during online learning. Furthermore, this chaining scheme ensures that synaptic gates *G1* through G_N are activated in order during potentiating plastic events. Excitatory connections from *G2* to *G1* ensure that long term depression occurs in the opposite order, that is, synaptic gates are disabled from G_N through *G1*.

By implementing learning rules with NCN assemblies (as opposed to hardware in the learning Neurosynaptic Core), one also gains a high degree of parameterization and flexibility. For example, the Hebbian learning assembly can be extended to use history buffers similar to the STDP assembly. This modification allows Hebbian learning to occur over a broader time window. The STDP assembly can also be parameterized so that learning only occurs after bursty behavior, as some of the literature has indicated the impor-

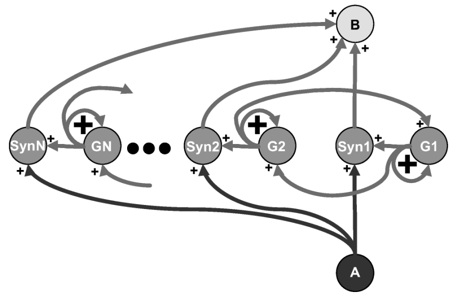


Figure 13. *NCN assembly emulating chain of synapses.*

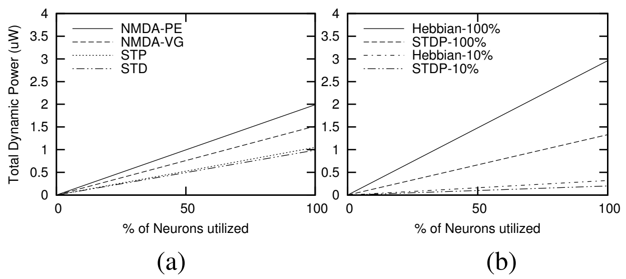


Figure 14. *Dynamic power dissipated by NCN assemblies (a) Emulating complex neural dynamics (b) Emulating learning mechanisms.*

tance of bursting over single spike behavior [19, 28]. Finally, NCN assemblies can be utilized to explore and implement new learning mechanisms. For example, there has been some interest in triple-spike-STDP learning (as opposed to the classic pair-of-spikes STDP explored in this paper), since some research argues it better fits biological experiment data [30].

5 Power and Area Overheads

In this section, we evaluate the power and area overheads of the NCN assemblies.

5.1 NCN Assembly Power Overhead

Figure 14(a) shows the overheads incurred in dynamic power of the Neurosynaptic Core when complex neuronal behaviors (NMDA-like prolonged effects, STP, and NMDA-like voltage-dependent synapses) are emulated using the equivalent NCN assemblies.

We assume a firing rate of 10Hz (comparable to biological neurons [16]) for each of the 256 NCNs available on the Neurosynaptic Core. Furthermore, we assume that each of the spikes consumes 45pJ of energy [4]. This means that without any LLIF neurons emulating complex behaviors, the Neurosynaptic Core dissipates 0.115μW of dynamic power. The amount of dynamic power consumed increases linearly as the percentage of NCNs emulating complex neuronal behaviors on a single Neurosynaptic Core increases, since each NCN assembly includes at least one NCN neuron with a strong recurrent connection to retain its state through self-excitation. Because of this self-exciting loop, the NCN will fire once per cycle until it becomes inhibited. Here, we also assume that the sparse random inhibitory inputs ensure that on average, the effect of the STP NCN lasts for 25 ms (25 cycles on the Neurosynaptic Core), while the effects relating to the NMDA receptors (prolonged signalling effects and voltage-dependent synapses) last for 50 ms (50 cycles). NCN assemblies emulating the prolonged effect of NMDA receptors incur the maximum dynamic power overhead; however, even when 100% of the NCNs are emulating NMDA assemblies, the total dynamic power is a modest 2μW.

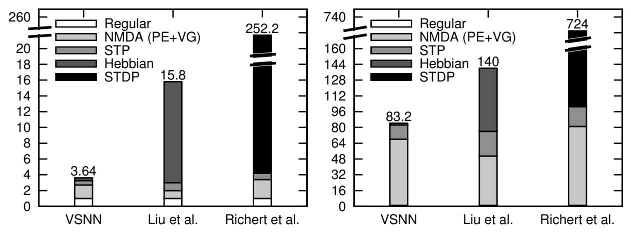
Figure 14(b) shows similar results for the two learning mechanisms that we emulate using NCN assemblies. For both of these learning mechanisms, we plot a learned rate of 10% and 100%. A learned rate of 10% means that out of all the STDP NCN assemblies, only 10% have effectively learned a pattern (and thus, their synaptic gating NCN *G* fires at a rate of 1kHz due to the recurrent connection). From this figure we see that emulating Hebbian learning incurs more power overhead than STDP learning. This is because the Hebbian NCN assembly requires only 2 extra neurons, one of which must fire at 1kHz to maintain its learned state, while the STDP assembly utilizes 8 extra neurons, of which only one fires at 1kHz (thus, resulting in a smaller average firing rate across all neurons). We note that this means 64 Hebbian learning NCN assemblies can fit on a single core, while only 25 STDP assemblies fit on the core. However, even when all of the 256 neurons on the Neurosynaptic Core are utilized to emulate Hebbian learning with 100% learning rate, the total dynamic power is estimated to be only 3μW.

As Seo et al. [36] do not explicitly report the dynamic power consumption of their learning design, we cannot compare the power overhead incurred by the proposed NCN assemblies that emulate learning against the Neurosynaptic Core with plastic synapses. However, we note that the key semantic difference between IBM's learning and non-learning designs is that the learning design must compare spike times on every spiking event. This requires additional hardware per NCN as well as a custom transposable SRAM that must be accessed twice for each spike, imposing an area and dynamic power penalty for the learning design. We do not plot absolute leakage power since, as explicitly stated by the authors, the non-learning Neurosynaptic Core design sought to optimize dynamic power rather than static power[1] [4]. However, to the first order it can be assumed that leakage will scale linearly with the additional area required for NCN assemblies.

5.2 SNN Model Overheads

Finally, we examine the area and power overheads for complete neural network models. We examine the VSNN

[1] [4] does not provide enough detail for a quantitative evaluation of static power.



(a) (b)

Figure 15. *(a) Area overhead for the three SNN models implemented with NCN assemblies. (b) Dynamic power overhead for the three SNN models implemented via NCN assemblies.*

outlined in this paper, as well as two other recent neural network models which make use of both complex neuronal behaviors and learning mechanisms. Liu and She utilize complex neuronal behaviors and learning mechanisms to construct a dynamical memory network capable of Hebbian learning, storing and recalling memory states [23]. Richert et al. have also developed a spiking neural network simulator which also makes use of such behaviors and STDP learning to model the V1, V2, and MT cortical areas [31].

In the VSNN model, we generate the appropriate NCN assemblies necessary for our network to be compatible with the Neurosynaptic Core hardware. Observing this translated network, we can determine the area overhead by counting the extra NCNs needed to emulate Hebbian learning, STP, and NMDA effects. By simulating this translated network, we can also determine the dynamic power overhead associated with these NCN assemblies for the entire system. Since we do not have access to the full system models described by Liu [23] and Richert [31], we make two basic assumptions based on our VSNN to estimate the power and area overheads. First, the average firing rate of both models (without NCN assemblies overheads), like the VSNN, is 10 Hz. Second, the average number of learned synapses that have converged to 1, like the VSNN, is 10%.

Area: In Figure 15(a), we see the area overheads (i.e. amount of extra NCNs needed) for all three models, normalized to the number of "regular" neurons (i.e. the number of neurons in the network if complex behaviors and learning mechanisms are part of the modeled neurons). Our VSNN increases in area by 264%, largely due to the overhead needed to implement voltage-dependent synapses and prolonged effects associated with NMDA. Liu's SNN model shows fairly similar trends, though a larger number of Hebbian-learned synapses increases the area even more so. Finally, Richert's SNN model would require 252x the number of neurons if it were to be deployed on the Neurosynaptic Core. This is largely due to the fact their model utilizes STDP learning on a large number of synapses, each of which requires several additional NCNs (see Figure 11).

Power: Combining the NCN assembly area overhead described above with the power overhead analysis presented in Figures 14(a) and 14(b), we estimate the dynamic power overheads associated with the VSNN as well as Liu and Richert's SNN models. In Figure 15(b), we again normalize to the number of "regular" neurons in the models - here, making the assumption that each "regular" neuron is firing at 10Hz with an energy consumption of 45pJ per spike. In the VSNN, the majority of the dynamic power comes from the NMDA-emulating NCNs, though the STP assemblies are also fairly significant. In Liu's network, more power is attributed to the Hebbian learning assemblies, again, as a result of the high degree of learning connectivity. Finally, we see that in Richert's model, nearly all dynamic power comes from the STDP NCN assemblies.

These results lead us to several conclusions. First, for neural models such as Richert's that deploy STDP learning pervasively, direct hardware support may be merited due to the substantial area and power overheads of assembly-based emulation of STDP learning. One possible vehicle for such support is IBM's learning Neurosynaptic Core. Second, however, we see that hardware support for learning alone is not sufficient, since support for NMDA and STP/STD would still require the assemblies proposed in this paper. We leave detailed quantitative evaluation of the tradeoffs of direct hardware support for learning to future work, pending further disclosure of the power and area characteristics of the learning Neurosynaptic Core. Finally, it is clear that for SNN models with topographically organized sparse connectivity, such as our VSNN model, the proposed NCN assemblies on a non-learning substrate are an attractive option.

6 Conclusion

Looking ahead to an age where Moore's law will provide more transistors than can be concurrently active under the von Neumann computer architecture, computer designers must now consider novel low-power alternative models of computation. Neuromorphic architectures, by distributing data locally among processing elements and utilizing energy efficient event-driven computation, may indeed be the low-power computing paradigm of the future. However, as evidenced by our first principle arguments as well as a state-of-the-art neuromorphic hardware prototype from IBM, these systems will likely be built on a set of simple, yet extremely energy-efficient, primitives. As a result, a semantic gap emerges between the simple processing primitives provided by hardware and the complex neuronal behaviors and learning mechanisms required by large scale cortical models.

In this paper, we have demonstrated that linear leaky-integrate-and-fire neurons–the heavily simplified processing primitives of the IBM Neurosynaptic Core–are capable of emulating complex neuronal behaviors and established learning mechanisms, effectively bridging the semantic gap in neuromorphic systems. Our results have shown that assemblies of Neurosynaptic Core Neurons (NCNs) can achieve functional equivalence to more complicated neuron models that would be far more expensive to implement digital hardware. As our results also show, the power overhead associated with these NCN assemblies is reasonable and proportional to neural network complexity, since often it is the case that only a subset of modeled neurons require the complex behaviors emulated via the proposed assemblies, while the remaining neurons can benefit from the ultra-low power of the simplified NCNs. Given the techniques outlined in this paper, it is clear that neuromorphic architectures composed of highly simplified and extremely energy efficient primitives are not only applicable to traditional artificial neural network applications, but also for large scale cortical models.

Acknowledgment

The authors would like to thank Dharmendra Modha, Paul Merolla, Steve Esser, our anonymous reviewers, and our paper shepherd Mark Oskin for their helpful comments and review of this manuscript. This work was supported by the IBM subcontract A0854200 from the DARPA SyNAPSE Grant to IBM.

References

[1] The facets project. http://facets.kip. uni-heidelberg. de, 2010. retrieved: Oct, 2011.

[2] A universal spiking neural network architecture (spinnaker). http://apt.cs.man.ac.uk/ projects/ SpiNNaker/, 2010. retrieved: Oct, 2011.

[3] Y. Amit and J. Walker. Recurrent network of perceptrons with three state synapses achieves competitive classification on real inputs. *Frontiers in Comp. Neuroscience*, 6, 2012.

[4] J. Arthur et al. Building block of a programmable neuromorphic substrate: A digital neurosynaptic core. In *IJCNN*, 2012.

[5] Y. Ben-Ari et al. Gaba$_a$, nmda and ampa receptors: a developmentally regulated 'mnage trois'. *Trends in Neurosciences*, 20(11):523 – 529, 1997.

[6] T. Binzegger, R. Douglas, and K. Martin. A quantitative map of the circuit of cat primary visual cortex. *J. Neurosci.*, 24(39):8441–8453, Sep 2004.

[7] H. Brew and I. Forsythe. Two voltage-dependent k+ conductances with complementary functions in postsynaptic integration at a central auditory synapse. *The Journal of Neuroscience*, 15(12):8011–8022, 1995.

[8] Y. Dan and M.-M. Poo. Spike timing-dependent plasticity of neural circuits. *Neuron*, 44(1):23–30, Sep 2004.

[9] M. de Kamps and F. van der Velde. From artificial neural networks to spiking neuron populations and back again. *Neural Networks*, 14(6-7):941–954, 2001.

[10] R. Dingledine et al. The glutamate receptor ion channels. *Pharmacological Reviews*, 51(1):7–62, 1999.

[11] S. El Hihi and Y. Bengio. Hierarchical recurrent neural networks for long-term dependencies. *NIPS*, 8:493–499, 1996.

[12] H. Esmaeilzadeh et al. Dark silicon and the end of multicore scaling. In *ISCA*, pages 365–376, 2011.

[13] A. Finkelstein et al. Computational challenges of systems biology. *Computer*, 37(5):26–33, 2004.

[14] C. Gros. Neural networks with transient state dynamics. *New Journal of Physics*, 2007.

[15] A. Hashmi et al. Automatic abstraction and fault tolerance in cortical microachitectures. In *ISCA'11*, pages 1–10, 2011.

[16] M. Hausser et al. The beat goes on: Spontaneous firing in mammalian neuronal microcircuits. *The Journal of Neuroscience*, 24(42):9215–9219, 2004.

[17] A. Hodgkin and A. Huxley. A quantitative description of membrane current and its application to conduction and excitation in nerve. *The Journal of physiology*, 117(4):500, 1952.

[18] K. Hynna and K. Boahen. Silicon neurons that burst when primed. *IEEE International Symposium on Circuits and Systems*, pages 3363–3366, May 2007.

[19] E. Izhikevich. Which model to use for cortical spiking neurons? *Trans. on Neural Networks*, 15(5):1063 –1070, 2004.

[20] R. Kempter et al. Hebbian learning and spiking neurons. *Physical Review E*, 59(4):4498–4514, 1999.

[21] P. Lansky et al. The parameters of the stochastic leaky integrate-fire neuronal model. *J. of comp. neuro.*, 21(2), 2006.

[22] M. Lee and S. Cho. Mobile gesture recognition using hierarchical recurrent neural network with bidirectional long short-term memory. In *UBICOMM*, pages 138–141, 2012.

[23] J. Liu and Z. She. A spike-timing pattern based neural network model for the study of memory dynamics. *PLoS One*, 4(7):e6247, 2009.

[24] H. Markram, Y. Wang, and M. Tsodyks. Differential signaling via the same axon of neocortical pyramidal neurons. *PNAS*, 95:5323–5328, 1998.

[25] B. McNaughton. Long-term synaptic enhancement and short-term potentiation in rat fascia dentata act through different mechanisms. *The J. of Phy.*, 324(1):249–262, 1982.

[26] L. Medsker and L. Jain. *Recurrent Neural Networks: Design and Applications*. The CRC Press International Series on Computational Intelligence. CRC PressINC, 2000.

[27] J. Misra and I. Saha. Anns in hardware: A survey of two decades of progress. *Neurocomput.*, 74, 2010.

[28] A. Nere et al. A neuromorphic architecture for object recognition and motion anticipation using burst-stdp. *PLoS ONE*, 7(5):e36958, 05 2012.

[29] L. Pape, J. de Gruijl, and M. Wiering. Democratic liquid state machines for music recognition. *Studies in Computational Intelligence*, pages 191–215, 2008.

[30] J.-P. Pfister and W. Gerstner. Triplets of spikes in a model of spike timing-dependent plasticity. *Journal of Neuroscience*, 26(38):9673–9682, 2006.

[31] M. Richert et al. An efficient simulation environment for modeling large-scale cortical processing. *Frontiers in Neuroinformatics*, 5(19):PMC3172707, 2011.

[32] M. Rigotti et al. Attractor concretion as a mechanism for the formation of context representations. *NeuroImage*, 52(3):833–847, 2010.

[33] F. Rosenblatt. The perceptron: A perceiving and recognizing automaton, project para. Technical Report 85-460-1, Cornell Aeronautical Laboratory, 1957.

[34] U. Seiffert. Artificial neural networks on massively parallel computer hardware. *Neurocomputing*, 57:135–150, 2004.

[35] W. Senn and S. Fusi. Learning only when necessary: Better memories of correlated patterns in networks with bounded synapses. *Neural Comput.*, 17(10):2106–2138, 2005.

[36] J. Seo et al. A 45nm cmos neuromorphic chip with a scalable architecture for learning in networks of spiking neurons. In *Custom Integrated Circuits Conference (CICC), 2011 IEEE*, pages 1 –4, sept. 2011.

[37] O. Temam. A defect-tolerant accelerator for emerging high-performance applications. In *ACM/IEEE International Symposium on Computer Architecture (ISCA)*, June 2012.