A SPIKING BIDIRECTIONAL ASSOCIATIVE MEMORY NEURAL NETWORK

MELISSA JOHNSON

Thesis submitted to the University of Ottawa
in partial fulfillment of the requirements for the
Doctorate in Experimental Psychology

School of Psychology
Faculty of Social Sciences
University of Ottawa

© Melissa Johnson, Ottawa, Canada, 2021

ABSTRACT

Spiking neural networks (SNNs) are a more biologically realistic model of the brain than traditional analog neural networks and therefore should be better for modelling certain functions of the human brain. This thesis uses the concept of deriving an SNN from an accepted non-spiking neural network via analysis and modifications of the transmission function. We investigate this process to determine if and how the modifications can be made to minimize loss of information during the transition from non-spiking to spiking while retaining positive features and functionality of the non-spiking network. By comparing combinations of spiking neuron models and networks against each other, we determined that replacing the transmission function with a neural model that is similar to it allows for the easiest method to create a spiking neural network that works comparatively well. This similarity between transmission function and neuron model allows for easier parameter selection which is a key component in getting a functioning SNN. The parameters all play different roles, but for the most part, parameters that speed up spiking, such as large resistance values or small rheobases generally help the accuracy of the network. But the network is still incomplete for a spiking neural network since this conversion is often only performed after learning has been completed in analog form.

The neuron model and subsequent network developed here are the initial steps in creating a bidirectional SNN that handles hetero-associative and auto-associative recall and can be switched easily between spiking and non-spiking with minimal to no loss of data. By tying everything to the transmission function, the non-spiking learning rule, which in our case uses the transmission function, and the neural model of the SNN, we are able to create a functioning SNN. Without this similarity, we find that creating SNN are much more complicated and require much more work in parameter optimization to achieve a functioning SNN.

PREFACE

This thesis is comprised of one manuscript (Chapters 2) plus an introduction (Chapter 1) and an overall discussion (Chapter 3). The manuscript (Chapter 2) is written in preparation for submission. The authorship of this manuscript is: Melissa Johnson and Sylvain Chartier.

ACKNOWLEDGEMENTS

First and foremost, I must give my thanks to my thesis supervisor, Dr. Sylvain Chartier, for all the support and encouragement he has provided me over the years. This thesis would not be possible without all the time and effort he has contributed. His feedback has been invaluable, and my work is so much better for it.

I must also thank my thesis committee, Dr. Denis Cousineau, Dr. Brent Doiron, Dr. Richard Naud, and Dr. Jean-Philippe Thivierge for their time, effort, and feedback. They will probably never realize how much I truly respect them and their work.

The friendships I made while attending university and working on this thesis are priceless, both in furthering my work, and, probably more importantly, keeping my sanity. While there have been too many wonderful people to list that I've met though my studies, a special thanks needs to go out to: Sara De La Salle and Anthony Murkar -- the very first two people I met at Ottawa and stayed friends with throughout our time there and hopefully beyond. And there's the amazing Laurence Morissette who has listened to me in good times and bad, always having a supportive word and commiserating shoulder. And thanks to Farooq Kamal for pushing me in new areas and being there as friend. To all my friends, co-workers, and colleagues, thank you.

Finally, I need to thank my fantastic parents, France and Gordon Johnson who gave me a love of learning and the support to keep going no matter how frustrated I got. And finally, my wonderful sister Alice Mather who willingly proofread so many of my earlier drafts for both papers and this thesis, despite having no knowledge of neural networks – although now she probably knows more than she ever wanted to.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABBREVIATIONS

ANN	Artificial neural network
ASN	Adaptive spiking neuron model
BAM	Bidirectional associative memory
CIF	Cubic integrate-and-fire neuron model
CNN	Convolutional neural networks
EIF	Exponential integrate-and-fire neuron model
FFDNN	Feed forward deep neural networks
IF	Integrate-and-fire neuron model
LIF	Leaky integrate-and-fire neuron model
LTD	Long-term depression
LTP	Long-term potentiation
NDRAM	Non-linear recurrent associative memory
NDRBAM	Non-linear recurrent bi-directional associative memory
NSNN	Non-spiking neural network
ODE	Ordinary differential equation
QIF	Quadratic integrate-and-fire neuron model
ReLU	Rectified linear unit
RNN	Recurrent neural network
SCNN	Spiking convolutional neural networks
SNN	Spiking neural network
SRM	Spike response model
STDP	Spike time dependent plasticity

CHAPTER 1: GENERAL INTRODUCTION

1.1 Introduction

The two main purposes behind artificial neural networks (ANNs) are: (1) to understand how the brain works and (2) to improve computer functionality. ANNs are modeled after the brain, and this modelling has been used to study how different cognitive and biological findings may occur via brain function. In regard to studying the brain, ANNs have been used to study such things as Alzheimer's (Świetlik & Białowąs, 2019), prosopagnosia (Morissette et al., 2012; Vandermeulen et al., 2011), and symbol grounding (Hermann et al., 2017; Kiela et al., 2017). In terms of computer advancement, neural networks have had great success in fields such as recognition (Chan et al., 2016; Coşkun et al., 2017; Lee et al., 2017; Li et al., 2019; Zheng et al., 2017) and prediction (Coley et al., 2019; Cui et al., 2018; El_Jerjawi & Abu-Naser, 2018; Guo et al., 2017; Sadek et al., 2019). But neural networks are still behind the human brain for problems involving reasoning, language, and common sense. In fact, neural networks are unable to learn basic behaviours of simple animals (Zador, 2019). It is hoped that the third generation of neural networks, spiking neural networks (SNNs), will offer more advancement in areas such as creating an artificial intelligent agent, computer functionality, and of course, understanding the brain.

Despite SNNs being theoretically computationally more powerful than other types of artificial neural networks, in practice they have not outperformed their counterparts. Presently, the networks most used in practice are second generation analog neural networks that have simplified the brain dynamics. In these networks, a single node can represent a group of neurons and the firing pattern is simplified to the average firing rate. But there is power in having multiple methods of encoding information -- including both rate based and temporal --

something that can be done with SNNs, and therefore making SNNs necessary (Ghosh-Dastidar & Adeli, 2009). There are biological findings that are only explained with spiking. For example, Maass (1997) argues that when fast computations are needed, such as visual pattern analysis and pattern classification performed by humans, the spiking implementation of the rate based interpretation of non-spiking neural networks (NSNN) is questionable; these networks would need too much time to produce a firing sample to make a decision. The problem with SNNs presently is that there are still many open questions in how they work, such as how learning is done and how spike times are used. While biological studies are occurring to understand the issues of how the brain learns and codes information, research on SNNs can also help in understanding how spikes, or action potentials, manifest intelligence.

This thesis looks at the requirements needed to create a functioning bidirectional spiking neural network. Bidirectionality involves a group of one or more neurons that send their action potentials to another group of one or more neurons, and this second group in turn sends their action potentials back to the first group. Bidirectionality has been found in the brain, for example Nägerl et al. (2004) found bidirectional activity between CA1 pyramidal neurons. Bidirectionality also has many uses in image processing and pattern recognition due to its ability to associate pairs of items together (Acevedo-Mosqueda et al., 2013). In this thesis, the creation of a bidirectional associative memory (BAM) spiking neural network is done via conversion of a non-spiking BAM, specifically the nonlinear dynamic recurrent bidirectional associative memory neural network (NDRBAM). Through preliminary studies, it was found that the transmission function of the NDRBAM can be converted into leaky integrate and fire neuron model similar to currently accepted neural models such as the quadratic integrate and fire model (QIF). Therefore,

the NDRBAM is considered a good network to use for converting a working analog neural network into a spiking neural network.

In Chapter 2, we extend previous work (Brea et al., 2013, 2013; Chartier & Boukadoum, 2011; Hunsberger & Eliasmith, 2016; Maass & Natschläger, 1998; Meunier & Paugam-Moisy, 2005, 2005; Wärnberg, 2014; Yu et al., 2014; Zambrano et al., 2017) by studying how to create a bidirectional SNN, specifically how a neuron model should be matched to the non-spiking counterpart's transmission function. The focus is on converting neural networks into spiking neural networks via studying neuron models that are both similar and dissimilar to the transmission function, and determining what their effects are on recall. This study is done by (1) creating a neuron model based on the transmission function; this allows the network to be used as both a spiking and non-spiking network without requiring any changes, (2) comparing different neuron models and their effects on the resultant spiking neural network, and (3) exploring parameter selection's overall influence in the resultant SNNs. This methodology is unique in two aspects: first it creates its own neuron model to aid in the study of the effects of replacing the transmission function with neuron models. And second, it has multiple comparisons between different neural models and network architectures to show just how the neuron model, and parameters, affect recall in different networks.

Chapter 1 concludes with some background information on neural networks, specifically SNNs and BAM. Chapter 2, containing a paper for publication, is then presented. Chapter 3 concludes with an overall discussion, including a section on future works.

1.2 Background

Neural networks have been around for years, and over that time they have undergone many changes. In fact, SNNs are considered the third generation of ANNs. This background

starts with a brief history of ANNs, which moves onto understanding bidirectional neural networks. Since the goal is to create a spiking bidirectional neural network, a basic overview of SNNs is also provided. There are many unknowns in SNNs, such as how learning actually occurs, which will also be touched upon in this section. Finally, there is also a quick review of previous research into spiking bidirectional associative memory.

1.2.1 Brief History of ANNs

The origin of ANNs is rooted in biology, specifically in how the brain functions. And while the application of ANNs has expanded greatly over the years, the core principle of being biologically inspired has never changed.

In the 1940s, McCulloch and Pitts (1943) published a landmark paper that heralded the start of the first generation of ANNs. These networks where characterized by their "all-or-none" neural activity; they either fired or did not, depending on if voltage passed a set threshold gate. The idea of these networks with nodes and threshold gates lead to a number of networks, most notably the Perceptron (Rosenblatt, 1958). These original ANNs allowed computers to perform task by using examples, something that had never been done before (Bottou, 2017). Unfortunately, for the first generation of ANNs, including the Perceptron, Minsky and Papert (1969) mathematically argued this area of research would not progress. There proof of limitations in the networks lead to a lull in research on ANNs for almost 20 years despite their analysis not necessarily reflecting the whole field of neural networks (Bottou, 2017; Jain et al., 1996; Olazaran, 1996).

A resurgence occurred in the 1980s, thanks to the development of better neural networks, such as that done by Hopfield (1982) and new learning algorithms such as back-propagation (McClelland et al., 1986; Rumelhart et al., 1986; Werbos, 1988). With this resurgence, came the

second generation of neural networks. These networks are no longer digital “all-or-none” networks but analog, allowing for a range of inputs and outputs. This second generation of neural networks were much more robust than the first generation, capable of solving a wider range of problems using fewer nodes (DasGupta & Schnitger, 1994). These networks also brought in the concept of activation functions which are abstractions of neural firing rate. There are many different second generation neural networks, including the non-linear dynamic recurrent associative memory network (NDRAM).

While research into, and usage of, second generation neural networks is still growing, a third generation is slowly growing as well: SNNs. In the second generation, a node can represent a group of neurons and neural firing is abstracted into a firing rate but in SNNs, a node represents a single neuron, and action potentials, or spikes, form the output. While this might initially seem to be going back to the first generation’s “all-or-none” firing, it is actually much more elaborate because these networks include other important aspects of neurons, such as how spikes might encode and decode data, firing times and rates, and other biologically inspired information (Johnson & Chartier, 2017). In fact, the first and second generations of neural networks are simplified models of SNNs. By removing some of the abstraction, SNNs are not only at least as powerful as previous generations, they can solve the same problems with fewer nodes. They are also computationally more powerful with regard to temporal coding (Maass, 1996, 1997b).

Unfortunately, while SNNs’ power has been shown mathematically, accessing the benefits for machine learning has been slow and there are still many questions about how it is done. This thesis aims to help develop new SNNs in hopes of eventually providing possible answers to some of those questions, or at least producing better SNNs by trying to derive a SNN

from a second generation ANN. Before studying SNNs though, we shall first review a second generation ANN, specifically the BAM.

1.2.2 Bidirectional Neural Networks.

There is evidence of bidirectionality from both marine species (Guan et al., 2002) and mammals (Malleret et al., 2010). There is also growing evidence of the reciprocal nature of neural assemblies for action words and related body parts (Kiefer & Pulvermüller, 2012; Pulvermüller, 2005; Pulvermüller, Hauk, et al., 2005; Pulvermüller, Shtyrov, et al., 2005). The architecture of BAM is bidirectional so that it handles paired-data associations; or hetero-associative memories. Input from one item in the pair affects connections to the nodes of the other item in the pair and vice versa. This reciprocal connection between nodes then allows for input from either part of the pair to activate the other pair for recall, as opposed to associative memory where recall is only in one direction. An example to explain hetero-associative memories is remembering a paper's title by the author, and being able to do the reverse -- remembering the author given the paper's title (Acevedo-Mosqueda et al., 2013; Chartier & Boukadoum, 2006; Kosko, 1988).

One of the first artificial bidirectional networks was created by Kosko (1988) in hopes of remedying some of the disadvantages of the Hopfield network (Acevedo-Mosqueda et al., 2013). To handle hetero-associative memories, Kosko used two Hopfield networks so that pairs of items can recall each other. BAM is considered to be more biologically accurate than Hopfield's network (Bhatia, 2016; Zhuang et al., 1993) because of this back and forth between the nodes. Kosko's BAM had issues though, such as low memory capacity, many spurious attractors, and susceptibility to noise.

Chartier and Boukadoum (2006) extended Kosko's method for bidirectional memory. First, while Kosko used two networks, there is only one weight matrix; the transpose of the one weight matrix was used to pass items back to the first network. Chartier and Boukadoum use two distinct weight matrices that allow for associations to be made between inputs of different sizes. Next, they used the NDRAM network (Chartier & Proulx, 2005) which supports larger memory load and is less susceptible to spurious attractors than the Hopfield network. These changes allowed this new BAM to keep the improvements of NDRAM regarding memory, spurious attractors, and noise, while also keeping the ability to handle hetero-associative memories, and in fact increased functionality by allowing pairs to be different sizes. This BAM neural network is referred to as the NDRBAM for non-linear dynamic recurrent bidirectional associative memory.

1.2.2.1 NDRBAM.

The NDRBAM is a bidirectional associative memory neural network that can recall hetero- and auto-associative patterns. The network was originally developed as an auto-associative recurrent network (Chartier & Proulx, 2005) that was then modified for bidirectional associative memory (Chartier & Boukadoum, 2006)

Neural networks are defined by three assumptions: architecture, transmission function, and learning rule. The architecture of a neural network is characterized by how many layers there are in a network, how many nodes there are in a layer, and how the nodes are connected. The transmission function is a mathematical equation that defines the nodes themselves and used in decoding information. In other words, the transmission function defines how information is processed. In some cases, such as NDRBAM, the transmission function is also a part of the learning rule and therefore necessary in determining how the weight matrices change and for

decoding input to retrieve the related pattern. The learning rule changes the weight matrices based on the information provided.

Architecture. The NDRBAM network architecture consists of two layers connected together such that each node in the first layer passes information to every node in the other layer, and vice versa. These layers have weight matrices, $\boldsymbol{W}$ and $\boldsymbol{V}$, that are used to pass results to the other layer (see Fig. 1).

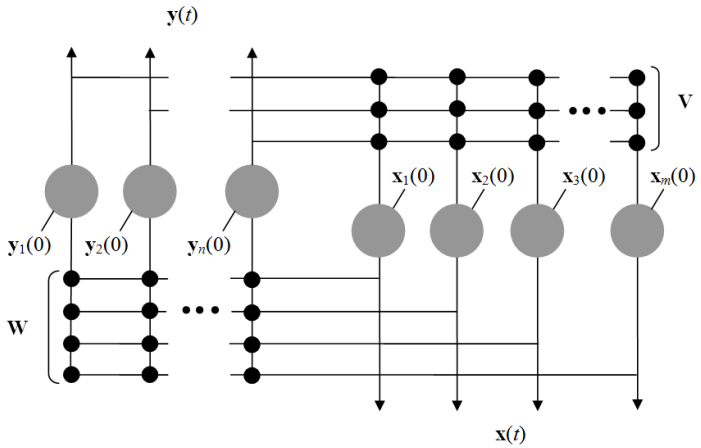

Figure 1: NDRBAM network architecture which consists of two fully connected layers such that all nodes in one layer are connected to all the nodes in the other layer.

Transmission Function. The transmission function is based on the classic Verhulst equation, extended to a cubic form with saturating limits at ±1 (Chartier et al., 2008). The transmission functions are defined by the following two equations:

$$\forall\, i = 1, \ldots, N, y_i(t+1) \quad = f\big(\boldsymbol{W_i x}(t)\big) \qquad (1)$$
$$= \begin{cases} 1 & if\ \boldsymbol{W_i x}(t) > 1 \\ -1 & if\ \boldsymbol{W_i x}(t) < -1 \\ (\delta + 1)\boldsymbol{W_i x}(t) - \delta \boldsymbol{W_i x}^3(\mathrm{t}) & else \end{cases}$$

$$\forall\, i = 1, \ldots, M, x_i(t+1) \quad = f\big(\boldsymbol{V_i y}(t)\big) \qquad (2)$$
$$= \begin{cases} 1 & if\ \boldsymbol{V_i y}(t) > 1 \\ -1 & if\ \boldsymbol{V_i y}(t) < -1 \\ (\delta+1)\boldsymbol{V_i y}(t) - \delta \boldsymbol{V_i y}^3(\mathrm{t}) & else \end{cases}$$

where N and M are the number of nodes in each layer and therefore the weight matrices, $\boldsymbol{W}$ and $\boldsymbol{V}$, are of size $N \times M$ and $M \times N$ respectively, and when indexed with i, that represents the specific row used in the calculations. This allows for associated patterns to be of different sizes. **x** and **y** are the nodes in the network such that at iteration t the layer contents are represented by $\boldsymbol{x}(t)$ and $\boldsymbol{y}(t)$. The parameter δ is a general transmission parameter set to a fixed value between 0 and 0.5 to assure fixed-point behaviour (Chartier et al., 2008). This transmission function is used because it has no asymptotic behaviour when $0 < \delta < 0.5$ and is therefore useable during learning and recall. The saturating limit of [-1, 1] at the two attractors allows it to be comparable to a sigmoid-type function.

Learning Rule. The learning rule is Hebbian/Anti-Hebbian as expressed by the following equations:

$$\boldsymbol{W}(k+1) = \boldsymbol{W}(k) + \eta(\boldsymbol{y}(0) - \boldsymbol{y}(t))(\boldsymbol{x}(0) + \boldsymbol{x}(t))^{\mathrm{T}} \qquad (3)$$

$$\boldsymbol{V}(k+1) = \boldsymbol{V}(k) + \eta(\boldsymbol{x}(0) - \boldsymbol{x}(t))(\boldsymbol{y}(0) + \boldsymbol{y}(t))^{\mathrm{T}} \qquad (4)$$

where η represents the learning parameters and k is the learning trial. The T is a standard symbol for the transpose operator.

The initial inputs are $\boldsymbol{x}(0)$ and $\boldsymbol{y}(0)$ while $\boldsymbol{x}(t)$ and $\boldsymbol{y}(t)$ are the state vectors after t iterations through the network in that particular learning trial, k. For auto-associative memory, where the recalled items is identical to the input item, $\boldsymbol{y}(0) = \boldsymbol{x}(0)$, the learning rule can be simplified to:

$$\boldsymbol{W}(k+1) = \boldsymbol{W}(k) + \eta(\boldsymbol{x}(0)\boldsymbol{x}^{\mathrm{T}}(0) - \boldsymbol{x}(t)\boldsymbol{x}^{\mathrm{T}}(t)) \quad (5)$$

Note for equation 5, for auto-associative memory, using equation 4 instead of 3 will produce the same results. In other words, in auto-associative memory, **W=V.**

Based on equations 3 and 4, the weights can only converge when $\boldsymbol{y}(t)=\boldsymbol{y}(0)$ and $\boldsymbol{x}(t)=\boldsymbol{x}(0)$. Therefore, the learning rule is linked to the network's output and a different transmission function will change the learning rule results. In order for the association to be stored as a fixed point, η must be set according to (Chartier et al., 2008):

$$\eta < \frac{1}{2(1-2\delta)Max[N,M]}; \quad \delta \neq \frac{1}{2} \quad (6)$$

While the NDRBAM is an excellent network, it is still a second generation analog network which means that its nodes are representative of a group of neurons in the brain, and that its transmission function translates into a mean firing rate. This NSNN is not sufficient to fully understand or mimic brain functions because it is missing an important piece of brain functions, spikes. A bidirectional SNN, which is biologically more realistic and can take advantage of spike times would be a better model (VanRullen et al., 2005).

1.2.3 Spiking Neural Networks

As mentioned previously, theoretically SNNs are computationally more powerful and biologically more realistic than previous generations of ANNs. Therefore, they should be able to better advance our understanding of the brain and how it works. For example, in an analog neural network each node represents groups of neurons, but studies using single cell recordings have

found the existence of neurons that respond to single objects (Kiefer & Pulvermüller, 2012). This specificity of how individual nodes act, and interact with each other, cannot be mimicked in second generation ANNs since nodes in ANNs, at best, groups of neurons. Unfortunately, as there is still much that is not understood about the brain, our ability to produce effective and efficient SNNs has been unable to match our success with second generation ANNs.

SNNs still have an architecture just like their non-spiking counterparts, but the transmission function is replaced with a neuron model and learning must be adjusted to handle spikes, or action potentials. While in the first generation, learning may not be directly tied with the transmission function, in the second generation and in SNNs, learning and transmission are related. In SNNs, the neuron model will affect learning because the neuron models control how input is integrated to produce action potentials.

1.2.4 Neuron Models.

One of the first neuron models created was introduced by Lapicque in 1907, back when neuroscience was still in its infancy (Abbott, 1999). The neuron model Lapicque introduced was the integrate-and-fire (IF) neuron model; a model still widely used today in varying forms. Since so little was known about neurons back then, the IF was modelled after an electrical circuit and it was assumed that such a neuron would have some kind of threshold to indicate when an action potential would occur. It was not until 1952 when Hodgkin and Huxley dissected a giant axon from a squid (loligo) that the effect of the ionic flow on the cell membrane was better understood (Hodgkin & Huxley, 1952a, 1952b, 1952c, 1952d; Hodgkin et al., 1952). These studies of the giant axon also lead to the biologically realistic Hodkgin-Huxley model (Hodgkin & Huxley, 1952d). A thorough explanation of the Hodgkin-Huxley neural model can be found in *Spike neural models Part I: The Hodgkin-Huxley model* (Johnson & Chartier, 2017).

While both the IF model and the Hodgkin-Huxley model attempt to explain and model neurons in the brain, they are also extremely different models, both in terms of complexity and biological realism. Over the years, many models have been developed that fall somewhere between these two models. IF models have been modified to produce leaky integrate-and-fire models since neurons have a "leak" component – without input, the neuron membrane potential will revert back to resting potential. Izhikevich's model allows for the wide variability in firing patterns of biological models via simple parameter changes (Izhikevich, 2003). In the spike response model (SRM), they analyzed and tried to approximate, the Hodgkin-Huxley model to develop a more realistic version of the IF model (Kistler et al., 1997).

Overall, there is a lot of variability in model complexity and realism; for further information on this see *Spike neural models part II: Abstract neural models* (Johnson & Chartier, 2017, 2018). But for all the variability in neurons, the overall goal of mimicking specific aspects of real neurons is the same, and this means there is also a lot of overlap in features and functions of the neuron models. In some cases, a set, or family, of models can be grouped together as a canonical model (Carandini & Heeger, 2012; Hoppensteadt & Izhikevich, 2002). A canonical model is like a template of a model: with just parameter adjustments, this template can be transformed into any model in its family. But minimizing models to a canonical model may result in the loss of distinct and interesting features necessary for the explanation of the results (Chirimuuta, 2014). Therefore, instead of reducing groups of models to a canonical model, it may be more useful to look at the different features the models have. Choosing models and their features is discussed in Izhikevich (2004). There are many different possible features to consider, including how long it would take to compute the chosen model, how it responds to increasing levels of input, and what spike patterns it can produce.

Every neuron model has advantages and disadvantages, so the goal is to figure out the best balancing point for these differences for the research. While being biologically accurate is important (Kaplan & Craver, 2011; Levy & Bechtel, 2013) abstraction is also a necessary tool for understanding the concept being modelled. For example, in a contest for models to accurately predict spike times, all the winning entries were simple abstractions of neurons (Gerstner & Naud, 2009). This example shows that despite abstractions of many general neuron features and properties, the specific features being studied can still be biologically realistic. Abstraction does not mean that the model contains falsehoods or inaccuracy, just that the specific details of the neuron are summarized if those details are not needed for the research (Gerstner & Naud, 2009; Levy & Bechtel, 2013). Abstraction also allows for easier analysis of the network (Gerstner et al., 2014).

There are two general types of excitable neuron models: Type I which are integrate models (see subsection for more information), and Type II which are resonant models. Both types of models work together with resonant neurons keeping the network stable while integrate neurons allow it to respond to external input (Muresan & Savin, 2007). As the main concern of this thesis is creating long term associated memories and this works with external input, using Type II neurons makes little sense, so models used are selected from Type I.

One of the simplest and best known models of neurons is the leaky integrate-and-fire (LIF) model, but this model does not always follow Type I neurons; therefore unexpected responses may occur with its use (Badel et al., 2008). Nonlinear integrate-and-fire models of Canonical Type I are generally preferred for modeling. The two most common nonlinear, Canonical Type I models are the quadratic integrate-and-fire (QIF) and the exponential integrate-

and-fire (EIF). The QIF is easy to analyze but the EIF is more biologically realistic in the way it handles action potentials (Badel et al., 2008).

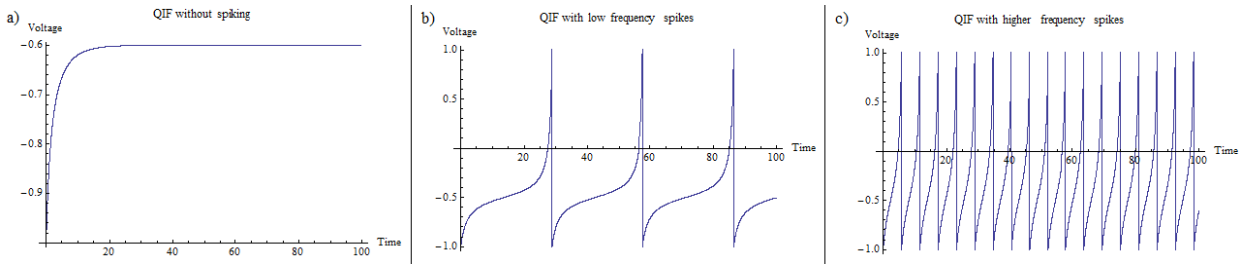

Figure 2: Example of the QIF model at different stable states. Resting value is -1, critical is 0 and α=1. Input is a constant value of 0.24 (left), 0.26 (middle), and 0.4 (right).

Type 1 Excitable Models. The main concern in Type I excitable models is how the firing pattern is related to the input. Canonical Type I excitable models are stable in both firing and non-firing states, and their firing frequency is related to the external input (see Fig. 2) (Ermentrout, 1996; Izhikevich, 2004). This type of model is understood fairly well (Hoppensteadt & Izhikevich, 2002).

Both QIF and EIF are Canonical Type I models because they are stable in both resting and tonic spiking modes. They also have spike latencies and activity dependent thresholds. Unfortunately, neither are biologically plausible because they are one dimensional and therefore cannot burst or exhibit spike frequency adaption without modifications. For more information on modeling with Type I excitable neurons, please see Hoppensteadt & Izhikevich (2002).

There are many reasons for different neuron models to be created, although generally it is to better understand a phenomenon. For example, Hodgkin-Huxley model was developed to understand how biological neurons work while Izhivkevich's model provides an easy way to study how firing patterns affect networks. The simple IF models are easy to use and analyze while still providing basic spiking ability. The next chapter, The Importance of Neural Model and

Parameter Selection, creates a new model for two reasons: to ease in the creation of a spiking NDRBAM and as part of an analysis of the importance of neuron model selection, and parameter selection, in creating spiking neural networks. The idea behind the creation of the new neuron model is twofold: (1) it allows us to create a network that can switch between spiking and non-spiking without changes, something that is unique to our network, and (2) it allows for a better comparison between neuron models and their effects on the network.

While there has been a lot of work in neural dynamics (Gerstner et al., 2014; Liljenström, 2015; Vogels et al., 2005), there has been little direct comparisons of results between different neural models and networks in comparison to transmission functions. There have been many studies that convert a non-spiking to a spiking neural network (Belatreche et al., 2006; He et al., 2019; Hunsberger & Eliasmith, 2015; Hunter et al., 2008; Maass & Natschläger, 1998; Meunier & Paugam-Moisy, 2005), and this is often done by replacing the transmission function with a neuron model, plus possibly other changes such as removing biases (Cao et al., 2015; Diehl et al., 2015; Folowosele et al., 2011; Rueckauer et al., 2016; Zambrano et al., 2017) or normalizing the weight matrices (Diehl et al., 2015; Rueckauer et al., 2016; Zambrano et al., 2017; Zambrano & Bohte, 2016). But we were unable to find any clear experimental study that shows how changing a neuron model affects the results in relation to both transmission function and original network.

The closest research to directly compare different neuron models is Zambrano et al. (2017). In this research, they compare previous state-of-the-art SNN performance with an SNN that uses the adaptive spiking neuron model (ASN). They found that their new SNN with ASN achieved the same accuracy as the analog counterpart, therefore performing better than previous

state-of-the-art SNNs. But ASN have additional features that the LIF neurons do not have, and it is those features that are the cause of the increased accuracy.

The features of neuron models also play a role in how those neurons work in the network. When Maass & Natschläger (1998) created a spiking Hopfield network, their theoretical work was based on simple IF neurons, but their simulation used more complex compartmental neurons. While they found that the network could perform associative recall, even with noisy versions of the stored pattern, the effect that these compartmental neurons had on recall is unknown. Their theoretical findings for a general IF neuron models found that delays needed to be precise, but this precision was not necessarily required for more advanced neuron models. Meanwhile, Wärnberg (2014) found that trying to duplicate Maass & Natschläger's (1998) work using an IF neuron model, it was difficult and often failed. This range of results and general difficulty in duplicating previous work leads to the conclusion that the neuron model, and properties of the model, have a large impact on the SNN. Some of these difficulties found, and the reasons for them, are discussed in the next chapter.

1.2.5 Coding.

Neuron models produce spikes - they take in input and produce output that is either spike or no spike. Within the brain, these spikes produce memory, meaning, thought. Neural coding, how neurons encode and decode data using spikes, is still an open question in neuroscience. Presently, the two most known and studied theories about coding are rate-based and temporal coding (Yu et al., 2018), although the true solution might be a combination of these two theories (Brette, 2015; Gerstner et al., 2014; Naud & Sprekeler, 2018; Stein et al., 2005; Yu et al., 2018).

1.2.5.1 Rate-based Coding

Rate-based is the traditional, and most common, method of coding (Brette, 2015; Gerstner et al., 2014), having been first report in 1926. By experimenting on a frog, Adrian & Zotterman (1926) found that end organs in the muscle send regular impulses and that the frequency of these impulses is dependent on the intensity of the stimulus. This is the basis of rate coding: the intensity of the stimulus is directly and positively related to the frequency of the action potentials. In general, the firing rate is a continuous function in time where the value returned at any point is the probability that a neuron will generate a spike at that time point (Bialek et al., 1991). Firing rate does not look at individual spikes or timing between spikes because it assumes that variations around the average occur because of noise and that noise has no use.

Rate-based coding uses the average (mean) firing rate to remove inter-spike interval variability, but there are multiple ways of calculating this average (Gerstner, 2002). The most general idea is that one takes the average number of spikes produced over a period of time (the time window), and this average is considered the rate. The actual calculation of this average can be done a few different ways, such as using the spike count which counts the number of spikes that occurred in a pre-defined period and divides that by the time period, or time dependent firing rate which uses repeated trials and averages the number of spikes that occur for each time period across those trials, producing the probability of a spike occurring at every point in time.

Rate-based coding is very robust to inter-spike interval noise because it averages out the spikes, regardless of timing between them (Stein et al., 2005; Yu et al., 2018). There have been studies that indicate that variations in timing are pure noise (London et al., 2010). Analog neural networks use rate-based coding. Specifically, in analog neural networks, the transmission

function essentially transmits the average firing rate. But, if the timing of each spike is important, temporal coding is needed.

1.2.5.2 Temporal Coding

There is evidence that indicates that timing of spikes is important (Caporale & Dan, 2008; Kasabov et al., 2012; Markram et al., 2011; Moller, 1999; Theunissen & Miller, 1995). Temporal encoding assumes that inter-spike variation carries information and therefore this variance cannot be discounted as just noise. Firing rate may still be important; for example, it may determine the rate of information being processed, but firing rate is not the content of the information (Brette, 2015). Considering the speed at which humans can respond to sensory input, timing, particularly first-spike time, could be very important (VanRullen et al., 2005).

The importance of the first-spike is just one type of temporal coding. Temporal coding uses features of spike trains that cannot be described with firing rate alone. This means that things like first-spike, inter-spike interval probability distribution, and precisely timed spikes are all considered part of temporal coding (Kostal et al., 2007).

1.2.5.3 Combination

While rate-based and temporal coding are the main two categories of coding, there is a growing body of evidence for using a combination of coding methodologies. Based on their research on decision-making tasks, Smith et al. (2019) conclude that it is possible that rate-coding is used to detect and signal possible conflicts while temporal coding stabilizes and amplifies task-relevant information. This idea of rate and temporal coding is further argued by Hurowitz (2020) based on their studies of value-based decision-making tasks with monkeys. Sanders et al. (2019) proposed that a spike train can simultaneously represent two independent

information streams. They argue that place cells in the hippocampus can use temporal order to express spatial relations and firing rate to represent non-spatial information.

While this thesis will not solve the coding problem, the problem impacts any SNN since the network must somehow process spike trains. As the spiking BAM becomes more biologically plausible and functional, different methods of coding can be studied.

1.2.6 Learning.

Regardless of coding method, there needs to be a method to allow for learning. The idea of plasticity between neurons, that connections between neurons can change, has been around for a long time. In the 1890s, Gotch and Horsley recorded the spinal cord and sciatic nerve in cats and monkeys while stimulating the cerebral cortex. Their recordings show initial facilitation followed by depression (Markram et al., 2011). Around this same time Cajal proposed that long-term memories could be produced via growth of new connections between existing neurons, rather than needing new neurons (Markram et al., 2011).

1.2.6.1 Hebbian/Anti-Hebbian Learning

The early 1900s produced many studies into neurons and how they work, such as those mentioned above for neuron models and coding. These historical works were eventually the basis for Hebb's seminal book *The Organization of behaviour: a neuropsychological theory* (Hebb, 1949). This book was influential in learning, producing the concept of Hebbian learning. The key component of Hebbian learning is:

> When an axon of cell A is near enough to excite a cell B and repeatedly or persistently takes part in firing it, some growth process or metabolic change takes place in one or both cells such that A's efficiency, as one of the cells firing B, is increased". (pg. 80)

The above statement was summarized as “cells that fire together wire together” by Shatz (1992) – although the colloquial expression is “what fires together, wires together” (Keysers & Gazzola, 2014).

Hebbian learning is an example of learning where weights are strictly increased or held steady (Gerstner et al., 2014). The general formulation that can generate a Hebbian learning rule is:

$$\frac{d}{dt}\boldsymbol{W}_{ij} = \boldsymbol{F}(\boldsymbol{W}_{ij}, \boldsymbol{v}_i, \boldsymbol{v}_j) \tag{7}$$

where $\frac{d}{dt}\boldsymbol{W}_{ij}$ is the change in synaptic strength between the presynaptic neuron $\boldsymbol{v}_i$ and the postsynaptic neuron $\boldsymbol{v}_j$. $\boldsymbol{F}$ is some function that uses the relationship between the pre- and postsynaptic neurons and their synaptic strength.

Learning rules also need some method to decrease synaptic strength otherwise synaptic strength would only ever be able to grow until all the connections become saturated at a maximum value (Gerstner & Kistler, 2002). Yet Hebb’s postulate only dealt with increasing synaptic strength not decreasing synaptic strength. Anti-Hebbian learning is the inverse of Hebbian learning, where correlated activation leads to a weakening of synaptic strength.

A simple example of Hebbian and anti-Hebbian learning is based off the Taylor series about $\boldsymbol{v}_i$=$\boldsymbol{v}_j$=0, expanding to get the bilinear term. Setting all constants to 0 except the bilinear term yields

$$\frac{d}{dt}\boldsymbol{W}_{ij} = c_{11}^{corr}\,\boldsymbol{v}_i\boldsymbol{v}_j \tag{8}$$

where, c_{11}^{corr} is a positive function (equation 9) whose value depends on $\boldsymbol{W}_{ij}$ and for anti-Hebbian learning c_{11}^{corr} is a similar negative function (equation 10). Synaptic weights fall between a range of 0 (not connected) and some maximum value w^{max}. Therefore, to bound the synaptic strength, we want the change to tend towards zero as the weight matrix approaches maximum for synaptic growth:

$$c_{11}^{corr} = y_2(w^{max} - \boldsymbol{W}_{ij})^{\beta} \tag{9}$$

where y_2 and β are positive constants; β is typically set to 1. For Anti-Hebbian learning, we can use:

$$c_{11}^{corr} = -y_0(\boldsymbol{W}_{ij})^{\beta} \tag{10}$$

where y_0 and β are positive constants; β is typically set to 1. The complete Hebbian/Anti-Hebbian learning rule, based on the Taylor Expansion is:

$$\frac{d}{dt}\boldsymbol{W}_{ij} = (y_2(w^{max} - \boldsymbol{W}_{ij})^{\beta} - y_0(\boldsymbol{W}_{ij})^{\beta})\boldsymbol{v}_i\boldsymbol{v}_j \tag{11}$$

The above is just one form of Hebbian learning. In this form, input to be learned must be orthogonal and the learning rule works best for a competitive network where a pattern is represented by a single node. There are many different rules in existence to handle different situations. The main requirement is that firing rates are correlated so that pairs of neurons that fire close together in time have their synaptic strength changed.

Hebbian/Anti-Hebbian learning rules are often used in neural networks. As mentioned above in the section about NDRBAM, the learning rule there is Hebbian/Anti-Hebbian. In

Legenstein et al. (2010), they implemented a Hebbian-based learning rule that simulates experimental results for selective learning within a population of cortical neurons. This learning rule used neuronal noise for parameter adaption, which means that, unlike the rule in NDRBAM which is rate based, theirs is temporal.

1.2.6.2 Spike Time Dependent Plasticity Learning

With Hebb's postulate, the idea of looking at temporal order instead of just frequency took research in a whole new direction. In the years following Hebb's book, numerous studies emerged testing the idea of long-term plasticity via causation firing. Still, it took nearly 20 years before there were experimental results supporting Hebb's postulate. The first true support came from Andersen's laboratory (Bliss & Gardner-Medwin, 1973; Bliss & Lømo, 1973; Markram et al., 2011) which showed that synaptic strength increases could last longer than previously shown, thus showing long term potentiation (LTP), as predicted by Hebb. Bliss et al. (1973) used tetanic simulation, spike trains with 15 spikes per second, for 15-20 seconds, to produce the LTP.

The repetitive clause of the postulate was not proven until 1975 when Douglas and Goddard (1975) showed that repetitive bursts were more efficient than the one long spike train that researcher previously used. By using kindling trains once a day for 5 days, they were able to discover that LTP lasted at least 12-days, and in some cases, over 3 months. They also found that each day, the strength of the potentiation increased until a maximum was reached near the end of the trials. Therefore, repetition and consistent bursts effectively produced LTP.

The reverse of LTP, long term depression (LTD) occurs when the synaptic strength between neurons weakens. LTD was first observed by Lynch et al. (1977) while trying to examine postsynaptic consequences of repetitive stimulation. They found that while potentiation occurred with pathways that they trained, or activated, inactive pathways underwent depression.

Early results for LTD were inconsistent though, with some labs finding LTD and some labs being unable to produce it (Bear, 1999). Eventually a way to reliably produce LTD was discovered, and is still used today, but this protocol is not biologically plausible (Markram et al., 2011). One of the main problems is a lack of understanding of the purpose of long-term depression; it is known to be a mechanism for memory, but exactly why is unknown (Bear, 1999).

As studies revealed the occurrence of LTP and LTD, researchers started to look closer at the timing of spikes and the effects of timing on potentiation and depression. Studies have found a clear temporal asymmetry in hippocampal plasticity for LTP and LTD (Fig. 3) (Bi & Poo, 1998; Zhang et al., 1998). Specifically, a synaptic input is potentiated if a presynaptic action potential precedes the postsynaptic action potential by at most 20 ms. A synaptic input is depressed if a presynaptic action potential follows a postsynaptic action potential by at most 20 ms. The asymmetry occurs in the degree of potentiation versus the degree of depression that occurs at a given time difference. The closer the timing between pre- and postsynaptic action potentials, the stronger the effect of depression or potentiation, but potentiation has much stronger effects at smaller time differences compared to the effects of depression for the same time difference. This asymmetric LTP and LTD can be implemented in the learning rule for SNNs.

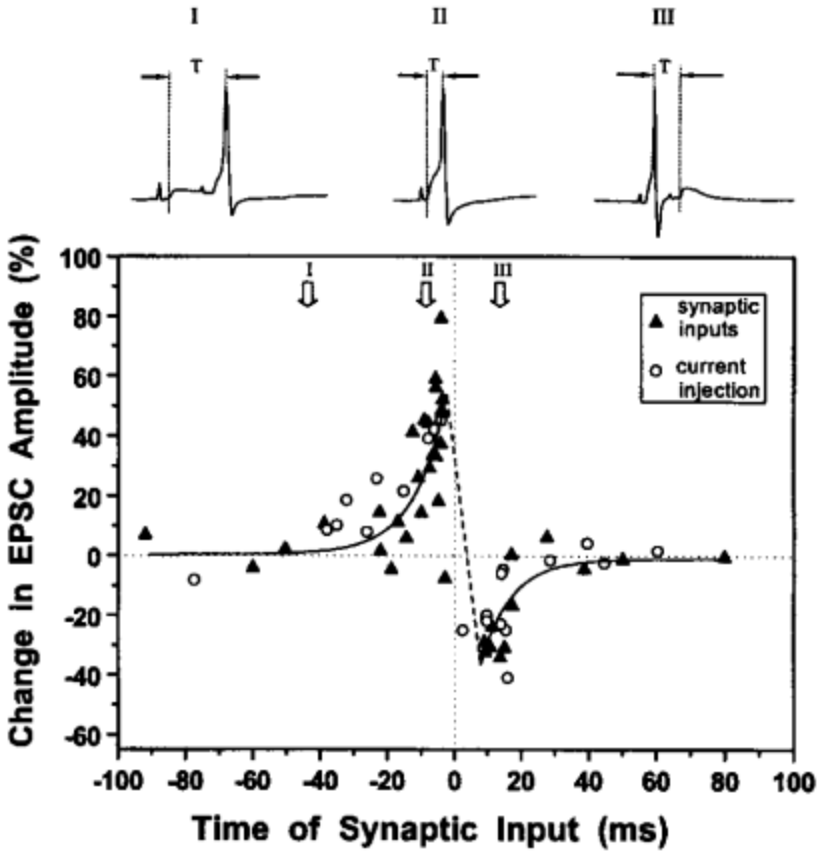

Figure 3: Critical window for synaptic depression and potentiation. Figure retrieved from Zhang et al. (1998)

Most models of STDP update using a pair-based update rule (Gerstner et al., 2014). This means that the change in synapse is based on the temporal difference of firing time: if the presynaptic spike occurs before the postsynaptic spike then the synapse strength increases, and this increase is larger the closer in time those spikes are. Conversely, if the postsynaptic spike fires before the presynaptic spike, then the synapse strength weakens, and strength of weakening is again related to how close together the firing times are between the pre- and postsynaptic neurons. A basic form of pair-based STDP equation is:

$$\begin{aligned} \Delta \boldsymbol{W} &= \boldsymbol{F}_{+}(\boldsymbol{W}) e^{\frac{-|\Delta t|}{\tau_{+}}} \quad at\ t_{post} \quad for\ t_{pre} < t_{post} \\ \Delta \boldsymbol{W} &= \boldsymbol{F}_{-}(\boldsymbol{W}) e^{\frac{-|\Delta t|}{\tau_{-}}} \quad at\ t_{pre} \quad for\ t_{pre} > t_{post} \end{aligned} \tag{12}$$

where Δt is the difference in timing between the pre- and postsynaptic spikes, $\tau_{\pm}$ are time constants and $\boldsymbol{F}_{\pm}(\boldsymbol{W})$ is a function used to adjust synaptic weights, dependent on current synaptic weights, preferably in a logarithmic manner. For $\boldsymbol{F}_{+}(\boldsymbol{W})$, this change would be positive while $\boldsymbol{F}_{-}(\boldsymbol{W})$ would have a negative change. Weight updates occur only after both pre- and postsynaptic neurons spike.

While pair-based spiking is commonly used, it is not necessarily biologically accurate because it does not replicate synaptic changes in biology. LTP is based on spike time, firing rate, and cooperation among neurons (Sjöström et al., 2001), not timing alone. Most studies look at many presynaptic fibers to produce LTP, possibly because one neuron is too weak to cause synaptic activation, therefore pair-based changes may miss necessary cooperation between fibers to produce synaptic changes.

1.2.7 Spiking BAMs

Despite there being a lot of studies showing bidirectionality in the brain (Dreyer & Pulvermüller, 2018; Guan et al., 2002; Kiefer & Pulvermüller, 2012; Malleret et al., 2010; Pulvermüller, Hauk, et al., 2005), within neural networks, there is still very little research on spiking bidirectional neural networks. Some work has been done via the lens of biological processes such as vision (Wysoski et al., 2010) or via engineering such as the memristive systems (Diederich et al., 2018) but compared to other networks, such as spiking convolutional neural networks, there is still a lot to be done.

In terms of creating SNN-BAMs, a first step can be to build on the analog bidirectional neural networks. It has been found that for deep SNNs, networks that are converted from their

analog counterparts achieve higher accuracy on MNIST data compared to deep SNNs that are trained directly (Taherkhani et al., 2020). Therefore, we know that the learned information can be accessed despite network changes. Unfortunately, the work on converting BAMS to SNN-BAMs has been minimal.

In Meunier & Paugam-Moisy (2005), they emulated a BAM with an SNN and showed how this emulation can contribute to our understanding of the brain, specifically how the SNN-BAM can model intermodal priming. To mimic the cognitive effect being studied, their network is composed of two lower layers, representing visual and auditory inputs, and an upper layer which ultimately produces the output. The upper and lower layers are fully connected between each other but there are no connections between the two lower layers themselves. Learning was done off-line with the original non-spiking BAM. They used the SRM to replace the nodes in their SNN. They do not compare how their network does against other spiking or non-spiking networks, but instead focus on how such an SNN can help expand our understanding of the brain and psychology.

A very different approach was used in Zamani et al. (2010) to create a spiking BAM. Training in this spiking version of BAM was done by a genetic algorithm with co-evolutional technique. The spiking neurons used were a set of heterogeneous neurons of different types and classes, all based on Izhikevich's cortical spiking neuron model. Patterns presented to the network were translated into a set of spikes with pre-defined timings. Two populations were evolved simultaneously but separately and integrated after training to form the spiking BAM. Their results showed that SNNs can have better recall and storage capacity than the related NSNN, but their methods are not biologically feasible, and recall is very separate from training.

The next chapter extends work in SNN-BAMs (Meunier & Paugam-Moisy, 2005; Zamani et al., 2010) by creating a spiking bidirectional neural network that can switch back and forth between spiking and non-spiking. This work also focuses on how the network uses the spiking component to recall information, and how parameters affect recall; both of which are the focus of this thesis.

CHAPTER 2: THE IMPORTANCE OF NEURAL MODEL AND PARAMETER SELECTION IN CREATING SPIKING NEURAL MODELS

Melissa G. Johnson[1], **Sylvain Chartier**[1]

[1] University of Ottawa, Ottawa ON, NJ K1N 6N5, Canada

2.1 Abstract

Symbol grounding is a natural phenomenon of human cognition that allows one to learn and associate symbols with their meanings. Spiking neural networks (SNNs) should naturally ground symbols, but our knowledge of how SNNs work is still in its infancy. This paper studies the effects of neural model and parameter selection on SNNs using the unique perspective of comparing similarities and differences. Three networks -- Hopfield, non-linear dynamic recurrent associative memory (NDRAM), and non-linear dynamic recurrent bidirectional associative memory (NDRBAM) -- are combined with three neural models -- leaky integrate-and-fire (LIF), quadratic integrate-and-fire (QIF), and a new cubic integrate-and-fire (CIF) -- to create 9 SNNs that are compared. There are minimal differences in results between neural models when combined with the Hopfield network although those spiking Hopfield networks generally perform poorly. Parameter selection may improve performance but finding optimal parameters can be difficult. For NDRAM and NDRBAM, CIF and QIF have the best performance results and parameter selection is straight forward to achieve acceptable results. In general, it is found that networks work best if the neural model used is similar to the original transmission function, such as is the case for QIF and CIF and the NDRAM and NDRBAM transmission functions. This similarity between transmission function and neural model allows for better results and makes good parameter selection easier. When the neural network's transmission function and neural model do not match; good parameter selection may still allow the network to achieve

acceptable results but there is a smaller set of parameters that work well, so finding them can be difficult.

Key words: spiking, recall, Hopfield, bidirectional, neural models

2.2 Introduction

The idea behind grounding in cognition is that we learn symbols and their meanings based on our interactions with, and observations of, the world. In other words, we are not born with a collection of cells that know what a "grandmother" is. Instead, our brain learns the symbol "grandmother" and what it means from our interactions with our grandmothers and from observing the interactions between other people and "grandmothers" (Barsalou, 1999, 2008). Our perceptions, actions, emotions, and interactions with the real world are all used to help provide meaning to symbols (Binder et al., 2016; Mei et al., 2016), and this is done by neurons in the brain interacting with each other. Most of the research into grounding uses non-spiking neural networks (NSNN), such as Mei et al. (2016) and Malinowski et al. (2015) but NSNNs are not biologically realistic due to the fact that nodes can represent groups of neurons with a variety of firing rates. NSNNs are also rate based which is an issue for biological realism. Spiking neural networks (SNNs), are more biologically realistic and should be naturally grounded (Pulvermüller, 2018).

SNNs, despite theoretically being computationally more powerful than NSNNs (Maass, 1996) are still not as widely used in real world applications as NSNNs. This may be because, at present, our understanding of spikes, specifically of learning, is still in its infancy. A current trend around this problem has been to convert a NSNN into an SNN. Learning is completed by the NSNN and the SNN is used for recall or classification (He et al., 2019; Maass & Natschläger, 1997; Meunier & Paugam-Moisy, 2005; Rueckauer et al., 2016). This methodology implies that information learned while the network was a NSNN is accessible to the SNN despite the change in how information is decoded (transmission function to neural model). But how, or why, this change allows data that is encoded one way be decoded another way is never discussed.

Artificial neural networks (ANNs) are described by three assumptions: architecture, learning rule, and transmission function. When converting NSNNs to SNNs, the architecture is generally not changed. Learning, done by the NSNN, produces a weight matrix which is then used by the SNN. The transmission function, generally part of the NSNN node, is replaced with a neural model. But the transmission function is a critical component of ANNs, so changing it to a neural model is a significant change yet there is not much research into the effects that occur due to this change. These effects can be compounded for NSNNs that use the transmission function in their learning algorithm because the transmission function is a component in the encoding-decoding relationship which is modified when switching to an SNN.

When selecting neural models, researchers have minimized the differences between transmission function and neural model but they appear to do so without ever studying the effects of said conversion. For conversion of convolutional neural networks (CNNs) to spiking convolutional neural networks (SCNN), researchers often replace a rectified linear unit (ReLU) function with an integrate-and-fire (IF) or leaky integrate-and-fire (LIF) neural unit since the ReLU function and the IF neuron integrate input in a similar fashion. If encoding and decoding have matching dynamics, and there are no changes in network architecture, grounded information should translate to the SNN. But for networks that do not use ReLU functions, such as recurrent neural networks (RNNs) for associative memory like the Hopfield ANN and the nonlinear dynamic recurrent bidirectional associative memory ANN (NDRBAM) this replacement may not be straight forward.

This paper investigates the effects of matching the neural model with the original transmission function when converting a NSNN to an SNN by comparing the performance of the Hopfield network and two versions of NSNNs that use the cubic transmission function, a

feedforward associative memory network called non-linear dynamic recurrent associative memory (NDRAM) and its bidirectional equivalent (NDRBAM). This combination allows us to compare networks with the same architecture (Hopfield and NDRAM) or the same transmission functions (NDRAM and NDRBAM). In addition to the three networks, we use three different neural models so that we can develop a clearer understanding of the scope that this conversion has on the grounding problem. Specifically, we created a new neural model, the cubic integrate-and-fire (CIF), which matches the transmission of the NDRAM/NDRBAM. Two other neural models are also used, one that is similar to the CIF, the quadratic integrate-and-fire (QIF), and one that is dissimilar, the LIF. These three models are all simple leaky-integrate-and-fire type of neurons to avoid the compounding factors that additional neuron model features will have on the networks. Additional features make it harder to compare the different networks because we would have to parse out the effects of those features. For example, adding a simple delay would add the additional difficulty of delay effects which is affected by the length of the delays. Keeping the networks and neural models simple allow us to produce a full set of meaningful comparisons to determine how SNNs are affected by neural models, and their parameters.

For our complete comparisons, we have similar/dissimilar in neural models where QIF and CIF are similar and LIF is dissimilar, similar/dissimilar in architecture where Hopfield and NDRAM are similar and NDRBAM is dissimilar, and similar/dissimilar in original transmission function where NDRAM and NDRBAM and similar and Hopfield is dissimilar. This multiple configurations of similarity and dissimilarity allows us to determine if the effects on recall are because of properties of the neural model or properties of the ANN. By changing just the transmission function of the network while leaving everything else alone, we are isolating the role of the transmission function and neural model in the problem of grounding. This type of

thorough comparison between SNNs expands our knowledge in how to effectively create SNNs and how to select parameters to improve overall performance. The introduction of the new neural model CIF allows for more in-depth comparisons and also allows for the creation of a network that can be switched back and forth between spiking and non-spiking with minimal changes.

This paper is organized as follows: section 2.3 contains previous research into converting NSNNs into SNNs. Section 2.4 contains information about CIF model based on the cubic transmission function of the NDRBAM. Three simulations are performed to delve into the issues of converting NSNNs to SNNs and how converting the NSNN affects grounding. The simulation in section 2.5 shows how neural models affect auto-associative recall. The next simulation, described in section 2.6, confirms that the SNNs still work for hetero-associative recall. Section 2.7 has the final simulation which shows how parameter selection affects recall. Finally, we discuss future work and conclude our findings in the section 2.8.

2.3 Background

It is generally agreed that the learning algorithm needs to be compatible with the encoding/decoding strategy (Brea et al., 2013; Lengyel et al., 2005; Taherkhani et al., 2020), but there is no single precise mechanism for linking these items in SNNs. In fact, learning in general is one of many open areas of research in SNNs. Unlike SNNs though, we have a good understanding of how NSNNs work, and have successfully used NSNNs for a variety of tasks such as: identifying diseases like Alzheimer's (Świetlik & Białowąs, 2019) and breast cancer (Sepandi et al., 2018); modelling different disorders such as prosopagnosia (Morissette et al., 2012; Vandermeulen et al., 2011); modelling human behaviours such as language acquisition (Poveda & Vellido, 2006); many classification tasks such as sound classification (Emmerson et al., 1991; Wu et al., 2018) and facial recognition (Er et al., 2002; Khashman, 2009; Khorasani et

al., 1994; Ma & Khorasani, 2004); and NSNNs are used throughout robotics (Ding & Chan, 1996; Ishiguro et al., 1992; Kawato et al., 1988; Mao & Hsia, 1997; Sasaki et al., 2016).

Because learning in SNNs is a problem, therefore many strategies for creating SNNs involve converting a NSNN into an SNN, leaving the learning (or training) to occur with the NSNN. It has been found that for deep SNNs, networks that are converted from their NSNN counterparts achieve higher accuracy on MNIST data compared to deep SNNs that are trained directly (Taherkhani et al., 2020). Therefore, we know that the learned information can be accessed despite network changes. The issue of grounding is more nuanced though: exactly what changes are affecting the information and how?

Research into replacing the transmission function with a spiking neural model in CNNs and feed forward deep neural networks (FFDNNs) have produced a fair number of results for classification tasks (Cao et al., 2015; Diehl et al., 2015; He et al., 2019; Hunsberger & Eliasmith, 2015; Zambrano & Bohte, 2016). In these studies, the CNNs use ReLU which gets replaced with IF neurons (Cao et al., 2015; Diehl et al., 2015), LIF neurons (Hunsberger & Eliasmith, 2015; Zambrano & Bohte, 2016), and in some cases the spike response model (SRM) (Zambrano et al., 2017) since these neurons are a good fit in dynamics for replacing the ReLU (Hunsberger & Eliasmith, 2015). Often the SCNNs have a slight degradation in performance over the optimized CNN or FFDNN, although when Zambrano & Bohte (2016) used an adaptive neural model, their networks performed identically to their NSNN counterpart.

The conversion of CNN to SCNN often have changes other than just replacing the transmission function, such as removing biases (Cao et al., 2015; Diehl et al., 2015; Folowosele et al., 2011; Rueckauer et al., 2016; Zambrano et al., 2017) or normalizing the weight matrices (Diehl et al., 2015; Rueckauer et al., 2016; Zambrano et al., 2017; Zambrano & Bohte, 2016).

One of the few papers that do not make any such changes as part of the conversion is Hunsberger & Eliasmith (2016). Like other research, they use ReLU in the network, change max pooling to average pooling, and remove local response normalization layers of the network, but these changes are done to the base CNN before training. The only difference between their base (non-spiking) network and the SCNN is that they replaced the ReLU with LIF neurons. They studied how training the CNN with noise can improve the performance of the related SCNN. The overall message from all these studies is that for classification tasks using FFDNNs or CNNs, the transmission function can be replaced with a spiking neuron model and the network will still produce acceptable results.

However, the immediate effect of changing that neuron model is never studied. The closest research to studying different neuron models is Zambrano et al. (2017). In their research, they compare the previous state-of-the-art SNNs performance with an SNN that uses the adaptive spiking neuron model (ASN). They found that their new SNN with ASN achieved the same accuracy as the NSNN counterpart, therefore performing better than the previous state-of-the-art SNN. But ASN have additional features that LIF models do not have, such as adaption, and it is those features that cause the increased accuracy. We have not found any study, until this one, about how, or if, the dynamics of the neuron model affect decoding data/data retrieval.

While there has been a lot of work done in classification and converting CNNs and FFDNNs to SNNs, the research in converting associative memory and RNNs has been lacking and less unified. One of the earlier examples of converting a NSNN into an SNN was done by Maass & Natschläger (1998). They started with a Hopfield network to determine the weight matrix which they scaled and set negative weights as inhibitory synapses. They replaced their nodes with a complex neuron model; each node was turned into a compartmental neuron with

122 compartments. There were also 3 synapses between each pair of neurons and these synapses each had their own delays. Input patterns had some items flipped from 1 to -1 and vice versa as part of their noisy simulation. They found that the network could perform associative recall, even with these noisy versions of the stored pattern. It is unknown, though, what effect the complexity of the neuron model had on recall. Their theoretical findings for a general IF neuron model found that delays needed to be precise, but this precision is not necessarily required for more advanced neuron models. While Maass & Natschläger (1998) indicated that their network simulation works, it is unclear precisely how accurate it was. Meanwhile, Wärnberg (2014) found that duplicating this work with different parameters and an IF model was difficult and often failed. This range of results, and general difficulty in duplicating previous work, leads to the conclusion that the neuron model, and properties of the model, have a large impact on the performance of the SNN.

Another common approach to creating SNNs is to create a new network with both learning and recall such as what Brea et al. (2013) did. They proposed a generic learning rule that uses spiking and can be matched to dynamics of a wide range of neuron models. Therefore, by making modifications of the learning rule dependent on the chosen neuron model, one can keep the dynamics of learning matched to that of recall. But while their goal was a generic learning rule, it was only tested with the SRM. They did test their learning rule with a variety of parameters though, to show how those parameters affect recall.

With associative memory, there is auto-associative, such as the Hopfield network, and there is hetero-associative memory. Specifically, bidirectional associative memory neural networks (BAMs) can perform both auto- and hetero-associative tasks. BAMs are more biologically realistic than feedforward recurrent neural networks. There have been some initial

steps in creating BAM-SNNs such as by Meunier & Paugam-Moisy (2005) and Zamani et al. (2010).

In Meunier & Paugam-Moisy (2005), they converted a BAM to an SNN and showed how the SNN can contribute to our understanding of the brain; specifically how the SNN-BAM can model intermodal priming. To mimic the cognitive effect being studied, their network is composed of two lower layers, representing visual and auditory inputs, and an upper layer which produces the output. The upper and lower layers are fully connected between each other but there are no connections between the two lower layers themselves. Learning is done off-line with the original BAM NSNN. They then use the SRM to replace the nodes in their SNN. They do not compare how their network does against other spiking or non-spiking networks, but instead focus on how such an SNN can help expand our understanding of the brain and psychology.

A very different approach is used in Zamani et al. (2010) to create a spiking BAM. Training in this spiking version of BAM is done by a genetic algorithm with co-evolutional technique. The spiking neurons used are a set of heterogeneous neurons of different types and classes, all based on Izhikevich's cortical spiking neuron model. Patterns presented to the network are translated into a set of spikes with pre-defined timings. Two populations are evolved simultaneously but separately. These two populations are integrated after training to form the spiking BAM. Their results show that SNNs are more powerful than the related NSNN, but their methods are not biologically feasible, and recall is very separate from training. It is also a completely unique network with no NSNN counterpart.

No matter the network, grounding is a function of how the information is learned, and that information can only be accessed if the decoding takes learning into consideration. This tying of encoding and decoding is done for NSNNs, and by keeping the neuron models used in

SCNN similar to CNN nodes, it has been understood within the SCNN field. This paper extends previous work (Brea et al., 2013; Hunsberger & Eliasmith, 2016; Maass & Natschläger, 1998; Meunier & Paugam-Moisy, 2005; Wärnberg, 2014; Zambrano et al., 2017) by researching associative memory tasks with SNNs and comparing how different neuron models affect recall with the focus on the relationship between the neuron model and the original transmission function. An added benefit of this work is creating a network that can be switched from non-spiking to spiking without having to make any changes since the neuron model is tied to the transmission function. Parameter selection is also investigated to determine how the parameters of the neuron model affect performance; this investigation is done in a detail we have not found in previous research. Through minimizing the effects of changing the transmission function to a neuron model, we argue that the information is grounded; the encoding and decoding of learned material is transferred between both spiking and non-spiking networks.

2.4 Proposal

We propose creating a neuron model that is directly derived from the cubic transmission function of the NDRAM/NDRBAM. This approach allows the model and transmission function to be as similar as possible and has the added benefit of creating a network that functions as both a NSNN and an SNN.

2.4.1 Cubic Integrate-and-Fire (CIF)

NDRBAM is a hetero-associative memory network, but when used with auto-associative tasks, it becomes the nonlinear dynamic recurrent neural associative memory network (NDRAM) (Chartier & Boukadoum, 2006). NDRBAMs (and NDRAMs) are capable of learning bi-polar and non-bipolar patterns, have a higher memory capacity, and develop less spurious attractors than the Hopfield network (Chartier & Proulx, 2005). The NDRAM/NDRBAM transmission

function, seen in equation 13, is a cubic function based off the Verhulst equation. The transmission function has 3 fixed points, z_1, z_2, and z_3, where z_1 and z_3 are stable fixed points while z_2 is an unstable fixed point. The parameter a controls the local minimum/maximum, which subsequentially controls the slope of the curve. In order for the 3 fixed points to be stable/unstable as mentioned above, a needs to be negative. In NDRAM/NDRBAM, these parameters are set such that: z_1=-1, z_2=0, z_3=1, and a=-1.

$$\frac{d\boldsymbol{x}}{dt} = f(\boldsymbol{x}) = a(\boldsymbol{x} - z_1)(\boldsymbol{x} - z_2)(\boldsymbol{x} - z_3) \tag{13}$$

This transmission function can be modified to create a nonlinear neural model, the CIF, which is shown in equation 14.

	$if\ \frac{d\boldsymbol{v}}{dt} = f(\boldsymbol{v}) > u_{threshold}, \frac{d\boldsymbol{v}}{dt} = f(\boldsymbol{v}) = u_{reset}$ $else\ \frac{d\boldsymbol{v}}{dt} = f(\boldsymbol{v}) = \alpha(\boldsymbol{v} - u_{rest})(\boldsymbol{v} - u_{critical})(\boldsymbol{v} - u_{max}) + R\boldsymbol{I}(t)$	(14)

The parameter a is replaced by α and controls the rheobase (minimum current amplitude needed to produce an action potential). Both $u_{threshold}$ and u_{reset} are spiking specific parameters, such that $u_{threshold}$ is the maximum voltage researched before the voltage is reset to u_{reset}. The value of $u_{threshold}$ needs to be between $u_{critical}$ and u_{max} to guarantee that it is reached. u_{rest} is the membrane resting potential and $u_{critical}$ is the voltage that the membrane potential needs to achieve to guarantee an action potential. The third fixed point, u_{max} controls where along the x-axis the rheobase is; adjusting the location of u_{max} adjusts how the voltage changes as inputs are

integrated. An exceptionally large u_{max} essentially translates the CIF into a QIF by putting the rheobase halfway between u_{rest} and $u_{critical}$.

Input is a function of time represented by $\boldsymbol{I}$. The equation for $\boldsymbol{I}$ is shown in equation 15. Input is multiplied by resistance, a constant represented by R.

$$\boldsymbol{I}_{i,t} = \sum \boldsymbol{W}_{j,i}\boldsymbol{S}_{j,t} + \boldsymbol{p}_{i,t} \tag{15}$$

Input uses the weight matrix, $\boldsymbol{W}$, that is produced during learning. $\boldsymbol{W_{j,i}}$ is the weight between the j^{th} neuron to the i^{th} neuron. An external pattern presented to the network is represented with $\boldsymbol{p}$ while internal spiking, is the node spiking or not, is represented by $\boldsymbol{S}$. The time step is t. Both internal and external spiking are functions of time.

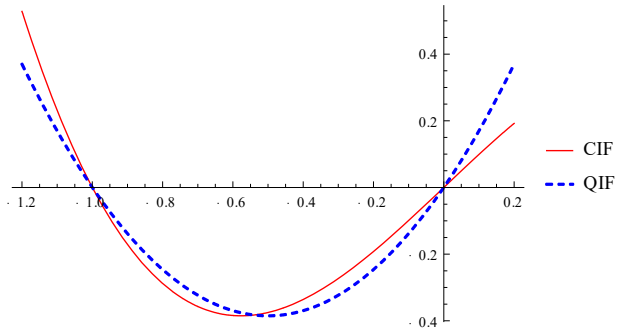

Figure 4: Comparison figure showing the curves of CIF and QIF with resting potential set to -1, critical value set to 0, and rheobase set to $\frac{2}{3\sqrt{3}}$.

The CIF is similar to the QIF as shown in Fig. 4. The LIF, QIF, and CIF are all leaky integrate-and-fire neural models, meaning that they integrate input, but without any input their voltage will slowly revert to, and stay at, the resting voltage. These three models are also Class I neural models (see Izhikevich (2004) for more information). All Class I neural models have a

stable resting state where they do not fire and a stable firing state where they continuously fire. In other words, below a certain threshold, constant input into a Class I neural model does not produce any action potentials, instead it produces a steady membrane potential, this membrane potential is positively related to the input. Once input passes the rheobase, Class I neurons continually fire, and the rate of firing is positively related to the input voltage such that the higher the input, the faster the neuron fires. These stability properties are shown in Fig. 5 and Fig. 6.

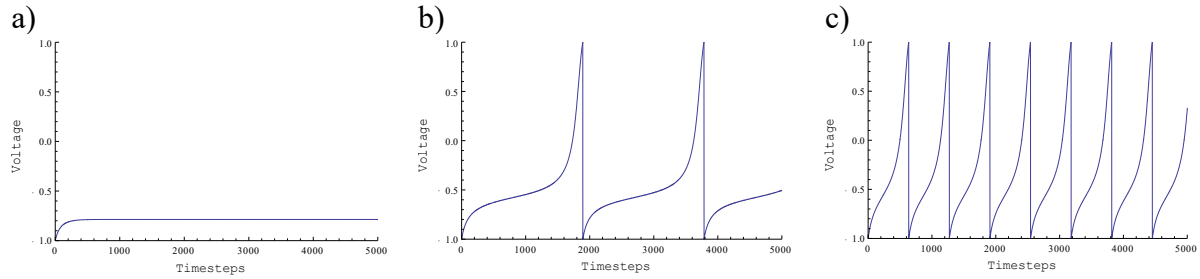

Figure 5: Canonical properties of CIF. Parameters of CIF are: u_{rest}=-1, $u_{critical}$=0, u_{max}=1, R=1, α=-1, and the threshold for spiking is 1. Constant input is used for all three: 0.3 (left), 0.4 (middle), and 0.5 (right).

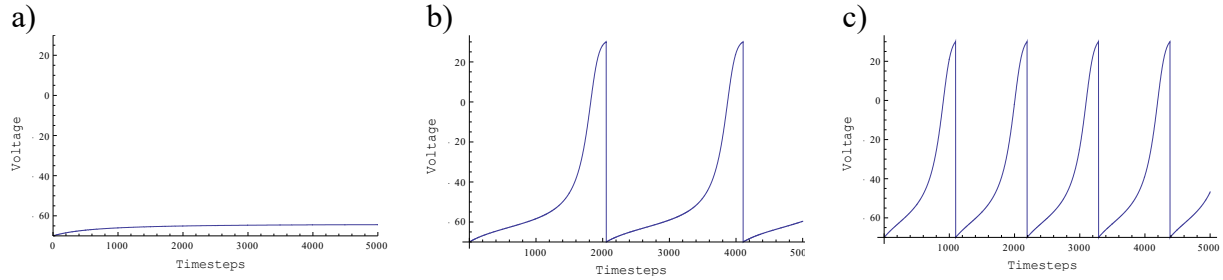

Figure 6: Canonical properties of CIF. Parameters of CIF are: u_{rest}= -70, $u_{critical}$=-55, u_{max}=30, R=1, α=-0.002, and the threshold for spiking is 30. Constant input is used for all three. From left to right, input is set to: 1 (left), 2 (middle), and 4 (right).

Like other nonlinear integrate and fire models, the CIF model also has properties such as spike latency, bi-stability, and threshold variability (Izhikevich, 2004; Mihalaş & Niebur, 2009).

2.5 Simulation 1: Auto-associative Recall

The general purposes of the first simulation are to show that (1) the CIF works as a neuron model, comparable to the accepted QIF, and (2) relationship between neuron model and transmission function influences the recall results of the SNN. This is done using the similar/dissimilar framework mentioned above with an auto associative task that all three networks, Hopfield, NDRAM, and NDRBAM, can perform as NSNNs.

It is hard to compare networks with different neuron models because performance is heavily influenced by model properties. There are a lot of different models to choose from, ranging from LIF simplicity to Huxley-Hodgkin biological realism (Johnson & Chartier, 2017, 2018). Ultimately, the most simple models are used here because (a) more complex models will take longer to run (Izhikevich, 2004), (b) it has been shown that the simplest models can be modified to replicate biological findings (Gerstner & Naud, 2009), and (c) more complex models have more parameters that need to be fitted and will have a bearing on the results making it impossible to determine what exact properties of the model caused the results to vary. Therefore, the three models chosen (LIF, QIF, and CIF) are chosen for their simplicity and for how they are related to each other in the similar/dissimilar framework. It is the effect that these similarities and differences have on the network's results that show the effect of matching the neuron model with the transmission function. LIF is dissimilar from CIF while QIF is similar to CIF. Both the LIF and QIF models are accepted neuron models in current literature. The LIF is the most commonly used neuron model and therefore there are many resources on it, such as Gerstner, (2002) and Johnson & Chartier (2018). While less common than the LIF, the QIF is still a well-known model and information can be found on it at Gerstner (2002) and Izhikevich (2007). The CIF is a new model described in section 2.4.

2.5.1 Methodology

The first step is learning, and as per previous research (Cao et al., 2015; Diehl et al., 2015; Hunsberger & Eliasmith, 2015; Maass, 1997a; Maass & Natschläger, 1998; Meunier & Paugam-Moisy, 2005; Rueckauer et al., 2016; Zambrano & Bohte, 2016), learning is done offline using the NSNN learning rules.

$$\boldsymbol{W}_{\text{Hopfield:}i,j} = \frac{1}{n}\sum_{\boldsymbol{p}=\boldsymbol{1}}^{\boldsymbol{n}} \boldsymbol{p}_i \boldsymbol{p}_j \tag{16}$$

The Hopfield network uses the one-shot learning rule shown in equation 16. The number of patterns is represented by n, and each individual pattern (all n of them) is represented by $\boldsymbol{p}$. The $\boldsymbol{p}_i$ and $\boldsymbol{p}_j$ are the ith and jth nodes of pattern $\boldsymbol{p}$ respectively.

$$\boldsymbol{W}_{\text{NDRBAM}}(k+1) = \boldsymbol{W}_{\text{NDRBAM}}(k) + \eta(\boldsymbol{y}(0) - \boldsymbol{y}(t))(\boldsymbol{x}(0) + \boldsymbol{x}(t))^{\text{T}} \tag{17}$$

$$\boldsymbol{V}_{\text{NDRBAM}}(k+1) = \boldsymbol{V}_{\text{NDRBAM}}(k) + \eta(\boldsymbol{x}(0) - \boldsymbol{x}(t))(\boldsymbol{y}(0) + \boldsymbol{y}(t))^{\text{T}} \tag{18}$$

$$\boldsymbol{W}_{\text{NDRAM}}(k+1) = \boldsymbol{W}_{\text{NDRAM}}(k) + \eta(\boldsymbol{x}(0)\boldsymbol{x}^{\text{T}}(0) - \boldsymbol{x}(t)\boldsymbol{x}^{\text{T}}(t)) \tag{19}$$

In auto-associative tasks, the NDRBAM learning rules (equations 17 and 18) and the NDRAM learning rule (equation 19) are equivalent. This means that in the case of auto-associative tasks, $\boldsymbol{W}_{\text{NDRBAM}}=\boldsymbol{V}_{\text{NDRBAM}}=\boldsymbol{W}_{\text{NDRAM}}$, assuming patterns and order that patterns are learned are identical.

The initial inputs are $\boldsymbol{x}(0)$ and $\boldsymbol{y}(0)$ while $\boldsymbol{x}(t)$ and $\boldsymbol{y}(t)$ are the state vectors after t iterations through the network, per each learning trial, k. In the case of auto-associative memory, $\boldsymbol{x(0) = y(0)}$ and $\boldsymbol{x(t)=y(t)}$. The learning rules for NDRAM and NDRBAM are applied to all patterns repeatedly k times, or for k learning trials, until the algorithm converges to a solution. In

this case, *t* always equals one, since for each learning trial, *k*, the input goes though the network only once. The parameter η represents the learning parameter and is set to 0.01 in all simulations. All patterns are learned in each learning trial, but order of patterns is randomized.

Random variations in learning order can affect recall; to avoid this being a potential cause of different results between NDRAM and NDRBAM, and because in auto-associative tasks their learning rules are equivalent, learning is only done using the NDRBAM learning rules. Therefore, the weight matrices that are used for recall are the same for NDRBAM and NDRAM. The only difference between the spiking NDRAM and spiking NDRBAM is the architecture.

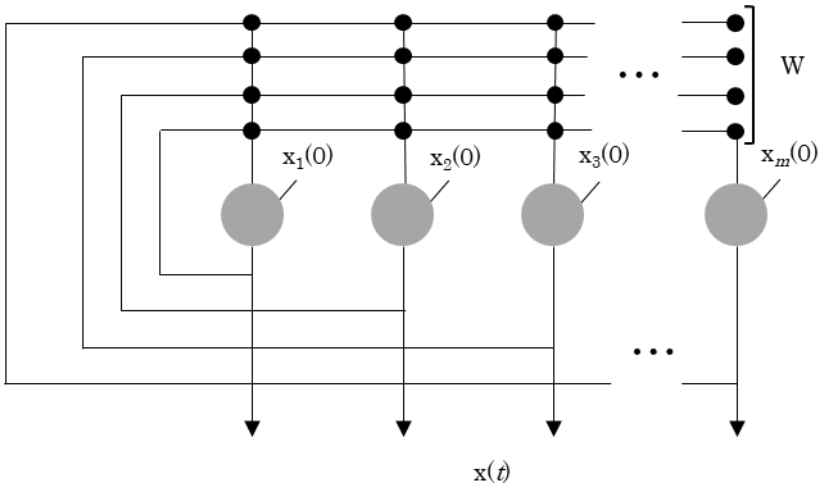

Figure 7: Fully connected recurrent auto-associative neural network architecture. The network architecture contains one layer of nodes that are fully connected to each other.

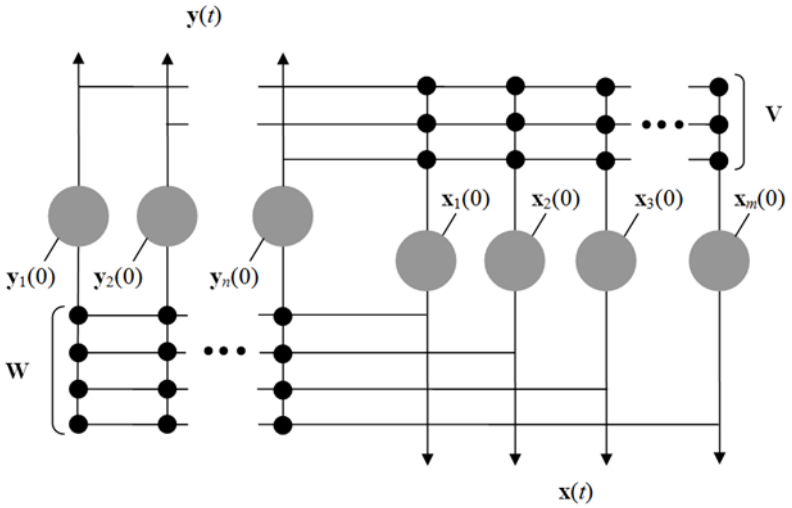

Figure 8: Fully connected recurrent hetero-associative neural network architecture. This network contains two layers and the nodes in each layer are fully connected to the nodes in the other layer.

Both the Hopfield network and NDRAM have the fully connected recurrent auto-associative networks as shown in Fig. 7. Each node passes information to every node in the network though the associated weight matrix.

NDRBAM is two fully connected layers combined, as shown in Fig. 8. In NDRBAM, each node in one layer passes information to every node in the other layer via the associated weight matrices.

For more information on the Hopfield network see Hopfield (1982, 1984) and Hopfield & Tank (1985). The transmission function for NDRAM and NDRBAM is briefly discussed in section 2.4 with the equation given in equation 13. As mentioned in section 2.4, parameters are number set such that: z_1=-1, z_2=0, z_3=1, and a=-1. For more information on NDRAM see Chartier et al. (2005), Chartier & Proulx (2001), and Chartier & Proulx (2005). For more information on NDRBAM see Chartier & Boukadoum (2006).

LIF $$\boldsymbol{v}(t) = u_{rest} + R\boldsymbol{I}(t) + (\boldsymbol{v}(t-1) - (u_{rest} + R\boldsymbol{I}(t)) * e^{-\frac{\delta}{\tau}} \quad (20)$$

QIF $$\boldsymbol{v}(t) = \boldsymbol{v}(t-1) + (\alpha(\boldsymbol{v}(t-1) - u_{rest})(\boldsymbol{v}(t-1) - u_{critical}) + R\boldsymbol{I}(t))\delta \quad (21)$$

CIF $$\begin{aligned}\boldsymbol{v}(t) &= \boldsymbol{v}(t-1) \\ &+ (\alpha(\boldsymbol{v}(t-1) - u_{rest})(\boldsymbol{v}(t-1) - u_{critical})(\boldsymbol{v}(t-1) \\ &- u_{max}) + R\boldsymbol{I}(t))\delta\end{aligned} \quad (22)$$

Where, for all 3 neural modes:

$$if\ \boldsymbol{v}(t) > u_{critical}\ then\ v(t) = u_{rest}$$

Once learning is done, the networks are converted into SNNs by replacing the transmission functions with neural models. The three neural models, LIF, QIF, and CIF, whose equations are shown in equations 20 to 22, require parameter selection distinct to each model. Parameter selection is crucial because it has been shown that it can have a large impact on how well the network performs at its recall task (Wärnberg, 2014). The parameters used in this simulation are not optimized. Instead, the parameters are selected so that the networks all have the same resting potential, rheobase, and critical value. These parameter values are related to the original NSNN NDRAM/NDRBAM. In both CIF and QIF, the models can produce a voltage higher than the critical value, which displays as a spike (see Figures 5 and 6) but the LIF cannot. Instead, the LIF is said to spike when the voltage reaches a threshold value. To avoid differences in spiking between the QIF, CIF, and LIF, the threshold for the spike in the LIF, and subsequently the CIF and QIF, is set to the same value as the critical value in the QIF and CIF. In all cases, the resting potential is -1. The critical value, and therefore the threshold, is 0. CIF has the extra parameter u_{max} which is set to 1. The local maximum in the NDRAM/NDRBAM transmission function is the rheobase, which equals $\frac{2}{3\sqrt{3}}$. To reach this rheobase, CIF sets α=-1 while in QIF α= $\frac{8}{3\sqrt{3}}$. In LIF, the rheobase is set by setting the resistance, R=$\frac{3\sqrt{3}}{2}$ and τ=3.5. The δ is set to 0.1.

These parameters are chosen because they are essentially identical to the parameters used in the NDRAM/NDRBAM transmission function. To make sure that the patterns can cause the neurons to spike, all stimuli are strengthened by a factor of 5 in all models. This means that in QIF and CIF, R =5 and in LIF $R=\frac{15\sqrt{3}}{2}$. The influence of a spike lasts for 25 time steps.

Simulation 1 compares the Hopfield, NDRAM, and NDRBAM neural networks. Hopfield is chosen as one network because there has been a lot of research using it for spiking associative memory networks (Maass, 1997a; Maass & Natschläger, 1998; Tanaka et al., 2005; Wärnberg, 2014). The NDRAM/NDRBAM networks are chosen because the transmission function can be modified into a neural model and because of their similarity; the NDRBAM reduces to the NDRAM for auto-associative tasks (Chartier & Boukadoum, 2006). To confirm that neuron model changes are not compounded because of differences in network architecture, Hopfield and NDRAM have the same architecture while NDRBAM has a different architecture. NDRBAM and NDRAM have the same original transmission function and learning rule, which is different from the Hopfield network. These similarities and differences will be used to determine if any changes in performance are due to the relationship between the neuron model and the transmission function.

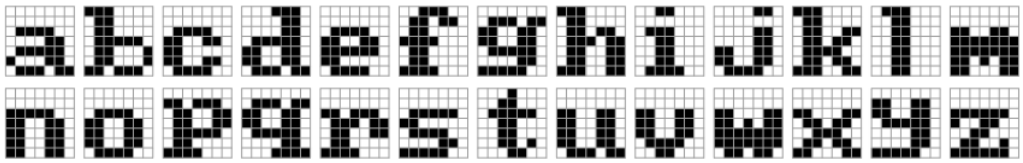

Figure 9: Input used in all simulations; squares represent constant values of -1 (white) and 1 (black).

All networks learn the exact same items, between 1-7 randomly selected lower-case letters as shown in Fig. 9; this represents 2%-14% memory load; memory load is calculated by the number of patterns divided by the number of nodes in the network. The letters shown are translated into a 49-node vector with white being represented by -1 and black by 1. After learning is finished, non-spiking recall is performed as per Hopfield (1982) for Hopfield and Chartier & Boukadoum (2006) for NDRAM and NDRBAM. This step is done to confirm that the non-spiking versions works as expected, and therefore learning, and the related matrices, are valid. The non-spiking results can be found in Appendix A.

For the spiking recall results used in the results section, the transmission function of the NSNN is replaced with the neuron models CIF, QIF, and LIF. These neuron models use the parameters described above. No other changes are made to the networks. The individual network's weight matrix is untouched from the NSNN which means that the Hopfield network has a different weight matrix than NDRAM and NDRBAM but NDRAM and NDRBAM use the same weight matrix. All other network parameters are discussed above. Random noise is introduced into the recall pattern by randomly flipping 0-9 nodes (0%-18% noise level). The same noisy patterns are used in recall for all networks. To translate the input from a non-spiking network of using a [-1,1] vector to a spiking network, the input is set as a constant of -1 (inhibitory) or 1 (excitatory) for each node, for 100 time steps. After this initial presentation time, the external inputs are removed by setting the external input to 0. The networks continue to run with internal input for another 200 time steps. For NDRAM and Hopfield, internal input is the network's previous time step's spiking results but for NDRBAM, the internal input is the opposite layer's previous time step output; the first layer's output goes to the second layer and the vis versa. All trials are repeated 100 times. Results presented here are the averages of those

trials. Because the NDRBAM has two layers that pass information back and forth, only the first layer is presented with the external input; the other layer only uses internal input; there is no external input presented to the second layer.

It needs to be noted that in non-spiking auto-associative memory, NDRAM and NDRBAM are equivalent and will therefore produce the same results if there are no random variations between them, as is the case here. This equivalency cannot be assumed for their spiking forms. For recall in non-spiking format, the external input is presented to the network initially and then the network passes only its internal pattern back and forth between layers (NDRBAM) or to itself (NDRAM) without any external influences. Because both NDRAM and NDRBAM are using the same transmission function and weight matrix, they will produce the same answer, although time steps needed to reach the final solution are reduced for NDRBAM. In the SNN form, the external input is presented for a specific period of time, 100 time steps in our simulations, before being removed (set to 0). In NDRAM, there is only 1 layer, therefore for those 100 time steps, that external input is always presented along with the network's own output. In NDRBAM, there are two layers working concurrently but the external input is only presented to one of the layers for 100 time steps. The second layer's input consists of only the output from the first layer's previous time step. This difference in how the networks receive external input affects their input and subsequently their output. Once the external input is removed, NDRAM and NDRBAM behave exactly the same since they use the same weight matrix and neuron model, but it cannot be assumed that those first 100 time steps did not create enough difference in the internal pattern to have no overall effect on recall, despite NDRAM/NDRBAM being resistant to spurious attractors and fairly robust to noise. Because randomization is controlled via using the same weight matrix and using the same external input,

any differences in results between NDRAM and NDRBAM will be because of the difference that external input presentation causes to internal inputs. When the NDRAM is compared to Hopfield, both are single layer networks and differences in output are only caused by how the different weight matrices are affected by the neuron models. Hopfield and NDRBAM differences are because of both the architecture differences and weight matrices differences.

A network is said to accurately recall a pattern if at some point in the recall, during the previous 25 time steps, only the correct nodes, and all the correct nodes, produce a spike. There is no optimization in data or parameters because we do not care if our results are better than the NSNN results or best spiking models done to date. Rather the goal is to study how the different neuron models affect the results considering their relationship with the original network and the relationships between networks in terms of architecture and transmission function. Evaluation of overall performance is only used in terms of comparison between the different network/neuron model combinations.

The networks used in this simulation are small with only 49 nodes but this should not be a deterrent to applying the results in future, larger simulations. The NSNN can be expanded to larger sizes without changes in results assuming proportion of memory load and noise are the same, therefore we would expect similar results for our SNNs done with this methodology. Noise levels and memory loads are kept small because Hopfield has a low memory load ability. Also, preliminary work showed that without any optimization, spiking can have a lower performance than NSNN, therefore having high noise or memory load could produce no correct recall items which would be difficult to analysis for differences. All simulations are written in Wolfram Mathematica 9.

2.5.2 Auto-Associative Recall Results

2.5.2.1 Hopfield

		LIF - Memory Load		
N		**2%**	**8%**	**14%**
O	**0%**	78.0%	18.3%	7.0%
I	**6%**	78.0%	16.8%	5.7%
S	**12%**	78.0%	15.8%	5.9%
E	**18%**	78.0%	16.3%	5.0%
		QIF - Memory Load		
N		**2%**	**8%**	**14%**
O	**0%**	78.0%	19.5%	6.6%
I	**6%**	78.0%	19.0%	5.0%
S	**12%**	78.0%	16.3%	6.0%
E	**18%**	78.0%	16.8%	4.9%
		CIF - Memory Load		
N		**2%**	**8%**	**14%**
O	**0%**	78.0%	19.5%	6.6%
I	**6%**	78.0%	19.0%	5.0%
S	**12%**	78.0%	16.3%	6.0%
E	**18%**	78.0%	16.8%	4.9%

Table 1: Overall Hopfield spiking performance for auto-associative tasks, separated by neuron model.

As seen in Table 1, initial results for the spiking Hopfield networks show poor performance; it is assumed that with just 1 pattern (2% memory load), recall level should be close to 100% based on previous research (Maass, 1997a; Maass & Natschläger, 1998). This did not happen here, although these results do farther confirm the difficulties found in Wärnberg (2014). This poor performance is explored farther in Appendix B. Despite the poor results, analysis can still be performed because the goal is to determine how the different neuron models affect performance instead of overall accuracy. All three neuron models used with the Hopfield network have the same issues in performance, so the issue is not neuron model specific but network specific; this is farther confirmed since neither NDRAM nor NDRBAM have the same performance issues (see sections 2.5.2.2 and 2.5.2.3).

QIF and CIF perform identically. LIF's performance is similar to QIF/CIF. In general, QIF and CIF may be slightly more susceptible to memory load, while LIF may be more susceptible to noise.

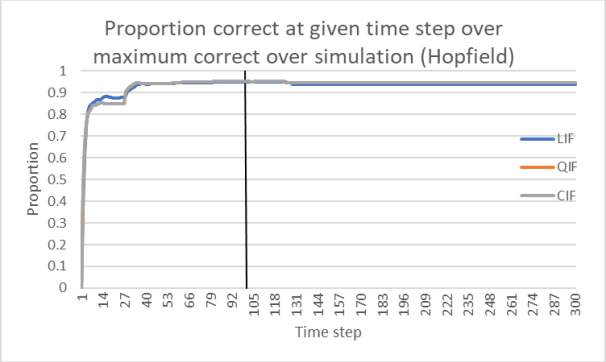

Figure 10: Out of all items correctly recalled, number of items correctly recalled at each time step, divided by total number of items correctly recalled for the Hopfield network. All noise and memory load levels used. Black vertical line indicates when external stimulus is removed.

When recall is successful, most of those items are recalled during the first 100 time steps, while the external input is present. Any noise that is cleared from the input is cleared very quickly. This is clearly shown in Fig. 10, which has optimal performance occurring within the first 100 time steps while the noisy external input is present.

While it makes sense that with no noise, correct recall happens very quickly, this quick recall happens in all levels of noise, as seen in Fig. 11. All three networks show a "jump" in accuracy at around the 30 time step mark, implying noise level itself does not affect how the network cleans data for recall.

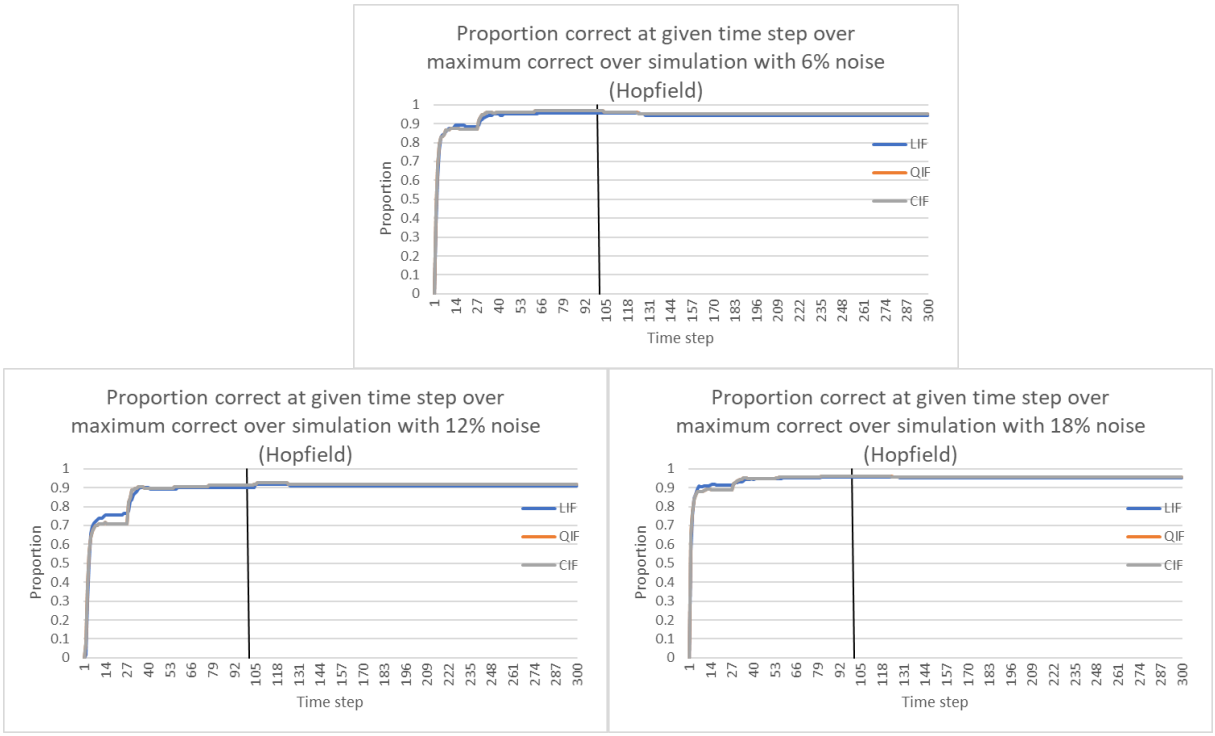

Figure 11: Proportion of correctly recalled patterns at each time step, out of all correctly recalled patterns. Each graph incudes the three neural models combined with the Hopfield network. Top graph shows results with 6% noise, bottom left at 12% noise, and bottom right at 18% noise. Results flattened over all memory load levels. Black vertical line indicates when external stimulus is removed.

Figure 11 shows the proportion of correctly recalled patterns per time step out of all correctly recalled patterns. If all patterns that the network can recall correctly are done so at a particular point of time, that would be 100% -- this is never reached. The fact that proportion of items being correctly recalled at one time never reaches 100% means that at some point in time, a pattern is recalled correctly but than “forgotten” or recalled incorrectly at a subsequent point in time.

2.5.2.2 NDRAM

NDRAM does not have the same issue with the inputs that the Hopfield network has. There is a slight decrease in performance of spiking over non-spiking (non-spiking shown in

Appendix A, Table A.1). This slight decrease in performance is expected considering the lack of optimization.

		LIF - Memory Load		
N		**2%**	**8%**	**14%**
O	**0%**	100.0%	100.0%	99.3%
I	**6%**	100.0%	100.0%	99.4%
S	**12%**	100.0%	99.5%	98.7%
E	**18%**	100.0%	99.3%	95.0%
		QIF - Memory Load		
N		**2%**	**8%**	**14%**
O	**0%**	100.0%	100.0%	99.3%
I	**6%**	100.0%	100.0%	99.4%
S	**12%**	100.0%	99.5%	98.7%
E	**18%**	100.0%	99.3%	95.0%
		CIF - Memory Load		
N		**2%**	**8%**	**14%**
O	**0%**	100.0%	100.0%	99.3%
I	**6%**	100.0%	100.0%	99.4%
S	**12%**	100.0%	99.5%	98.9%
E	**18%**	100.0%	99.3%	95.0%

Table 2: Overall NDRAM spiking performance for auto-associative tasks, separated by neuron model.

Interestingly, LIF is more similar to QIF and CIF in this network compared to Hopfield; there is an overall average total difference of 0.04% difference between QIF and LIF, 0.02% between CIF and QIF (in Hopfield, these two are identical), and 0.02% difference between CIF and LIF. Because the only differences between the three networks are the neuron models, the fact that the differences are so small imply that the differences are not significant and using any of these neuron models with these parameters will provide equivalent results.

All three neuron models produce near identical performance in these networks. The majority of items correctly recalled are only recalled after the noisy input is removed. Unlike the Hopfield network, the NDRAM does not do the majority of its cleaning of data while the external inputs are present. The NDRAM also reaches, and stays at, 100% recall for items it does

recall, implying that once an item is correctly recalled, it will be continually passed correctly through the network at each time step.

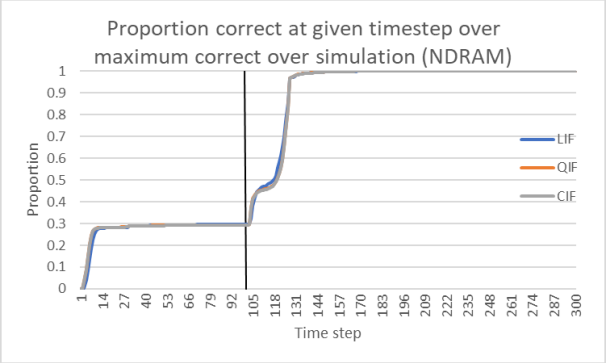

Figure 12: Proportion of correctly recalled patterns at each time step, out of all correctly recalled patterns for the NDRAM network. Flattened over all noise and memory load levels. Black vertical line indicates when external stimulus is removed.

2.5.2.3 NDRBAM

FIRST LAYER					SECOND LAYER				
		LIF - Memory Load					LIF - Memory Load		
N		**2%**	**8%**	**14%**	**N**		**2%**	**8%**	**14%**
O	**0%**	100.0%	97.5%	98.6%	**O**	**0%**	100.0%	97.5%	98.6%
I	**6%**	100.0%	97.3%	97.3%	**I**	**6%**	100.0%	97.3%	97.3%
S	**12%**	100.0%	96.5%	95.4%	**S**	**12%**	100.0%	96.5%	95.4%
E	**18%**	100.0%	94.3%	90.0%	**E**	**18%**	100.0%	94.3%	90.0%
		QIF - Memory Load					**QIF - Memory Load**		
N		**2%**	**8%**	**14%**	**N**		**2%**	**8%**	**14%**
O	**0%**	100.0%	100.0%	99.3%	**O**	**0%**	97.8%	98.4%	100.0%
I	**6%**	100.0%	100.0%	99.4%	**I**	**6%**	97.5%	97.6%	100.0%
S	**12%**	100.0%	99.5%	98.7%	**S**	**12%**	96.5%	95.7%	100.0%
E	**18%**	100.0%	99.3%	95.0%	**E**	**18%**	94.8%	90.0%	100.0%
		CIF - Memory Load					**CIF - Memory Load**		
N		**2%**	**8%**	**14%**	**N**		**2%**	**8%**	**14%**
O	**0%**	100.0%	97.8%	98.4%	**O**	**0%**	100.0%	97.8%	98.4%
I	**6%**	100.0%	97.5%	97.7%	**I**	**6%**	100.0%	97.5%	97.7%
S	**12%**	100.0%	96.5%	95.6%	**S**	**12%**	100.0%	96.5%	95.6%
E	**18%**	100.0%	95.0%	90.3%	**E**	**18%**	100.0%	95.0%	90.3%

Table 3: Overall NDRBAM spiking performance for auto-associative tasks, separated by neuron model and layer. The first layer (left) receives external input as well as internal input from the second layer. The second layer (right) only receives internal input from the first layer.

The NDRBAM has the largest differences in accuracy across the three neural models, although these differences are still quite small. The overall difference between CIF and QIF is 0.08% (layer 1) and 0.06% (layer 2) (with CIF leading). The dissimilar differences, those between QIF and LIF (0.13% for layer 1 and 0.15% for layer 2) and CIF and LIF (0.21% for both layers) is more than the similar (CIF-QIF) differences. There might be a trend towards better performance in the similar neuron models, but the difference is still very small and therefore probably insignificant with these parameters.

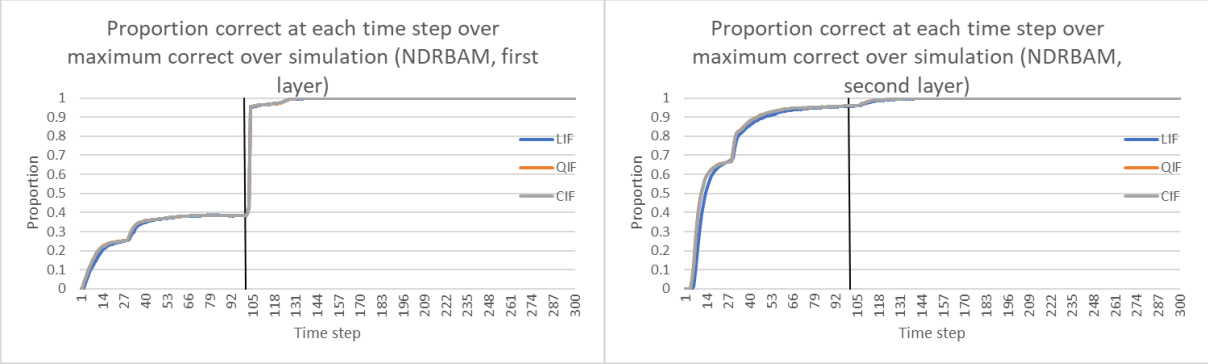

Figure 13: Proportion of correctly recalled patterns at each time step, out of all correctly recalled patterns for the NDRBAM first layer (left, receives external input) and second layer (right, does not receive external input). All noise and memory load levels used. Black vertical line indicates when external stimulus is removed.

The first layer works in a similar fashion as NDRAM, producing the most accurate results after the external input is removed. The second layer is able to clean out most noise from the first layer's internal input, despite receiving an incorrect pattern from the first layer.

Like NDRAM, NDRBAM also appears to keep items it recalls correctly in memory for the duration of the simulation.

2.5.3 Discussion

There are three main differences between the networks: overall performance, clearing of noise, and retaining correct output in memory. For all three issues, Hopfield is different when compared to NDRAM and NDRBAM.

Hopfield is the only network that has an issue recalling select input, and therefore is unable to reach 100% accuracy. The fact that the network can recall perfectly in the non-spiking version, and that the NDRAM network does not have the same issues when spiking implies that there is something unique about the conversion to spiking for the Hopfield network. This affects all three neuron models in the same way. All three neuron models though, are matched to NDRAM's and NDRBAM's transmission function. This difference in performance between the networks implies that matching neuron model and transmission function is in fact an important component of creating spiking neural networks, at least when converting from NSNNs.

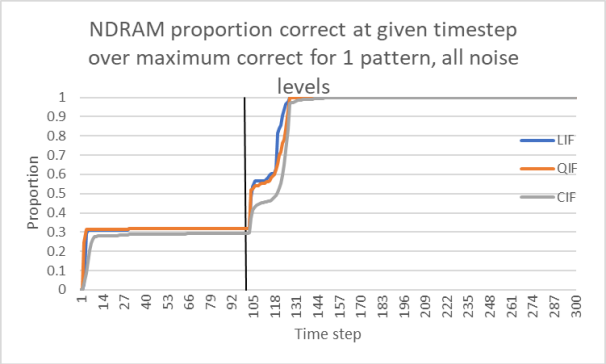

Figure 14: NDRAM recall proportion of correctly recalled items out of all correctly recalled items per time step for lowest memory load (2% - 1 pattern learned)

Another interesting difference between networks is when noise is cleared from the network; specifically, that the Hopfield produces its best results while the noisy external inputs are still present while NDRAM and NDRBAM require the noisy external inputs to be removed

first. The fact that once the noisy external input is cleared, NDRBAM cleans up the noise quicker than NDRAM is logical because the second layer has essentially already filtered the noise out, so the first layer immediately gets cleaner data right away. The reasons for the differences between NDRAM/NDRBAM and Hopfield are less clear. It could be argued that Hopfield's low performance is a misleading factor in the cleanup of data. The spiking Hopfield networks appear to only really recall 1 item that is learned; as soon as multiple patterns are learned, recall performance drastically decreases. But, when looking at spiking NDRAM percentage correct over total correct at each time step (Fig. 14), we see that even with just 1 pattern, NDRAM still clears noise after pattern removal. Due to the fact that performance for 1 pattern is identical at all noise levels implies that noise is not a factor. In general, the fact that Hopfield does perform badly while NDRAM and NDRBAM do not ties with the fact that the neural model parameters are set to match the transmission function of NDRAM/NDRBAM, not Hopfield. This tie between transmission function and neural model appears to be necessary.

Finally, a more subtle difference is keeping a correct pattern in the network. Once a pattern is correctly recalled, both NDRAM and NDRBAM keep the correct pattern alive in memory; they are able to keep recalling it for the remainder of the simulation. The Hopfield network recalls most patterns for the remainder of the simulation, but there appears to be about 10% of the patterns that, while they can be correctly recalled at some point, that pattern is not maintained in memory.

There are some minor differences between NDRAM and NDRBAM – NDRAM has overall better recall but NDRBAM clears noise from the network faster – which indicate that the different network architectures and subsequent input presentation differences do have a small influence over recall accuracy. But the main differences occur between NDRAM/NDRBAM and

the Hopfield networks. Hopfield is dissimilar in transmission function and this is reflected in how it performs with the different neural models.

2.6 Simulation 2: Hetero-associative

Auto-associative tasks are easier than hetero-associative tasks. Yet, even in auto-associative task, NDRBAM had a lower performance compared to NDRAM and its corresponding NSNN which brings up the question of if hetero-associative tasks will work, and how well. This simulation confirms that spiking NDRBAM can still work with hetero-associative tasks without serious performance degradation. This simulation also adds to the comparison of the effects that neuron models can have on spiking neural networks for hetero-associative tasks.

2.6.1 Methodology

Of the three general networks used in this paper, only the NDRBAM is a bidirectional network capable of performing hetero-associative tasks. The methodology here is almost identical to that used in simulation, including parameters used for the neuron models. The only differences are that only NDRBAM is used, memory load levels increased to 2-20% (1-10 patterns learned and recalled), and both upper- and lowercase letters are learned.

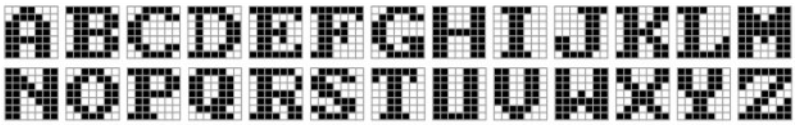

Figure 15: Associated input used in simulation 2. Squares represent constant values of -1 (white) and 1 (black).

Patterns include the lowercase letters shown in Fig. 9 associated with their corresponding uppercase letters shown in Fig. 15. As with simulation 1, learning is done offline (non-spiking)

and non-spiking recall results can be found in Appendix A, Table A.2. Weight matrices created during learning are used in recall without any modification.

2.6.2 Recall Results

FIRST LAYER – LOWER CASE						SECOND LAYER – UPPER CASE					
		LIF – MEMORY LOAD						LIF – MEMORY LOAD			
NOISE		**2%**	**8%**	**14%**	**20%**	**NOISE**		**2%**	**8%**	**14%**	**20%**
	0%	100.0%	98.3%	95.4%	90.4%		**0%**	100.0%	98.3%	95.4%	90.4%
	6%	100.0%	97.3%	94.0%	86.6%		**6%**	100.0%	97.3%	94.0%	86.6%
	12%	100.0%	95.8%	90.9%	76.6%		**12%**	100.0%	95.8%	90.9%	76.6%
	18%	100.0%	93.0%	83.0%	62.5%		**18%**	100.0%	93.3%	83.0%	62.5%
		QIF – MEMORY LOAD						QIF – MEMORY LOAD			
NOISE		**2%**	**8%**	**14%**	**20%**	**NOISE**		**2%**	**8%**	**14%**	**20%**
	0%	100.0%	98.5%	95.7%	90.8%		**0%**	100.0%	98.5%	95.7%	90.8%
	6%	100.0%	97.8%	94.6%	87.3%		**6%**	100.0%	97.8%	94.6%	87.3%
	12%	100.0%	96.5%	91.3%	77.7%		**12%**	100.0%	96.5%	91.3%	77.7%
	18%	100.0%	94.3%	84.1%	64.8%		**18%**	100.0%	94.3%	84.1%	64.8%
		CIF – MEMORY LOAD						CIF – MEMORY LOAD			
NOISE		**2%**	**8%**	**14%**	**20%**	**NOISE**		**2%**	**8%**	**14%**	**20%**
	0%	100.0%	98.5%	95.7%	91.0%		**0%**	100.0%	98.5%	95.7%	91.0%
	6%	100.0%	97.8%	94.7%	87.3%		**6%**	100.0%	97.8%	94.7%	87.3%
	12%	100.0%	96.5%	91.6%	77.8%		**12%**	100.0%	96.5%	91.6%	77.8%
	18%	100.0%	94.5%	84.0%	65.1%		**18%**	100.0%	94.5%	84.0%	65.1%

Table 4: Overall NDRBAM spiking performance for hetero-associative tasks, separated by neuron model and layer. The first layer (left) receives external input (lowercase letter) as well as internal input from the second layer. The second layer (right) only receives internal input from the first layer.

Results from all three SNNs are similar, although there is a slightly larger difference between neuron models than observed in simulation 1. Both CIF and QIF show either the same or better performance than LIF, and while individual differences are small, the overall differences are larger than seen in simulation 1. CIF averages 0.9% (layers 1 and 2) better than

LIF while QIF averages 0.8% (layers 1 and 2) better than LIF. Meanwhile, the average difference between CIF and QIF is still very small at only 0.1% (layers 1 and 2).

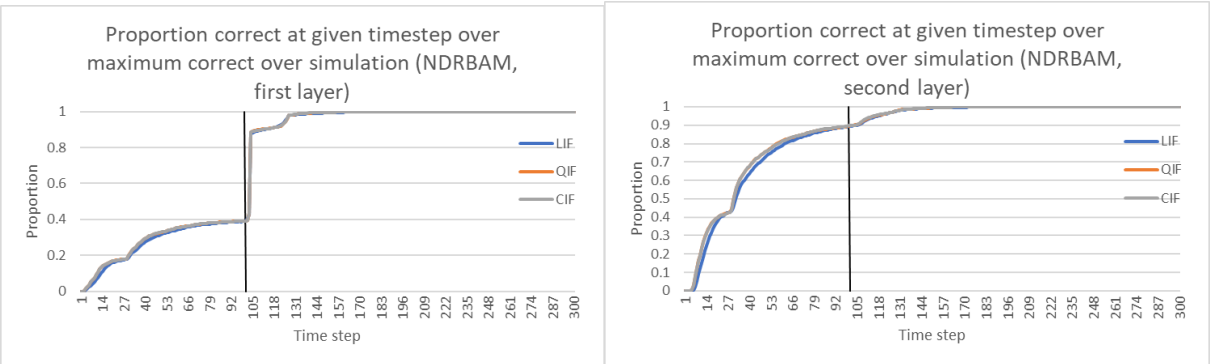

Figure 16: Spiking NDRBAM, the first layer (right, receives external input) and the second layer (left, does not receive external input), proportion correct at any time step over total patterns correctly recalled.

As in Simulation 1, the second layer is able to clean up data despite having incorrect input from the first layer, and the first layer requires the noisy external input to be removed before reaching maximum accuracy. Once a pattern is cleaned, it appears to be retained in memory for the life of the network if no other input is given (Fig. 16).

2.6.3 Discussion

Hetero-associative tasks are harder than auto-associative tasks, therefore the fact that the network performs worse in this simulation than simulation 1 is expected. But it should be noted that the differences between spiking and NSNN in both simulation 1 and 2 are comparable, leading to the idea that while hetero-associative task is harder; spiking does not increase the complexity.

This simulation shows slightly clearer evidence of neural model influence, as it shows greater difference between the three neural models. The CIF neural model has the best performance while the LIF has the worse, which supports matching neural model with

transmission function. This is not conclusive evidence though, since the differences are still small and parameter selection is done based on NSNN NDRAM/NDRBAM.

2.7 Simulation 3: Parameter Manipulation

There appears to be a slight effect of neural model, but the results are from one set of parameters, and the strongest evidence is still very minimal, and it is from the hetero-associative NDRBAM simulation, which cannot confirm if the reason for better CIF performance is because of its similarity to the original transmission function alone.

Simulation 3 delves into the neuron models more by using a variety of different parameters to see how parameter selection affects the results. This is in conjunction with the neural network (Hopfield, NDRAM, NDRBAM) differences, which allows us to study how differences in original NSNN affect performance of spiking networks with similar and dissimilar neuron models.

2.7.1 Methodology

Simulation 3 methodology is similar to simulation 1 in the networks used, input patterns (lowercase letters from Fig. 9), and original NSNNs are used for off-line learning. The main difference in simulation 3 is neuron model parameter selection. Four parameters, the rheobase, resting and critical values, δ, and resistance are set to a variety of values to determine their effects on performance of the networks. Three large and 3 small values are used for the rheobase (2, 4, 6 for large and 0.2, 0.4, and 0.6 for small). The rheobase is set by modifying α in CIF and QIF and R in LIF. The original resting and critical values (-1 and 0) are used, as well as a larger set (-1, 1). In the CIF model, the third fixed point, u_{max}, is set to 1 and 3 respectively. The rheobase and resting/critical values affect the size and shape of the curve that the voltage follows, while the δ affects how large or small the changes in voltage are. In this case 4 different

values are used, 0.001, 0.01, 0.1, and 1. Finally, the resistance is set to low (0.01, 0.05, 0.1, 0.5), medium (1, 5, 11, 15) and high (50, 100, 150, 200). In all three neuron models, resistance is set by *R*; in LIF this is multiplied by the value needed to determine the corresponding rheobase.

In all trials, 2 randomly chosen lowercase letters are learned. The effects of memory load are not being considered here, so 2 items allow us to make sure it can recall multiple items without straining the memory load. It is expected that parameter selection itself will produce a variety of results ranging from floor to ceiling. Noise is set to 0-16% (0-8 pixels flipped). Between the different parameters and noise options, there are 1728 unique combinations (576 per noise level) per the 9 network/model combinations and each combination is run 100 times. All networks use the same input for learning and recall.

Simulation 1 and 2 show that peak performance is reached quickly, either while external input is present (Hopfield) or after it is removed (NDRAM/NDRBAM), therefore length of time the simulations are run for is reduced. The external input pattern is presented for the first 50 time steps before being set to 0, then the network continues to run for another 100 time steps making the network run for a total of 150 time steps.

Learning is done offline and all parameters used in learning are the same as simulation 1. To confirm learning was performed correctly, all items at all noise levels are recalled in the non-spiking networks. All 3 NSNN recalled perfectly regardless of noise level.

2.7.2 Results

Because of the sheer volume of parameter combinations, the focus of the analysis here is how the parameter changes affect the different neural network/model combinations. Interactions are presented to illustrate these effects in more detail.

2.7.2.1 Resistance

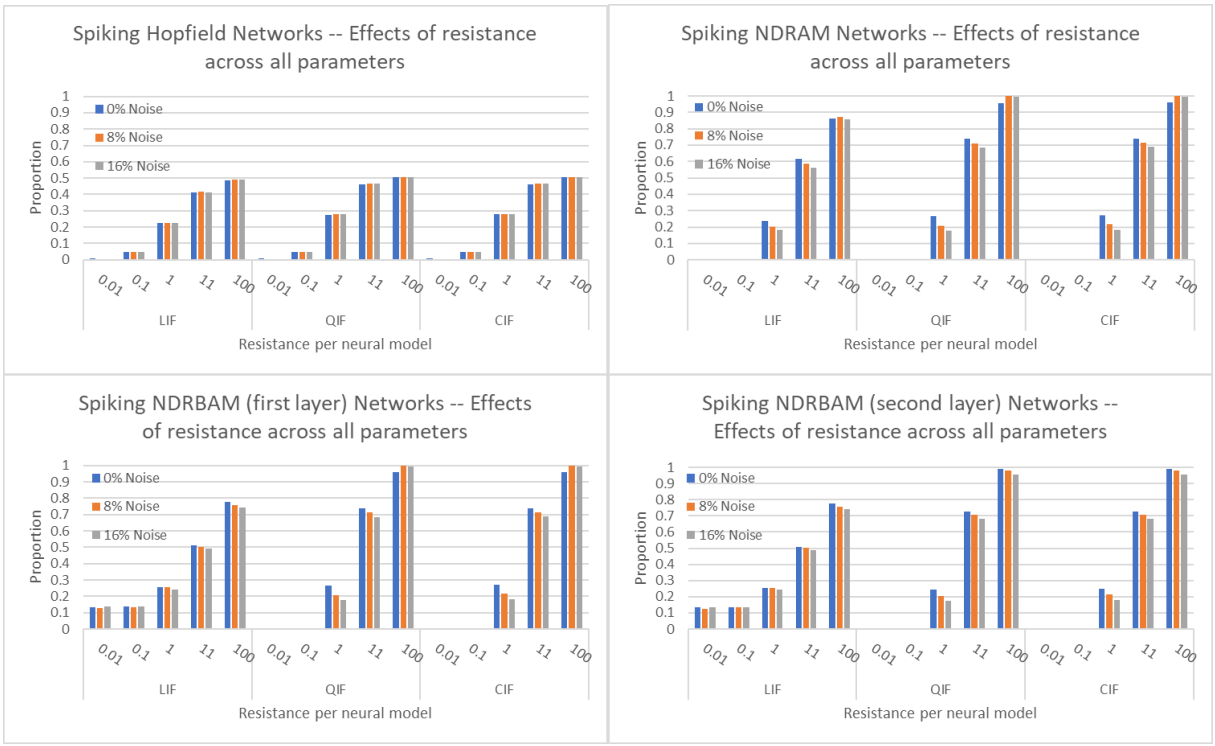

Figure 17: Effects of increasing resistance on network performance for all spiking network combinations. All other parameters flattened.

There is a clear effect, across all networks and neuron models, that as resistance increases, accuracy increases (see Fig. 17). The effect of resistance is not uniform across neuron models and networks though; NDRAM and NDRBAM experience a larger increase in accuracy with increased resistance values. Within NDRAM and NDRBAM, QIF and LIF experience larger increases in accuracy than LIF. At low values of resistance, NDRAM and NDRBAM cannot recall, except in the case of NDRBAM LIF. Low values of resistance decrease input voltage of external and internal sources, therefore making it harder to reach the critical value while high values do the reverse, making it easier to reach the critical value needed to produce a spike.

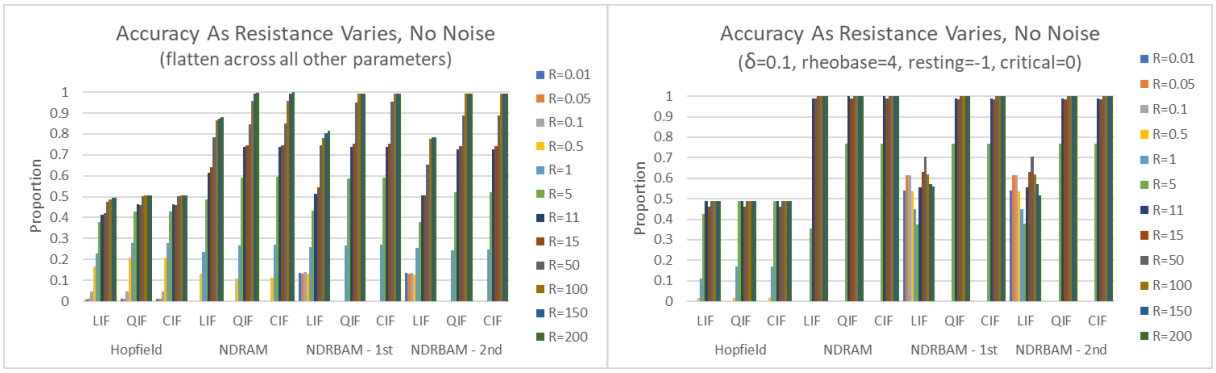

Figure 18: Displaying how there is an interaction between resistance, network, and select parameters. Proportion of items recalled correctly when resistance is increased across all networks. 0% noise used. Flattened across all other parameters (left) and with δ=0.1, rheobase=4, resting =-1, and critical=0 (right).

A more in-depth look at resistance shows that there is an interaction between resistance and neuron model. Specifically, the LIF model sometimes produces more chaotic results as shown in the right-hand side of Fig. 18 for NDRBAM.

2.7.2.2 Resting and Critical Values

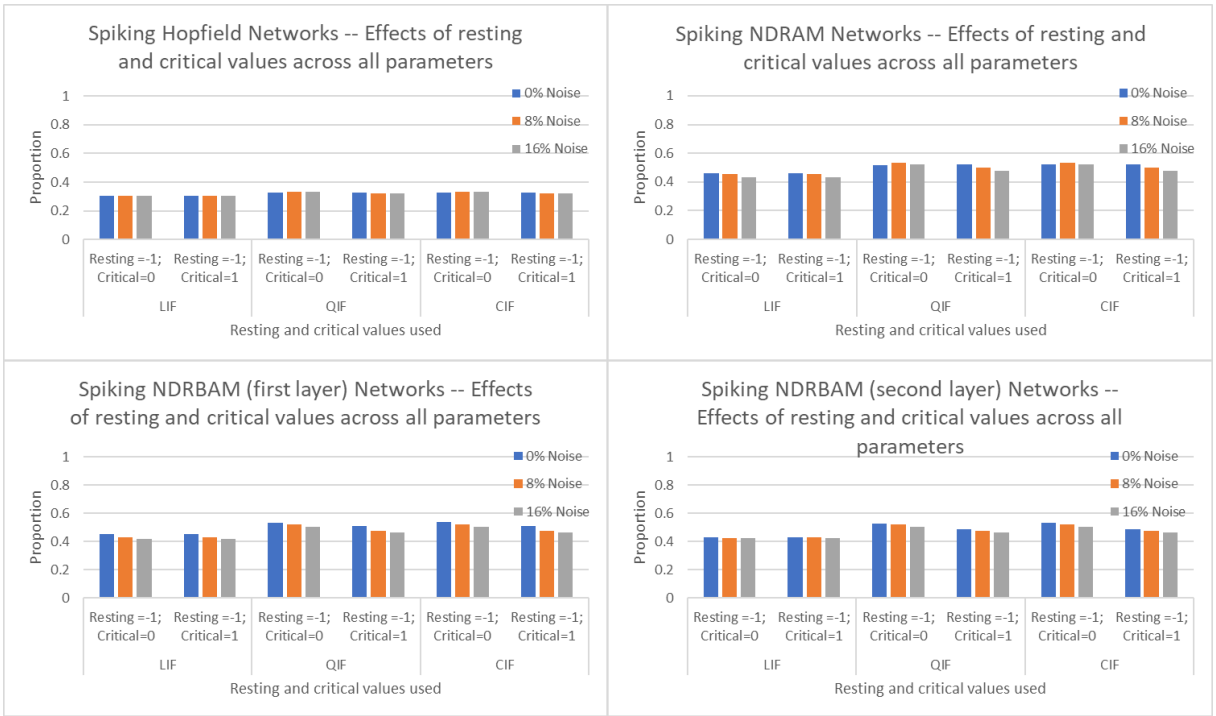

Figure 19: Effects of resting and critical values for all spiking network combinations. All other parameters flattened.

Figure 19 shows that the resting and critical values do not appear to have much effect in the overall performance, although the closer values (-1 and 0) trends towards having better performance.

2.7.2.3 Rheobase

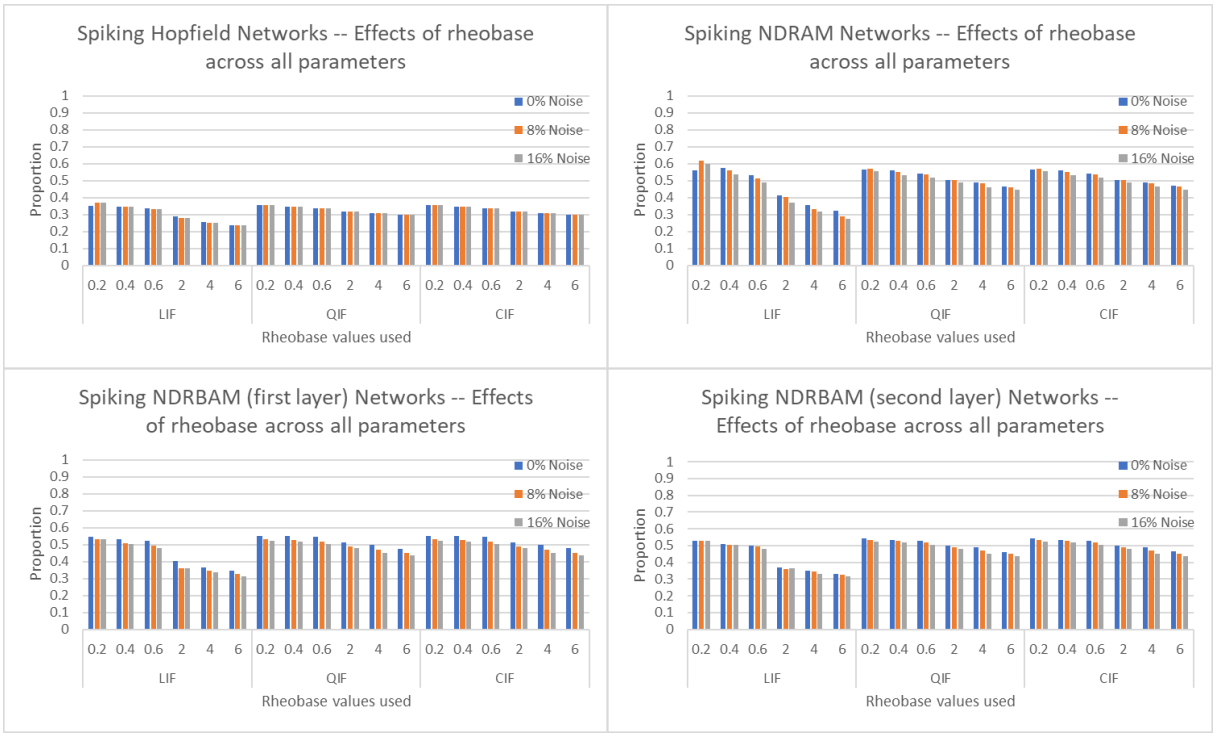

Figure 20: Effects of rheobase size for all spiking network combinations. All other parameters flattened.

As values increase for the rheobase, performance tends to decrease. This effect of the rheobase is seen across all networks and neuron models (Fig. 20) although the effect appears to be felt more by the LIF model than the CIF or QIF models. NDRAM LIF appears to be the most affected by changes in the rheobase.

2.7.2.4 δ

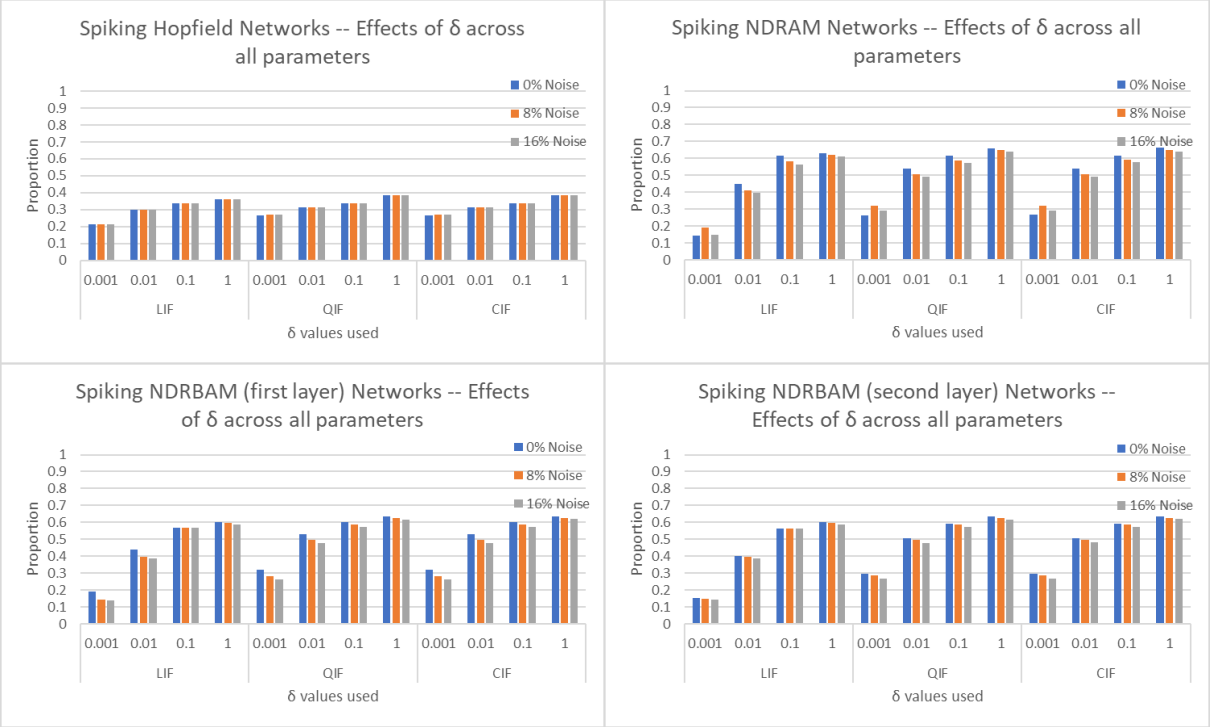

Figure 21: Effects of δ on all spiking neural network combinations. All other parameters flattened.

The δ has a clear effect with the larger δ values producing better results (Fig. 21). There does not to appear to be an interaction between δ and neuron model (in each network type, CIF, QIF, and LIF appear to change equally), but there is one between δ and network. The Hopfield networks show the smallest change in performance as δ increases. The NDRBAM and NDRAM networks show both show larger increases in accuracy as δ increases compared to Hopfield.

2.7.2.5 Networks & Neural Models

The timing of when the networks produce accurate results is similar to Simulation 1's findings. As seen in Figure 22, NDRAM and NDRBAM produce their most accurate results after the noisy external input is removed while Hopfield can produce accurate results with the noisy

external input still present. Here, Hopfield network shows some increase even after the external input is removed though, which was not shown in Simulation 1.

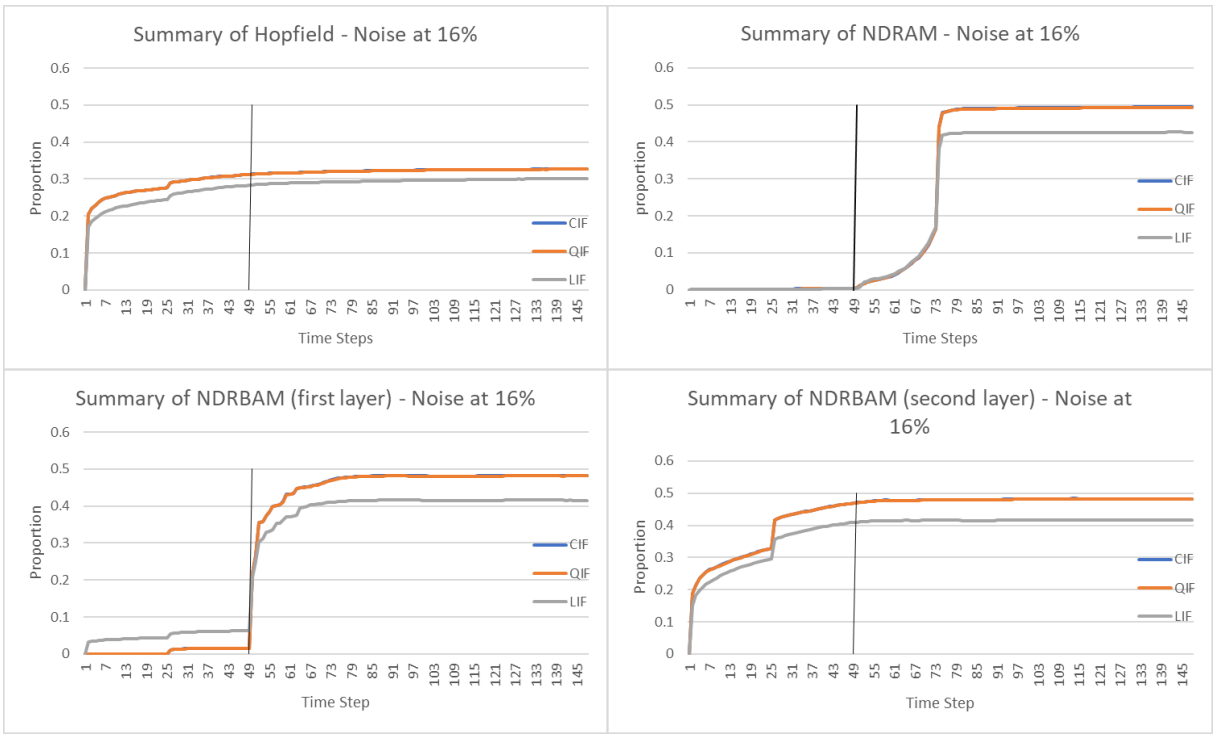

Figure 22: Proportion correct over total correct at each time step across networks. Hopfield on top right, NDRAM top left, and NDRBAM on bottom right (first layer) and left (second layer). Noise is set to 16% and flattened across all other parameters. Black vertical line is when noisy external input is removed.

The values for δ interacts with the Hopfield network accuracy over time. Figure 23 shows the Hopfield network's accuracy across the lifetime of the simulation and how with small δ it takes longer to reach peak accuracy but continues to improve in performance after the external data is removed (top left). With the larger δ (bottom right), it removes whatever noise it can to recall at its peak accuracy almost immediately. It should be noted that in Fig. 23, it appears that the Hopfield network does the worse when δ=0.001, but this is misleading due to interactions with other parameters. In fact, as seen in Table 5, Hopfield only reaches its best recall performance when δ=0.001. For the Hopfield network, the problem may be that the simulation

time was reduced too much. If it had run longer, the lower values of δ may ultimately show better results than the larger δ values. With the low δ values, the network takes longer to converge on a solution, therefore more time could benefit those parameters.

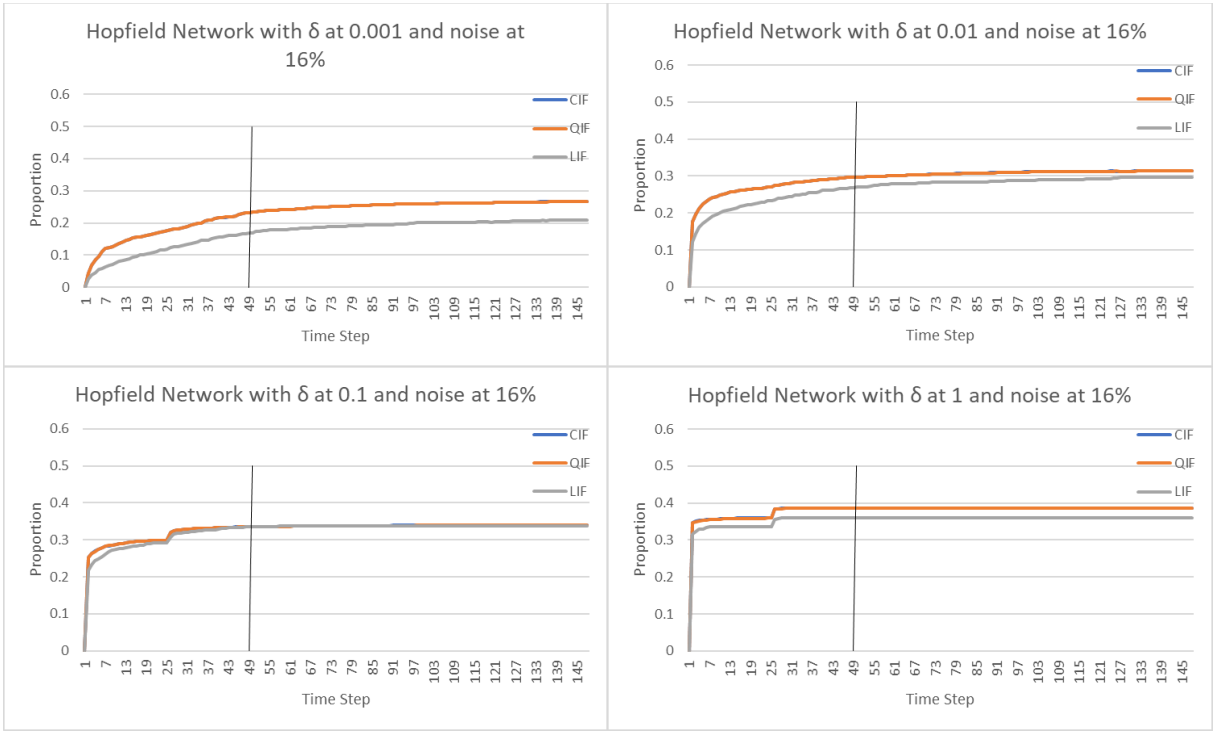

Figure 23: Hopfield network over the 4 different δ options (δ=0.001 top left, δ=0.01 top right, δ=0.1 bottom left, and δ=1 bottom right) and noise at 16%. All other parameters flattened. Black vertical line indicates when external pattern was removed.

			HOPFIELD			NDRAM		
R	**Δ**	**A**	**LIF**	**QIF**	**CIF**	**LIF**	**QIF**	**CIF**
High	**1**	**Large**	51.0%	51.0%	51.0%	100.0%	100.0%	100.0%
High	**1**	**Small**	51.0%	51.0%	51.0%	100.0%	100.0%	100.0%
High	**0.1**	**Large**	49.0%	49.0%	49.0%	100.0%	100.0%	100.0%
High	**0.1**	**Small**	49.0%	49.0%	49.0%	100.0%	100.0%	100.0%
High	**0.01**	**Large**	49.5%	49.5%	49.5%	94.9%	100.0%	100.0%
High	**0.01**	**Small**	49.5%	49.5%	49.5%	100.0%	100.0%	100.0%
High	**0.001**	**Large**	46.7%	52.5%	52.5%	24.4%	69.7%	70.1%
High	**0.001**	**Small**	43.6%	51.6%	51.6%	60.9%	61.0%	61.3%
Medium	**1**	**Large**	42.8%	51.0%	51.0%	64.4%	89.3%	90.3%
Medium	**1**	**Small**	51.0%	51.0%	51.0%	99.2%	99.4%	99.4%
Medium	**0.1**	**Large**	38.8%	40.4%	40.4%	59.3%	65.4%	66.4%
Medium	**0.1**	**Small**	48.3%	48.3%	48.3%	98.0%	98.5%	98.6%
Medium	**0.01**	**Large**	27.9%	39.5%	39.5%	3.8%	59.6%	59.6%
Medium	**0.01**	**Small**	45.4%	46.1%	46.1%	70.8%	74.7%	74.7%
Medium	**0.001**	**Large**	11.5%	25.8%	25.8%	0.0%	0.0%	0.0%
Medium	**0.001**	**Small**	22.2%	22.5%	22.5%	0.0%	0.0%	0.0%
Low	**1**	**Large**	1.7%	11.4%	11.4%	0.0%	0.0%	0.0%
Low	**1**	**Small**	19.0%	19.5%	19.5%	15.1%	15.6%	16.7%
Low	**0.1**	**Large**	1.0%	1.5%	1.5%	0.0%	0.0%	0.0%
Low	**0.1**	**Small**	16.1%	16.8%	16.8%	11.2%	10.5%	10.6%
Low	**0.01**	**Large**	0.0%	1.2%	1.3%	0.0%	0.0%	0.0%
Low	**0.01**	**Small**	7.6%	8.6%	8.6%	0.0%	0.0%	0.0%
Low	**0.001**	**Large**	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
Low	**0.001**	**Small**	3.5%	4.4%	4.4%	0.0%	0.0%	0.0%

Table 5: Summary of Hopfield and NDRAM results for all high (50, 100, 150, 200), medium (1, 5, 11, 15), low (0.01, 0.05, 0.1, 0.5) resistance (R), all levels of δ, and large (2, 4, 6) and small (0.2, 0.4, 0.6) rheobase (A) when there is no noise and resting/critical values at -1, 0, 1

			NDRBAM – FIRST LAYER			NDRBAM – SECOND LAYER		
R	**Δ**	**A**	**LIF**	**QIF**	**CIF**	**LIF**	**QIF**	**CIF**
High	**1**	**Large**	99.7%	100.0%	100.0%	99.7%	100.0%	100.0%
High	**1**	**Small**	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%
High	**0.1**	**Large**	64.1%	100.0%	100.0%	63.2%	100.0%	100.0%
High	**0.1**	**Small**	63.5%	100.0%	100.0%	63.4%	100.0%	100.0%
High	**0.01**	**Large**	93.3%	97.3%	97.3%	83.8%	97.3%	97.3%
High	**0.01**	**Small**	97.3%	97.3%	97.3%	97.3%	97.3%	97.3%
High	**0.001**	**Large**	13.4%	100.0%	100.0%	3.3%	100.0%	100.0%
High	**0.001**	**Small**	97.1%	100.0%	100.0%	88.1%	100.0%	100.0%
Medium	**1**	**Large**	63.9%	89.3%	90.3%	63.1%	89.3%	90.3%
Medium	**1**	**Small**	98.8%	99.2%	99.3%	98.8%	99.2%	99.3%
Medium	**0.1**	**Large**	52.7%	65.1%	66.0%	52.8%	64.1%	64.1%
Medium	**0.1**	**Small**	53.3%	97.8%	97.8%	52.5%	97.8%	97.8%
Medium	**0.01**	**Large**	3.8%	59.5%	59.5%	0.3%	51.5%	51.9%
Medium	**0.01**	**Small**	70.2%	73.9%	73.9%	60.6%	73.9%	73.9%
Medium	**0.001**	**Large**	0.0%	0.8%	0.9%	0.0%	0.0%	0.0%
Medium	**0.001**	**Small**	6.1%	3.1%	3.1%	0.9%	0.1%	0.1%
Low	**1**	**Large**	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
Low	**1**	**Small**	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
Low	**0.1**	**Large**	53.5%	0.0%	0.0%	52.5%	0.0%	0.0%
Low	**0.1**	**Small**	53.1%	0.0%	0.0%	52.7%	0.0%	0.0%
Low	**0.01**	**Large**	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
Low	**0.01**	**Small**	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
Low	**0.001**	**Large**	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
Low	**0.001**	**Small**	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%

Table 6: Summary of NDRBAM results (both layers) for all high (50, 100, 150, 200), medium (1, 5, 11, 15), low (0.01, 0.05, 0.1, 0.5) resistance (R), all levels of δ, and large (2, 4, 6) and small (0.2, 0.4, 0.6) rheobase (A) when there is no noise and resting/critical values at -1, 0, 1

The Hopfield network has an overall worse performance than the other two networks. In fact, the top accuracy that Hopfield reaches is 52.5% while both NDRAM and NDRBAM can achieve 100% accuracy (see Tables 5 and 6). The Hopfield network reaches its highest accuracy levels when the δ is 0.001 (smallest value used), and the resistance is at the high levels (50, 100, 150, 200). There are neuron model differences for the rheobases where CIF and QIF reach peak accuracy regardless of rheobase while LIF requires the rheobase to be at the small level. The

NDRAM and NDRBAM perform better with large δ and resistance values; and at those levels, the rheobase has no effect. NDRAM and NDRBAM perform very well over select parameters while Hopfield will perform, albeit poorly, over a wider range of parameters.

There is also an interaction with resistance, specifically with the LIF neuron model and spiking NDRBAM network. At low resistance, the NDRBAM is still able to recall some items using the LIF neural model but has no recall for CIF or QIF. Once the resistance reaches medium to high levels though, QIF and CIF neural models have a definite increase in performance over LIF. This low resistance results do not occur with any of the NDRAM network.

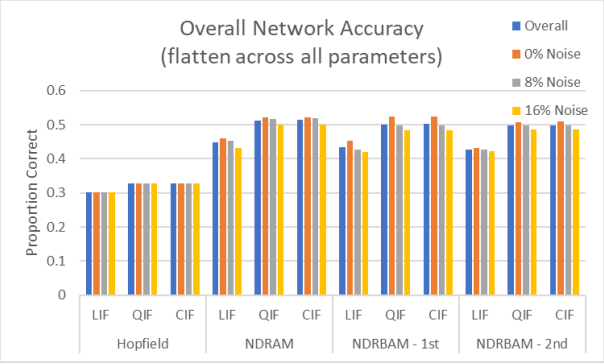

Figure 24: Comparison of overall network accuracy between all 9 network/neural model combinations for different levels of noise. All other parameters flattened.

Overall, QIF and CIF perform better in all networks than LIF regardless of parameters (see Fig. 24); except at low level resistance (see Fig. 17). In the Hopfield network, there is no difference between CIF and QIF and the difference between LIF and CIF/QIF is small. Compared to Hopfield network, there is a larger difference between LIF and CIF/QIF in both NDRAM and NDRBAM. The difference between overall performance of QIF and CIF is minor in NDRAM and NDRBAM. For optimal results using parameters studied here, resistance needs to be set high, such as 200. There does not seem to be a downside in setting the resistance too

high, although it might not be biologically realistic. For NDRAM and NDRBAM, larger δ values (δ =1) work. The results for δ levels and the spiking Hopfield networks are more complex because while best results happened at lowest value, the overall best performance across all parameters is at the highest level. This is probably due to simulation run time, therefore we suggest using the lower δ values with longer run times. Optimal results for the rheobase values are not as clear. While the high values (2, 4, 6) work better than low (0.2, 0.4, 0.6), it is unclear if it is strictly higher is better. At high resistance, all rheobase value results become equal. The resting and critical values do not have much effect with these parameters tested, although it would be recommended to set resting and critical values to -1 and 0, at least for NDRAM and NDRBAM, to keep them tied to the original NSNN. Keeping the resting and critical values the same as the original transmission function parameters allows the network to be quickly and easily switched between from spiking to NSNN.

2.7.3 Discussion

Most of the main effects of the parameters are expected. Increasing the resistance value increases how often the neuron spikes which increases the accuracy of the network. The rheobase affects the voltage needed to spike; a larger rheobase means the voltage needs to be larger to induce a spike, reducing the spike rate and therefore accuracy. The simulations did not find any substantial effect of the resting and critical values but this could be because the difference between the two options used is too small. The most interesting findings are the interactions between the parameters studied and the neural models and networks.

2.7.3.1 Neural Model Comparisons

The LIF neural model is the least susceptible to rheobase and δ changes. Two reasons for this are: how the LIF is implemented and issues with the LIF as an excitatory neuron that does

not actually spike. The LIF is a logarithmic function where the rheobase and resistance are tied together. The rheobases and resistance are mapped to the same variable in the equation which may explain why the network reacts to the resistance differently than the other two neural models. Additionally, the lack of an explicit spiking mechanism can cause uncommon responses to input (Badel et al., 2008), which might be what is being reflected in the NDRBAM network (Fig. 18).

QIF and CIF are similar to each other and this is reflected in how they perform with the different variables. Overall, there is a smaller difference between neuron models in the Hopfield network than the other two networks. This difference implies that the Hopfield network is not as susceptible to changes in neuron models, possibly because those neuron models are not related to the Hopfield's original NSNN form. The minor differences in neuron models are amplified in NDRAM and NDRBAM, probably because these models are more similar to the original functions and therefore the slight differences produce larger effects.

2.7.3.2 Neural Network Comparisons

The spiking Hopfield networks perform poorly regardless of the parameters although there are signs of improvements. For many of the trials, the spiking Hopfield networks only recalls one of the two learned patterns no matter what patterns are used as input. But with the lowest δ used and with high resistance values, the network is able to overcome this to recall correctly over 50% of the time, therefore it has to at least occasionally be able to recall both learned patterns correctly. The Hopfield network also shows signs of continued improvement after the external input is removed with small time steps. It is possible that if the simulation ran for longer and had a lower δ value the task dependency issues found in simulation 1 could be removed. While the Hopfield network produces results for many different parameters, producing

good results is much more difficult and requires precise parameter selection, thus the replicating issues found in Wärnberg (2014) and confirming the that parameter selection needs to be done carefully for the Hopfield network (Maass & Natschläger, 1998).

Because NDRAM and NDRBAM have the same transmission function, the difference in architecture is the only thing that accounts for the differences in performance. The NDRBAM has two layers that pass information back and forth, with only one layer receiving noisy external input compared to single layer NDRAM. This change is significant enough to produce different internal patterns between NDRAM and NDRBAM and thus producing different results.

The NDRBAM-LIF network is a special case that produces results when the δ value is small which is implies there is an interaction between both the uniqueness of LIF and the architecture of NDRBAM. In general, the parameter selection has smaller influences on the LIF network than CIF or QIF, and on Hopfield than NDRAM and NDRBAM.

Overall, it appears that both the architecture and the transmission function/neural model relationship play a role in network performance. Hopfield's network is dissimilar in transmission function and has the overall most dissimilar results of the three networks while the three neuron models within Hopfield perform the most alike. This implies that there is a connection between the transmission function and the neuron model; specifically, that when they are not tied, there is less effect of parameters and differences. NDRAM and NDRBAM have the transmission function tied into the learning rule, therefore the neuron models may have a greater connection to this learning rule than Hopfield's which may also account for some performance differences. Hopfield does not have significant neural model differences, NDRAM and NDRBAM do, with better performance for neural models that match the transmission function. This implies that the grounding between learning and recall is better preserved by the similar neuron model.

Parameter selection is also easier when matching neuron model and transmission function. NDRAM and NDRBAM work well when the neuron model parameter selection is based on transmission function, but for the Hopfield network parameters will need to be optimized carefully to get best performance; something that is not achieved here.

2.8 Conclusion

Grounding is the theory that we associate symbols with meaning learned through our experiences with the world. It is encoded in our brain via our neurons and how they function. This process of grounding data is translated into neural networks though their architecture, encoded via the learning rule, and decoded via the transmission function. But as we progress from NSNNs to SNNs, we are modifying those networks without considering how the different components interact to produce the learned material.

This paper looks at how converting the transmission function to a neuron model affects the network's ability to recall. Specifically, it looks at how the relationship between the original transmission function and the new neuron model impacts recall and how the parameters used in the neuron model affect grounding. It does this by introducing a new neuron model, the CIF, which is derived from the transmission function of the NDRAM/NDRBAM. It then compares the Hopfield, NDRAM, and NDRBAM networks that are converted to spiking networks by replacing the nodes with either a LIF, QIF, or CIF neuron model. Parameters are subsequently changed to observe how they affect the neuron model and the overall accuracy of the SNNs.

While ultimately it can always be argued that the parameters of the neuron play a huge rule in recall accuracy, we show that having a neuron model related to the transmission function makes a difference, although it does not have to be tied directly to the transmission function like CIF is. Neuron models similar to the transmission function allow for easier parameter selection

and a generally better performance. This effect of the neuron model is particularly noticeable with the fact that the CIF and QIF, nearly identical models, perform identically for the Hopfield network and produce similar results as the LIF Hopfield network. But in NDRAM and NDRBAM, there are differences between CIF and QIF and those two neural models produce better results than LIF. Simulation 3 has the most noticeable neuron model differences, producing the clearest case for the importance of neuron model-transmission function interaction. Simulation 3 also shows that parameter selection can be used to cover bad neuron model selections, although parameter selection needs to be carefully optimized in this case.

One of the more interesting findings is how poorly Hopfield does, both because of the task dependence found in simulation 1, and the fact that with the majority of parameters, generally only 1 pattern can be correctly recalled as observed in simulation 3. Finding parameters that work when there is no starting point, such as a transmission function, may be difficult. Starting with networks that have a transmission function that is easy to convert to neuron models may quickly produce spiking neural networks that can compete with the NSNNs currently in use.

2.8.1 Future Work

How the external input is learned can have an impact on recall. Research in SCNN has found that training the NSNN with dirty input improved the SCNN performance (Hunsberger & Eliasmith, 2015; Wu et al., 2018). It has also been found that within NDRBAM, using an expanded dataset that contains noisy images of items to be recalled improved performance (Johnson & Chartier, 2014). Therefore, future work should look at the role that training with noise has on the spiking NDRAM/NDRBAM.

Another issue with data is the conversion from non-spiking to spiking. The above simulations could be redone using different methods of data transformation to see how that affects the results. Once learning is also incorporated, datasets that take advantage of both time and rate should be used to better understand how the network, and therefore the brain, grounds data.

SNNs will never reach their full potential of being able to deal with temporal data until they can learn via spikes. Preliminary work has already begun on this problem; in Johnson (2017) the learning rule in the NSNN is modified to show that it can successfully model spike-time dependent learning. The next step is to take both the learning rule and neural model to build a fully functioning SNN. While this approach is similar to Brea et al. (2013) for matching learning and the neuron model, there are some key differences. In Brea et al. (2013), they proposed a generic learning rule that would need to be matched via parameter selection to whatever neuron model is also used in the SNN. There is no relationship between the new SNN and any previous SNN and NSNN. Our proposal would match a NSNN, specifically the NDRAM/NDRBAM to the SNN. It is known that in the NDRAM/NDRBAM this combination of learning and transmission are effective, therefore there is a base expectation that it can be easily converted to a spiking network. Additionally, creating an SNN derived from a NSNN should eliminate some of the potential parameter selection errors since both are tied back to known NDRBAM parameters which we have shown here makes parameter selection easier.

The CIF neuron used should be investigated further as well. The second stable fixed point could potentially have important ramifications to how the neuron model works. As a stable fixed point, it essentially produces a maximum voltage that the neuron model can achieve. Adjusting this point closer/farther from the middle unstable fixed point adjusts both curve of the function

and the spike voltage after passing the critical threshold. How these properties affect the network needs to be studied. In SCNN, having a ASN model has proven to improve network accuracy and efficiency (Zambrano et al., 2017; Zambrano & Bohte, 2016) so it would be interesting to see if this point can be used to duplicate some of those benefits.

CHAPTER 3: GENERAL DISCUSSION

3.1 Discussion

Unfortunately, there are still a lot of unknowns about how the brain works, and subsequently, how SNNs can be built and optimized to achieve the theoretical benefits that mathematical analysis has found. What is known is that NSNN are just abstractions, or simplifications, of SNNs. By focusing on creating all parts of an SNN from a non-spiking network and studying how the changes between spiking and non-spiking affect recall, we are presenting a unique means of studying and producing SNNs in hopes of providing more functional SNNs, and eventually answers to some of the unknowns. Converting an analog network into a spiking network is not new (He et al., 2019; Hunsberger & Eliasmith, 2015; Rueckauer et al., 2016), but the in-depth study of how specific neural models combined with different networks, and subsequently the parameters of the neural model, affect the network, is new. This in-depth study is combined with a new neural model based on the NDRBAM transmission function, allowing the new network to be able to switch back and forth from non-spiking to spiking with minimal loss of information. The future plan is to create an efficient and powerful spiking neural network that can surpass NSNN in hetero-associative recall tasks, both for efficiency, and for complexity of the tasks.

Chapter 2, “The Importance of Neural Model and Parameter Selection”, shows that having a neural model that is related to the transmission function allows for better recall in the neural network. Presently, most networks that convert from a non-spiking to spiking ANN have tried to keep the dynamics between the transmission function and the neural model the same, but this is the first clear comparison of neuron models and their overall effects on the network. Interestingly, because both the CIF and QIF have similar performance overall, there does not

seem to be a need for an exact duplicate of the transmission function for the neural model, but they should be similar. There needs to be similarity of neural model and transmission function, otherwise LIF should have reached the same performance levels as CIF and QIF in the NDRAM networks. The fact that the Hopfield network does not have the same difference in performance between QIF/CIF and LIF also indicates the importance of matching neural model with transmission function, otherwise benefits of those neuron models can be lost.

Good parameter selection can overcome a lot of issues due to poor neuron model choice, and with optimal parameters, LIF still can perform quite well. Ideal parameter selection may make the LIF firing patterns similar to QIF/CIF since they are all excitatory, class I neurons and therefore there is a lot of similarity between them. The main difference though is the ease in finding good working parameters for similar (QIF/CIF) neurons versus dissimilar (LIF) neurons for matching the transmission function.

What is more interesting is that LIF is less susceptible to changes in parameter selection while QIF and CIF, despite being more susceptible, are susceptible in ways that improve performance. More often, the CIF and QIF neuron models outperformed the LIF model with different parameters. When CIF/QIF did not show a better performance than LIF, generally all three networks performed poorly, and the results would not have been acceptable anyway.

The difference in performance between the NDRBAM/NDRAM and the Hopfield network is also quite interesting. The overall performance of the Hopfield is very poor while NDRBAM/NDRAM performance is comparable to the non-spiking versions. This difference in performance between Hopfield and NDRAM implies that the relation between the transmission function and the neuron model is a significant influence and that it is not just the different neuron models alone. There is a relationship between CIF and QIF and the NDRAM and NDRBAM that

does not exist with the Hopfield network, and this is shown by performance differences between all three neuron models.

Learning and the relationship between learning and the transmission function is probably a key reason why the relationship between the neuron model and the transmission function is so important since no modifications were done to the weight matrices after learning occurred in the NSNN. The possible importance of learning implies that to maximize the relationship between learning and the neuron model, the learning rule needs to be tied to the neuron model, or both should be derived from the same source, in our case, the transmission function.

3.2 Future Work

In the paper presented here, we deal with creating an SNN, ignoring the problem of learning. There has been preliminary work done on learning in (M. Johnson & Chartier, 2017) where we create a learning rule derived from the original NDRBAM learning rule. What is needed next, and is being studied now, is the combination of those two projects to create one SNN that uses the derived learning rule and neural model. To test the network, input patterns also need to extend beyond the common input that is used in the NSNN so that spike time is incorporated. This has been one of the biggest drawbacks of SNNs so far -- testing is still comparing with input patterns that analog networks have been perfected. SNNs are theoretically capable of more complex input due to the added dimensionality of time and therefore need their own input patterns to be tested on.

It would be interesting to study if the incorporation of transmission and learning rule is needed for optimal SNNs. It is logical that there needs to be a direct tie for spiking because there cannot be a change in neuron models between learning and recall, but the exact effect of transmission function on learning is unknown.

Implementing the bidirectional SNN in a robotic framework that is capable of interacting with its environment will farther the issue of learning symbols and their meaning (grounding) and determining if and how meaning is understood in an agent with artificial intelligence.

3.3 Conclusion

Spiking neural networks are complicated. They are complicated because there are still many unknowns in how they work, including how they learn (Brea et al., 2013; Taherkhani et al., 2020; Yu et al., 2014) and how spikes are used (Boerlin & Denève, 2011; Gerstner & Hemmen, 1994; Kasabov, 2010). They are complicated because there are many features in neurons (Izhikevich, 2004), network connectivity (Moradi et al., 2013; Pastorelli et al., 2018), and even what input data is used (Mohemmed et al., 2013), that can affect their performance. They are complicated because there are many different models of neurons and learning that have been developed individually that need to be matched dynamically to create a functioning network.

This thesis breaks up the complexity of SNNs by developing a spiking neural network though the basic premise of modifying an existing, working, NSNN. This creation of an SNN was done by analyzing the NDRBAM network, a bidirectional associative memory neural network with a cubic transmission function and show how its transmission function can be adapted for spiking.

The transmission function of the NDRBAM can easily be modified into a Type I excitatory neural model, the CIF, which is similar to the QIF. This allows the network to easily switch between spiking and none spiking with minimal loss of information. In studying the conversion of the transmission function to a neuron model, we have shown the effect that a simple neuron model change can have on SNNs. Specifically, by keeping the neuron model

similar to the transmission function, the SNN produces more accurate recall results without the parameter selection issues that have been a problem in the past (Maass & Natschläger, 1998; Wärnberg, 2014). This is one of only a few studies that directly compares the effects that simple neuron models have on recall in associative memory neural networks.

There is still a lot of work to be done on SNNs, particularly on spiking bidirectional neural networks. This thesis shows important starting steps in creating such a network, and more importantly, how unique proprieties of both the neuron model and learning rule can impact the SNN. By modifying the NDRBAM, we attempted to minimize the loss of functionality done by the conversion and create a network that can readily handle both spiking and non-spiking data. At present, learning and recall are disjointed, so the next step is to create a network where both learning and recall are done via the SNN.

REFERENCES

Abbott, L. F. (1999). Lapicque's introduction of the integrate-and-fire model neuron (1907). *Brain Research Bulletin*, *50*(5–6), 303–304. https://doi.org/10.1016/S0361-9230(99)00161-6

Acevedo-Mosqueda, M. E., Yáñez-Márquez, C., & Acevedo-Mosqueda, M. A. (2013). Bidirectional associative memories: Different approaches. *ACM Computing Surveys (CSUR)*, *45*(2), 18. https://doi.org/10.1145/2431211.2431217

Adrian, E. D., & Zotterman, Y. (1926). The impulses produced by sensory nerve-endings: Part II. The response of a Single End-Organ. *The Journal of Physiology*, *61*(2), 151. https://doi.org/10.1113/jphysiol.1926.sp002281

Badel, L., Lefort, S., Berger, T. K., Petersen, C. C. H., Gerstner, W., & Richardson, M. J. E. (2008). Extracting non-linear integrate-and-fire models from experimental data using dynamic I–V curves. *Biological Cybernetics*, *99*(4–5), 361–370. https://doi.org/10.1007/s00422-008-0259-4

Barsalou, L. W. (1999). Perceptual symbol systems. *Behavioral and Brain Sciences*, *22*(4), 577–660. https://doi.org/10.1017/s0140525x99002149

Barsalou, L. W. (2008). Grounded cognition. *Annu. Rev. Psychol.*, *59*, 617–645. https://doi.org/10.1146/annurev.psych.59.103006.093639

Bear, M. F. (1999). Homosynaptic long-term depression: A mechanism for memory? *Proceedings of the National Academy of Sciences*, *96*(17), 9457–9458. https://doi.org/10.1073/pnas.96.17.9457

Belatreche, A., Maguire, L. P., Mcginnity, M., & Wu, Q. X. (2006). Evolutionary Design of Spiking Neural Networks. *New Mathematics & Natural Computation*, *2*(3), 237–253. https://doi.org/10.1142/S179300570600049X

Bhatia, S. (2016). The dynamics of bidirectional thought. *Thinking & Reasoning*, *22*(4), 397–442. https://doi.org/10.1080/13546783.2016.1187205

Bi, G., & Poo, M. (1998). Synaptic modifications in cultured hippocampal neurons: Dependence on spike timing, synaptic strength, and postsynaptic cell type. *The Journal of Neuroscience*, *18*(24), 10464–10472. https://doi.org/10.1523/JNEUROSCI.18-24-10464.1998

Bialek, W., Rieke, F., Steveninck, R. de R. van, & Warland, D. (1991). Reading a neural code. *Science*, *252*(5014), 1854–1857. https://doi.org/10.1126/science.2063199

Binder, J. R., Conant, L. L., Humphries, C. J., Fernandino, L., Simons, S. B., Aguilar, M., & Desai, R. H. (2016). Toward a brain-based componential semantic representation. *Cognitive Neuropsychology*, *33*(3–4), 130–174. https://doi.org/10.1080/02643294.2016.1147426

Bliss, T. V., & Gardner-Medwin, A. R. (1973). Long-lasting potentiation of synaptic transmission in the dentate area of the unanaesthetized rabbit following stimulation of the perforant path. *The Journal of Physiology*, *232*(2), 357. https://doi.org/10.1113/jphysiol.1973.sp010274

Bliss, T. V. P., & Lømo, T. (1973). Long-lasting potentiation of synaptic transmission in the dentate area of the anaesthetized rabbit following stimulation of the perforant path. *The Journal of Physiology*, *232*(2), 331–356. https://doi.org/10.1113/jphysiol.1973.sp010273

Boerlin, M., & Denève, S. (2011). Spike-Based Population Coding and Working Memory (Spike-Based Population Coding and Working Memory). *PLoS Computational Biology*, *7*(2), e1001080. https://doi.org/10.1371/journal.pcbi.1001080

Bottou, L. (2017). Foreword. In *Perceptrons: An Introduction to Computational Geometry*. The MIT press. http://direct.mit.edu/books/book/3132/chapter/85832/Foreword

Brea, J., Senn, W., & Pfister, J.-P. (2013). Matching Recall and Storage in Sequence Learning with Spiking Neural Networks. *The Journal of Neuroscience*, *33*(23), 9565–9575. https://doi.org/10.1523/JNEUROSCI.4098-12.2013

Brette, R. (2015). Philosophy of the Spike: Rate-Based vs. Spike-Based Theories of the Brain. *Frontiers in Systems Neuroscience*, *9*. https://doi.org/10.3389/fnsys.2015.00151

Cao, Y., Chen, Y., & Khosla, D. (2015). Spiking deep convolutional neural networks for energy-efficient object recognition. *International Journal of Computer Vision*, *113*(1), 54–66. https://doi.org/10.1007/s11263-014-0788-3

Caporale, N., & Dan, Y. (2008). Spike Timing–Dependent Plasticity: A Hebbian Learning Rule. *Annual Review of Neuroscience*, *31*, 25–46. https://doi.org/10.1146/annurev.neuro.31.060407.125639

Carandini, M., & Heeger, D. J. (2012). Normalization as a canonical neural computation. *Nature Reviews Neuroscience*, *13*(1), 51–62. https://doi.org/10.1038/nrn3136

Chan, W., Jaitly, N., Le, Q., & Vinyals, O. (2016). Listen, attend and spell: A neural network for large vocabulary conversational speech recognition. *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 4960–4964. https://doi.org/10.1109/ICASSP.2016.7472621

Chartier, S., Helie, S., Boukadoum, M., & Proulx, R. (2005). SCRAM: Statistically converging recurrent associative memory. *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.*, *2*, 723–728. https://doi.org/10.1109/IJCNN.2005.1555941

Chartier, S., & Proulx, R. (2001). A new online unsupervised learning rule for the BSB model. *IJCNN'01. International Joint Conference on Neural Networks. Proceedings (Cat. No.01CH37222)*, *1*, 448–453. https://doi.org/10.1109/IJCNN.2001.939061

Chartier, Sylvain, & Boukadoum, M. (2006). A bidirectional heteroassociative memory for binary and grey-level patterns. *Neural Networks, IEEE Transactions On*, *17*(2), 385–396. https://doi.org/10.1109/TNN.2005.863420

Chartier, Sylvain, & Boukadoum, M. (2011). Encoding Static and Temporal Patterns with a Bidirectional Heteroassociatie Memory. *Journal of Applied Mathematics*, *2011*. https://doi.org/10.1155/2011/301204

Chartier, Sylvain, & Proulx, R. (2005). NDRAM: Nonlinear Dynamic Recurrent Associative Memory for Learning Bipolar and Nonbipolar Correlated Patterns. *IEEE Transactions on Neural Networks*, *16*(6), 1393–1400. https://doi.org/10.1109/TNN.2005.852861

Chartier, Sylvain, Renaud, P., & Boukadoum, M. (2008). A nonlinear dynamic artificial neural network model of memory. *New Ideas in Psychology*, *26*(2), 252–277. https://doi.org/10.1016/j.newideapsych.2007.07.005

Chirimuuta, M. (2014). Minimal models and canonical neural computations: The distinctness of computational explanation in neuroscience. *Synthese*, *191*(2), 127–153. https://doi.org/10.1007/s11229-013-0369-y

Coley, C. W., Jin, W., Rogers, L., Jamison, T. F., Jaakkola, T. S., Green, W. H., Barzilay, R., & Jensen, K. F. (2019). A graph-convolutional neural network model for the prediction of

chemical reactivity. *Chemical Science*, *10*(2), 370–377. https://doi.org/10.1039/C8SC04228D

Coşkun, M., Uçar, A., Yildirim, Ö., & Demir, Y. (2017). Face recognition based on convolutional neural network. *2017 International Conference on Modern Electrical and Energy Systems (MEES)*, 376–379. https://doi.org/10.1109/MEES.2017.8248937

Cui, Z., Ke, R., Pu, Z., & Wang, Y. (2018). Deep bidirectional and unidirectional LSTM recurrent neural network for network-wide traffic speed prediction. *ArXiv Preprint ArXiv:1801.02143*.

DasGupta, B., & Schnitger, G. (1994). The power of approximating: A comparison of activation functions. *MATHEMATICAL RESEARCH*, *79*, 641–641. https://doi.org/10.1.1.52.263

Diederich, N., Bartsch, T., Kohlstedt, H., & Ziegler, M. (2018). A memristive plasticity model of voltage-based STDP suitable for recurrent bidirectional neural networks in the hippocampus. *Scientific Reports*, *8*(1), 1–12. https://doi.org/10.1038/s41598-018-27616-6

Diehl, P. U., Neil, D., Binas, J., Cook, M., Liu, S.-C., & Pfeiffer, M. (2015). Fast-classifying, high-accuracy spiking deep networks through weight and threshold balancing. *2015 International Joint Conference on Neural Networks (IJCNN)*, 1–8. https://doi.org/10.1109/ijcnn.2015.7280696

Ding, H., & Chan, S. P. (1996). A real-time planning algorithm for obstacle avoidance of redundant robots. *Journal of Intelligent and Robotic Systems*, *16*(3), 229–243. https://doi.org/10.1007/bf00245422

Douglas, R. M., & Goddard, G. V. (1975). Long-term potentiation of the perforant path-granule cell synapse in the rat hippocampus. *Brain Research*, *86*(2), 205–215. https://doi.org/10.1016/0006-8993(75)90697-6

Dreyer, F. R., & Pulvermüller, F. (2018). Abstract semantics in the motor system?–An event-related fMRI study on passive reading of semantic word categories carrying abstract emotional and mental meaning. *Cortex*, *100*, 52–70. https://doi.org/10.1016/j.cortex.2017.10.021

El_Jerjawi, N. S., & Abu-Naser, S. S. (2018). Diabetes prediction using artificial neural network. *International Journal of Advanced Science and Technology*, *121*, 54–64.

Emmerson, M. D., Damper, R. I., Hey, A. J. G., & Upstill, C. (1991). Fault tolerance and redundancy of neural nets for the classification of acoustic data. *Acoustics, Speech, and Signal Processing, 1991. ICASSP-91., 1991 International Conference On*, 1053–1056. https://doi.org/10.1109/icassp.1991.150529

Er, M. J., Wu, S., Lu, J., & Toh, H. L. (2002). Face recognition with radial basis function (RBF) neural networks. *IEEE Transactions on Neural Networks*, *13*(3), 697–710. https://doi.org/10.1109/tnn.2002.1000134

Ermentrout, B. (1996). Type I membranes, phase resetting curves, and synchrony. *Neural Computation*, *8*(5), 979–1001. https://doi.org/10.1162/neco.1996.8.5.979

Folowosele, F., Vogelstein, R. J., & Etienne-Cummings, R. (2011). Towards a cortical prosthesis: Implementing a spike-based hmax model of visual object recognition in silico. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, *1*(4), 516–525. https://doi.org/doi.org/10.1109/jetcas.2012.2183409

Gerstner, W. (2002). *Spiking neuron models: Single neurons, populations, plasticity*. Cambridge University Press. https:// doi.org/doi.org/10.1017/CBO9780511815706

Gerstner, W., & Hemmen, J. L. van. (1994). How to Describe Neuronal Activity: Spikes, Rates, or Assemblies? In J. D. Cowan, G. Tesauro, & J. Alspector (Eds.), *Advances in Neural*

Information Processing Systems 6 (pp. 463–470). Morgan-Kaufmann. http://papers.nips.cc/paper/850-how-to-describe-neuronal-activity-spikes-rates-or-assemblies.pdf

Gerstner, W., & Kistler, W. M. (2002). Mathematical formulations of Hebbian learning. *Biological Cybernetics*, *87*(5–6), 404–415. https://doi.org/10.1007/s00422-002-0353-y

Gerstner, W., Kistler, W. M., Naud, R., & Paninski, L. (2014). *Neuronal dynamics: From single neurons to networks and models of cognition*. Cambridge University Press. https://doi.org/10.1017/CBO9781107447615

Gerstner, W., & Naud, R. (2009). How Good Are Neuron Models? *Science*, *326*(5951), 379–380. https://doi.org/10.1126/science.1181936

Ghosh-Dastidar, S., & Adeli, H. (2009). Third Generation Neural Networks: Spiking Neural Networks. In *Advances in Computational Intelligence* (Vol. 116, pp. 167–178). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-03156-4

Guan, Z., Giustetto, M., Lomvardas, S., Kim, J.-H., Miniaci, M. C., Schwartz, J. H., Thanos, D., & Kandel, E. R. (2002). Integration of Long-Term-Memory-Related Synaptic Plasticity Involves Bidirectional Regulation of Gene Expression and Chromatin Structure. *Cell*, *111*(4), 483–493. https://doi.org/10.1016/S0092-8674(02)01074-7

Guo, H., Tang, R., Ye, Y., Li, Z., & He, X. (2017). DeepFM: A factorization-machine based neural network for CTR prediction. *ArXiv Preprint ArXiv:1703.04247*.

He, H., Shang, Y., Yang, X., Di, Y., Lin, J., Zhu, Y., Zheng, W., Zhao, J., Ji, M., & Dong, L. (2019). Constructing an Associative Memory System Using Spiking Neural Network. *Frontiers in Neuroscience*, *13*, 650. https://doi.org/10.3389/fnins.2019.00650

Hebb, D. O. (1949). *The organization of behavior: A neuropsychological theory*. Wiley.

Hermann, K. M., Hill, F., Green, S., Wang, F., Faulkner, R., Soyer, H., Szepesvari, D., Czarnecki, W. M., Jaderberg, M., Teplyashin, D., Wainwright, M., Apps, C., Hassabis, D., & Blunsom, P. (2017). Grounded Language Learning in a Simulated 3D World. *ArXiv:1706.06551 [Cs, Stat]*. http://arxiv.org/abs/1706.06551

Hodgkin, A. L., & Huxley, A. F. (1952a). Propagation of electrical signals along giant nerve fibres. *Proceedings of the Royal Society of London. Series B, Biological Sciences*, 177–183. https://doi.org/10.1098/rspb.1952.0054

Hodgkin, A. L., & Huxley, A. F. (1952b). The components of membrane conductance in the giant axon of Loligo. *The Journal of Physiology*, *116*(4), 473. https://doi.org/10.1113/jphysiol.1952.sp004718

Hodgkin, A. L., & Huxley, A. F. (1952c). Movement of Sodium and Potassium Ions During Nervous Activity. *Cold Spring Harbor Symposia on Quantitative Biology*, *17*, 43–52. https://doi.org/10.1101/SQB.1952.017.01.007

Hodgkin, Alan L., & Huxley, A. F. (1952). A quantitative description of membrane current and its application to conduction and excitation in nerve. *The Journal of Physiology*, *117*(4), 500–544. https://doi.org/10.1113/jphysiol.1952.sp004764

Hodgkin, Allan L., & Huxley, A. F. (1952). The dual effect of membrane potential on sodium conductance in the giant axon of Loligo. *The Journal of Physiology*, *116*(4), 497–506. https://doi.org/10.1113/jphysiol.1952.sp004719

Hodgkin, Ao L., Huxley, A. F., & Katz, B. (1952). Measurement of current-voltage relations in the membrane of the giant axon of Loligo. *The Journal of Physiology*, *116*(4), 424. https://doi.org/10.1113/jphysiol.1952.sp004716

Hopfield, J. J. (1982). Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences*, *79*(8), 2554–2558. https://doi.org/10.1073/pnas.79.8.2554

Hopfield, J. J. (1984). Neurons with graded response have collective computational properties like those of two-state neurons. *Proceedings of the National Academy of Sciences*, *81*(10), 3088–3092.

Hopfield, J., & Tank, D. (1985). "Neural" computation of decisions in optimization problems. *Biological Cybernetics*, *52*(3), 141–152. https://doi.org/10.1007/BF00339943

Hoppensteadt, F., & Izhikevich, E. (2002). Canonical Neural Models. In *Brain Theory and Neural Networks* (2nd ed., pp. 181–186). The MIT press.

Hunsberger, E., & Eliasmith, C. (2015). Spiking deep networks with LIF neurons. *ArXiv Preprint ArXiv:1510.08829*.

Hunsberger, E., & Eliasmith, C. (2016). Training Spiking Deep Networks for Neuromorphic Hardware. *ArXiv:1611.05141 [Cs]*. https://doi.org/10.13140/RG.2.2.10967.06566

Hunter, R., Cobb, S., & Graham, B. P. (2008). Improving associative memory in a network of spiking neurons. *International Conference on Artificial Neural Networks*, 636–645. https://link.springer.com/chapter/10.1007/978-3-540-87559-8_66

Hurowitz, A. (2020). *Value and choice encoding via prefrontal neuron firing rate and spike timing* [PhD Thesis]. Icahn School of Medicine at Mount Sinai.

Ishiguro, A., Furuhashi, T., Okuma, S., & Uchikawa, Y. (1992). A neural network compensator for uncertainties of robotics manipulators. *IEEE Transactions on Industrial Electronics*, *39*(6), 565–570. https://doi.org/10.1109/41.170976

Izhikevich, E. M. (2003). Simple model of spiking neurons. *Neural Networks, IEEE Transactions On*, *14*(6), 1569–1572. https://doi.org/10.1109/TNN.2003.820440

Izhikevich, Eugene M. (2004). Which model to use for cortical spiking neurons? *IEEE Transactions on Neural Networks*, *15*(5), 1063–1070. https://doi.org/10.1109/tnn.2004.832719

Izhikevich, Eugene M. (2007). *Dynamical systems in neuroscience*. MIT press. https://doi.org/10.7551/mitpress/2526.001.0001

Jain, A. K., Jianchang Mao, & Mohiuddin, K. M. (1996). Artificial neural networks: A tutorial. *Computer*, *29*(3), 31–44. https://doi.org/10.1109/2.485891

Johnson, M., & Chartier, S. (2014). Increasing Accuracy in a Bidirectional Associative Memory through Expended Databases. In *Artificial General Intelligence* (pp. 53–62). Springer.

Johnson, M., & Chartier, S. (2017). Model Derived Spike Time Dependent Plasticity. In A. Lintas, S. Rovetta, P. F. M. J. Verschure, & A. E. P. Villa (Eds.), *Artificial Neural Networks and Machine Learning – ICANN 2017* (Vol. 10613, pp. 345–353). Springer International Publishing. https://doi.org/10.1007/978-3-319-68600-4_40

Johnson, M. G. (2017, May 18). *Modified Cubic Integrate and Fire Neural Model with Biologically Accurate Spike Timing* [Poster]. Interdisciplinary Conference in Psychology, University of Ottawa. https://gallery.mailchimp.com/ea333fe97679aed43cf767575/files/9377f7b9-3cdf-496d-82c4-6e704d4d3cc6/ICP_CIP_2017_Program_book.pdf

Johnson, M. G., & Chartier, S. (2017). Spike neural models Part I: The Hodgkin-Huxley model. *The Quantitative Methods for Psychology*, *13*(2), 105–119. https://doi.org/10.20982/tqmp.13.2.p105

Johnson, M. G., & Chartier, S. (2018). Spike neural models part II: Abstract neural models. *The Quantitative Methods for Psychology*, *14*(1), 1–16. https://doi.org/10.20982/tqmp.14.1.p001

Kaplan, D. M., & Craver, C. F. (2011). The Explanatory Force of Dynamical and Mathematical Models in Neuroscience: A Mechanistic Perspective. *Philosophy of Science*, *78*(4), 601–627. https://doi.org/10.1086/661755

Kasabov, N. (2010). To spike or not to spike: A probabilistic spiking neuron model. *Neural Networks*, *23*(1), 16–19. https://doi.org/10.1016/j.neunet.2009.08.010

Kasabov, N., Dhoble, K., Nuntalid, N., & Indiveri, G. (2012). Dynamic evolving spiking neural networks for on-line spatio- and spectro-temporal pattern recognition. *Neural Networks*. https://doi.org/10.1016/j.neunet.2012.11.014

Kawato, M., Uno, Y., Isobe, M., & Suzuki, R. (1988). Hierarchical neural network model for voluntary movement with application to robotics. *IEEE Control Systems Magazine*, *8*(2), 8–15. https://doi.org/10.1109/37.1867

Keysers, C., & Gazzola, V. (2014). Hebbian learning and predictive mirror neurons for actions, sensations and emotions. *Philosophical Transactions of the Royal Society B: Biological Sciences*, *369*(1644). https://doi.org/10.1098/rstb.2013.0175

Khashman, A. (2009). Application of an emotional neural network to facial recognition. *Neural Computing and Applications*, *18*(4), 309–320. https://doi.org/10.1007/s00521-008-0212-4

Khorasani, K., Cuffaro, A., & Grigoriu, T. (1994). A new learning algorithm for bidirectional associative memory neural networks. *, 1994 IEEE International Conference on Neural Networks, 1994. IEEE World Congress on Computational Intelligence*, *2*, 1115–1120 vol.2. https://doi.org/10.1109/ICNN.1994.374339

Kiefer, M., & Pulvermüller, F. (2012). Conceptual representations in mind and brain: Theoretical developments, current evidence and future directions. *Cortex*, *48*(7), 805–825. https://doi.org/10.1016/j.cortex.2011.04.006

Kiela, D., Conneau, A., Jabri, A., & Nickel, M. (2017). Learning Visually Grounded Sentence Representations. *ArXiv:1707.06320 [Cs]*. https://doi.org/10.18653/v1/N18-1038

Kistler, W. M., Gerstner, W., & Hemmen, J. L. van. (1997). Reduction of the Hodgkin-Huxley equations to a single-variable threshold model. *Neural Computation*, *9*(5), 1015–1045. https://doi.org/10.1162/neco.1997.9.5.1015

Kosko, B. (1988). Bidirectional associative memories. *Systems, Man and Cybernetics, IEEE Transactions On*, *18*(1), 49–60. https://doi.org/10.1364/ao.26.004947

Kostal, L., Lansky, P., & Rospars, J.-P. (2007). Neuronal coding and spiking randomness. *European Journal of Neuroscience*, *26*(10), 2693–2701.

Lee, S.-M., Yoon, S. M., & Cho, H. (2017). Human activity recognition from accelerometer data using Convolutional Neural Network. *2017 Ieee International Conference on Big Data and Smart Computing (Bigcomp)*, 131–134.

Legenstein, R., Chase, S. M., Schwartz, A. B., & Maass, W. (2010). A Reward-Modulated Hebbian Learning Rule Can Explain Experimentally Observed Network Reorganization in a Brain Control Task. *Journal of Neuroscience*, *30*(25), 8400–8410. https://doi.org/10.1523/JNEUROSCI.4284-09.2010

Lengyel, M., Kwag, J., Paulsen, O., & Dayan, P. (2005). Matching storage and recall: Hippocampal spike timing–dependent plasticity and phase response curves. *Nature Neuroscience*, *8*(12), 1677–1683. https://doi.org/10.1038/nn1561

Levy, A., & Bechtel, W. (2013). Abstraction and the Organization of Mechanisms. *Philosophy of Science*, *80*(2), 241–261. https://doi.org/10.1086/670300

Li, G., Tang, H., Sun, Y., Kong, J., Jiang, G., Jiang, D., Tao, B., Xu, S., & Liu, H. (2019). Hand gesture recognition based on convolution neural network. *Cluster Computing*, *22*(2), 2719–2729. https://doi.org/10.1007/s10586-017-1435-x

Liljenström, H. (Ed.). (2015). *Advances in Cognitive Neurodynamics (IV)*. Springer Netherlands. https://doi.org/10.1007/978-94-017-9548-7

London, M., Roth, A., Beeren, L., Häusser, M., & Latham, P. E. (2010). Sensitivity to perturbations in vivo implies high noise and suggests rate coding in cortex. *Nature*, *466*(7302), 123–127. https://doi.org/10.1038/nature09086

Lynch, G. S., Dunwiddie, T., & Gribkoff, V. (1977). Heterosynaptic depression: A postsynaptic correlate of long-term potentiation. *Nature*, *266*(5604), 737–739. https://doi.org/10.1038/266737a0

Ma, L., & Khorasani, K. (2004). Facial expression recognition using constructive feedforward neural networks. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, *34*(3), 1588–1595. https://doi.org/10.1109/tsmcb.2004.825930

Maass, W. (1996). Lower bounds for the computational power of networks of spiking neurons. *Neural Computation*, *8*(1), 1–40. https://doi.org/10.1162/neco.1996.8.1.1

Maass, W. (1997a). Fast sigmoidal networks via spiking neurons. *Neural Computation*, *9*(2), 279–304. https://doi.org/10.1162/neco.1997.9.2.279

Maass, W. (1997b). Networks of spiking neurons: The third generation of neural network models. *Neural Networks*, *10*(9), 1659–1671. https://doi.org/10.1016/s0893-6080(97)00011-7

Maass, W., & Natschläger, T. (1997). Networks of spiking neurons can emulate arbitrary Hopfield nets in temporal coding. *Network: Computation in Neural Systems*, *8*(4), 355–371. https://doi.org/10.1088/0954-898X_8_4_002

Maass, W., & Natschläger, T. (1998). Emulation of Hopfield networks with spiking neurons in temporal coding. In *Computational Neuroscience* (pp. 221–226). Springer. https://doi.org/10.1007/978-1-4615-4831-7_37

Malinowski, M., Rohrbach, M., & Fritz, M. (2015). Ask Your Neurons: A Neural-Based Approach to Answering Questions about Images. *2015 IEEE International Conference on Computer Vision (ICCV)*, 1–9. https://doi.org/10.1109/ICCV.2015.9

Malleret, G., Alarcon, J. M., Martel, G., Takizawa, S., Vronskaya, S., Yin, D., Chen, I. Z., Kandel, E. R., & Shumyatsky, G. P. (2010). Bidirectional Regulation of Hippocampal Long-Term Synaptic Plasticity and Its Influence on Opposing Forms of Memory. *Journal of Neuroscience*, *30*(10), 3813–3825. https://doi.org/10.1523/JNEUROSCI.1330-09.2010

Mao, Z., & Hsia, T. C. (1997). Obstacle avoidance inverse kinematics solution of redundant robots by neural networks. *Robotica*, *15*(1), 3–10. https://doi.org/10.1017/s0263574797000027

Markram, H., Gerstner, W., & Sjöström, P. J. (2011). A history of spike-timing-dependent plasticity. *Frontiers in Synaptic Neuroscience*, *3*, 4. https://doi.org/10.3389/fnsyn.2011.00004

McClelland, J. L., Rumelhart, D. E., & Group, P. R. (1986). Parallel distributed processing. *Explorations in the Microstructure of Cognition*, *2*, 216–271. https://doi.org/10.7551/mitpress/5237.001.0001

McCulloch, W. S., & Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The Bulletin of Mathematical Biophysics*, *5*(4), 115–133. https://doi.org/10.1007/bf02478259

Mei, H., Bansal, M., & Walter, M. R. (2016). Listen, Attend, and Walk: Neural Mapping of Navigational Instructions to Action Sequences. *Thirtieth AAAI Conference on Artificial Intelligence*, *30*(1), 7.

Meunier, D., & Paugam-Moisy, H. (2005). A Spiking Bidirectional Associative Memory for Modeling Intermodal Priming. *Proceedings of the IASTED International Conference on Neural Networks and Computational Intelligence*, 25–30.

Mihalaş, Ş., & Niebur, E. (2009). A Generalized Linear Integrate-and-Fire Neural Model Produces Diverse Spiking Behaviors. *Neural Computation*, *21*(3), 704–718. https://doi.org/10.1162/neco.2008.12-07-680

Minsky, M., & Papert, S. (1969). An introduction to computational geometry. *Cambridge Tiass., HIT*.

Mohemmed, A., Schliebs, S., Matsuda, S., & Kasabov, N. (2013). Training spiking neural networks to associate spatio-temporal input–output spike patterns. *Neurocomputing*, *107*, 3–10. https://doi.org/10.1016/j.neucom.2012.08.034

Moller, A. R. (1999). Review of the roles of temporal and place coding of frequency in speech discrimination. *Acta Oto-Laryngologica*, *119*(4), 424–430. https://doi.org/10.1080/00016489950180946

Moradi, S., Imam, N., Manohar, R., & Indiveri, G. (2013). A Memory-Efficient Routing Method for Large-Scale Spiking Neural Networks. In *2013 European Conference on Circuit*

Theory and Design, ECCTD 2013—Proceedings (p. 4). https://doi.org/10.1109/ECCTD.2013.6662203

Morissette, L., Chartier, S., Vandermeulen, R., & Watier, N. (2012). Depth of treatment sensitive noise resistant dynamic artificial neural networks model of recall in people with prosopagnosia. *Neural Networks*, *32*, 46–56. https://doi.org/10.1016/j.neunet.2012.02.008

Muresan, R. C., & Savin, C. (2007). Resonance or Integration? Self-Sustained Dynamics and Excitability of Neural Microcircuits. *Journal of Neurophysiology*, *97*(3), 1911–1930. https://doi.org/10.1152/jn.01043.2006

Nägerl, U. V., Eberhorn, N., Cambridge, S. B., & Bonhoeffer, T. (2004). Bidirectional Activity-Dependent Morphological Plasticity in Hippocampal Neurons. *Neuron*, *44*(5), 759–767. https://doi.org/10.1016/j.neuron.2004.11.016

Naud, R., & Sprekeler, H. (2018). Sparse bursts optimize information transmission in a multiplexed neural code. *Proceedings of the National Academy of Sciences*, *115*(27), E6329–E6338. https://doi.org/10.1073/pnas.1720995115

Olazaran, M. (1996). A Sociological Study of the Official History of the Perceptrons Controversy. *Social Studies of Science*, *26*(3), 611–659. https://dx.doi.org/10.1177/030631296026003005

Pastorelli, E., Paolucci, P. S., Simula, F., Biagioni, A., Capuani, F., Cretaro, P., De Bonis, G., Cicero, F. L., Lonardo, A., Martinelli, M., Pontisso, L., Vicini, P., & Ammendola, R. (2018). Gaussian and exponential lateral connectivity on distributed spiking neural network simulation. *2018 26th Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP)*, 658–665. https://doi.org/10.1109/PDP2018.2018.00110

Poveda, J., & Vellido, A. (2006). Neural Network Models for Language Acquisition: A Brief Survey. In E. Corchado, H. Yin, V. Botti, & C. Fyfe (Eds.), *Intelligent Data Engineering and Automated Learning – IDEAL 2006* (pp. 1346–1357). Springer. https://doi.org/10.1007/11875581_160

Pulvermüller, F. (2005). Brain mechanisms linking language and action. *Nature Reviews Neuroscience*, *6*(7), 576–582. https://doi.org/10.1038/nrn1706

Pulvermüller, F. (2018). Neurobiological mechanisms for semantic feature extraction and conceptual flexibility. *Topics in Cognitive Science*, *10*(3), 590–620. https://doi.org/10.1111/tops.12367

Pulvermüller, F., Hauk, O., Nikulin, V. V., & Ilmoniemi, R. J. (2005). Functional links between motor and language systems. *European Journal of Neuroscience*, *21*(3), 793–797. https://doi.org/10.1111/j.1460-9568.2005.03900.x

Pulvermüller, F., Shtyrov, Y., & Ilmoniemi, R. (2005). Brain signatures of meaning access in action word recognition. *Journal of Cognitive Neuroscience*, *17*(6), 884–892. https://doi.org/10.1162/0898929054021111

Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, *65*(6), 386–408. https://doi.org/10.1037/h0042519

Rueckauer, B., Lungu, I.-A., Hu, Y., & Pfeiffer, M. (2016). Theory and tools for the conversion of analog to spiking convolutional neural networks. *ArXiv Preprint ArXiv:1612.04052*.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, *323*(6088), 533–536. https://doi.org/10.1038/323533a0

Sadek, R. M., Mohammed, S. A., Abunbehan, A. R. K., Ghattas, A. K. H. A., Badawi, M. R., Mortaja, M. N., Abu-Nasser, B. S., & Abu-Naser, S. S. (2019). Parkinson's Disease Prediction Using Artificial Neural Network. *International Journal of Academic Health and Medical Research*, *3*(1), 1–8.

Sanders, H., Ji, D., Sasaki, T., Leutgeb, J. K., Wilson, M. A., & Lisman, J. E. (2019). Temporal coding and rate remapping: Representation of nonspatial information in the hippocampus. *Hippocampus*, *29*(2), 111–127. https://doi.org/10.1002/hipo.23020

Sasaki, K., Noda, K., & Ogata, T. (2016). Visual motor integration of robot's drawing behavior using recurrent neural network. *Robotics and Autonomous Systems*, *86*, 184–195. https://doi.org/10.1016/j.robot.2016.08.022

Sepandi, M., Taghdir, M., Rezaianzadeh, A., & Rahimikazerooni, S. (2018). Assessing Breast Cancer Risk with an Artificial Neural Network. *Asian Pacific Journal of Cancer Prevention : APJCP*, *19*(4), 1017–1019. https://doi.org/10.22034/APJCP.2018.19.4.1017

Shatz, C. J. (1992). The developing brain. *Scientific American*, *267*(3), 60–67. https://doi.org/10.1038/scientificamerican0992-60

Sjöström, P. J., Turrigiano, G. G., & Nelson, S. B. (2001). Rate, Timing, and Cooperativity Jointly Determine Cortical Synaptic Plasticity. *Neuron*, *32*(6), 1149–1164. https://doi.org/10.1016/S0896-6273(01)00542-6

Smith, E. H., Horga, G., Yates, M. J., Mikell, C. B., Banks, G. P., Pathak, Y. J., Schevon, C. A., McKhann, G. M., Hayden, B. Y., & Botvinick, M. M. (2019). Widespread temporal coding of cognitive control in the human prefrontal cortex. *Nature Neuroscience*, 1–9. https://doi.org/10.1038/s41593-019-0494-0

Stein, R. B., Gossen, E. R., & Jones, K. E. (2005). Neuronal variability: Noise or part of the signal? *Nature Reviews Neuroscience*, *6*(5), 389–397. https://doi.org/10.1038/nrn1668

Świetlik, D., & Białowąs, J. (2019). Application of Artificial Neural Networks to Identify Alzheimer's Disease Using Cerebral Perfusion SPECT Data. *International Journal of Environmental Research and Public Health*, *16*(7). https://doi.org/10.3390/ijerph16071303

Taherkhani, A., Belatreche, A., Li, Y., Cosma, G., Maguire, L. P., & McGinnity, T. M. (2020). A review of learning in biologically plausible spiking neural networks. *Neural Networks*, *122*, 253–272. https://doi.org/10.1016/j.neunet.2019.09.036

Tanaka, H., Morie, T., & Aihara, K. (2005). Associative Memory Operation in a Hopfield-type Spiking Neural Network with Modulation of Resting Membrane Potential. *2005 International Symposium on Nonlinear Theory and Its Applications*.

Theunissen, F., & Miller, J. P. (1995). Temporal encoding in nervous systems: A rigorous definition. *Journal of Computational Neuroscience*, *2*(2), 149–162. https://doi.org/10.1007/BF00961885

Vandermeulen, R., Morissette, L., & Chartier, S. (2011). Modeling prosopagnosia using dynamic artificial neural networks. *The 2011 International Joint Conference on Neural Networks*, 2074–2079. https://doi.org/10.1109/IJCNN.2011.6033482

VanRullen, R., Guyonneau, R., & Thorpe, S. J. (2005). Spike times make sense. *Trends in Neurosciences*, *28*(1), 1–4. https://doi.org/10.1016/j.tins.2004.10.010

Vogels, T. P., Rajan, K., & Abbott, L. F. (2005). Neural Network Dynamics. *Annual Review of Neuroscience*, *28*(1), 357–376. https://doi.org/10.1146/annurev.neuro.28.061604.135637

Wärnberg, E. (2014). *Implementation and Robustness of Hopfield Networks with Spiking Neurons* [Degree Project in Engineering Physics, First Cycle, KTH Royal Institute of Technology]. http://www.diva-portal.org/smash/record.jsf?pid=diva2:753649

Werbos, P. J. (1988). Generalization of backpropagation with application to a recurrent gas market model. *Neural Networks*, *1*(4), 339–356. https://doi.org/10.1016/0893-6080(88)90007-X

Wu, J., Chua, Y., Zhang, M., Li, H., & Tan, K. C. (2018). A spiking neural network framework for robust sound classification. *Frontiers in Neuroscience*, *12*, 836. https://doi.org/10.3389/fnins.2018.00836

Wysoski, S. G., Benuskova, L., & Kasabov, N. (2010). Evolving spiking neural networks for audiovisual information processing. *Neural Networks*, *23*(7), 819–835. https://doi.org/10.1016/j.neunet.2010.04.009

Yu, Q., Li, H., & Tan, K. C. (2018). Spike timing or rate? Neurons learn to make decisions for both through threshold-driven plasticity. *IEEE Transactions on Cybernetics*, *49*(6), 2178–2189. https://doi.org/10.1109/TCYB.2018.2821692

Yu, Q., Tang, H., Tan, K. C., & Yu, H. (2014). A brain-inspired spiking neural network model with temporal encoding and learning. *Neurocomputing*, *138*, 3–13. https://doi.org/10.1016/j.neucom.2013.06.052

Zador, A. M. (2019). A critique of pure learning and what artificial neural networks can learn from animal brains. *Nature Communications*, *10*(1), 3770. https://doi.org/10.1038/s41467-019-11786-6

Zamani, M., Sadeghian, A., & Chartier, S. (2010). A bidirectional associative memory based on cortical spiking neurons using temporal coding. *The 2010 International Joint Conference on Neural Networks (IJCNN)*, 1–8. https://doi.org/10.1109/IJCNN.2010.5596806

Zambrano, D., & Bohte, S. M. (2016). Fast and efficient asynchronous neural computation with adapting spiking neural networks. *ArXiv Preprint ArXiv:1609.02053*.

Zambrano, D., Nusselder, R., Scholte, H. S., & Bohte, S. (2017). Efficient computation in adaptive artificial spiking neural networks. *ArXiv Preprint ArXiv:1710.04838*.

Zhang, L. I., Tao, H. W., Holt, C. E., Harris, W. A., & Poo, M. (1998). A critical window for cooperation and competition among developing retinotectal synapses. *Nature*, *395*(6697), 37–44. https://doi.org/10.1038/25665

Zheng, H., Fu, J., Mei, T., & Luo, J. (2017). Learning multi-attention convolutional neural network for fine-grained image recognition. *Proceedings of the IEEE International Conference on Computer Vision*, 5209–5217. https://doi.org/10.1109/ICCV.2017.557

Zhuang, X., Huang, Y., & Chen, S.-S. (1993). Better learning for bidirectional associative memory. *Neural Networks*, *6*(8), 1131–1146. https://doi.org/10.1016/S0893-6080(09)80024-5

APPENDIX A – Non-spiking Recall Performance

The recall tasks in sections 2.5 and 2.6 for the spiking networks were also performed on the original NSNN to confirm that the initial weight matrices were correct. The same pixel flips for noise were used in the NSNN that were used for the spiking networks. Simulation 3's results from section 2.7 are not displayed because all three non-spiking networks achieved perfect accuracy.

A.1 Simulation 1

		MEMORY LOAD					
		2%		8%		14%	
		Hopfield	NDR(B)AM	Hopfield	NDR(B)AM	Hopfield	NDR(B)AM
N		100.0%	100.0%	74.8%	100.0%	32.0%	100.0%
O	0%	100.0%	100.0%	73.5%	100.0%	30.7%	99.9%
I	6%	100.0%	100.0%	71.0%	99.8%	29.4%	99.3%
S	12%	100.0%	100.0%	68.0%	99.5%	23.9%	99.3%
E	18%	100.0%	100.0%	74.8%	100.0%	32.0%	100.0%

Table A.1: Corresponding non-spiking results for Hopfield, NDRAM, and NDRBAM using the same data as spiking results in section 2.5.

A.2 Simulation 2

FIRST LAYER – LOWER CASE						SECOND LAYER – UPPER CASE					
		MEMORY LOAD						MEMORY LOAD			
N		2%	8%	14%	20%	N		2%	8%	14%	20%
O	0%	100.0%	99.7%	100.0%	100.0%	O	0%	100.0%	100.0%	100.0%	100.0%
I	6%	100.0%	99.7%	99.3%	99.5%	I	6%	100.0%	100.0%	99.3%	99.5%
S	12%	100.0%	99.7%	98.4%	95.2%	S	12%	100.0%	100.0%	98.3%	95.3%
E	18%	100.0%	97.7%	92.5%	86.2%	E	18%	100.0%	98.0%	92.5%	86.2%

Table A.2: Corresponding non-spiking results for NDRBAM, first and section layer, using the same data as spiking results in section 2.6.

APPENDIX B – Hopfield Issues

In section 2.5, the recall for the Hopfield network was below expectations, especially for single pattern recall where there should be 100% accuracy. Extra analysis was done to determine why these results were obtained.

It appears that the spiking Hopfield is displaying task dependency. Specifically, when the input had more spiking nodes then non-spiking, the network recalled the inverse of that pattern. The letters d, g, k, h, m, and w all have more spiking nodes than non-spiking ones and these are the letters that always failed in Hopfield, even when they were the only letter learned.

In Simulation 1, there were 1200 distinct patterns learned, 292 (24.3%) were these problem letters; 22 from the 1 pattern trials (22%), 89 from the 4 pattern trials (22.25%) and 181 from the 7 pattern trials (25.86%) which is equal to the percentage of problem letters in the alphabet (23%). Tables B.1-B.3 show what the results would be if these “bad” patterns were removed.

HOPFIELD				
		LIF - Memory Load		
NOISE		**2%**	**8%**	**14%**
	0%	100.0%	23.5%	9.1%
	6%	100.0%	21.5%	7.3%
	12%	100.0%	20.3%	7.7%
	18%	100.0%	20.9%	6.7%
		QIF - Memory Load		
NOISE		**2%**	**8%**	**14%**
	0%	100.0%	25.1%	8.7%
	6%	100.0%	24.4%	6.6%
	12%	100.0%	20.9%	7.9%
	18%	100.0%	21.5%	6.6%
		CIF - Memory Load		
NOISE		**2%**	**8%**	**14%**
	0%	100.0%	25.1%	8.7%
	6%	100.0%	24.4%	6.6%
	12%	100.0%	20.9%	7.9%
	18%	100.0%	21.5%	6.6%

Table B.1: Overall Hopfield spiking performance for auto-associative tasks (section 2.5.2.1) with "problem" letters are removed, separated by neuron model.

NDRAM				
		LIF - Memory Load		
NOISE		**2%**	**8%**	**14%**
	0%	100.0%	100.0%	99.0%
	6%	100.0%	100.0%	99.2%
	12%	100.0%	99.4%	98.5%
	18%	100.0%	99.4%	95.2%
		QIF - Memory Load		
NOISE		**2%**	**8%**	**14%**
	0%	100.0%	100.0%	99.0%
	6%	100.0%	100.0%	99.2%
	12%	100.0%	99.4%	98.7%
	18%	100.0%	99.7%	95.4%
		CIF - Memory Load		
NOISE		**2%**	**8%**	**14%**
	0%	100.0%	100.0%	99.0%
	6%	100.0%	100.0%	99.2%
	12%	100.0%	99.4%	98.7%
	18%	100.0%	99.4%	95.4%

Table B.2: Overall NDRAM spiking performance for auto-associative tasks (section 2.5.2.2) with "problem" letters are removed, separated by neuron model.

NETWORK 1					NETWORK 2				
		LIF - Memory Load					LIF - Memory Load		
N		**2%**	**8%**	**14%**	**N**		**2%**	**8%**	**14%**
O	**0%**	100.0%	99.7%	98.8%	**O**	**0%**	100.0%	99.7%	98.8%
I	**6%**	100.0%	99.4%	98.5%	**I**	**6%**	100.0%	99.4%	98.5%
S	**12%**	100.0%	98.7%	97.3%	**S**	**12%**	100.0%	98.7%	97.3%
E	**18%**	100.0%	98.1%	94.0%	**E**	**18%**	100.0%	98.1%	94.0%
		QIF - Memory Load					**QIF - Memory Load**		
N		**2%**	**8%**	**14%**	**N**		**2%**	**8%**	**14%**
O	**0%**	100.0%	99.7%	98.8%	**O**	**0%**	100.0%	99.7%	98.8%
I	**6%**	100.0%	99.7%	98.5%	**I**	**6%**	100.0%	99.7%	98.5%
S	**12%**	100.0%	98.7%	97.1%	**S**	**12%**	100.0%	98.7%	97.1%
E	**18%**	100.0%	98.4%	93.8%	**E**	**18%**	100.0%	98.4%	93.8%
		CIF - Memory Load					**CIF - Memory Load**		
N		**2%**	**8%**	**14%**	**N**		**2%**	**8%**	**14%**
O	**0%**	100.0%	99.7%	98.8%	**O**	**0%**	100.0%	99.7%	98.8%
I	**6%**	100.0%	99.7%	98.5%	**I**	**6%**	100.0%	99.7%	98.5%
S	**12%**	100.0%	98.7%	97.1%	**S**	**12%**	100.0%	98.7%	97.1%
E	**18%**	100.0%	98.7%	94.0%	**E**	**18%**	100.0%	98.7%	94.0%

Table B.3: Overall NDRBAM (first and second layers) spiking performance for auto-associative tasks (section 2.5.2.3) with "problem" letters are removed, separated by neuron model.

The reason for Hopfield's issue with recalling high-spiking patterns is not known, but it is possible that it is because of the parameter selection. It has been found in Wärnberg (2014) that parameter selection for the Hopfield network with simply neural models like LIF can make a huge difference in recall. Parameter selection is not optimized in any of the simulations presented here. In chapter 2's simulations 1 and 2 (sections 2.5 and 2.6) the parameters were selected based on the original NDRAM transmission function. Simulation 3 (section 2.7) looks at a variety of parameters but is by no means an exhaustive list of possible parameter combinations. Chapter 2's simulation 3 did show that certain parameters improved the Hopfield network performance, therefore it is believed that ideal parameters can be found that allow the Hopfield network to work regardless of input. Specifically, consider δ, which Hopfield performed best at the lowest value studied; if a lower value was tested, this might help solve the task dependency seen here.

The neuron models may also be the problem. The models used here are very simple models, when Maass & Natschläger (1998) successfully produced a spiking Hopfield network, they used a more complex model for their simulation and subsequently mention that for less complex models, parameters, (in their case, specifically delays), may need to be carefully selected.