Learning and Leveraging Neural Memories

by

Sean Aubin

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Applied Science
in
Systems Design Engineering

Waterloo, Ontario, Canada, 2018

Author's Declaration

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

Learning in the Neural Engineering Framework (NEF) and the Semantic Pointer Architecture (SPA) has been recently extended beyond the supervised Prescribed Error Sensitivity (PES) to include the unsupervised Vector Oja (Voja). This thesis demonstrates how the combination of these learning rules can be used to learn associative memories. Moreover, these techniques are used to provide explanations of two behaving cognitive phenomena that are modeled with spiking neurons. First, the standard progression of cognitive addition strategies from counting to memorization, as occurs in children, is modelled as a transfer of skills. Initially, addition by counting is performed in the slow basal ganglia based system, before being overtaken by a rapid cortical associative memory as a type of pre-frontal, cortical consolidation. Second, a word-pair recognition task, where two distinct types of word-pairs are memorized, is modelled. The Voja learning rule is modified to match temporal lobe magnetoencephalography (MEG) data generated by each word-pair type observed during the task. This empirically grounds the associative memory model, which has not been possible using other cognitive modeling paradigms. The distinct implementation of Voja for each area, pre-frontal and temporal, demonstrates the different roles that the areas perform during learning.

Acknowledgements

I would like to thank

- **Chris Eliasmith** and **Terrence C. Stewart**, for their leadership of the Computational Neuroscience Research Group, as well as suggestions and encouragement through periods of scientific struggle.
- **Jelmer Borst** for explaining his MEG experiment and not letting me give up on matching the data.
- **Aaron Voelker**, for his mathematical insights in understanding the constraint of Voja.
- **Bryan Tripp, Jeff Orchard, and Chris Eliasmith** for providing feedback on drafts of this thesis.
- **Psychology & Neuroscience StackExchange** for helping me orient myself in the domain of Cognitive Science.
- My **labmates** and **friends**, for being such supportive, engaging, and thought-provoking people.
- My parents, **Robert** and **Elizabeth**, for providing a basis from which to strive.
- **Robin** for being my Complice over the last three years.
- **Will** and **Michelle** for being compassionate constants in my life.

Table of Contents

List of Figures

Chapter 1

Introduction

Two of the greatest challenges facing cognitive modelers are scalability and adaptability. Scalability refers to the ability to keep reusing the same system, the brain, to switch between tasks. Adaptibility refers to the ability to leverage previous experience when performing similar tasks. Building a model able to replicate various cognitive behaviours across multiple tasks is difficult enough, without giving it the ability to improve over time. This undertaking becomes even more daunting when trying to constrain the model biologically by basing its communication on spiking neurons, as well as making sure learning only requires information local to the neuron.

One attempt at satisfying biological constraints while scaling up is the Semantic Pointer Architecture Unified Network (Spaun), which is currently the largest behaving model of the human brain (Choo, 2018). Spaun has a range of cognitive skills, including serial working memory, induction and reinforcement learning. As input, it interprets 224x224 images with its vision system. As output, it controls a simulated arm. Spaun uses these skill to perform a variety of tasks, include list memorization, Raven's Progressive Matrices and the n-arm bandit task. In addition to these tasks, Spaun is able to follow instructions for combining/chaining different tasks together. However, Spaun lacks the ability to permanently improve its performance from previously experienced cognitive tasks.

This cognitive skill, wherein performance improves as tasks are rehearsed and practiced, is the focus of this thesis.

1.1 Improving with Practice

Previous attempts at improving with practice in a neural model include DeWolf and Eliasmith, 2013, wherein a simple motor skill (assigning a tone heard to a specific output) is consolidated in the cortex with practice. However, it is not clear how the skill allowing this simple perceptual one-to-one mapping could be extended for complex symbolic relations. This thesis scales the concept of improving with practice to enable consolidation while using complex representations, thus explaining more biological and cognitive phenomena.

In this thesis, two tasks using complex representations are modelled to understand the mechanisms behind improving with practice in different areas of the brain. First, the developmental progression of cognitive addition approaches, from a counting-based to recall-based strategy, is modeled. Second, a word-pair associative recognition task modeled. In the addition task, the improvement with practice is thought to originate in the pre-frontal cortex. Whereas in the word-pair task, the improvement is thought to occur in the temporal lobe. By contrasting these two models the mechanisms required for improving with practice are better understood in terms of their inputs and goals.

1.1.1 Thesis Structure

This thesis is structured as follows. The remainder of Chapter 1 describes the Addition Strategy Progress (ASP) and Word-Pair Recognition (WPR) tasks. Chapter 2 introduces the Neural Engineering Framework (NEF) and the Semantic Pointer Architecture (SPA) and discusses how they characterize the

neural representation of symbols, as well as the learning of associative memories. In Chapter 3, the ASP task is modeled, replicating the developmental behaviour and fMRI correlates of evolving addition strategies. Chapter 4 modifies the Voja learning rule used in association memory learning to match the experimental MEG signals of the WPR task. Chapter 5 summarizes the results and limitations of the different learning rules, as well as suggesting paths for future research.

1.1.2 Addition Strategy Progression

Mathematical Development in Children

The developmental transition from counting to memorization strategies when performing addition is an excellent example of improving with practice. This transition, like the acquisition of most mathematical skills throughout development, is not a simple linear progression (Sarnecka and Carey, 2008). Children transition through various strategies before converging on memorization, as shown in Table 1.1. The *Counting* strategy, modelled in Section 3.2, involves taking the largest addend and counting the amount of the remaining addend. For example, if performing 3+2, you would count from 3 twice: "3, 4, 5". The *Recall* strategy, modelled in Section 3.3, involves retrieving the addition fact from memory nearly instantaneously. As *Counting* is replaced by *Recall*, both the reaction times and error rates decrease, as shown in Table 1.2 and 1.3. This decrease in reaction times and error is also modelled in Section 3.3.

Although other strategies have been identified, this thesis focuses on the *Counting* and *Recall* strategies, since they present the most significant developmental change. Additionally, for the sake of simplicity, only sums under 10 are considered.

Addition-by-counting has already been implemented in Spaun. However

TABLE 1.1: Percentage of addition strategy use by grade level (summarized from Siegler, 1987).

Grade level	Counting	Recall	Guess or no response	Other
Kindergarten	30 %	16 %	30 %	24 %
Grade 1	38 %	44 %	8 %	10 %
Grade 2	40 %	45 %	5 %	11 %

TABLE 1.2: Median solution times (seconds) per addition strategy use by grade level (summarized from Siegler, 1987).

Grade level	Counting	Recall
Kindergarten	6.0 s	3.9 s
Grade 1	6.9 s	2.1 s
Grade 2	3.9 s	1.8 s

TABLE 1.3: Percentage of errors per addition strategy use by grade level (summarized from Siegler, 1987).

Grade level	Counting	Recall
Kindergarten	19 %	29 %
Grade 1	4 %	17 %
Grade 2	3 %	7 %

Spaun has no means of improving in terms of accuracy or speed when presented with similar tasks (Choo, 2018). Consequently, a cortical association mechanism is implemented in Chapter 3 to allow for memorization of previously seen addition-by-counting problems.

The transition between *Counting* and *Recall* was previously modelled using ACT-R (Lebiere, 1999). However, the ACT-R model uses a symbolic abstraction to explain the memorization process, wherein the mapping from addend to a set of addition facts is learned according to a probabilistic learning rule. Thus, unlike the model in Chapter 3, the ACT-R model has no grounding in biological neurons and thus limited neuroanatomical mapping. For example, ACT-R can relate activation of a module to a certain brain area, but the dynamics of this activation are not possible. This lack of constraints limit the

explanatory power of the ACT-R model.

1.1.3 Word-Pair Recognition

The word-pair recognition task, described initially by Borst et al. (2013), requires a subject to study word pairs such as SPARK+METAL and DOOR+CAR. During training, the subject is prompted with one word from the pair (DOOR) and must provide the matching word (CAR). A day later, the subject is tested to differentiate previously seen target pairs (SPARK+METAL) from foils composed of words seen during training (SPARK+CAR). Given that the foils are composed of training words that have been re-paired into combinations not seen during training, they are called re-paired foils (RPFoil). The practiced-based nature of the WPR task differentiates it from previously modelled list memorization tasks which rely on internal rehearsal (e.g. Gosmann (2018)). In the WPR, pairs are learned via prompt, whereas list memorization relies on a single type of presentation with no quiz-like prompts.

The WPR task relies on a similar association mechanism to the memorization strategy of the ASP task. They both combine a pair of inputs into a memory. However, there is no obvious iterative procedure for determining the correct result of a recall, thus ASP and WPR are not perfectly analogous.

In the WPR task, two different sets of word pairs with different associative fan, FAN1 and FAN2, are studied. Associative "fan" refers to the number of associations an item has with other items in memory. The distinction is explained with examples in Table 1.4 and reiterated in the next paragraph while describing the training procedure. Word-pairs are studied during the task by prompting the subject with a single word from the pair and waiting for all relevant responses.

For example, using the words from Table 1.4, if the subject is prompted with SPARK, they should respond with METAL. The fact that SPARK is only

TABLE 1.4: Stimuli types for word-pair recognition task. FAN1 and FAN2 pairs are presented during training. During testing, FAN1 and FAN2 pairs must be distinguished from RPFoil1 and RPFoil2 pairs. The "Half-matching pairs" column lists pairs seen during training with a word in common with the presented pair.

Stimuli type	**Example word-pairs**	**Half-matching pairs**
FAN1	METAL+SPARK	None
	TREE+BRAIN	None
FAN2	FILE+WHEEL	FILE+BIKE
		BIKE+WHEEL
	DOOR+CHAIR	DOOR+CAR
		CHAIR+CAR
RPFoil1	METAL+BRAIN	METAL+SPARK
		TREE+BRAIN
RPFoil2	FILE+CAR	FILE+WHEEL
		FILE+BIKE
		DOOR+CAR
		CHAIR+CAR

ever associated to METAL during training and no other word, makes SPARK+METAL a FAN1 pair.

If the subject is prompted with a word from a FAN2 pair, instead of a FAN1 pair as in the previous example, they must provide both paired words. For example, still using the words from Table 1.4, if the subject is prompted with the word FILE, they should respond with both WHEEL and BIKE. FILE requires two responses because it is part of two word-pairs (FILE+WHEEL and FILE+BIKE). The fact that FILE, WHEEL and BIKE each have exactly two associated pairs, as shown in Table 1.4, makes each word-pair (FILE+WHEEL, FILE+BIKE, BIKE+WHEEL) a FAN2 word-pair.

During recognition testing, both pairs seen during training are shown, combined with RPFoils from each word-pair type. Thus, four types of stimuli are seen at test time: FAN1 targets, FAN2 targets, FAN1 RPFoil and FAN2 RPFoil. For the sake of brevity, these will be referred to as: FAN1, FAN2, RPFoil1 and RPFoil2. These distinct stimuli give rise to different MEG responses, as well as different error rates and reaction times, as discussed in Chapter 4.

Chapter 2

Methods

This chapter introduces the NEF, SPA, and the learning rules used to implemented the tasks presented in Chapter 1.

2.1 Neural Engineering Framework

At its core, the NEF is a mathematical tool for translating dynamic functions defined over vector spaces into networks of spiking neurons and weights (Eliasmith and Anderson, 2003). The models presented in this thesis rely specifically on the NEF concepts of encoding and decoding.

2.1.1 Encoding

Encoding defines how a vector $\mathbf{x}(t)$ can be represented by the spiking activity of a neuron population. Each neuron is i assigned an encoding vector $\mathbf{e}_i$, which translate from N-dimensional representation space to firing rates. They can be understood as a preferred direction in the vector space. Neurons are also assigned a gain α_i, and a background current J_i^{bias}. These parameters define the translation of the input vector into input current $J_i(t)$. This input current then drives a neural nonlinearity $G_i[\cdot]$ which converts the input current into spikes. In this thesis, the neural nonlinearity is a Leaky Integrate-and-Fire (LIF) neuron model used to convert the input current into a spike

train $s_i(\mathbf{x}(t))$.

$$s_i(\mathbf{x}(t)) = G_i\left[\, J_i(\mathbf{x}(t))\,\right], \quad J_i(\mathbf{x}(t)) = \alpha_i\, \mathbf{e}_i \cdot \mathbf{x}(t) + J_i^{\text{bias}} \tag{2.1}$$

The spike train can be converted into a firing rate via filtering. This filter is modeled as convolution with a low-pass filter $h(t)$ which is a decaying exponential modelled after the postsynaptic current.

$$a_i(t) = (s_i * h)(t) \tag{2.2}$$

In this thesis, neuron properties will often be defined in terms of the neuron's tuning curve x-intercept, x^{cept} (from now on referred to as "the intercept of the neuron" or simply "the intercept") and max firing rate a^{max} where:

$$\begin{aligned} x^{\text{cept}} &< \mathbf{e} \cdot \mathbf{x} \text{ when } a_i(\mathbf{x}) = 0 \\ a^{\text{max}} &= a(x) \text{ when } \mathbf{e} \cdot \mathbf{x} = 1. \end{aligned} \tag{2.3}$$

These properties, visualized in Figure 2.1, can be easily used to derive the gain and bias of Equation 2.1, but are more useful for discussing encoder learning rules in Sections 2.5.2 & 4.2.1. A neural population that encodes a state-space as defined by the NEF in this thesis are referred to as an ensemble. All non-neural components of models, such as abstracted inputs and outputs, is referred to as "nodes".

2.1.2 Decoding

Decoding defines how to translate the filtered spikes trains from an ensemble into a vector via temporal decoding by using the filtered result from Equation 2.2 scaled by a decoding vector $\mathbf{d}_i$:

$$\hat{\mathbf{x}}(t) = \sum_i \mathbf{d}_i\, a_i(t). \tag{2.4}$$

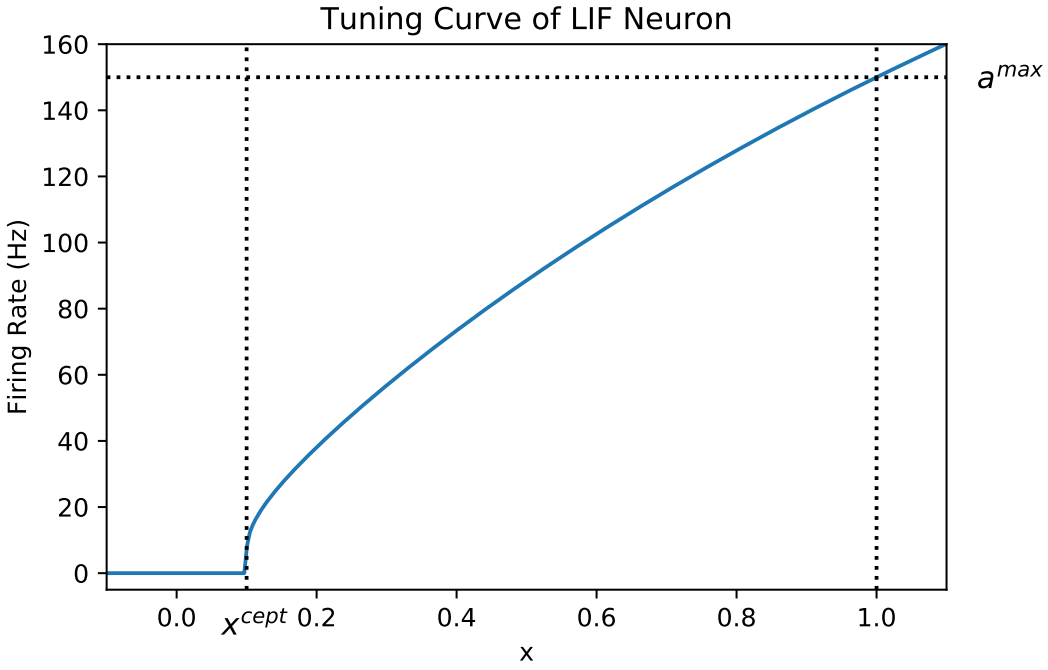

FIGURE 2.1: LIF neuron tuning curve demonstrating the locations of x^{cept} and a^{max}. The dotted line at $x = 1$ represents when $e \cdot x = 1$ for Equation 2.3, since $e = 1$.

The decoders $\mathbf{d}_i$ are typically found using regularized least squares optimization to minimize the error[1] over the range of inputs $\mathbf{x}$:

$$\int \|\mathbf{x} - \hat{\mathbf{x}}\|^2 \, d\mathbf{x}. \tag{2.5}$$

To decode an arbitrary function, the decoding error can be calculated using the desired function:

$$\int \left\| f(\mathbf{x}) - \hat{f}(\mathbf{x}) \right\|^2 d\mathbf{x}. \tag{2.6}$$

Connections between ensembles are weight matrices ω defined by

$$\omega_{ij} = \alpha_i \mathbf{e}_i \cdot \mathbf{d}_j \ .$$

Additionally, for the special case where the output is a linear function of the input, the gain term can be put into the weight matrix directly as $\mathbf{L}$. Thus,

[1] Many other metrics and optimization methods can be used.

given an output $\mathbf{Lx}$, $\omega_{ij} = \alpha_i \mathbf{d}_j \mathbf{L}\mathbf{e}_i$. Thus, the input current J to neuron i from neuron j is:

$$J_i(t) = \sum_j \omega_{ij} a_j(t) + J_i^{\text{bias}}. \tag{2.7}$$

Together, the concepts of encoding and decoding allow for the construction of spiking neural networks, composed of interconnected ensembles, to manipulate vectors spaces using arbitrary functions.

2.2 Semantic Pointers for Representing Symbols

The NEF has been used to form the basis for a cognitive architecture called the Semantic Pointer Architecture (SPA). The SPA includes elements for action selection, working memory, and proposes a general neural representation called a Semantic Pointer (SP). SPs are compressed, neurally implemented representations. Eliasmith (2013) suggests that such compressed representations are generated and used in the motor, cognitive, and perceptual systems.[2]

This thesis focuses on using SPs as a symbol-like representations. In Chapter 3, SPs are used to represent the digits to be added. In Chapter 4, SPs represent the words to be paired and recognized.

2.3 Dynamics

Often it is desirable to operate on the state represented by the ensemble over time. For example, in Section 4.3.2, a neural population is needed to integrate over an input value over time. To define these dynamics and compute them neurally, we first consider a state variable $x(t)$ with some desired non-linear dynamics:

$$\frac{dx}{dt} = g(x). \tag{2.8}$$

[2]For a more complete introduction to using SPA for large-scale cognitive modeling, see Sharma, Aubin, and Eliasmith (2016).

These dynamics can be mapped onto recurrent decoders. Specifically, to satisfy Equation 2.8 given the filter $h(t)$ from before, the recurrent decoders can be solved for as $f(x) = \tau_s g(x) + x$. This dynamic state representation is used in the SPA model working memory in Section 3.2, as well as the WPR evidence accumulator in Section 4.3.2.

2.4 Matching MEG data

In this thesis, it is assumed an MEG signal is analogous to summing the post-synaptic potentials of the neural population from which the MEG signal is thought to originate (Ahlfors and Wreh, 2015). Additionally, it is assumed the dendrites from which the post-synaptic potential originates are roughly parallel and thus their individual contributions are purely constructive. The precise mapping from NEF neuron responses to the MEG signal is outside the scope of this thesis. Instead, the relative positioning of responses given different inputs is emphasized over their exact estimated current values. Specifically, as will be discussed further in Section 4.2, the WPR task creates a greater response for FAN1 inputs compared to FAN2 inputs. Thus, the goal of Section 4.2 will not be to match the exact current amount in a neural ensemble, but instead to create a neural ensemble whose sum of firing rates is greater for FAN1 inputs than FAN2 inputs.

2.5 Associative Memories

2.5.1 Designing Associative Memories

The NEF concepts of encoding and decoding can be used to create heteroassociative memories, such that a set of input vectors are mapped to a different set of output vectors by a neural ensemble (Stewart, Tang, and Eliasmith, 2011;

Gosmann, Voelker, and Eliasmith, 2017). This is achieved by creating an array of ensembles, such that:

1. Encoders for each ensemble only respond to a single input vector
2. Mutual inhibition between each ensemble of neurons computing a Winner-Take-All (WTA) function
3. Decoders are chosen to map the winning ensemble onto the desired output vector

An example network implementing these features is shown in Figure 2.2. However, this associative memory design requires knowing the input space

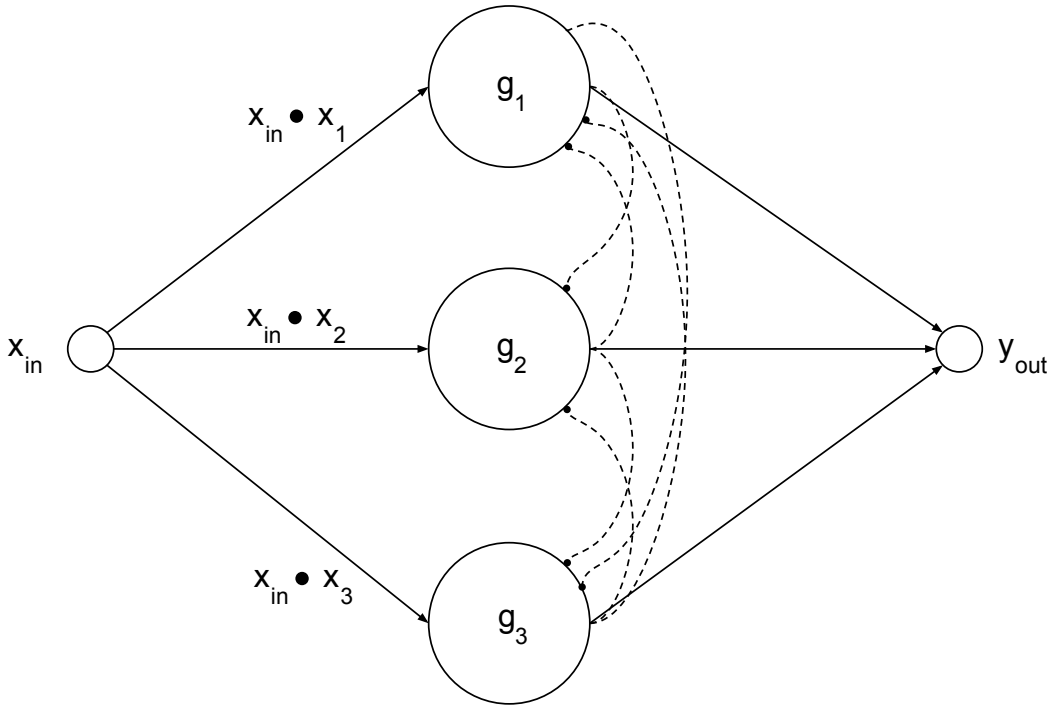

FIGURE 2.2: An example Winner-Take-All heteroassociative memory. Each group of neurons g represents a specific input x. The groups inhibit each other proportional to their activation, as shown by the dashed connections, so only the highest activated group outputs a value to y_{out}.

and output space ahead of time.

2.5.2 Learning Associative Memories

Instead of "hard-coding" these memories, it is possible to learn them from experience for better biological and psychological plausibility. A variety of different learning approaches have been proposed over the last 40 years for associative memories Willshaw, Buneman, and Longuet-Higgins, 1969; Wu and Pados, 2000, however none meet the needs for learning an associative memory in the NEF in a biologically plausible manner in the NEF. Specifically, none are biologically plausible (learned online in continuous time with information local to the neuron) while allowing for neurons representing multiple dimensions.

By combining the supervised decoder Prescribed Error Sensitivity (PES; MacNeil and Eliasmith, 2011) learning rule with an unsupervised encoder learning rule, Knight et al. (2016) have shown it is possible to learn such a discontinuous high-dimensional function in a spiking neural network in a scalable and efficient manner.

PES defines how decoders ($\mathbf{d}_i$ from Equation 2.4) are modified given the filtered spike train a_i resulting from $\mathbf{x}$ described in Equation 2.1, as well as an error signal $\mathbf{E}$ and a learning rate κ:

$$\Delta \mathbf{d}_i = \kappa \mathbf{E} a_i. \tag{2.9}$$

For a discontinuous, high-dimensional function, such as a heteroassociative memory mapping a set of inputs to a distinct set of outputs, PES alone is insufficient. As explained below, this is due to the difficulty of mapping distinct errors onto a decoder.

In heteroassociative memories, each input generates a distinct error vector $\mathbf{E}$, despite activating similar neurons. Thus, given that the change of decoder $\Delta \mathbf{d}_i$ for neuron i is proportional to the firing rate a_i in Equation 2.9, firing for multiple inputs with distinct errors $\mathbf{E}$ causes the decoder change to overwrite

the previously learned change. This overwriting causes "catastrophic forgetting". This phenomena is shown explicitly later in Figure 3.6. This forgetting could be overcome if each neuron fired selectively for a single input, since there would only be a single $\mathbf{E}$ associated to a_i.

Instead of choosing encoders for better neuron input selectivity, encoders can be learned from the presented inputs using Vector Oja (Voja; Voelker, Crawford, and Eliasmith, 2014). Given neuron j and its firing rate a_j in response to an input $\mathbf{x}$ described in Equation 2.1, a learning rate η and an input $\mathbf{x}$, the encoder $\mathbf{e}_j$ is adjusted according to Voja as follows:

$$\Delta \mathbf{e}_j = \eta a_j (\mathbf{x} - \mathbf{e}_j). \tag{2.10}$$

Voja can also be defined as a variant of its namesake, Oja's rule (Oja, 1989), which is a normalized version of Hebbian learning. Oja operates on the connections between two populations of neurons. Substituting $\mathbf{e}_i$ with the row weights ω_i, x for the pre-synaptic activity $\mathbf{b}$ and letting $s = \frac{1}{a_i}$ gives

$$\Delta \omega_i = \kappa a_i (\mathbf{b} - s a_i \omega_i) \ ,$$

which is the single-row Oja update rule.

Voja Convergence

To guarantee convergence, the x-intercepts of the neurons in the population are chosen to be more than the maximum similarity of all inputs. Otherwise, in Equation 2.10 a_j would respond to multiple inputs and $\mathbf{e}_j$ could be pulled between those two inputs. Without setting x^{cept} properly, a_i will cause the decoders to adapt to multiple distinct errors $\mathbf{E}$ and "catastrophic forgetting" will occur.

The conditions for convergence are shown explicitly in Figure 2.3 where two neurons, n_1, n_2 [3], with the respective intercepts $x_1^{\text{cept}} = \cos\ \frac{\pi}{3}$ and $x_2^{\text{cept}} = \cos\ \frac{\pi}{6}$ are moved using Voja given two stimuli $\mathbf{x}_1, \mathbf{x}_2$ where $\mathbf{x}_1 \cdot \mathbf{x}_2 = \frac{\pi}{5}$. The plots in Figure 2.3 explain why n_2 with intercept $\cos\ \frac{\pi}{6} > \cos\ \frac{\pi}{5}$ will converge, while n_1 with intercept $\cos\ \frac{\pi}{3} < \cos\ \frac{\pi}{5}$ does not. Before any stimuli is pre-

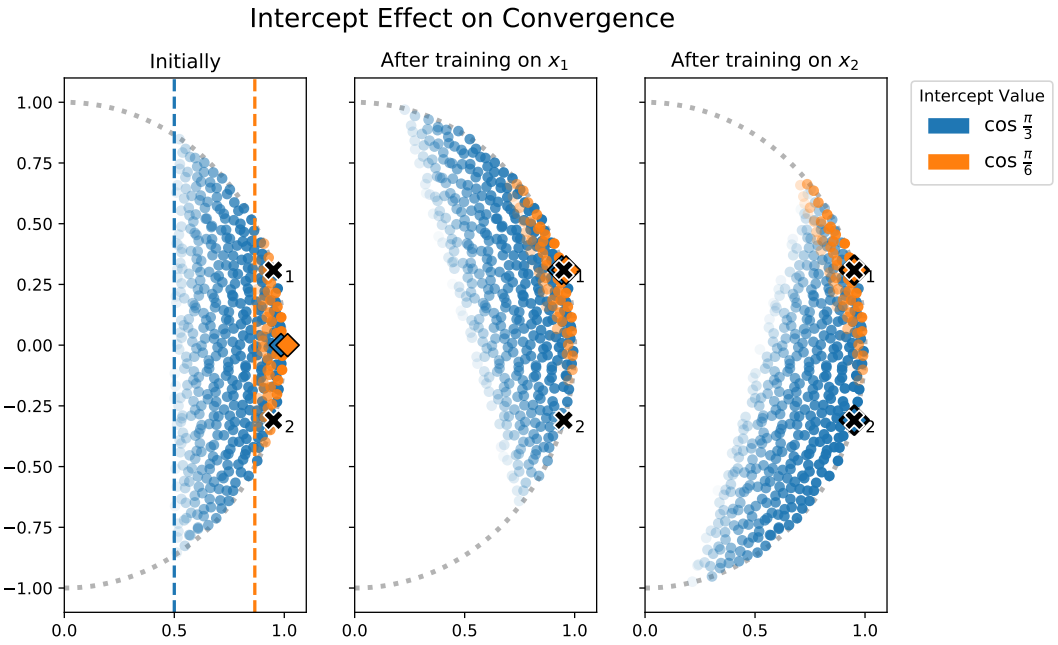

FIGURE 2.3: Contrasting Voja convergence behaviour between two neurons with different intercepts with their receptive fields delineated by dashed lines. Coloured circles represent the neuron firing rate at the given point via opacity, where translucence indicates less firing. Stimuli are represented by a black x, while the encoder of both neurons is represented by diamonds of the respective colours with slightly displaced so they do not overlap. See text for further details.

sented, as shown in Figure 2.3 in the plot titled "Initially", the encoders of both neurons are $\mathbf{e}_1 = \mathbf{e}_2 = [1, 0]$ and are thus equidistant from possible input, since $\mathbf{e}_1 \cdot \mathbf{x}_1 = \mathbf{e}_2 \cdot \mathbf{x}_2 = \frac{\pi}{10}$.

When $\mathbf{x}_1$ is presented, both encoders converge to it, as shown in the plot titled "After training on x_1". The encoders converge such that $\mathbf{e}'_{1,2} \to \mathbf{x}_1$ to minimize $(\mathbf{x} - \mathbf{e})$ in Equation 2.10. Note, that if $\mathbf{x}_2$ were presented first, $\mathbf{e}_{1,2}$

[3] Sometimes written as $n_{1,2}$.

would have converged to it instead. Recall from Figure 2.3, that a neuron will fire for given $x^{\text{cept}} < \mathbf{e} \cdot \mathbf{x}$, which given $x_1^{\text{cept}} < x_2^{\text{cept}} < \mathbf{e}_{1,2} \cdot \mathbf{x}_2$ causes $a_{1,2} > 0$. This is shown visually in the Figure, as a coloured "receptive field" for each neuron, where $\mathbf{x}_{1,2}$ are both inside the coloured area. The non-convergence only becomes evident after the setup of this initial adaptation.

When x_2 is presented, only n_1 with intercept $x_1^{\text{cept}} < \mathbf{e}_1' \cdot \mathbf{x}_2$ causing $a_1 > 0$ and will move again. While n_2 with intercept $\mathbf{e}_2' \cdot \mathbf{x}_2 > x_1^{\text{cept}}$ will remain converged on the first input, since $a_2 = 0$. This result is shown in Figure 2.3 in the plot titled "After training on x_2". The only way for both neurons to converge to either x_1 without ever firing for x_2 is if the similarity between stimuli vectors was less than the minimum intercept, such that $x_{1,2}^{\text{cept}} < \cos\frac{\pi}{3} < \mathbf{x}_1 \cdot \mathbf{x}_2$.

As shown by Knight et al. (2016), this similarity constraint can be optimally satisfied when mapping one set of vectors to another by leveraging a solution to the kissing problem. Specifically, the Leech lattice allows for the selection of 196560 24 dimensional vectors each separated by an angle $\theta \leq \frac{\pi}{3}$. However, both the ASP and WPR task require mapping combination of vectors to another set of vectors: ASP requires mapping two addend vectors to the sum, and WPR requires mapping two word components to a word pair. Once vectors are combined, there is no predictable limit to their similarity and this optimal solution cannot be used.

The following chapter shows how these methods for building symbol-like representations and implementing learning rules can be combined to give a model capturing typical childhood addition strategy progression from those reliant on counting to those reliant on a faster memorization of sums.

Chapter 3

Learning Addition via Memorization

3.1 Introduction

This chapter applies the concepts presented in Chapter 2 to model the transition between counting-based and recall-based addition strategies described as the ASP task in Chapter 1. First, a model of the counting strategy is presented. This is followed by an extension to the counting model, inspired by psychological evidence, which allows the system to memorize the counting strategy results. This memorization allows for quick recall-based answers. Finally, the anatomical mapping of these circuits and their implications for dyscalculia, a disorder causing calculation difficulties, are discussed.

3.2 Modeling the Counting Strategy

As mentioned in Chapter 1, Spaun is able to perform addition using the counting strategy. However, using the full version of Spaun, which has 6.6 million neurons (Choo, 2018), to study only this one task would be impractical. Spaun is computationally expensive to run, given that it also performs 11 other tasks not being studied. As a result, the network has been re-implemented with the following abstractions:

- The Visual system is abstracted into an output node outputting two addend digit SPs directly instead of translating them from images
- The Motor system is abstracted into an input node accepting digit SPs instead of a simulated arm

Each SP representing a digit is chosen randomly from a 10 dimensional orthonormal basis. This was chosen to give a defined bound on the similarity between concatenated pairs of pointers, which are used as the input to the memory in Section 3.3 when the recall strategy is introduced, given the limitations of the Voja learning rule described in Section 2.5.2. However, the digit SP representations are still chosen randomly from this basis to limit implied prior knowledge. As in development, the only knowledge the system has is how to count, which is a skill contained in the Incrementing Memory. The Incrementing Memory is a designed associative memory, as described in Section 2.5.1, where the input is a digit SP and the output is the incremented version of the input SP.

A network, named Slow-Net due to it operating at the speed of sub-vocal rehearsal, is constructed to carry out addition-by-counting strategy. Slow-Net iteratively follows steps controlled by the default basal ganglia and thalamus action selection system in the SPA (Stewart, Choo, and Eliasmith, 2010). The first step of this process is to load the addends from the vision node into working memory. The largest addend is loaded into the "Count result" memory module, while the smaller addend is loaded into "Total counts to take". Once these memories are initialized with the addition problem, the steps are enumerated below and illustrated in Figure 3.1, with each step corresponding to arrows annotated with the corresponding digit, are followed:

1. Route digits from memory to the Incrementing Memory
2. Transform the digit using the Incrementing Memory

3. Overwrite the old digit in working memory with the transformed digit

4. If “Counts finished” equal “Total counts to take”, then output “Count result” to the motor output as the final answer

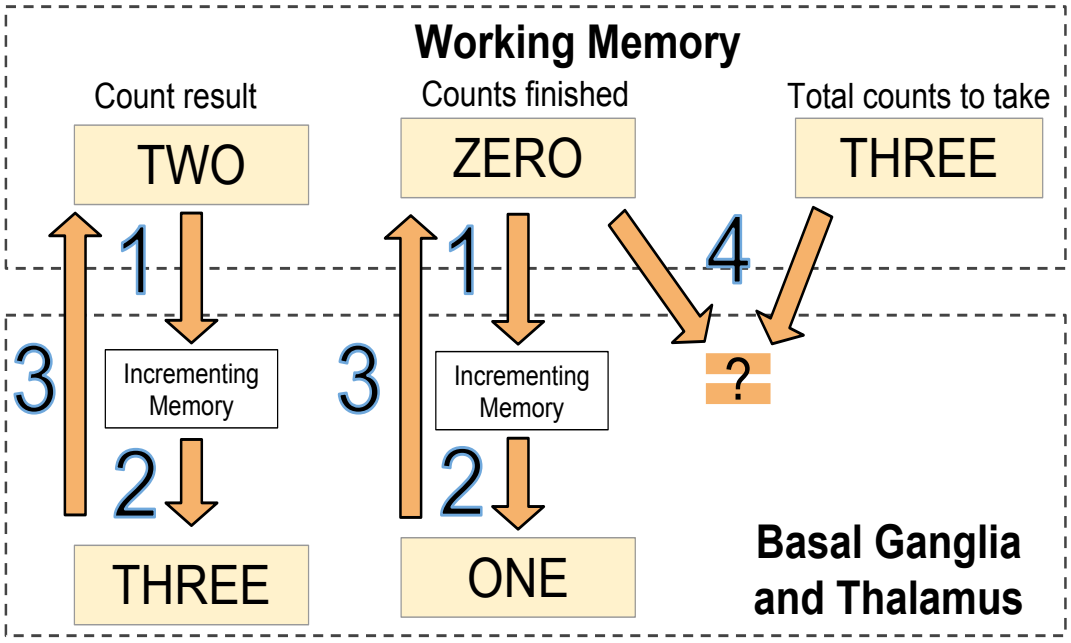

FIGURE 3.1: Overview of a single step of the addition-by-counting procedure computing 2+3 performed by Slow-Net. Each number, colored in blue, pertains to a step enumerated in the text.

The goal of Fast-Net, presented in the next section, is to memorize a similar function, such that the inputs are the same addends presented to Slow-Net and the output is the sum.

3.3 Memorization via Practice

To model the recall strategy of addition, a memory is learned to replace the Slow-Net process. As described in Section 2.5.2, heteroassociative memories can be learned by applying Voja learning rule to adjust encoders and the PES learning rule to adjust decoders. Thus, to implement the recall strategy, a heteroassociative memory named Fast-Net is learned.

As shown in Figure 3.2, Fast-Net is placed in parallel with Slow-Net, thus inputs are presented to both networks simultaneously. The error signal required by PES is provided by calculating the difference between Fast-Net and Slow-Net outputs. The error is only propagated once Slow-Net outputs an answer. This internal control of the dopaminergic error signal can be thought of as a type of metalearning (Doya, 2002) or controlling how to learn. Such feedback could also come from the environment (e.g., in the form of a teacher correcting the student who is drilling addition facts), but this extension is outside the scope of this thesis.

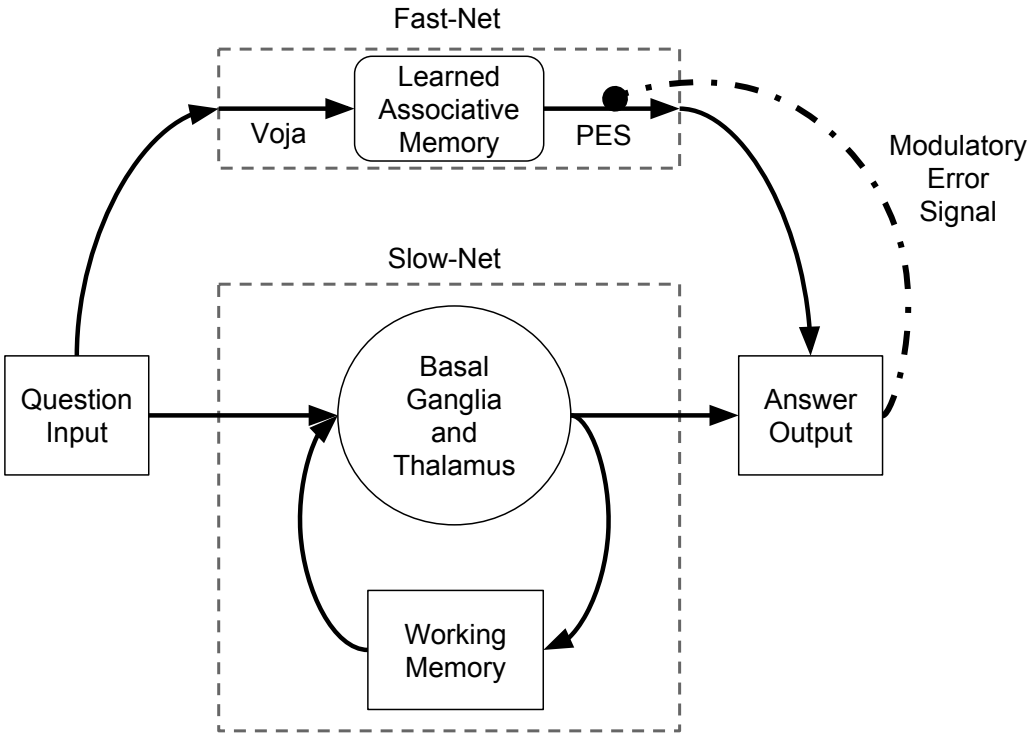

FIGURE 3.2: High-level model architecture, featuring the parallel Slow-Net and Fast-Net. The input is provided to both networks simultaneously. The Fast-Net is an associative memory learned using the Voja and PES learning rules which memorizes a mapping from addends to a sum. The Slow-Net iteratively calculates a sum via working memory manipulation. The Fast-Net learns its responses via a modulatory error signal projected from the output of the Slow-Net.

When the similarity of the output of Fast-Net and the set of possible numerical outputs surpasses an arbitrarily set similarity threshold τ_{num}, Fast-Net

outputs an answer to the motor system and interrupts Slow-Net via inhibition. This decision is carried out by a separate basal-ganglia loop, as shown in Figure 3.3. In this way, Fast-Net learns from the output of Slow-Net until it can take over the function computed.

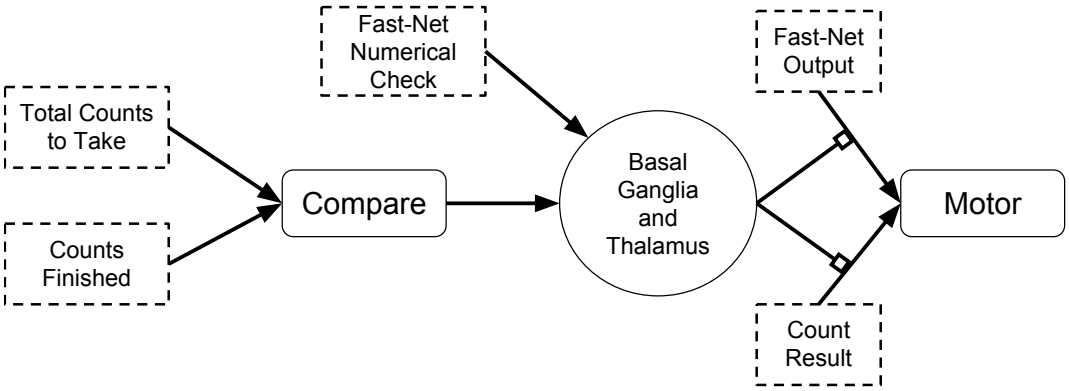

FIGURE 3.3: Answer Output network from Figure 3.2 expanded to show routing accomplished by Basal Ganglia and Thamalus. Routing connections, which enable normal neural connections, are shown as box-headed arrows. (Counts to Take · Counts Finished) is fed into the Basal Ganglia to detect if the Slow Net has output an answer. Fast-Net Numerical Check = Fast-Net Output · (ONE + TWO + ... + NINE) − τ_{num} is also considered and can override the Slow-Net process. Depending on which case is satisfied first, the Thalamus routes the Fast-Net Output or Count-Result to the Motor output.

3.4 Results

The model was built and simulated using Nengo 2.1 (Bekolay et al., 2014), while the results were plotted using Seaborn 0.7.1 (Waskom et al., 2016). Code for the simulations and plots are available at `github.com/Seanny123/counting_to_addition`.

3.4.1 Slow-Net Counting Performance

The results of the Slow-Net, which implement the counting strategy, are shown in Figure 3.4.

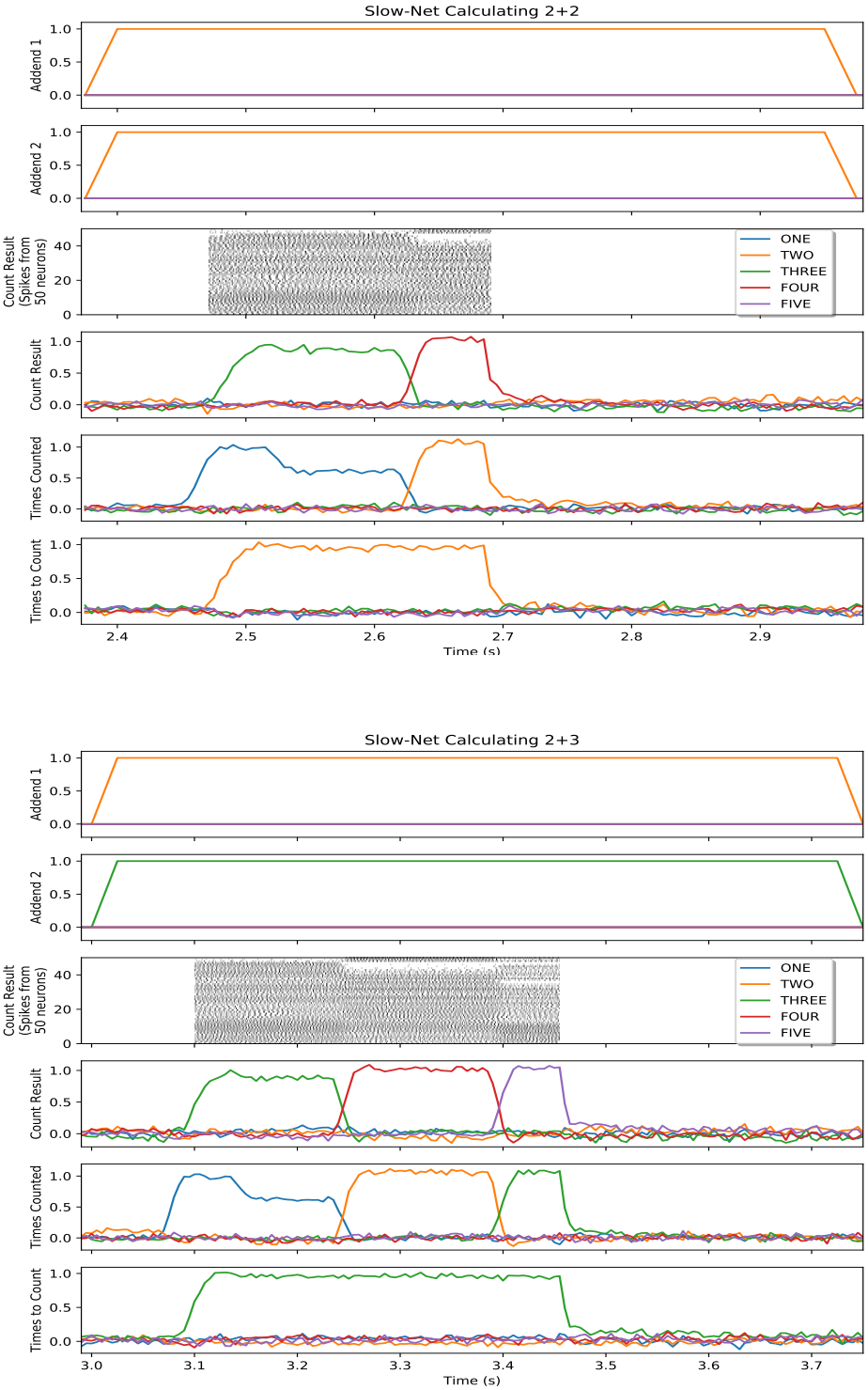

FIGURE 3.4: The Slow-Net (counting network) answering $2 + 2$ and $2 + 3$. Line plots show similarity between neural activity in the area and the ideal spiking pattern for a Semantic Pointer digit over time. As shown in "Count Result", the model iterates through intermediate digits before reaching the answer.

In the Figure, Slow-Net solves two sums, $2 + 2$ (occurring between 2.2s and 2.8s) and $2 + 3$ (occurring between 3s and 3.6s). To compute $2 + 2$, as described in Section 3.2, the first step is to load the values for this computation into memory. "Times to Count" is assigned 2. The "Count Result" is assigned the incremented value of $2 \rightarrow 3$, since values are incremented before being loaded into memory to save an iteration step. "Times Counted" is assigned 1, since "Count Result" has already been incremented once. Although not shown explicitly in the figure, the contents of "Times Counted" and "Time to Count" are compared before proceeding to the next increment. After both "Count Result" and "Time Counted" are incremented, "Times Counted" and "Time Counted" are equal. Thus, the final "Count Result" of 4 is output as the answer, the process stops and the memories are cleared. The same steps occur for $2 + 3$, but with different initial memory values.

3.4.2 Fast-Net Memorization Performance

The learning rate of Fast-Net can be tuned according to developmental speed. At a high rate, sums are memorized after a single presentation. To ensure Fast-Net would learn the transform of two concatenated addends at all, and to demonstrate the necessity of Voja, it was first trained in isolation with a high learning rate, as shown in Figures 3.5 and 3.6. The latter figure was created by selecting the 100 decoder weights from each ensemble with the largest absolute derivative during the learning period and shows how encoders learned using Voja allow for decoders weights to converge.

3.4.3 Full Network Performance

Given the function of Fast-Net was confirmed in isolation in Figure 3.5 & 3.6, the next goal is to match the decrease in errors and reaction time with practice, as discuss Section 1.1.2.

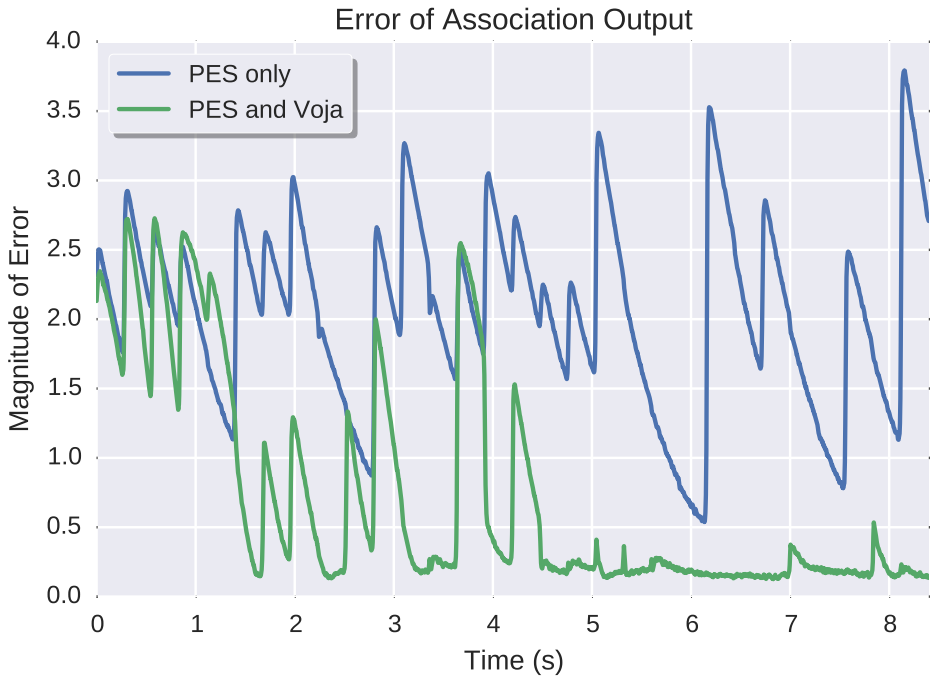

FIGURE 3.5: Fast-Net output error magnitude while being simulated separately from Slow-Net and learning the sum transformation given inputs of two addends concatenated. A new set of addends is shown every 300ms.

For simulation in the full network, a lower learning rate was chosen to emulate the gradual learning seen in human behaviour, but fast enough for the learning to converge after a short amount of simulation time. A training epoch consisted of 20 addition questions shown in a random order. As expected, the error decreased significantly after each example and there is a uniform decrease after each epoch, as shown in Figure 3.7.

Once the error magnitude decreases past the arbitrary threshold of 0.5 mentioned in Section 3.3, the reaction times plateau to the minimum time it takes to recall the correct answer from Fast-Net. In the case of the simulation in Figure 3.8, this is approximately 7 repeats of the 20 addition problems. The reaction times decreasing matches the psychological data trend of decreasing reaction times in Table 1.2.

FIGURE 3.6: Fast-Net decoder weights changing over time while being simulated separately from Slow-Net and learning the sum transformation given inputs of two addends concatenated. Ensemble decoders learned using only the PES rule never converge. However, if learned using Voja with PES, ensemble decoders quickly settle to a stable value.

In over 100 trials, the Slow-Net only failed three times to produce a correct answer and instead over-counted. These three failures are omitted from Figure 3.8, as they are considered as outliers. Although this failure rate is not unreasonable for children, these failures could either be corrected by tweaking the model further or by implementing an introspective error monitoring mechanism, such as the one being investigated by Thorgeirsson, Stewart, and Eliasmith (2018).

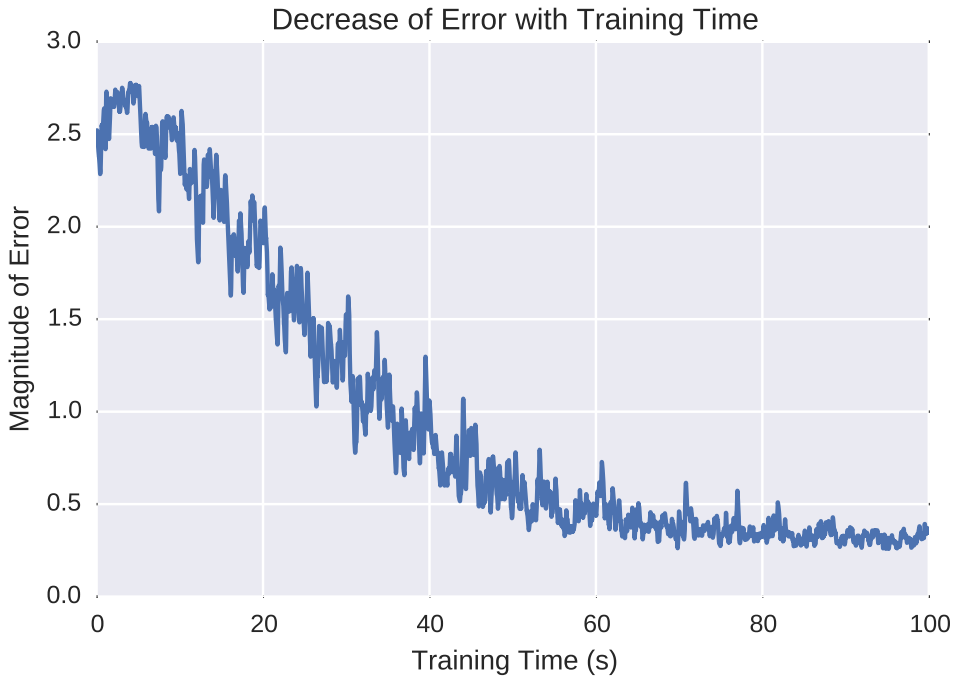

FIGURE 3.7: Error magnitude in the Fast-Net decreasing with training received from the Slow-Net feedback of each trial.

3.5 Neuroanatomical Mappings and Dyscalculia

Spaun's mapping of counting (Eliasmith et al., 2012) associates parietal areas with stable, learned transformations, while prefrontal areas are more for transient, working-memory manipulations. Given this mapping and the transition from Fast-Net to Slow-Net seen in the model, activation in humans while performing addition should transition from prefrontal to parietal areas with practice. This is supported by changes in activation during mental calculation differing with age.

Specifically, Rivera et al. (2005) have shown age is positively correlated with parietal fMRI activation and inversely correlated with prefrontal and hippocampal brain areas, as well as the use of the dorsal basal ganglia area. Using the model, this can be framed as older children abandoning the Slow-Net (requiring loading of instructions into the hippocampus, control of prefrontal work-memory resources by the basal ganglia), in favour of the memorized

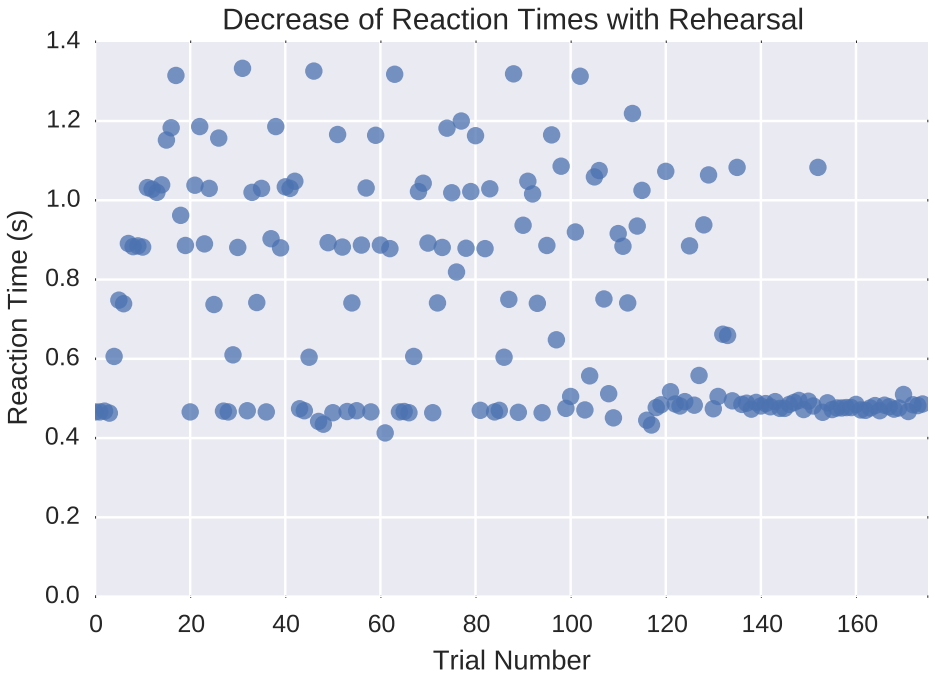

FIGURE 3.8: Reaction times decreasing with rehearsal as the Fast-Net takes over for the Slow-Net for increasingly more additions. Note that these reaction times do not take into account motor planning for communicating the result and are thus much faster than those seen in humans.

parietal transforms similar to the Fast-Net. Additionally, this transition from pre-frontal to parietal is not exclusive to large developmental timescales. In Ischebeck et al. (2007), subjects were shown complex multiplication problems while being imaged with fMRI. Even with only 24 training exposures, previously seen multiplications activated parietal areas, whereas novel multiplication problems activated prefrontal areas.

Section 1.1.2 discusses typical development of children, however there are individuals who are diagnosed with the learning disability dyscalculia. Dyscalculia manifests itself as a difficulty acquiring arithmetic skills. Neurologically, individuals with dyscalculia show greater activation of the prefrontal cortex, compared to parietal areas (Kucian and Aster, 2015). Although this model makes no claims about the origins of dyscalculia, given its lack of a direct cause and frequent comorbidities (Rubinsten and Henik, 2009), it does offer

an explanation as to why this compensation occurs. Given a malfunctioning parietal learning system, the transition from working memory resources in the prefrontal cortex never occurs. This malfunction could be due to a variety of reasons, such as an incorrectly modulated error signal, noisy input or inaccurate feedback. Regardless of the cause, the result of a progression from iterative processes to recall would not be seen in a malfunctioning network.

Chapter 4

Matching Associative Memory MEG Signals

On the surface, memorizing addition results is similar to recognizing previously studied word-pairs. In both cases, a pair of stimuli is consciously memorized via repeated trials. However, as discussed in Chapter 3, memorizing addition occurs in the prefrontal cortex, while according to Borst, Ghuman, and Anderson (2016) word-pair memorization occurs in the temporal cortex. It follows that separate brain areas may learn associations differently. Consequently, this chapter uses the MEG data from the recall memory component of a WPR task (Borst, Ghuman, and Anderson, 2016) to guide the modification of the associative memory model presented in the context of the ASP task in Section 3.3. This modified model gives insight into the different function of these two brain areas.

4.1 Modeling Word-Pair Recognition

The WPR SPA model is shown in Figure 4.1. The task words are represented as random 32-dimensional SPs. Similar to the counting network of Chapter 3, the visual system and motor system are abstracted away into two word SP inputs and a single word SP output. According to EEG (Borst and Anderson, 2015) and MEG (Borst, Ghuman, and Anderson, 2016) data analysis,

the recognition of a word-pair happens in three stages between initial visual encoding of the words and motor actuation outputting the word-pair classification:

1. **Familiarity:** determine if either component of the pair has not been studied before

2. **Recall:** access the learned memory

3. **Representation and Decision:** make a decision based on the output of the learned memory

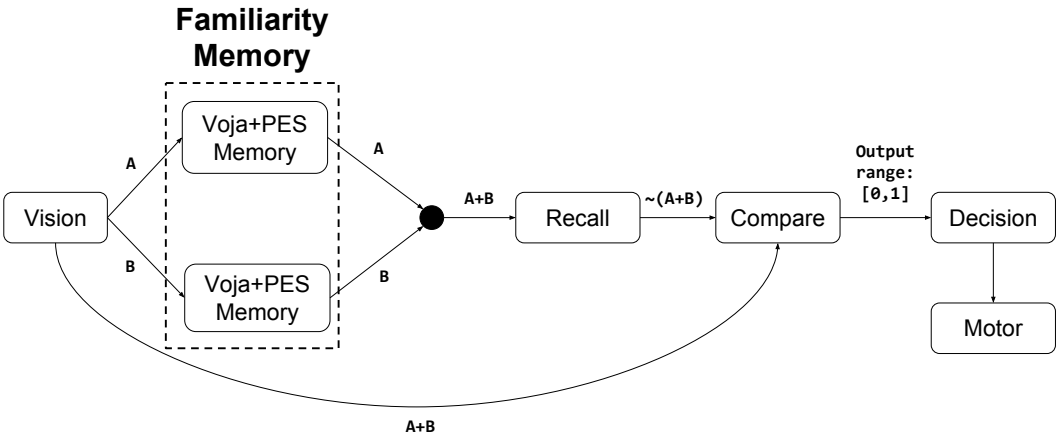

FIGURE 4.1: High-level model architecture. Bold connection labels show information flow for "word" pair **A+B**. See text for details.

Each stage is represented by different neuron populations in the SPA model. Although this chapter briefly discusses the familiarity and decision network, the focus of this chapter is the recall population and how its output influences the final decision. Specifically, the recall population is designed to match its MEG response to the available experimental data given different input types. Additionally, the recall population's influence on the reaction time and error rate, as shown in Figures 4.2, 4.11 and 4.10 respectively, is discussed.

The training task, where the pairs are learned initially, is not the focus of this thesis and is thus not explicitly modeled[1]. Instead, the training task is only used to inform what training data (targets only) is given to the model.

All models in this chapter were built and simulated using Nengo 2.7 (Bekolay et al., 2014), while the results were plotted using Seaborn 0.9.0 (Waskom et al., 2016). Code for the simulations and plots are available at `github.com/Seanny123/nengo_learn_assoc_mem`. Nengolib 0.4.2 (Voelker, 2018) was used for encoder initialization, as described in Section 4.3.1. The data in the following subsections is taken from the test phase which consisted of 14 blocks with 64 randomly ordered trials (16 FAN1 targets + 16 FAN2 targets + 16 RPFoil1 + 16 RPFoil2) each administered to 18 subjects.

4.2 Modeling the Familiarity and Recall Stages

As shown in Figure 4.1, the first stage after decoding words from visual input is the familiarity stage. This stage ensures only targets and RPFoils reach the recall stage. It is motivated by the significantly lower reaction times for novel foils (foils composed of words not seen during training) compared to RPFoils in Borst and Anderson (2015). Although not explicitly modelled in this thesis, this filtering behaviour is accomplished by two auto-associative memories (mapping previously seen inputs to themselves and mapping unseen inputs to noise), trained using Voja and PES to recognize the words seen during training. The outputs of these memories are fed into a separate decision network that triggers a response for novel foils by taking the dot product between the input and the memory output. A similar method is used and explicitly modelled when making a decision based off the recall memory output.

[1] Although modeling the task should only require an associative memory using the Voja and PES learning rules

The inputs into the recall memory come from the familiarity module's two parallel memories whose output vectors are added together and normalized. Vector addition was chosen as the word pair encoding, given that the operator had to be order invariant (unlike the concatenation used in Chapter 3), as well as lossless (unlike circular convolution which is the operator used in SPA for compression (Eliasmith, 2013)) for ease of understanding and learning [2].

To translate recall into a decision, the recall memory output is compared with the vision input by calculating the dot product between the two vectors. This is represented by the Compare network in Figure 4.1. The returned magnitude of this operation acts as a proxy for recall confidence. A strongly recalled memory with a large dot-product is considered a target, while a weakly recalled memory with a smaller dot-product is considered a foil.

As described in Section 2.4, it is assumed that an MEG signal is analogous to summing the firing rates of a neural population. The experimental MEG data used for comparison to neural models in Figure 4.2 is taken from Borst, Ghuman, and Anderson (2016). The MEG signal is generated by averaging over the processed MEG responses of 18 subjects. Additionally, each subject MEG response was averaged over 14 test blocks. Further information on the experimental testing procedure can be found in Borst, Ghuman, and Anderson (2016).

All model MEG signals (Figures 4.3, 4.4, 4.6 & 4.8) in this Chapter are produced in a similar manner, but with only a single test block. 64 randomly ordered trials composed of 16 FAN1 targets + 16 FAN2 targets + 16 RPFoil1 + 16 RPFoil2 are input into an ensemble with all learning disabled. The MEG signal for each of these trials is acquired by summing over the spikes from an ensemble during the 300ms per-trial presentation period. The MEG signals are then averaged per trial type over the 300ms presentation window before being plotted.

[2] Lossy compression would make inputs more similar, making associations harder to learn

The MEG signals to match, shown in Figure 4.2, are thought to originate from the temporal lobe where both the familiarity and recall steps of the task occur. Matching the general MEG signal pattern (FAN1 and RPFoil1 higher than FAN2 and RPFoil2) is emphasized over matching the exact electric current values.

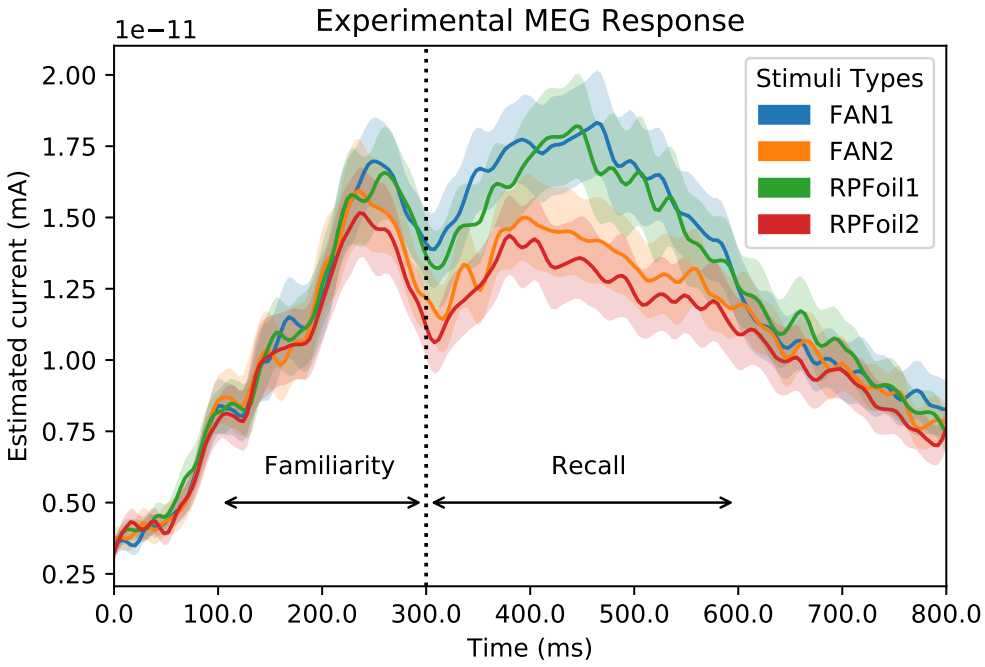

FIGURE 4.2: The experimental MEG response of the recall (before 300ms) and familiarity (after 300ms) steps of the word-pair recognition task. Shaded areas represent 95% boot-strapped confidence intervals.

The first "bump", as labeled in the Figure 4.2, originates from the familiarity process. It is matched by the previously mentioned pair of memories trained with Voja and PES, as shown in Figure 4.3. The second "bump" after the dotted line cannot be matched in the same way, as shown in Figure 4.4 and explained below.

The "Recall" memory's role in Figure 4.1 is essentially that of an associative memory that associates an input word-pair to itself. However training a memory with Voja and PES, as performed in Chapter 3 does not give the

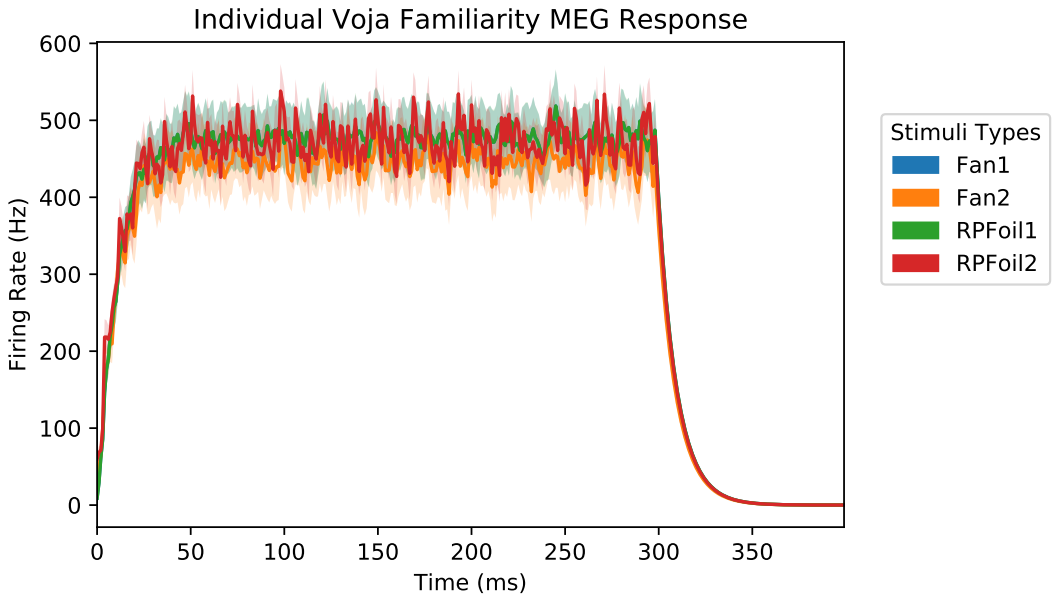

FIGURE 4.3: The MEG response of the WPR task familiarity stage using an associative memory trained with Voja. All input types overlap similar to the experimental data. Shaded areas represent 95% boot-strapped confidence intervals.

desired MEG response. Specifically, a population trained with Voja (intercepts set to 0.3 and with 5 repetitions of the FAN1 and FAN2 target vectors during training) will respond more to FAN2 than FAN1 pairs, as shown in Figure 4.4. This is the opposite of the response observed experimentally.

The greater MEG response for FAN2 vs. FAN1 vectors, seen in Figure 4.4, occurs because encoders learned using Voja are drawn to the more "clustered" FAN2 pairs. Where "clustering", given a set of M-dimensional vectors $V = \{\mathbf{v_1}, \mathbf{v_2} \ldots \mathbf{v_n}\}$, is defined as:

$$\begin{aligned} \text{clust}\,(V) &= \frac{1}{n} \sum_{V} V \cdot \overline{V} \\ \overline{V} &= \left\| \frac{1}{n} \sum_{i=1}^{n} V_n \right\|. \end{aligned} \tag{4.1}$$

Intuitively, FAN2 pairs are more clustered than FAN1 pairs, by virtue of FAN2 pairs reusing words, causing the pairs to be more similar to each-other. This is shown more explicitly by plotting the clustering 16 FAN1 and 16 FAN2 pairs

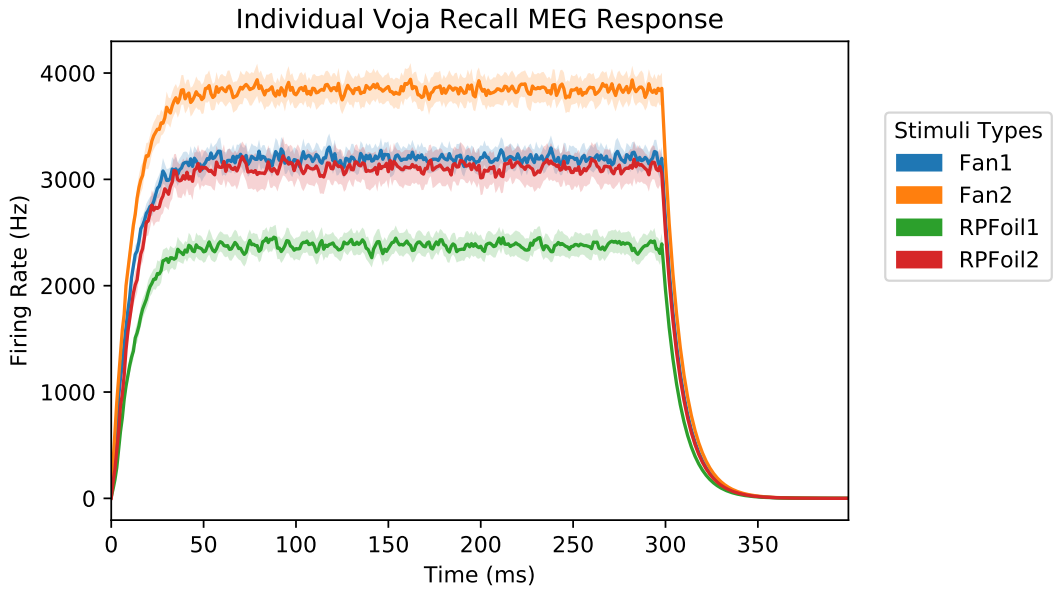

FIGURE 4.4: The MEG response of the WPR task recall stage using an associative memory trained with Voja failing to match the experimental data. Shaded areas represent 95% boot-strapped confidence intervals.

of increasing dimensions in Figure 4.5. Regardless of the dimensionality used, FAN2 is significantly more clustered than FAN1.

To summarize, Voja moves encoders towards more frequently activated sections of the representation space. This results in encoders responding more to clustered FAN2 vectors, whereas the experimental data shows more response for the spread out FAN1 vectors.

4.2.1 Prioritizing Novelty with Mixed Voja

Given Voja's dysfunction, a new encoder learning rule prioritizing novel, unfamiliar, previously unseen vectors is required. This is essentially the opposite of Voja, thus the learning rule Neg Voja is proposed. As implied by the name, Neg Voja is the Voja learning rule, but with a negative learning rate. In addition, it includes a normalization term. Without this normalization, a negative learning rate would move all encoders outside of the stimuli radius, ensuring

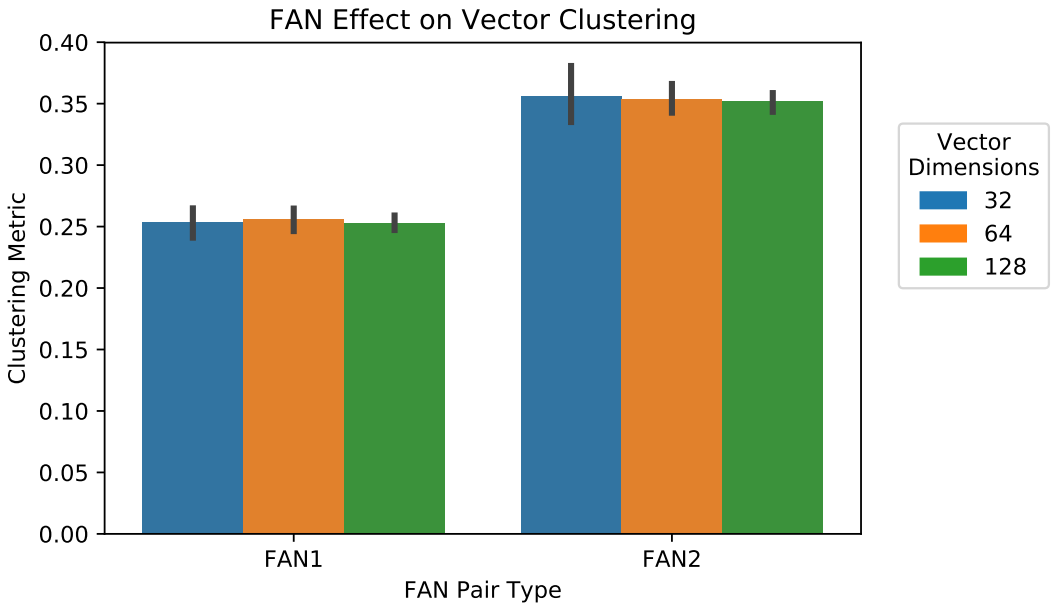

FIGURE 4.5: Comparison of FAN1 vs. FAN2 word-pair set clustering averaged across 20 distinct random seeds. Error bars represent 95% boot-strapped confidence intervals.

neurons would never fire for any possible stimulus. To keep the encoders inside the stimuli radius, the encoders are normalized to radius length r after being updated. In the case of representing semantic pointers, which are of unit length, $r = 1$. To accomplish this normalization, Equation 2.10 is modified as follows:

$$\Delta \mathbf{e}_j = r \cdot \langle \mathbf{e_j} + \eta a_j (\mathbf{x} - \mathbf{e}_j) \rangle \,, \tag{4.2}$$

where $\langle \mathbf{x} \rangle = \frac{\mathbf{x}}{\|\mathbf{x}\|}$.

Given a high enough learning rate and long enough training time, the learning rule describe in Equation 4.2 causes encoders to diverge until no encoders fire for any previously seen input. This is a problematic scenario. Instead, encoders should be distributed between converging to and diverging from inputs. To satisfy this need, an alternative Mixed Voja is proposed in Equation 4.3. It modifies both the the rate a_j and the stimuli distance $(\mathbf{x} - \mathbf{e}_j)$

factors of the original formulation of Neg Voja in Equation 4.2:

$$\delta = \begin{cases} 0, & \text{if } \|\mathbf{x} - \mathbf{e}_j\| > \delta^{\max}, \\ (\mathbf{x} - \mathbf{e}_j), & \text{otherwise.} \end{cases}$$
$$a_j^{\max} = a_j(r) \tag{4.3}$$
$$\Delta \mathbf{e}_j = \eta \left(\frac{a_j}{a_j^{\max}} - \tau \right) \delta.$$

To compensate for the heterogenous maximum firing rates of neurons, Mixed Voja uses the ratio of the neuron firing rate with its maximum firing rate $\frac{a_j}{a_j^{\max}}$. This ratio is thresholded by a scalar constant $0 < \tau < 1$. The expression $(\frac{a_j}{a_j^{\max}} - \tau)$ means neurons firing close to their maximum rate converge to the stimuli x, while neurons not firing as much diverge.

This modification to the rate term of Voja a_j necessitates modifications to the distance expression $(\mathbf{x} - \mathbf{e}_j)$ as well. In Voja, when $a_j \to 0$, $\Delta \mathbf{e}_j \to 0$, which means only firing neurons are affected. This is no longer the case in Equation 4.3. Instead, as $a_j \to 0$, $\Delta \mathbf{e}_j \to -\tau$. This means all neurons firing such that $\frac{a_j}{a_j^{\max}} < \tau$ would diverge to maximize $(\mathbf{x} - \mathbf{e}_j)$. Consequently, to limit the effect area of x, the stimuli distance is thresholded by the scalar constant $\delta^{\max}$ such that $\|\mathbf{x} - \mathbf{e}_j\| > \delta^{\max}$. This causes deviating encoders to converge upon $x \pm \delta^{\max}$.

Both the firing ratio τ and the scalar distance threshold $\delta^{\max}$ are chosen to match the MEG data, as discussed in Section 4.3.1.

Biological Plausibility

Compared to Voja, Mixed Voja could also be seen as equally biologically plausible, given that it only relies on information local to the neuron. However, whether these computations, specifically the normalization of $\Delta \mathbf{e}_j$, the firing ratio calculation $\frac{a_j}{a_j^{\max}}$ and the thresholding via $\delta^{\max}$, are able to be performed

in a single neuron is a topic for further investigation. Additionally, whether an analogue to this learning rule exists in neuroscience is unclear.

4.3 Results of Voja Alternatives

4.3.1 MEG response

Two models are simulated to contrast Neg Voja and Mixed Voja. Each model consists of a "memory" consisting of a neural ensemble of 500 neurons with one of the learning rules applied to the encoders. The model is trained by showing each word pair of a target word set (16 FAN1 targets + 16 FAN2 targets) a single time to the memory. The MEG response is determined as described in Section 4.1.

As shown in Figure 4.6, both Neg Voja and Mixed Voja are able to match the MEG response seen during the task using hand-tuned parameters shown in Table 4.1.

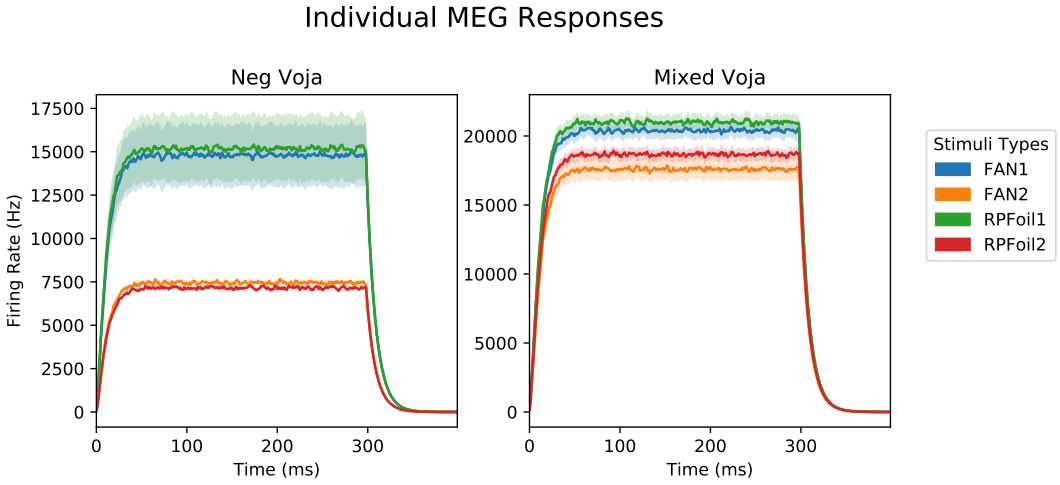

FIGURE 4.6: The MEG responses of Neg Voja and Mixed Voja hand-tuned recall models given a single word-set. Although exact responses differ, the overlap of foil and targets for each FAN are maintained. Shaded areas represent 95% boot-strapped confidence intervals.

TABLE 4.1: Parameter settings for Neg Voja and Mixed Voja distinguishing which parameters were set statically and which were tuned to create Figure 4.6. $U(x_1, x_2)$ signifies randomly sampled from the uniform distribution ranging from x_1 to x_2.

Learning Rule	Parameter Setting	Tuned or Static
Neg Voja	$\eta = 5\text{e}{-6}$	Tuned
	$x^{\text{cept}} = U(0, 1.5)$	Static
Mixed Voja	$\eta = -100$	Static
	$x^{\text{cept}} = 0.2$	Static
	$\tau = 0.1$	Tuned
	$\delta^{\text{max}} = 1$	Tuned

All other ensemble parameters use the Nengo and Nengolib defaults. Specifically, LIF neuron model parameters, as well as the distribution of the maximum firing rates, use the Nengo defaults. All neural ensembles are initialized with number-theoretic uniformly distributed encoders provided by Nengolib. These encoders, which are less clustered than uniformly sampled encoders, are shown in Figure 4.7. Encoder clumping would cause MEG results to vary more per learning instance as clumps migrated together, requiring more instances to confirm the same result.

Only Mixed Voja generalizes from a single word-set to multiple word sets when matching the MEG signal and using the previously hand-tuned parameters. As shown in Figure 4.8, the same Mixed Voja parameters match the MEG signal generated from multiple word-sets, while Neg Voja fails completely.

4.3.2 Behavioural responses

To match the WPR behavioural the learned encoders from Section 4.3.1 are used for the Recall memory in Figure 4.1, while the decoders of the Recall memory were trained during a single presentation of all target pairs using PES. The output of the trained Recall memory is then used as input to the

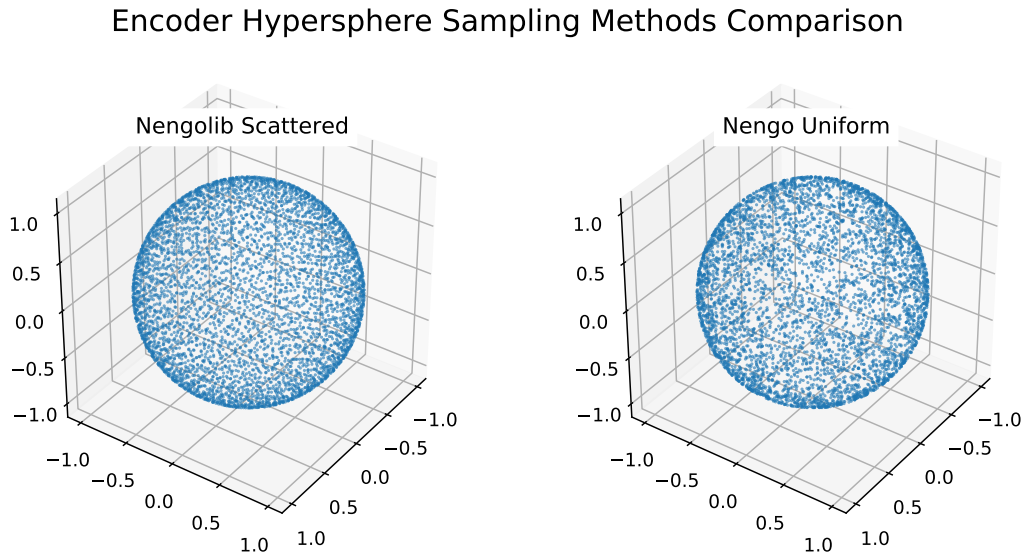

FIGURE 4.7: Comparison of encoders chosen by the Nengolib scattering process and the default Nengo uniform sampling. Fewer gaps and clusters can be seen in the superiorly uniform Nengolib sphere.

Compare network as described in Section 4.2. As shown in Figure 4.9, the scalar output of the Compare network is then used to classify whether the original input from vision was a target or a RPFoil.

The final decision is accomplished by integrating over time, comparing two conditions and choosing whichever is satisfied first:

$$c - p > \tau_{\text{foil}} \text{ indicates RPFoil}$$

$$1 - c > \tau_{\text{targ}} \text{ indicates Target},$$

where c is the output of the Compare network, while $p = 0.3$ is used to bias the model toward RPFoil or Target decision. The integral is performed by a neural integrator, where $g(x) = x(t)$ in Equation 2.8. Although the threshold constants τ_x could be learned during training using PES, for the sake of simplicity they are set heuristically in proportion to the mean target decision outcome ζ. Given the Compare network output $c^{\text{all}}(t) = \{c(t)_1, c(t)_2, \ldots c(t)_n\}$,

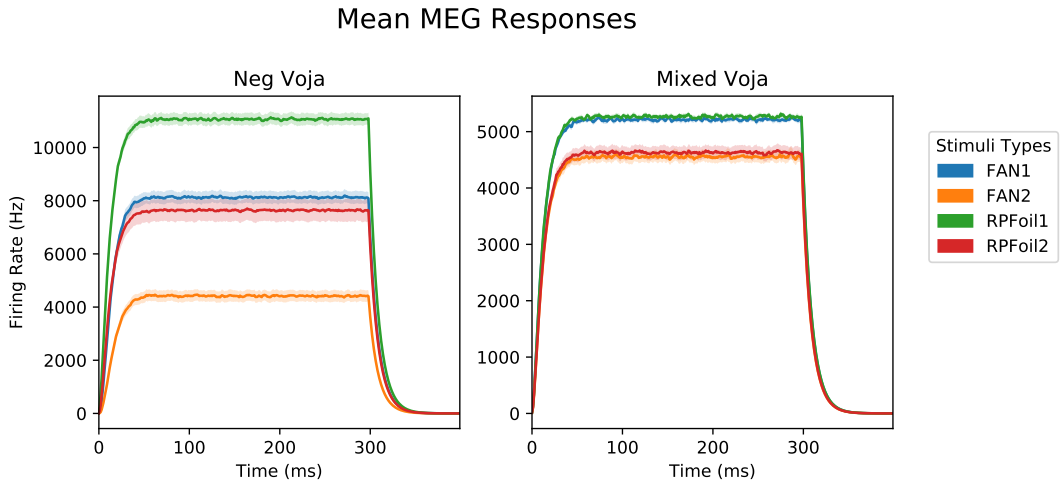

FIGURE 4.8: The MEG responses of the Neg Voja and Mixed Voja recall models averaged over 10 separate training iterations with different word sets. Only Mixed Voja created the same MEG response across different word sets.

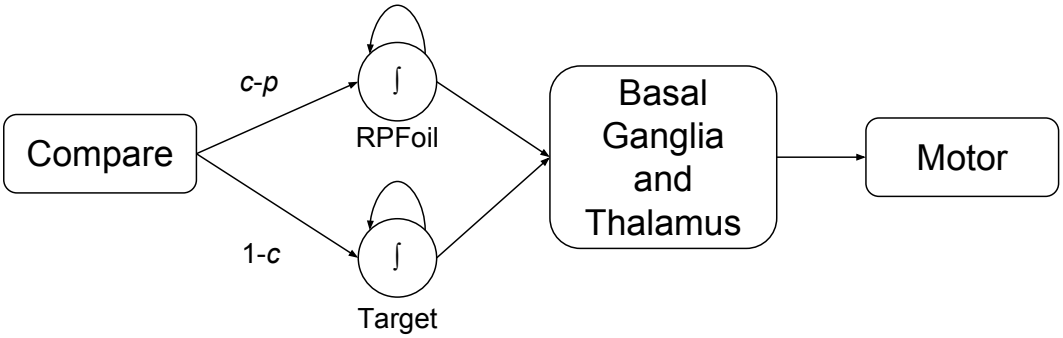

FIGURE 4.9: Module for decision making.

we can define the total foil and target integrator output for input $c(t)_n$ as:

$$c_n^{\text{foil}} = \int_0^{t_{\text{sim}}} c(t)_n - p \, \mathrm{d}t$$
$$c_n^{\text{targ}} = \int_0^{t_{\text{sim}}} 1 - c(t)_n \, \mathrm{d}t.$$

Thus, the mean of these total integrated outputs is:

$$\zeta_{\text{foil}} = \frac{1}{n}\sum_{i=1}^{n} c_i^{\text{foil}}$$
$$\zeta_{\text{targ}} = \frac{1}{n}\sum_{i=1}^{n} c_i^{\text{targ}}.$$

Allowing for the threshold constants to be set as:

$$\tau_{\text{foil}} = 1.6 * \zeta_{\text{foil}}$$
$$\tau_{\text{targ}} = 0.65 * \zeta_{\text{targ}}.$$

where t_{sim} is the simulation time for each target comparison. The τ_x scaling constant and simulation time were set arbitrarily.

Whichever condition is satisfied first causes the decision to be made. Typically, this decision would be translated into a motor action by the basal ganglia and thalamus performing a WTA operation on the two conditions. However, this operation, along with the possibility of neither condition being satisfied, is left out of the model.

Although data on the training accuracy during rehearsal is available, only test data was considered. This is due to the MEG signal only being collected for the testing phase. Thus, there is insufficient supporting evidence to determine how the encoders changed during training and how this would effect error rates during training. Additionally, as discussed briefly in Section 4.1, the behavioural performance during training is hypothesized to come from a different network trained in parallel that is not modeled in this thesis.

The Mixed Voja memory are not able to match the experimental error rate and reaction time data, as shown in Figures 4.10 & 4.11.

In the experimental data, FAN2 targets and foils had higher reaction times and error rates. This is expected, FAN2 targets and foils have more half-matching pairs causing confusion than FAN1 targets and foils, as shown in

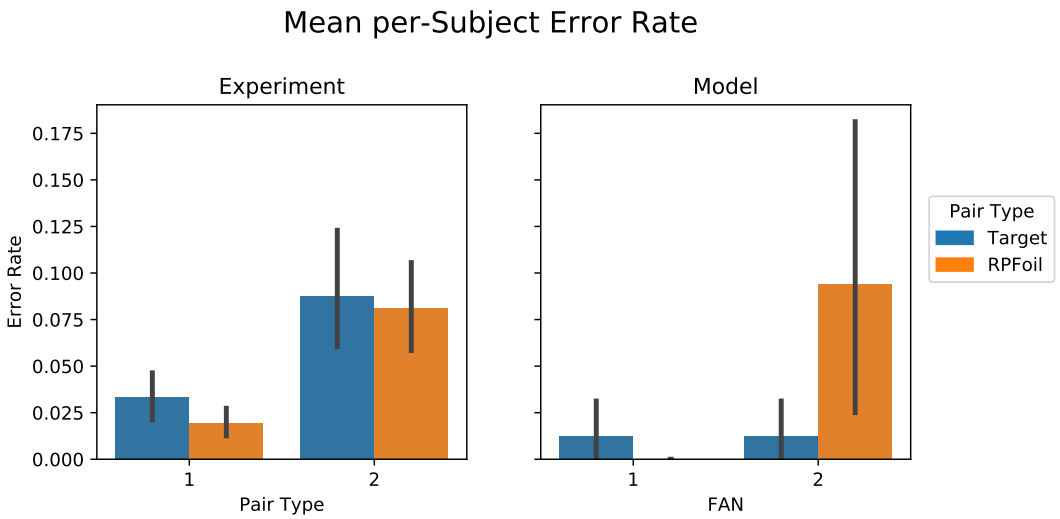

FIGURE 4.10: Average rate of word-pair misclassification. Experimental data uses the per-subject rate of over 14 blocks. Model uses an error-rate averaged over 10 different generated word-sets. Error-bars represent 95% boot-strapped confidence intervals.

Table 1.4. This half-matching pair confusion also hints at the clustering seen in Section 4.2. However, in the model, the reaction times were identical across stimuli types. Additionally, the model's error-rate was not able to match the experimental error-rate between FAN2 targets and RPFoils.

The most likely cause for these mismatches is the intercept selection and lack of recurrent connections.

The error-rate of each population was heavily dependent on the intercepts. If intercepts were set lower, the decoders learned would decode targets and RPFoils with the same confidence. Setting the intercept too high causes neurons to be so selective that they do not respond to any inputs. This can be overcome by adding more neurons, however the number required quickly becomes computationally unfeasible using traditional hardware. Finding an ideal distribution of intercepts or learning intercepts over time is beyond the scope of this thesis.

The error-rate and reaction-time are both related to the output confidence of the memory. However, the reaction-time is also connected to the output

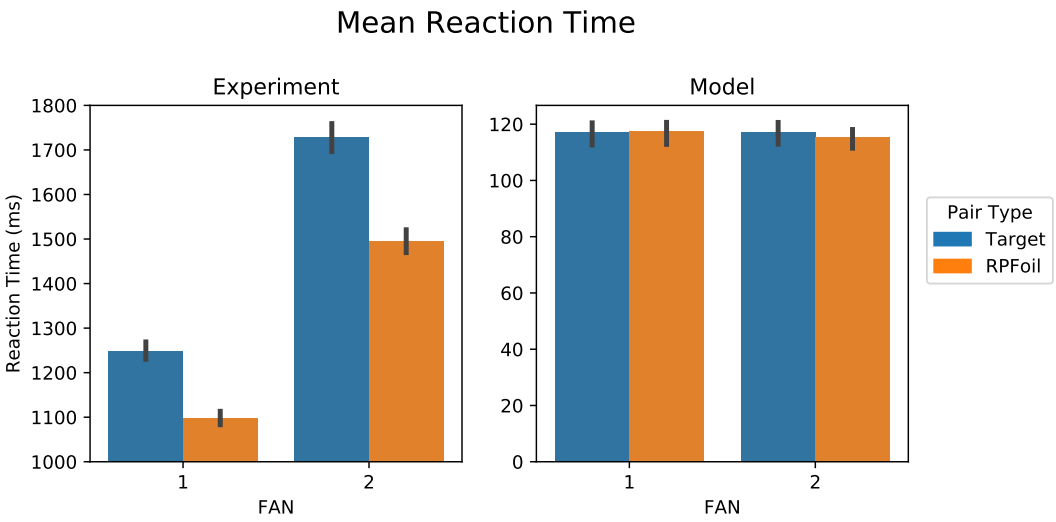

FIGURE 4.11: Average reaction-time for word-pair recognition. Experimental data is the per-subject rate of over 14 blocks. Model uses an error-rate averaged over 10 different generated word-sets. Error-bars represent 95% boot-strapped confidence intervals.

dynamics of the memory. Typically, "hard-coded" heteroassociative memories have recurrent connections enabling advance computations, such as Winner-Take-All (Gosmann, Voelker, and Eliasmith, 2017), which create non-linear outputs. These non-linear outputs may be what is required to match the reaction-time data.

Further model enhancement and exploration to take these changes into account to better match the behavioural data are covered in the next chapter.

Chapter 5

Discussion and Future Work

5.1 Discussion

In this thesis, the ASP and WPR results were modeled using spiking neural networks. These models suggest that associative memories are used differently in different parts of the brain allowing for cognitive systems to improve with practice. In the prefrontal lobe, where the recall addition strategy is thought to take place, convergence for quick transform memorization is prioritized. Whereas in the temporal lobe, where word-pair recognition occurred, novelty is prioritized over convergence. In both cases, encoder modification in response to stimuli is critical, but required modifications to match specific neural phenomena. It also seems unlikely that single encoder learning rule would satisfy both these specialized cases.

In the ASP case Voja is adequate, because the space of the inputs is limited. Specifically, it is limited to a concatenation of two digit SPs sampled from an orthonormal basis, despite the digits themselves and their combinations not being known ahead of time. It is assumed as prior knowledge that no symbol containing three concatenated digit symbols or a novel, non-orthonormal, digit input (such as a fraction or a decimal) will be presented. Thus, Voja and a specific setting of intercepts are sufficient to learn the mapping from two concatenated digit SPs to another digit SP. If Mixed Voja were applied, it

would move encoders to prepare for novel, unseen, inputs which would never come and may disrupt correctly converged encoders.

In contrast to ASP, the WPR task benefits from Mixed Voja, because there is no finite set of words. Additionally, there is no finite number of combinations. For example, FAN3 or FAN4 word-pairs could be input. The implicit assumption of Mixed Voja, that future inputs will be dissimilar from those already learned, is more likely to be beneficial in WPR than in ASP.

This contrast between modeling approaches implicitly claims the brain is able to manage memory based on the category of expected inputs (finite set of digits vs. infinite set of words) and the task (recall vs. recognition). Potential ways of validating this claim and other avenues for further investigation are covered in the next section.

5.2 Future Work

5.2.1 Different Vector Representations

In both the addition strategy transition task and the word-pair task, a tension exists between representational choices and neural network design.

In the counting strategies task, concatenation is chosen, but does not capture the subtleties of numerical representation. For instance, there is evidence that for numerical size comparisons, neurons are tuned to a log-scale and are sensitive to task saliency (Nieder and Dehaene, 2009).

In the word-pair recognition task, addition is chosen and the memory performs recall. Other functions could be computed by the associative memory and a variety of other vector combination operations could represent various memorization strategies. However, the many possible permutations

of function and encodings were not systematically explored. Creating high-dimensional vector spaces optimized for various representations is an open problem with no clear systematic approach.

Given the representational choices made, this thesis makes the strong claim of associative memory learning relying on different mechanisms depending on the nature of the association. Specifically, when the mapping of a limited set of two inputs to a novel output is memorized, Voja is used. When an unknown set of inputs are combined and recognized, Mixed Voja is used. This is supported by the WPR MEG results, however there is no MEG signal for the ASP task to justify the use of Voja. This data could be collected via an experiment of similar design to Borst et al. (2013), but altered to contrast the two types of learning.

For instance, instead of learning word-pairs, subjects could learn arithmetic facts using either, a different number base (binary, ternary), or a novel operation symbol ($x \circ y$ representing $2x + y$). One day after the subjects have practiced, the subjects would be tested with both previously studied facts and novel facts. If a stronger MEG signal is read from the parietal area for repeated facts than novel facts, then Voja is validated. Otherwise, if the recall occurs in a different brain area or a different MEG signal is seen, a new encoder learning rule should be investigated.

5.2.2 Improving Word-Pair Recognition Behavioural Responses

Although the WPR model was able to match the MEG signal by using the Mixed Voja learning rule, it did not to replicate the behavioural error-rates and reaction times. There are multiple avenues of investigation that might lead to the improvements necessary to address these behavioural constraints as well. Instead of assigning neuron encoders based on their firing rates, it would make more sense for the neuron's encoders to converge to or diverge

from a stimulus depending on the decoding error of the given input. This error signal could also be used to modify the intercepts in addition to the encoders.

For example, intercepts could be adjusted proportional to the decoding error, allowing the memory to remain flexible and able to learn new inputs. Given that intercepts greatly affect the accuracy of decoding, the different clustering and learning results of FAN1 and FAN2 may then be sufficient to create the difference in target reaction times and error-rates desired. However, the current dataset is limited given that it only provides MEG data during the test phase. This exploration of learning methods would be greatly helped by characterizing the changes to MEG responses during learning. For example, if changes in MEG response were correlated with recognition error, the error-driven approach to encoder learning would be validated.

As mentioned in Section 4.3.2, the memory designs in this thesis are single-layer feed-forward. However, "hard-coded" heteroassociative memories in the SPA typically use recurrent connections and multiple layers (Gosmann, Voelker, and Eliasmith, 2017). Multiple attempts at leveraging recurrent connections were made while writing this thesis.

For example, a learning rule which implemented a variant of the Bienenstock, Cooper, Munro (BCM) learning rule (Bienenstock, Cooper, and Munro, 1982) on all-to-all recurrent weights was proposed. In the implementation, recurrent connections were strengthened between neurons firing together for a certain input, while weakening connections where neuron firing was not correlated. However, none of the recurrent learning rules, including this recurrent BCM rule, offered any significant improvement in terms of robustness to noise or output confidence. This is possibly due to the fact that neural firing rates, instead of firing rate ratios from Mixed Voja, were used. Alternatively, maybe neuron firing should be considered in the context of the decoding error.

Finally, instead of recurrence, it may be worth investigating a multi-layer network where errors are propagated using a spiking version of Feedback

Alignment (Hunsberger and Eliasmith, 2017). Hypotheses aside, the challenge of identifying neurons tuned to each input and learning recurrent connections to compute functions, such as WTA, remains unsolved.

Regardless of representational and associative memory architectural assumptions made, this thesis has shown that the use of memories leveraging the Voja learning rule and their modification is a promising direction for learning critical components of cognitive models allowing them to improve with practice.

Bibliography

Ahlfors, S. P. and C. Wreh (2015). “Modeling the effect of dendritic input location on MEG and EEG source dipoles”. In: *Med Biol Eng Comput* 53.9, pp. 879–887.

Bekolay, Trevor et al. (2014). “Nengo: A Python tool for building large-scale functional brain models”. In: *Frontiers in Neuroinformatics* 7.48. ISSN: 1662-5196. DOI: 10.3389/fninf.2013.00048.

Bienenstock, Elie L, Leon N Cooper, and Paul W Munro (1982). “Theory for the development of neuron selectivity: orientation specificity and binocular interaction in visual cortex”. In: *Journal of Neuroscience* 2.1, pp. 32–48.

Borst, Jelmer P and John R Anderson (2015). “The discovery of processing stages: Analyzing EEG data with hidden semi-Markov models”. In: *NeuroImage* 108, pp. 60–73.

Borst, Jelmer P, Avniel S Ghuman, and John R Anderson (2016). “Tracking cognitive processing stages with MEG: A spatio-temporal model of associative recognition in the brain”. In: *NeuroImage* 141, pp. 416–430.

Borst, Jelmer P et al. (2013). “Stages of processing in associative recognition: Evidence from behavior, EEG, and classification”. In: *Journal of cognitive neuroscience* 25.12, pp. 2151–2166.

Choo, Xuan (2018). “Spaun 2.0: Extending the Worlds Largest Functional Brain Model”. PhD thesis. University of Waterloo. URL: http://hdl.handle.net/10012/13308.

DeWolf, Travis and Chris Eliasmith (2013). "A neural model of the development of expertise". In: *12th International Conference on Cognitive Modelling*. Ed. by Robert L. West & Terrence C. Stewart. Ottawa, Ontario: Carleton University, pp. 119–124.

Doya, Kenji (2002). "Metalearning and neuromodulation". In: *Neural Networks* 15.4, pp. 495–506.

Eliasmith, Chris (2013). *How to build a brain: A neural architecture for biological cognition*. New York, NY: Oxford University Press.

Eliasmith, Chris and Charles H. Anderson (2003). *Neural engineering: Computation, representation, and dynamics in neurobiological systems*. Cambridge, MA: MIT Press.

Eliasmith, Chris et al. (2012). "A large-scale model of the functioning brain". In: *Science* 338, pp. 1202–1205. DOI: 10.1126/science.1225266.

Gosmann, Jan (2018). "An Integrated Model of Context, Short-Term, and Long-Term Memory". PhD thesis. University of Waterloo.

Gosmann, Jan, Aaron R. Voelker, and Chris Eliasmith (2017). "A Spiking Independent Accumulator Model for Winner-Take-All Computation". In: *Proceedings of the 39th Annual Conference of the Cognitive Science Society*. London, UK: Cognitive Science Society. URL: https://mindmodeling.org/cogsci2017/papers/0405/index.html.

Hunsberger, Eric and Chris Eliasmith (2017). "Deep learning in spiking LIF neurons". In: *Cosyne Abstracts*. Salt Lake City USA. URL: http://cosyne.org/cosyne17/Cosyne2017_program_book.pdf.

Ischebeck, Anja et al. (2007). "Imaging early practice effects in arithmetic". In: *Neuroimage* 36.3, pp. 993–1003.

Knight, Jamie et al. (2016). "Efficient SpiNNaker simulation of a heteroassociative memory using the Neural Engineering Framework". In: *The 2016 International Joint Conference on Neural Networks (IJCNN)*. Vancouver, British Columbia: IEEE, pp. 5210–5217.

Kucian, Karin and Michael von Aster (2015). "Developmental dyscalculia". In: *European journal of pediatrics* 174.1, pp. 1–13.

Lebiere, Christian (1999). "The dynamics of cognition: An ACT-R model of cognitive arithmetic". In: *Kognitionswissenschaft* 8.1, pp. 5–19.

MacNeil, David and Chris Eliasmith (2011). "Fine-tuning and the stability of recurrent neural networks". In: *PloS one* 6.9, e22885.

Nieder, Andreas and Stanislas Dehaene (2009). "Representation of number in the brain". In: *Annual review of neuroscience* 32, pp. 185–208.

Oja, Erkki (1989). "Neural networks, principal components, and subspaces". In: *International journal of neural systems* 1.01, pp. 61–68.

Rivera, Susana M et al. (2005). "Developmental changes in mental arithmetic: evidence for increased functional specialization in the left inferior parietal cortex". In: *Cerebral cortex* 15.11, pp. 1779–1790.

Rubinsten, Orly and Avishai Henik (2009). "Developmental dyscalculia: heterogeneity might not mean different mechanisms". In: *Trends in cognitive sciences* 13.2, pp. 92–99.

Sarnecka, Barbara W and Susan Carey (2008). "How counting represents number: What children must learn and when they learn it". In: *Cognition* 108.3, pp. 662–674.

Sharma, Sugandha, Sean Aubin, and Chris Eliasmith (2016). "Large-scale cognitive model design using the Nengo neural simulator". In: *Biologically Inspired Cognitive Architectures* 17, pp. 86–100. ISSN: 2212-683X. DOI: http://dx.doi.org/10.1016/j.bica.2016.05.001. URL: http://www.sciencedirect.com/science/article/pii/S2212683X16300317.

Siegler, Robert S (1987). "The perils of averaging data over strategies: An example from children's addition." In: *Journal of Experimental Psychology: General* 116.3, p. 250.

Stewart, Terrence C., Xuan Choo, and Chris Eliasmith (2010). "Symbolic Reasoning in Spiking Neurons: A Model of the Cortex/Basal Ganglia/Thalamus

Loop". In: ed. by Stellan Ohlsson and Richard Catrambone. Portland, Oregon: Cognitive Science Society, pp. 1100–1105.

Stewart, Terrence C., Yichuan Tang, and Chris Eliasmith (2011). "A Biologically Realistic Cleanup Memory: Autoassociation in Spiking Neurons". In: *Cognitive Systems Research* 12, pp. 84–92. DOI: 10.1016/j.cogsys.2010.06.006. URL: http://dx.doi.org/10.1016/j.cogsys.2010.06.006.

Thorgeirsson, Sverrir, Terrence C. Stewart, and Chris Eliasmith (2018). "Analysis of Learning Action Selection Parameters in a Neural Cognitive Model". In: *20th International Conference on Cognitive Modelling*.

Voelker, A. R. (2018). *Nengolib: Additional extensions and tools for modelling dynamical systems in Nengo*. URL: https://github.com/arvoelke/nengolib.

Voelker, Aaron Russell, Eric Crawford, and Chris Eliasmith (2014). "Learning large-scale heteroassociative memories in spiking neurons". In: *Unconventional Computation and Natural Computation*. Ed. by Steffen Kopecki Oscar H. Ibarra Lila Kari. London, Ontario: Springer International Publishing.

Waskom, Michael et al. (2016). *seaborn: v0.7.0 (January 2016)*. DOI: 10.5281/zenodo.45133. URL: https://doi.org/10.5281/zenodo.45133.

Willshaw, David J, O Peter Buneman, and Hugh Christopher Longuet-Higgins (1969). "Non-holographic associative memory." In: *Nature*.

Wu, Yingquan and Dimitris A Pados (2000). "A feedforward bidirectional associative memory". In: *IEEE Transactions on Neural Networks* 11.4, pp. 859–866.