A Spiking Neural Model of the n-Back Task

Jan Gosmann (jgosmann@uwaterloo.ca)
Chris Eliasmith (celiasmith@uwaterloo.ca)
Centre for Theoretical Neuroscience, University of Waterloo
200 University Avenue West, Waterloo, ON, N2L 3G1, Canada

Abstract

We present a computational model performing the n-back task. This task requires a number of cognitive processes including rapid binding, updating, and retrieval of items in working memory. The model is implemented in spiking leaky-integrate-and-fire neurons with physiologically constrained parameters, and anatomically constrained organization. The methods of the Semantic Pointer Architecture (SPA) are used to construct the model. Accuracies and reaction times produced by the model are shown to match human data. Namely, characteristic decline in accuracy and response speed with increase of n is reproduced. Furthermore, the model provides evidence, contrary to some past proposals, that an active removal process of items in working memory is not necessary for an accurate performance on the n-back task.

Keywords: n-back task; neural engineering; computational neuroscience; vector symbolic architecture

Introduction

Reasoning about the world is a cognitive skill mediated by a large number of cognitive processes, including the ability to store information in working memory and quickly update this information in a controlled manner. The n-back task has been used extensively to investigate these features of cognitive processing. Thus, characteristic patterns in accuracy and reaction time data in this executive control test of working memory have been well validated.

Despite its wide use in experiments, only few computational models of this task exist. Here we propose the first model of the n-back task implemented in a spiking neural network. Our motivation is two-fold. First, we hope to gain a better understanding how the brain might process and update contents in working memory. Second, the model might lead to new insights about the interaction of different cognitive mechanisms in performing more complex working memory tasks. By using spiking neural networks we can tie many model parameters to biological constraints, diminishing the parameter space.

In the n-back task the test subject is presented with a list of stimuli (usually letters or spatial locations) one item at a time. For each stimulus the subject has to indicate whether he or she saw the current item exactly n positions before. As an example consider the sequence j-n-**j**-j-k. In a 2-back task the 'j' in bold would be a target, whereas the other letters in the sequence would be considered distractors.

A number of executive control processes, including updating, modification, maintenance, and matching of memory contents, are of importance in this task. Specifically, the sequence of recent stimuli has to be stored in working memory for some duration. For each new stimulus the list has to be updated. This requires rapid binding of the new item to its position in the list and an update of the position of remembered items. At the same time, older stimuli become irrelevant. Preserving them in memory could interfere with new stimuli and degrade performance. Because of this, an active removal or unbinding of old items from memory is often assumed (e.g., Juvina & Taatgen, 2007; Szmalec, Verbruggen, Vandierendonck, & Kemps, 2011). However, the model presented here shows that active removal is not essential. Finally, a recollection process is needed to recall the item n positions back and compare it to the current stimulus.

To construct this model, we employ the methods of the Semantic Pointer Architecture (SPA; Eliasmith, 2013). Among other things, the SPA proposes methods for representing symbol-like information, controlling the flow of information, and implementing serial working memory in biologically plausible spiking neural networks. It relies heavily on the Neural Engineering Framework (NEF; Eliasmith & Anderson, 2003) for constructing such networks.

This paper is organized as follows: First, we will give a short overview of the NEF. This is followed by a description of the methods used to represent symbol-like items and their positions in the model. Next we introduce the n-back model and present the simulation results. Finally, we conclude with a discussion of these results.

The Neural Engineering Framework

The Neural Engineering Framework (NEF) provides methods for implementing algorithms on abstract vector spaces in spiking neural networks (Eliasmith & Anderson, 2003). There are two key components to the NEF. First, it describes how an ensemble of neurons can form a distributed representation of a vector space. Second, it specifies how connections between neural populations can implement transformations and computations on vectors in those spaces.

Equation 1 states how a vector $\boldsymbol{x}(t)$, varying over time, is encoded by an ensemble of neurons. Each neuron i has a preferred direction or encoding vector $\boldsymbol{e}_i$, a gain α_i, and a background or bias current J_i^{bias}. From these parameters the neuron's input current $J_i(\boldsymbol{x}(t))$ is determined and passed through a non-linearity G_i that models the spiking response of a neuron to current input. The vector is thus encoded into the activity of the neurons $a_i(t)$.

$$a_i(t) = G_i[J(t)], \quad J(t) = \alpha_i \left(\boldsymbol{e}_i \cdot \boldsymbol{x}(t)\right) + J_i^{\text{bias}} \tag{1}$$

Different neuron models with a varying degree of realism can be used as the non-linearity G_i. For this model, we use spiking leaky integrate-and-fire (LIF) neurons as a good balance of biological realism and computational efficiency. In a

LIF neuron, the input currents are integrated over time with an exponential decay determined by the neuron's membrane time constant. Once a specified threshold is reached the neuron generates a spike. Afterwards it is reset to its starting voltage and held there for the duration of the absolute refractory period. This spiking neuron model produces a spike train of the form $a_i(t) = \sum_s \delta(t - t_{i,s})$ where $t_{i,s}$ is the time of spike s from neuron i.

To reconstruct, or decode, an encoded vector from the activity (i.e., the spike train) of the neuron, the spikes are convolved with a low pass filter $h(t)$ that accounts for the post-synaptic response of receiving neurons, and then weighted by linear decoding weights $\boldsymbol{d}_i$. To be clear, the low pass filter models the post-synaptic current produced in the post-synaptic neuron by an incoming action potential. A typical choice for this filter, also used here, is a decaying exponential with some synaptic time constant τ. The decoded value $\hat{\boldsymbol{x}}(t)$ is then given by

$$\hat{\boldsymbol{x}}(t) = \sum_i \boldsymbol{d}_i (a_i * h)(t). \tag{2}$$

The decoding weights $\boldsymbol{d}_i$ are obtained through a regularized least-squares optimization of the decoding error $\langle \|\hat{\boldsymbol{x}} - \boldsymbol{x}\| \rangle_{\boldsymbol{x}}$ over a set of inputs $\boldsymbol{x}$. Consequently, the representation of the vector x by the neural population a is defined by the combination of the encoding (Eq. 1) and decoding (Eq. 2) equations.

However, these equations only describe how a vector can be encoded and decoded in a single ensemble. To connect two ensembles in a communication channel, the synaptic weight matrix is given by the pre-synaptic decoders and post-synaptic encoders as $W_{ij} = \boldsymbol{e}_i \boldsymbol{d}_j^\top$. Thus, the input current of the post-synaptic neuron is obtained as $J_i(t) = \alpha_i W_{ij}(a_j * h)(t) + J_i^{\text{bias}}$.

Transformations, $f(\boldsymbol{x})$, of the represented vector across neural connections can be implemented analogously to the communication channel. Instead of obtaining the decoding weights with the least-squares optimization of $\langle \|\hat{\boldsymbol{x}} - \boldsymbol{x}\| \rangle_{\boldsymbol{x}}$ the decoding error $\langle \|\hat{\boldsymbol{x}} - f(\boldsymbol{x})\| \rangle_{\boldsymbol{x}}$ is used. Linear and low-order polynomials can be approximated best, whereas for high-order polynomials and less smooth function the approximation will be less precise. A desired accuracy can be reached in general by increasing the number of neurons in the ensemble (Eliasmith & Anderson, 2003).

Finally, it is possible to implement differential equations with recurrent connections. Note that the synaptic low-pass filter $h(t)$ will influence the recurrent connection dynamics. To implement $\frac{dx}{dt} = f(x)$ the connection weights for $f'(x) = \tau f(x) + x$ have to be computed with the approach given above (Eliasmith & Anderson, 2003, pp. 222–225).

Symbol-like representation

With the NEF we are able to represent vectors with neural ensembles and perform calculations on these with connections between neural populations. To represent structured, conceptual or symbolic information the SPA employs a specific type of Vector Symbolic Architecture (VSA; Gayler, 2003). Common to VSA approaches is that they represent individual concepts as vectors and combine vectors with nonlinear and linear operators to perform binding, all of which can easily be implemented by neurons with the NEF.

In addition to representing symbol-like concepts it has to be possible to perform appropriate syntactic operations on these concepts. The specific operator (i.e., circular convolution) we employ was first suggested by Plate (1995) for encoding syntactic structure in vector spaces and termed as Holographic Reduced Representations[1] (HRRs). The SPA provides a general characterization of compressed neural representations as 'semantic pointers'. Circular convolution is one of the compressive operations used in the SPA, and so the symbol-like representations used here are one example of semantic pointers.

Given two vectors $\boldsymbol{v}$ and $\boldsymbol{w}$ we perform a union like or superposition operation yielding a new vector $\boldsymbol{u}$ similar to both $\boldsymbol{v}$ and $\boldsymbol{w}$ by simple addition

$$\boldsymbol{u} = \boldsymbol{v} + \boldsymbol{w}. \tag{3}$$

Another important operation is the binding of vectors. This operation produces a new vector $\boldsymbol{u}$ which is dissimilar to both of the original vectors $\boldsymbol{v}$ and $\boldsymbol{w}$. As with HRRs, the SPA employs circular convolution defined as

$$\boldsymbol{u} = \boldsymbol{v} \circledast \boldsymbol{w} : \quad u_i = \sum_{j=1}^{D} v_j w_{(i-j) \bmod D}. \tag{4}$$

The binding operation can be undone (unbinding) by circular convolution with the involution of one of the operands.

$$\boldsymbol{v} \approx \boldsymbol{u} \circledast \boldsymbol{w}^{-1} \tag{5}$$

The involution is defined as

$$\boldsymbol{w}^{-1} = (w_1, w_D, w_{D-1}, \ldots, w_2)^\top. \tag{6}$$

Note that the unbinding operation produces an approximation of the original vector. The SPA includes a method for building biologically realistic clean up memories (Stewart, Tang, & Eliasmith, 2011). These compare the noisy vector with the clean vectors and then threshold the result with specially tuned neurons.

The binding and unbinding mechanisms allow us to recover certain concepts out of a superposition of bound items. Consider the vector $\boldsymbol{v} = \text{RED} \circledast \text{COLOR} + \text{SQUARE} \circledast \text{SHAPE}$. The color can be recovered as:

$$\begin{aligned} \boldsymbol{v} \circledast \text{COLOR}^{-1} &= \text{RED} \circledast \text{COLOR} \circledast \text{COLOR}^{-1} && (7)\\ &\quad + \text{SQUARE} \circledast \text{SHAPE} \circledast \text{COLOR}^{-1} && (8)\\ &\approx \text{RED} + \textit{noise} && (9)\\ &\approx \text{RED} && (10) \end{aligned}$$

[1] We relax the requirement that vectors are always renormalized. Given the noise inherent in the neural representation, this does not alter the behavior significantly. In addition, much of the normalization is automatically accounted for by neural saturation.

In the n-back task a list of items has to represented. Using a variation of the standard SPA approach, this is done by binding each item to a position vector with backward indexing (POS_1 indicates the most recent item):

$$\mathit{list} = \mathit{item}_1 \circledast \mathrm{POS}_1 + \mathit{item}_2 \circledast \mathrm{POS}_2 + \ldots + \mathit{item}_n \circledast \mathrm{POS}_n \tag{11}$$

The position vectors are defined as

$$\mathrm{POS}_i = \mathrm{CTX}^i \tag{12}$$

with

$$\mathrm{CTX}^i = \underbrace{\mathrm{CTX} \circledast \ldots \circledast \mathrm{CTX}}_{i \text{ times}}. \tag{13}$$

CTX is a random unitary vector, so that convolving with it does not change the norm of the vector being bound (for every $\boldsymbol{v}$ it has to hold that $\|\boldsymbol{v}\| = \|\boldsymbol{v} \circledast \mathrm{CTX}\|$).

Model

An overview of the model is given in Figure 1. It gets input through the *stimulus* ensemble in form of a semantic pointer which represents the parsed visual input (i.e., a consonant or spatial location). We do not model the visual system here as this is out of the scope of this paper (although see Eliasmith et al. (2012) for an example of doing so with the SPA).

The input is routed through a number of working memory ensembles. Each of these ensembles can be described as a gated difference integrator (see Fig. 2). This kind of integrator has a primary population of neurons acting as a standard recurrent integrator. Without any input, a recurrent connection with a long synaptic time constant ($\tau = 0.1\,\mathrm{s}$) ensures that the currently represented value does not change much over time (save for a small drift due to noise). Input is given through another neural ensemble, *gate*, which also receives the negative currently stored value. Thus, the input to the integrator population is the difference of the target value and current value. Furthermore, the gating population can be inhibited to disable any input to the integrator.

As the visual input is usually shown for a limited time only it is first stored in *memory* to have it available beyond the presentation. The currently remembered list of stimuli is stored in the *current-list* memory. The new list to remember is constructed in *updated-list* as

$$\mathit{updated\text{-}list} = \frac{1}{\sqrt{n}}\mathit{memory} + \sqrt{1-\frac{1}{n}}\mathit{current\text{-}list}. \tag{14}$$

This provides the best representation of the n-th item compared to other possible weightings of *memory* and *current-list*. It is possible that humans are not able to optimally weight the components or that the exact weighting depends on how much experience a person has with the task. When changing the weighting to a simpler rule (e.g., both components weighted by $1/\sqrt{2}$), the model performance decreases, but qualitatively remains the same.

The representational strength of items after the n-th position will decrease with each new item. Thus, older items will

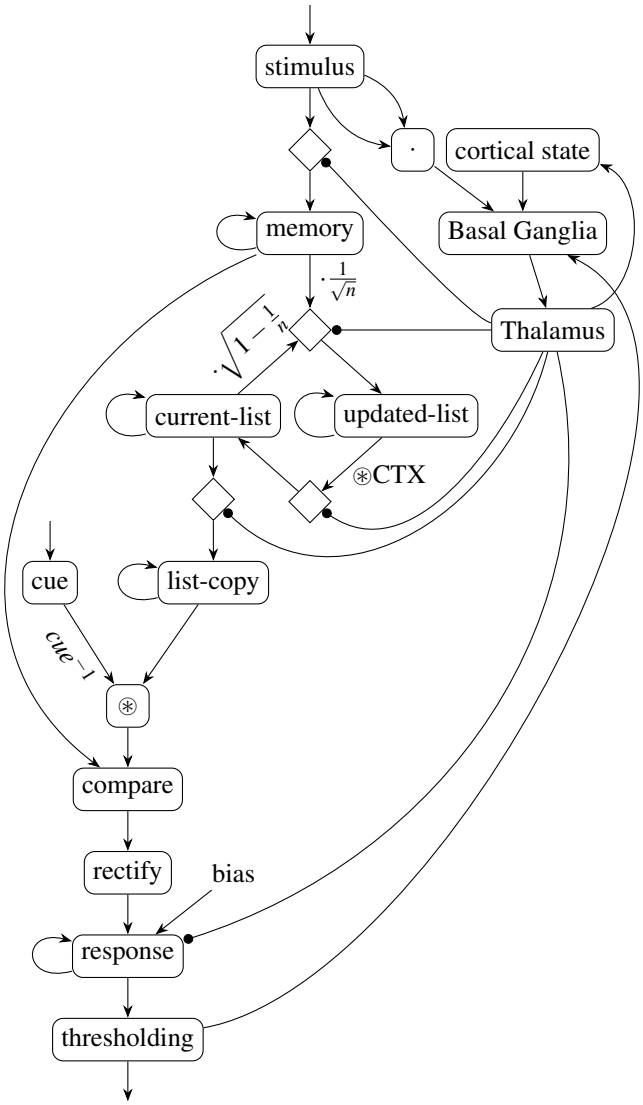

Figure 1: Overview of the n-back model. The labeled boxes represent neural ensembles with exception of *Basal Ganglia* which is a more complex network of multiple interconnected ensembles. The $\cdot$ and $\circledast$ ensembles output the dot product and circular convolution of their inputs. The gating ensembles of gated difference integrators are unlabeled diamonds. For clarity the difference connection back from the integrator is not shown. The flow of information is denoted with arrows (⟶). Transformations and computations are denoted along these connections. Inhibitory connections are indicated with full circles (—●).

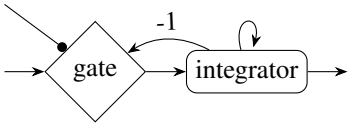

Figure 2: A gated difference integrator unit. It consists out of two neural populations *gate* and *integrator*. Arrows (⟶) represent NEF connections between ensembles. The line ending in a filled circle (⟶●) is a connection directly inhibiting the neurons of the targeted ensemble. See the text for additional details.

be automatically forgotten without an active removal mechanism.

Once the new list has been constructed, it is transferred from *updated-list* to *current-list* and a circular convolution with CTX is applied to update the position tags in the remembered list. In this convolution, one of the operands is constant allowing this transformation to occur efficiently in the connection weights (otherwise a dedicated neural ensemble with both operands as input would be needed).

A copy of the remembered list is also stored in *list-copy*. This allows the list in *current-list* to be updated while the model is still deciding on whether it saw the current item n positions before. Once it has given its answer, the content of *current-list* will be loaded into *list-copy*.

To provide an appropriate response, the desired n has to be given to the model (just as people are told the desired n). This information is encoded as CTX^n in *cue*. Using involution, the item at the n-th position in the sequence stored in *list-copy* is retrieved. This intermediate result is compared to *memory* in *compare* by computing a dot product. Values below zero will be clipped to zero in *rectify*.

The output of the *rectify* ensemble minus a bias is taken as evidence for a match if the value is positive and as evidence for a mismatch if the value is negative. The bias was set to $-\exp(\frac{n}{0.62}) - 0.2$. To form a final decision, the evidence is integrated in the *response* ensemble until either a positive or negative threshold is reached. This is consistent with neural mechanisms observed in decision making tasks (e.g., Wang, 2008). By reaching the decision threshold, the corresponding motor action would be triggered. The motor system is not part of the presented model and is out of the scope of this paper. Thresholds 0.5 and -0.9 were chosen for match and mismatch answers respectively. The bias and threshold parameters were obtained by trial-and-error.

As with SPA models in general, the routing is controlled by an action selection model of the basal ganglia and thalamus presented by Stewart, Choo, and Eliasmith (2010). To control routing in the n-back model, three states – ENCODE, WAIT, and TRANSFER – are used. For each state, a utility value is continuously calculated and the basal ganglia model selects the state with the highest utility value. The corresponding neurons in the thalamus are disinhibited and this leads to inhibition of cortical neural populations to route information accordingly. Table 1 lists how the utility values are calculated and how information is routed.

The model switches to the ENCODE state once input is available which is detected by calculating the squared length of *stimulus* using a dot product. A bias of 0.2 is added to ensure that this switch happens. In this state, the gate to *current-list* has to be inhibited to prevent the current list from being overridden while the new item is added to it.

Once the stimulus disappears, the utility for the ENCODE state decreases and the model switches to the WAIT state. As there is no input stimulus, the *memory* and *updated-list* gate have to be inhibited. Also, the *list-copy* gate will be inhibited to give the model time to provide an answer for the current trial.

Once an answer has been provided, which is detected by the thresholding of the *response* integrator, the state switches to TRANSFER. In the TRANSFER state the inhibition of the *list-copy* gate ends to allow the transfer of the content of *current-list*. The *response* integrator will be inhibited to prepare it for the next trial.

Most ensembles in the model represent 64 dimensional vectors with 3200 neurons. The dot product uses twice the number of neurons and the circular convolution uses 12 800 neurons. The *rectify* and *response* ensemble represent scalars with 50 neurons each. The thresholding ensemble represents a scalar with 100 neurons. 31 450 neurons are used in the basal ganglia and thalamus part of the model. Overall there are 92 250 neurons in the model. The connections between neurons are pre-calculated with the NEF methods and no learning occurs during simulation.

Apart from the basal ganglia and thalamus, all model components are assumed to be part of the cortex, and the neuron's model parameters were chosen accordingly. A membrane time constant of $\tau_{RC} = 20\,\mathrm{ms}$ and an absolute refractory period of 2 ms were used which are typical values for pyramidal cells in the cortex. During delay periods in memory tasks, maximal firing rates are typically around 80 Hz. But for computational efficiency the maximum firing rates in the model were uniformly chosen from 200 Hz to 400 Hz. The same results can be obtained with lower firing rates, but require a larger number of neurons, which increases simulation times. Recurrent connections were assumed to be of the NMDA type with a slow time constant of $\tau_{\mathrm{NMDA}} = 100\,\mathrm{ms}$. Inhibitory connections, assumed to be GABA-ergic, used a time constant of $\tau_{\mathrm{GABA}} = 8.48\,\mathrm{ms}$. Finally, for non-recurrent excitatory connections synapses of the glutamate type with a time constant of $\tau_{\mathrm{glut}} = 5\,\mathrm{ms}$ were assumed.

Results

To test the model, 48 instances of the model with different random number generator seeds were created. Each was run on a 1-, 2-, and 3-back random sequence consisting of 15 match and 30 mismatch trials ($45 + n$ trials overall). Lure trials, where the current item matches the one at position $n-1$

Table 1: The utility calculations for switching to different states and the routing actions taken in these states.

Cortical State	Utility Calculation	Routing
ENCODE	$stimulus \cdot stimulus + 0.2$	inhibit *current-list* gate; inhibit *response*
WAIT	$cortical\ state \cdot \text{ENCODE} + cortical\ state \cdot \text{WAIT}$	inhibit *memory*, *updated-list*, and *list-copy* gate
TRANSFER	$cortical\ state \cdot \text{TRANSFER} + \lvert thresholding \rvert$	inhibit *memory* and *updated-list* gate; inhibit *response*

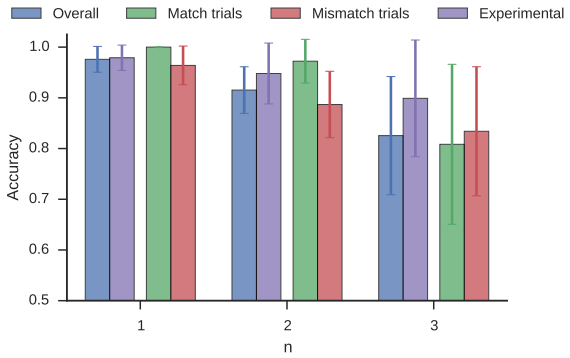

Figure 3: The average model accuracy (proportion of correct answers) given n. For each n, the overall model accuracy, experimental data, model accuracy in match trials, and model accuracy in mismatch trials is shown from left to right. The experimental data for comparison was taken from the practice session in Jonides et al. (1997). Error bars denote the standard deviation.

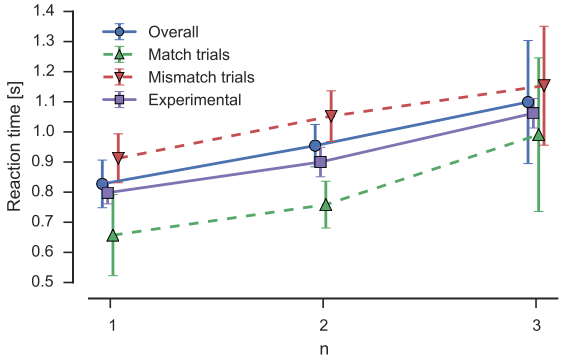

Figure 4: The average model reaction times (excluding trials without a response). Experimental data for comparison was taken from the practice session in Jonides et al. (1997). Error bars show the standard deviation.

or $n+1$, were allowed to occur. 20 different stimulus items analogous to the 20 consonants commonly used in the n-back task were generated, but as the model makes no assumptions about the stimulus modality, those items could also be interpreted as different spatial locations. In each trial, the current item was provided as input to *stimulus* for 0.5 s. The duration of a single trial was 2.5 s. Similar protocols are employed in n-back studies with human subjects (e.g., Jonides et al., 1997; Szmalec et al., 2011).

The response of the model was read out from the *response* population at the moment of switching the state to TRANSFER. A positive value indicated a 'match' response, whereas a negative value indicated a 'mismatch' response. In some rare trials (less than 5%) the model did not switch to the TRANSFER state because it did not gather enough evidence for one of the responses in time. These trials were counted as wrong answers and were excluded in the reaction time analysis.

The model reproduces the characteristic decline in accuracy with increasing n as shown in Figure 3. The standard deviation increases with n as in human studies. Reaction time data of the model is shown in Figure 4. With increasing n the reaction times increase. Moreover, reaction times in match trials are shorter than in mismatch trials. These results match the observations from human studies well (e.g., Jonides et al., 1997; Szmalec et al., 2011). Performing the same statistical analysis as in these human studies shows thas the effect of n on accuracy and reaction times is highly significant (accuracies: $F(3,48) = 36.2, p < 0.001$; reaction times: $F(3,48) = 48.0, p < 0.001$).

Discussion

To the best of our knowledge, we have presented the first biologically plausible, spiking neural network model able to perform the n-back task. The model is able to reproduce the well known decline in accuracy and increase in response times with increasing n. Slower reaction to mismatch trials than to match trials is also captured by the model.

Despite being implemented in spiking neurons it is straightforward to reuse parts of the model in other models related to list learning or working memory. This is a welcome feature of the model as it is unlikely that the brain has specialized subsystems for the n-back task. Also, in the n-back task itself, the model exhibits some flexibility. For example, it does not depend on a fixed timing of the stimuli and is robust to changing n on the fly. The latter can be done by changing the input to the *cue* ensemble and modulating the strength of the connections with a scaling dependent on n. This scaling of connection weights may correspond to the effect of dopamine in the prefrontal cortex, which is commonly taken to be modulatory.

However, we do not think that this model is complete with respect to all the processes involved in the n-back task. The current model only implements a recall based process, but there might also be a familiarity based process and a rehearsal process to keep the relevant items active in memory (Szmalec et al., 2011). To make matters more complicated, the con-

tribution of these additional processes might not be fixed but dynamically adjusted according to task demands (Botvinick, Braver, Barch, Carter, & Cohen, 2001). For example, if the number of lure trials is low, relying purely on familiarity can be quite accurate, but as the number of lure trials increases it becomes more important to recall the exact position of an item.

There is also the possibility that human subjects consciously or unconsciously employ different strategies in the n-back task. For this reason, Juvina and Taatgen (2007) build two different ACT-R models of the n-back task. The model presented here is similar to their low-control model which uses a time-tag approach. Here, however, we tag the serial position instead of the time an item occurred. It should be possible to tell these approaches apart by designing an n-back experiment with varying trial duration. This should leave our model mostly unaffected, but should be detrimental to the performance of the model by Juvina and Taatgen (2007).

The only other neural model of the n-back task to our knowledge was presented by Chatham et al. (2011). It is less biologically and psychologically plausible than the model presented here. First, the spiking LIF neurons of our model provide greater biological plausibility than rate neurons, by using a known mechanism of information transmission in the brain (i.e., action potentials). In addition, the Chatham et al. (2011) model relies on idealized computational functions (e.g., max for kWTA) not implemented in neurons, as well as several localist representations. Second, at the same time our model gives more insight into the high-level algorithm as it is explicitly formulated, whereas the model by Chatham et al. (2011) only implicitly learns it. Third, our model can dynamically switch between different n. In contrast to this, the model by Chatham et al. (2011) has to be trained for each specific n. While humans improve with training on the n-back task, they are also able to perform the task without any prior training.

It is often stated that the n-back task requires active inhibition or removal of irrelevant items. While there is evidence for active removal in some working memory tasks (Ecker, Oberauer, & Lewandowsky, 2014), this claim has not been investigated in the context of the n-back task. The performance of our model shows that an active removal process is *not* necessary in the n-back task. Furthermore, the model implies a two stage update process requiring a secondary memory population. First, the updated list is constructed and then the current list is replaced.

Notes

The source code for simulations and data analysis is available at https://github.com/ctn-archive/gosmann-cogsci2015/releases/tag/cogsci2015-paper. It has not been peer reviewed.

Acknowledgments

This work was supported by the Canada Research Chairs program, the NSERC Discovery grant 261453, Air Force Office of Scientific Research grant FA8655-13-1-3084, CFI and OIT. This work made use of SHARCNET an Compute Canada computer resources.

References

Botvinick, M. M., Braver, T. S., Barch, D. M., Carter, C. S., & Cohen, J. D. (2001). Conflict monitoring and cognitive control. *Psychological Review*, *108*(3), 624–652.

Chatham, C. H., Herd, S. A., Brant, A. M., Hazy, T. E., Miyake, A., O'Reilly, R., & Friedman, N. P. (2011, May). From an executive network to executive control: a computational model of the n-back task. *Journal of Cognitive Neuroscience*, *23*(11), 3598–3619.

Ecker, U. K., Oberauer, K., & Lewandowsky, S. (2014). Working memory updating involves item-specific removal. *Journal of Memory and Language*, *74*, 1–15.

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

Eliasmith, C., & Anderson, C. H. (2003). *Neural engineering: computation, representation, and dynamics in neurobiological systems*. Cambridge: MIT Press.

Eliasmith, C., Stewart, T. C., Choo, X., Bekolay, T., DeWolf, T., Tang, Y., & Rasmussen, D. (2012, November). A Large-Scale Model of the Functioning Brain. *Science*, *338*(6111), 1202–1205.

Gayler, R. W. (2003). Vector symbolic architectures answer Jackendoff's challenges for cognitive neuroscience. In *International Conference on Cognitive Science.*

Jonides, J., Schumacher, E. H., Smith, E. E., Lauber, E. J., Awh, E., Minoshima, S., & Koeppe, R. A. (1997, July). Verbal working memory load affects regional brain activation as measured by PET. *Journal of Cognitive Neuroscience*, *9*(4), 462–475.

Juvina, I., & Taatgen, N. A. (2007). Modeling control strategies in the n-back task. In *Proceedings of the 8th International Conference on Cognitive Modeling* (pp. 73–78).

Plate, T. A. (1995). Holographic reduced representations. *IEEE transactions on Neural networks*, *6*(3), 623–641.

Stewart, T. C., Choo, X., & Eliasmith, C. (2010). Dynamic behaviour of a spiking model of action selection in the basal ganglia. In D. Salvucci & G. Gunzelmann (Eds.), *Proceedings of the 10th International Conference on Cognitive Modeling* (pp. 235–240).

Stewart, T. C., Tang, Y., & Eliasmith, C. (2011, June). A biologically realistic cleanup memory: Autoassociation in spiking neurons. *Cognitive Systems Research*, *12*(2), 84–92.

Szmalec, A., Verbruggen, F., Vandierendonck, A., & Kemps, E. (2011). Control of interference during working memory updating. *Journal of Experimental Psychology: Human Perception and Performance*, *37*(1), 137–151.

Wang, X.-J. (2008, October). Decision making in recurrent neuronal circuits. *Neuron*, *60*(2), 215–234.