The Neural Executive: Can Gated Attractor Networks Account for Cognitive Control?

Jared Sylvester* and James A. Reggia

Department of Computer Science
University of Maryland
College Park, Maryland, U.S.A.
{jared,reggia}@cs.umd.edu

Abstract

AI systems can be grouped into two main categories: symbolic and neural. Each has advantages in certain types of tasks, and to date symbolic systems are typically superior at cognitive control and executive function. This paper presents a model using the GALIS framework which probes at this gap by successfully completing the n-Back cognitive control task at human-like levels using a neural approach. Several aspects of the model discussed here, in particular gated sequential attractor networks and behavior driven in part from the learned contents of instruction memory, allow it to behave in ways that are more typical of symbolic systems.

1 Introduction

Approaches to building intelligent systems can be broadly categorized into two camps: top-down, symbolic systems and bottom-up, neural ones. The schism between the two strategies, and the attendant debate over their relative superiority, has gone on almost as long as AI and cognitive modeling have been studied [12]. Neural systems currently excel at problems that make use of strengths like pattern matching, incremental learning, and processing noisy data [13]. However, they are less successful in domains which require high-level executive behaviors like representing the goals of a task or rules of an environment [11]. Conversely, symbolic systems can easily bind variables, create complex data structures, and perform global computations, allowing them to better perform high-level cognitive tasks like planning and reasoning.

Producing neural systems which are capable of higher-level cognitive behaviors has received increasing levels of interest from the AI community, but it remains an elusive goal [18]. It does not seem convincing that the neural paradigm is inherently less well-suited to carrying out higher cognitive functions. After all, biological neural networks are perfectly capable of doing so. Indeed these biological neural systems provide the only example we have of cognition. For example, a person can play a novel card game with little difficulty after having the rules presented to them briefly, and those rules can be encoded in a top-down AI system in a straightforward way. In contrast current artificial neural systems may need to witness thousands of iterations of the game before being able to play on its own.

What accounts for the divergent performances of neural and symbolic systems on executive cognition tasks? And how can this divergence be reconciled with the ability of naturally-occurring neural networks to formulate plans, select actions, shift attention, and carry out numerous other high-level cognitive control behaviors?

Researchers have been increasingly interested in examining this issue recently in attempts to understand the brain's computation from the bottom-up. Of particular interest here has been

*Corresponding author: http://www.cs.umd.edu/~jared/

the study of "cognitive control," an umbrella term for executive cognitive systems that manage other cognitive processes, such as working memory, attention, and inhibition. Several pioneering neural models explicitly incorporate aspects of cognitive control, such as for managing working memory [14].

However, many such neural models are hard-wired for the particular task for which they are designed [23], connection strengths are often set by hand without a learning procedure, and local conjunctive encodings are often used [6], specifying the exact sets of possible inputs and outputs and making adaptation to other situations, contexts or environments tricky. This specialization can make neural network models of cognitive control difficult to build because each model requires not only parameter tuning and other human supervision, but often construction from the ground up [7]. Even small changes in the task specifications can require large modifications to the architecture. For instance, Dehaene and Changeux's model for solving the Towers of London problem [4], while capable of an impressive amount of planning for a neural network, is incapable of solving the very similar Towers of Hanoi problem, or even of solving Towers of London using a method other than the greedy, depth-first search it has been constructed to execute.

One notable prior neural network model which does generalize to multiple tasks without making architectural changes is that of Rougier et al. [17]. These tasks revolve around identifying the single currently relevant feature (e.g., WCST). Rougier et al. frame their tasks as pattern classification problems, using iterative training to learn input-output mappings. In contrast, the system presented here uses one-shot learning, and frames each task as a procedure to be executed.

2 GALIS: A conceptual framework for modeling cortical cognitive control

The development of a general purpose, adaptive neural system, building on the successes of past specific implementations of cognitive control mechanisms, would be useful for a broad range of applications and in the study of our own cognition. Towards that end, we describe here an approach or philosophy for building models of cognitive control which we call GALIS, for "Gated Attractors Learning Instruction Sequences." GALIS is intended to be a general-purpose, adaptive neurocomputational architecture that learns how to perform tasks, including tasks that themselves involve learning, within which models for specific tasks can be instantiated. A GALIS model's behavior is determined in large part by the patterns that have been stored in its instruction memory. While it remains entirely a neurocomputational system, this introduces a similarity to the traditional von Neumann architecture computer.

GALIS is inspired by the organization and functionality of the cerebral cortex, although it is not intended to be a faithful neuroanatomical model of brain circuitry. We take inspiration from the organization of the cerebral cortex to create a general computational framework that can be used effectively to create a broad range of neural architectures for specific tasks. By developing methods for cognitive control which are not task or domain specific we hope to help bridge the gaps between artificial neural systems and both biological neural systems and artificial symbolic systems. We adopt three main hypotheses about how the cerebral cortex implements cognitive control.

First, the cerebral cortex is organized as a distributed network of interacting cortical regions [2, 22]. All aspects of working memory contents, both static information that captures task-specific details and dynamic procedures for performing a task, are stored within a network

of model regions. Model cortical regions must learn not only the facts about a specific instance of a task, but also the procedure or "software" that is needed to perform that task. This focus on making behavior largely dependent on patterns stored in the network's memory, rather than on the network's structure or "hardware," is a break from previous models of cognitive control, and is intended to make GALIS models more generalizable: their behavior can be changed by adjusting which sequences are learned rather than by adjusting the structure of the model itself.

Second, each region in the cortical network can usefully be viewed as an attractor neural network, i.e., as a dynamical system whose activity is continuously being driven towards certain preferred states. Attractor networks have been used previously in cognitive control models [5, 10], but usually they are limited to dealing with only fixed-point attractors. However, the attractors in GALIS are designed to enable switching between attractor states in ordered sequences. This is critical if procedural information is to be accommodated in memory: procedures by their very nature must be temporally extended. Various techniques have been used to add similar dynamism to attractor nets [3, 9, 28] but GALIS uses temporally asymmetric learning of intra-regional connection weights [24, 25].

Third, each cortical region can not only exchange information with other cortical regions in the form of activity patterns, but can also gate other regions' functions and interactions. By *gating* here we mean that a cortical region can turn on/off functions in other regions, or open/close the flow of information between other regions. There have been numerous theories posited for the biological structure underlying cortical gating [19, 21], but GALIS is agnostic about the particulars of gating in biological systems. Rather, we take the existence of some such mechanism as a given and implement them as direct gating interactions between model cortical regions and their connecting pathways.

3 *n*-Back Model

To demonstrate the GALIS approach, we next give an example model capable of performing the n-Back working memory task for $n \in \{1, 2, 3, 4, 5\}$. The model is able to perform without re-configuration or re-training the network when the value of n changes during the task. For a more detailed treatment, please refer to Sylvester et al. [26].

The n-Back task is of significant interest in cognitive psychology [15]. In it, the participant is presented with a stream of stimuli and must identify which of these is the same as the stimulus presented n steps earlier. For example, in a 3-back task the bold letters in the following sequence would be considered matches: *VHZ**V**XOLI**O**SAJX**A**O*.

3.1 Architecture

The top-level architecture for the GALIS n-Back model is shown in Figure 1. It consists of several interacting layers: the visual input layer, the n-input layer, the output nodes, the memory layer, the compare module, the context module, and the control module.

There are two input layers to the model. The visual input layer is set externally to represent the visual stimulus being presented during the current time step. The n-input layer is used to specify the current value of n so that the model knows which task version it is supposed to be performing. Which version is executed at any time depends only on changing this input.

A pair of nodes are used for outputting *match* and *no-match*. Which one is activated is a function of the output of the compare module, which compares the visual input layer to the current state of the working memory.

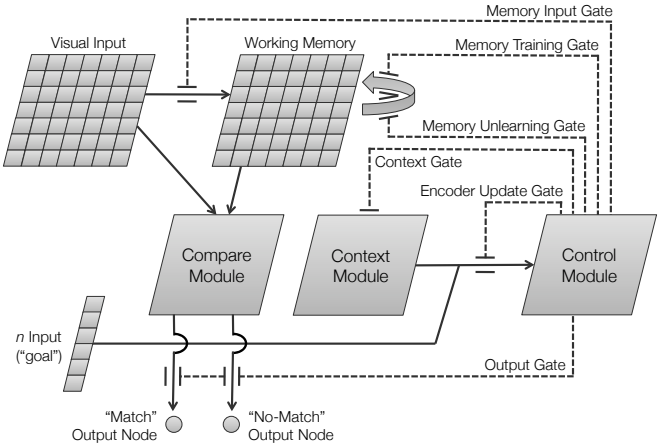


Figure 1: The GALIS model for the n-Back task. Thin, solid arrows denote one-to-one connections. The working memory layer is fully recurrently connected (broad arrow). Dotted lines are the outputs of the control module.

The context module allows the control module to keep track of what stage of processing it is in, fulfilling a similar but much simplified role as the Program Counter in a CPU. Processing each stimulus occurs in two stages: first the new stimulus is added to the working memory, then the working memory is searched to determine if that new stimulus is a match with the n-Back item. The control module both influences and is influenced by the context module, which enables it to set its own short-term sub-goals.

The working memory layer is a discrete Hopfield network forming an auto-associative memory. Like biological working memories, the GALIS working memory layer has limited capacity, high plasticity via one-shot learning, and close integration with executive systems. The working memory layer has been modified from standard Hopfield networks to allow it to recall patterns in a specified sequential order using dynamic thresholds, weight decay [16, 28] and temporally asymmetric weights [24, 25]. In addition to adding patterns to working memory, the network also has the capability to "unlearn" or partially "forget" stored patterns using an anti-Hebbian learning rule [8]. One could think of the unlearning procedure as the addition of an "erase" command to complement the typical "load" and "store" functions already present.

A key component is the control module, which is responsible for directing the operation of the rest of the system. Its outputs drive the six gates which govern flow of activity throughout the rest of the model. The control module's core is a second discrete Hopfield attractor network, called the "instruction sequence memory" (ISM). Like the working memory, the ISM stores sequences using temporally asymmetric weights. But where the working memory module stores visual stimuli, the control module stores the actions necessary for completing a task. Each action consists of opening and closing different gates to different degrees in order to influence the actions of the rest of the network. Both the working memory and ISM are based on the same weight update, input and state update rules. Reusing the same principles for both data and instruction storage makes GALIS particularly parsimonious. However, the ISM has been modified to store multiple sequences concurrently. Each sequence corresponds to a particular set of actions the model may need to perform during a task. For instance, one such sequence of actions for n-Back adds a new stimulus to working memory.

The gates which the control module adjusts modulates the flow of activity between layers

and regulates the weight updates in the working memory. These gates are: (*i*) the Memory Input gate, between the visual input layer and working memory layer which biases the memory's current state towards or away from the current stimulus; (*ii*) the Output gate which controls the flow of activity from the compare layer to the output nodes; (*iii*) the Memory Training and (*iv*) Memory Unlearning gates which control when the working memory learns and removes patterns, respectively, (*v*) the Context gate which regulates the state of the context module; and (*vi*) the Encoder Update gate which governs the inputs to the control module, so that it can control whether it begins a new action sequence or finish the sequence currently in progress. The effect of gates is multiplicative, such that the downstream activity of a gated connection is a product of its incoming activity and its current state. Gates are real-valued allowing them to amplify, damp, or inhibit their inputs, making them more nuanced than simply "open" and "closed."

3.2 Operation

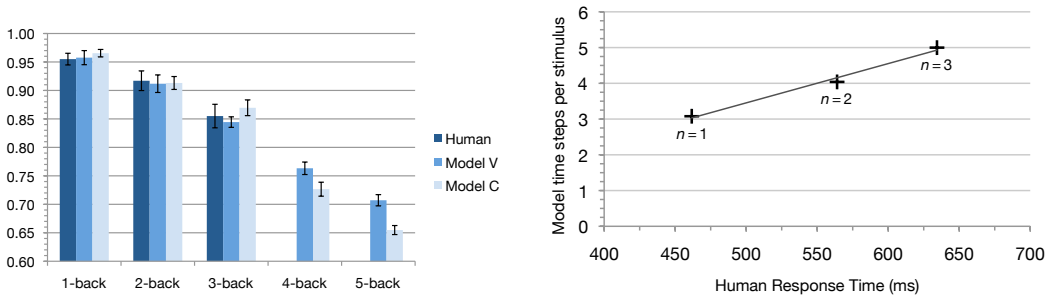
Each run of the model is divided into two phases: Controller Initialization and Task Execution. In the former the control module learns the instruction sequences necessary to perform the task using one-shot Hebbian learning. This training of the control module occurs only once, after which its weights remain fixed. In contrast, the working memory layer begins in a blank, untrained state, and has its weights updated multiple times as the trial progresses through the Task Execution phase. While the associations learned by the control module are determined by the human modeler, the learning that the working memory engages in during the task is entirely guided by the model itself, with the model determining when to add or remove patterns from memory.

The Controller Initialization phase occurs before the model is presented with any inputs or produces any outputs. For the n-Back task six instruction sequences are learned: one which adds a new stimulus to working memory as well as one each for comparing the current input to the 1^{st}, 2^{nd}, ..., 5^{th} item in memory. The network thus learns to perform the task for $n \in \{1, 2, 3, 4, 5\}$. Training is identical no matter which versions the model performs during the Task Execution phase. As a result, the model is capable of switching between versions of n-Back during trials, as dictated solely by its inputs and without any other adjustments being made.

During each step of processing in the Task Execution phase the model goes through the following operations directed by the control module. First, if either output node was activated in the previous time step a new stimulus will be presented, otherwise the inputs from the previous step are retained. Second, the state of the working memory is updated. Third, the output of the compare module is updated to reflect the new states of the working memory and visual input layers. Finally, the state of the context module is updated.

4 Results

After the GALIS n-Back model was trained to perform n-Back tasks of varying lengths (n=1 through n=5), it was presented with sequences of $30 + n$ stimuli. The first n are "preparatory stimuli," and the response to these is ignored, as they are for human subjects. For each trial, ten stimuli would be generated, and the sequence of inputs would then be drawn from these ten. (Using a small set of possible stimuli is typical of human n-Back experiments.) One third of the stimuli following the preparatory period were matches. No "lures" (stimuli which match those one position offset from the target) were used, as was the case for the human subjects whose performance we were attempting to match, which was drawn from Watter et al. [27].



(a) (b)

Figure 2: (a) Accuracy for human subjects and the computational model for all five versions of n-Back. (b) Correlation between human response time per stimulus and the average number of steps the model takes per stimulus.

In Figure 2, the model's average performance across all 30 stimuli over 250 trials is given and is also compared to that of human subjects on 1-, 2- and 3-back tasks. Two different variations of the model were tested. Model V used variable working memory decay rates (the larger n was, the smaller the decay rate used, so that for larger n values the working memory attempted to store more stimuli), while Model C used the same decay rate for all values of n.

Both models show that as n increased response accuracy decreased monotonically. For n=1,2,3 both models' results are not significantly different than human performance ($p = .05$); however the overall fit for Model V was closer. This is possible because different decay rates are most suitable for recalling sequences of different lengths [16, 24]. A lower decay rate allows longer sequences of visual stimuli to be successfully stored without deterioratiation. A higher decay rate reduces interference from older items, improving the ability to recall shorter sequences. It has previously been hypothesized that humans may adjust a working memory decay rate in order to control the length of sequences they are attempting to remember [1, 28].

Figure 2 does not show human results for n=4 and n=5 because they are not reported in Watter et al. [27]. Human subjects typically find 4- and 5-back to be extremely challenging [15]. Nonetheless, the GALIS n-Back model is trained to perform 4- and 5-back, and the simulation results are shown as model predictions. If humans really can adjust working memory decay to adapt to longer sequences, Model V's performance leads us to predict that subjects taking Watter's version of n-Back for n=4 and n=5 would see their performance drop off linearly to approximately 76.3% and 70.7%, respectively. Higher values of decay have more of an impact on larger values of n, since decay on patterns in working memory compounds each time a new item is learned. Keeping decay constant in Model C therefore disproportionally impacts performance for n=4,5, reducing Model C's accuracy on 5-back to no better than a biased coin toss. If humans cannot adjust working memory decay to suit the task then we would predict that their accuracy on 4-back to fall to 72.7%, and for them to be unable to perform 5-back at better than random accuracy.

In addition to matching human accuracy levels, the GALIS n-Back model also matches response times. The correlation between human response time and the number of time steps the model needed correlates well with the human response time data ($R^2 = 0.9888$).

Experiments were also run in which the value presented to the n-input layer changed mid-trial.

Input sequences were constructed in the same way as described at the beginning of this section, but the value of n changed between the 15^{th} and 16^{th} stimuli. Other than varying the n input no changes or adjustments were made to the model. After a short transition period lasting several stimuli, the model is capable of performing the newly selected version of the task as if it had been doing it all along. There is no long-lasting penalty from task-switching mid-trial, and the disruption in performance persists for no longer than n stimuli.

If the task is shifted to an easier version (i.e., n decreases), performance will climb monotonically until reaching the expected, baseline accuracy level of the new task variant. If the task is shifted to a harder version (i.e., n increases), the performance will temporarily drop below the expected level before climbing back to and settling at it. We hypothesize that this unexpected pattern has to do with an imbalance between interference and decay in the working memory when it is unexpectedly called on to begin recalling longer sequences, and note that it offers a testable prediction that could be critically evaluated with human subjects.

5 Discussion

The work described here offers further evidence that neural systems are capable of supporting the types of executive functioning typically associated with symbolic AI systems. Parameters did not need to be re-tuned in order to match human accuracy and response times on 1-, 2-, and 3-back. Additionally, testable predictions were produced regarding both 4- and 5-back performance and responses to intra-sequence changes to n. More significantly, this was accomplished in a novel way, by basing the behavior of model on both its structure and the content it had learned.

On the one hand, neither symbolic production rules nor explicit variable bindings were required to produce this human-like cognitive control behavior. On the other, GALIS does not need to hew too closely to physiological detail; for example no complex models of spiking neurons are required. This indicates that it may be possible to construct systems capable of higher-level cognitive functioning which neither completely eschew the basis of our own cognitive machinery nor need to mimic it too closely.

Though GALIS attractor networks operate in high-dimensional, continuous space, each attractor within that space can be seen as a discrete object [20]. As a result, gated attractor networks offer a balance between the continuous nature of neural networks and the discrete nature of symbolic systems, narrowing the gap between what is possible with neurocomputational systems and symbolic ones, providing the potential to produce symbolic-like behaviors using sub-symbolic systems.

The methods employed here also make it possible to "program" a neural network to a novel extent. This creates a second point of potential crossover between symbolic and sub-symbolic systems. It is true that at this point these neural programs had to be crafted by the modelers. Nevertheless, we feel that the approach of basing behavior on stored patterns is a valuable stepping-stone towards more autonomous systems in the same way that the Jacquard loom was a precursor to modern computing machinery. If behavior can be stored in memory then it can be more easily modified than if it was built into the architecture. And if it can be modified, we believe it can be generated autonomously during learning. In other words, GALIS moves away from systems whose behavior is a function of their *construction* and towards ones whose behavior is based on *instruction*.

References

[1] Erik Altmann and Wayne Gray. Forgetting to remember. *Psychological Science*, 13(1):27–33, 2002.

[2] Steven Bressler and Vinod Menon. Large-scale brain networks in cognition. *TiCS*, 14:277–290, 2010.
[3] Gorden Brown et al. Oscillator-based memory for serial order. *Psychological Review*, 107(1): 127–181, 2000.
[4] Stanislas Dehaene and Jean-Pierre Changeux. A hierarchical neuronal network for planning behavior. *PNAS*, 94(24):13293–8, 1997.
[5] Simon Farrell and Stephan Lewandowsky. An endogenous distributed model of ordering in serial recall. *Psychonomic Bulletin & Review*, 9(1):59–79, 2002.
[6] Michael Frank et al. Interactions between frontal cortex and basal ganglia in working memory. *Cognitive, Affective & Behavioral Neuroscience*, 1:137–160, 2001.
[7] Michael Frank et al. Hold your horses: Impulsivity, deep brain stimulation, and medication in Parkinsonism. *Science*, 318(5854):1309–12, 2007.
[8] John Hopfield et al. 'Unlearning' has a stabilizing effect in collective memories. *Nature*, 304(5922): 158–159, 1983.
[9] David Horn and Irit Opher. Temporal segmentation in a neural dynamic system. *Neural Computation*, 8(2):373–389, 1996.
[10] Matt Jones and Thad Polk. An attractor network model of serial recall. *Cognitive Systems Research*, 3(1):45–55, 2002.
[11] Gary Marcus. *The Algebraic Mind: Integrating Connectionism and Cognitive Science*. MIT Press, 2001.
[12] Marvin Minsky. Logical versus analogical or symbolic versus connectionist or neat versus scruffy. *Computation and Intelligence*, pages 647–674, 1995.
[13] Christian Omlin and Lee Giles. Symbolic knowledge representation in recurrent neural networks. In Ian Cloete and Jacek Zuruda, editors, *Knowledge Based Neurocomputing*, chapter 3, pages 63–115. MIT Press, 2000.
[14] Randall O'Reilly and Michael Frank. Making working memory work. *Neural Computation*, 18(2): 283–328, 2006.
[15] Adrian Owen et al. N-back working memory paradigm. *Human Brain Mapping*, 25(1):46–59, 2005.
[16] James Reggia et al. A simple oscillatory short-term memory. In *Proc. AAAI Symposium on Biologically Inspired Cognitive Architectures*, pages 103–108, 2009.
[17] Nicolas Rougier et al. Prefrontal cortex and flexible cognitive control: Rules without symbols. *PNAS USA*, 102(20):7338–43, 2005.
[18] Asim Roy. Connectionism, controllers, and a brain theory. *IEEE Transactions on Systems, Man and Cybernetics A*, 38:1434–1441, 2008.
[19] S. Sherman and R. Guillery. *Exploring the Thalamus and Its Role in Cortical Function*. MIT Press, 2006.
[20] Patrick Simen and Thad Polk. A symbolic/subsymbolic interface protocol for cognitive modeling. *Logic Journal IGPL*, 18(5):705–761, 2009.
[21] Wolf Singer. Dynamic formation of functional networks by synchronization. *Neuron*, 69:191–193, 2011.
[22] Olaf Sporns. *Networks of the Brain*. MIT Press, 2011.
[23] Terrence Stewart et al. Symbolic reasoning in spiking neurons. In *Proc. 32nd Ann. Conf. Cognitive Science Society*, pages 1100–1105, 2010.
[24] Jared Sylvester et al. A temporally asymmetric hebbian network for sequential working memory. In *Proc. 10th Int'l Conf. Cognitive Modeling*, pages 241–246, 2010.
[25] Jared Sylvester et al. Cognitive control as a gated cortical net. In *Proc. 2nd Int'l Conf. Biologically Inspired Cognitive Architectures*, pages 371–376, 2011.
[26] Jared Sylvester et al. Controlling working memory with learned instructions. *Neural Networks*, 41: 23–38, 2013.
[27] Scott Watter et al. The n-back as a dual-task. *Psychophysiology*, 38(6):998–1003, 2001.
[28] Ransom Winder et al. An oscillatory Hebbian network model of short-term memory. *Neural Computation*, 21(3):741–761, 2009.