Multilayer Context Reasoning in a Neurobiologically Inspired Working Memory Model for Cognitive Robots

Arthur S. Williams (asw3x@mtmail.mtsu.edu)
Center for Computational Science
Middle Tennessee State University
1301 E. Main St. Murfreesboro, TN 37132 USA

Joshua L. Phillips (Joshua.Phillips@mtsu.edu)
Department of Computer Science
Middle Tennessee State University
1301 E. Main St. Murfreesboro, TN 37132 USA

Abstract

The brain's working memory system relies heavily on the mesolimbic dopamine system and the delivery of reward signals. The interaction between the prefrontal cortex (PFC) and the basal ganglia are the main components simulated in working memory models. The Working Memory Toolkit (WMtk) is a framework that allows the incorporation of working memory into robotic/artificial systems. The HWMtk is built on top of WMtk by using holographic reduced representations for concept encoding. This allows end users to adopt the framework without the need to understand details of the algorithms involved. While the HWMtk captures human and animal performance on some cognitive tasks, tasks with multiple context layers are still problematic. We extended the HWMtk framework by adding a multilayer context reasoning working memory system. We tested our system on the AX-CPT task, 1-2-AX-CPT task and a 2-layer context task that is partially observable. Our results show that our model is capable of learning after a reasonable number of trials, thus making it amenable for comparison with human and animal performance data.

Keywords: artificial cognition; cognitive robotics; working memory; prefrontal cortex; dopamine; reinforcement learning

Introduction

In cognitive neuroscience, the working memory system enables the ability to hold multiple pieces of information in one's mind while continuing to process other information. In the human brain this system is supported by the prefrontal cortex and the mesolimbic dopamine system (O'Reilly & Munakata, 2000). Computational models seek to represent such systems through the use of artificial neural networks. The Working Memory Toolkit (WMtk) aimed at creating a working memory framework that could easily be incorporated in a robotic system (Phillips & Noelle, 2005). The WMtk used the delayed saccade task to demonstrate the effectiveness of the framework. The WMtk framework was improved to utilize Holographic Reduced Representations (HRRs) in order to encode representations of the environment as well as working memory concepts (Dubois & Phillips, 2017).

While the WMtk has been applied to various robotic tasks (Kawamura, Gordon, Ratanaswasd, Erdemir, & Hall, 2008; Busch, Skubic, Keller, & Stone, 2007; Erdemir et al., 2008; Wang, Tugcu, Hunter, & Wilkes, 2009), it often fails to solve *multilayer hierarchical problems* commonly explored in the cognitive sciences. Therefore, we have extended the use of HRRs within the WMtk to simulate multilayer hierarchical context reasoning in a working memory system. We show that our revised model can solve working memory tasks that are hierarchical in nature. Performance was tested on 3 tasks: 2-layer hierarchical maze, AX-CPT, and 1-2-AX-CPT. In the maze task, the agent sees a red/green cue in the outer context loop and then a blue/purple cue in the inner context loop and determines which memories to gate in or out. The AX-CPT task is a common hierarchical working memory task in the cognitive sciences, and shows our model's ability to handle multilayer context processing for future comparison with human/animal performance data. The 1-2-AX-CPT task adds an additional layer of context to the original AX-CPT task. We also compared our model's accuracy on the 1-2-AX-CPT task with a Long Short-Term Memory (LSTM) (Hochreiter & Schmidhuber, 1997) implementation. LSTM is the state-of-the-art working memory model in sequence learning, thus making it a natural competitor to our model. Furthermore, our model can can compete with LSTM in performance while also providing a more transparent viewing of working memory contents.

Background

Working Memory Architecture

Working memory provides the ability to hold on to task-relevant information needed for further processing and prevention of distractions/disruptions (O'Reilly & Munakata, 2000). In the prefrontal cortex (PFC), attractors provide a mechanism for robust active maintenance of information while counteracting the interference presented by ongoing processing. In previous models, the neural firing rate encoding of representations was handled by a complex multi-layer artificial neural network. In our model, we represent the encoding of representations in the form of holographic reduced representations (HRRs). The use of HRRs in our model reduces the complexity of the network down to a single layer since the HRRs provide the encoding instead of multiple layers within an artificial neural network (ANN).

The interaction between the prefrontal cortex and mesolimbic dopamine system is key to the emergent behavior of working memory. Additionally, the use of reinforcement learning

is also important as it pertains to working memory. Literature in the cognitive sciences suggest that learning is driven by rewards and punishments in response to the changes in expectation of future events (Schultz, Dayan, & Montague, 1997). The mesolimbic dopamine system is a neural substrate responsible for reinforcement learning in the brain. This system consists of the basal ganglia and ventral tegmental area (VTA). Recorded data from the firing of dopamine cells in monkeys show that dopamine cells fire in response to adjustments in expected future reward (Schultz et al., 1997). The basal ganglia is responsible for broadcasting dopamine signals to the PFC. The interaction between the PFC and basal ganglia creates a feedback loop where internal memory updates are chosen based on dopamine signals.

Our working memory model captures the PFC and dopamine system through the use of the temporal difference (TD) algorithm. Unlike LSTM, which uses a fully supervised feedback signal, TD learning simulates the modulation of the dopamine system through the use of semi-supervised TD error (δ) signals. These TD δ signals aim to simulate how the basal ganglia computes the firing of dopamine neurons. TD has traditionally been used to learn action selection, making it well suited for leaning internal action selection as well. This would consist of the opening or closing of circuits in the PFC for either flushing working memory contents or maintaining them. We will provide a more detailed explanation of the TD algorithm in a later subsection.

Working Memory Toolkit

The Working Memory Toolkit (WMtk) is a framework that was created for the purpose of providing a robot control system with the ability to utilize working memory (Phillips & Noelle, 2005). The utility of the WMtk has been tested on several robotic tasks. One task aimed to explore the feasibility of cognitive control systems in humanoid robots (Kawamura et al., 2008). Another task was a spatial memory task that required the robot to learn to associate perceptual information with that of a particular location in order solve the task (Busch et al., 2007). The next task required a robot to explore different internal scenarios before actually executing an external action(Erdemir et al., 2008). And a final task explored automatic scene recognition in regard to robotic navigation (Wang et al., 2009).

The Holographic Working Memory Toolkit (HWMtk) improved upon the Working Memory Toolkit by providing an interface between the distributive encoding (DE) of artificial neural networks and symbolic encoding (SE) (Dubois & Phillips, 2017). Through the use of holographic reduced representations (HRRs), the HWMtk was able to automate the SE/DE conversion problem. Holographic reduced representations (HRRs) are created using a vector of real values taken from a Normal distribution with mean zero and standard deviation $1/\sqrt{n}$, with n being the length of the vector (Plate, 1995). HRRs can be used to encode a particular concept within a model. Circular convolution allows for complex representations by combining two representations together into a single HRR of the same vector length. The circular convolution operation can be computed using Fast Fourier transforms which take $O(nlog(n))$ time. Another operation is correlation which uses a HRR as a key in order to retrieve information from complex HRR containing multiple combined HRR representations.

The HWMtk allows agents to learn to gate in appropriate cues in order to make appropriate action choices later in time. However, the HWMtk shows selective interference between policies contingent on hierarchically structured cues, suggesting future work improvement to the framework may be needed.

Cognitive Tasks

The AX Continuous Performance Test (AX-CPT) is a cognitive task that tests the context processing ability of an individual (Braver, Rush, Satpute, Racine, & Barch, 2005). In this task, an individual is presented with a pair of letters. The individual is only rewarded when they see the letter "A" followed by the letter "X" and press the right button. All other letter combinations are seen as distractors. The presentation of a letter sequence followed by an action marks the end of a trial. Performing well on this task requires correctly updating context information after each trial. Our model learns the correct button to press based on the sequence of letters presented to it. The agent was presented the A-X letter sequence 70% of the time and presented the B-X, A-Y, and B-Y sequences at a rate of 10% each. The particular letter sequence was determined randomly before each trial.

The 1-2-AX-CPT task is an extension of the AX-CPT task that adds an extra layer of context that must be maintained by the working memory system (Frank, Loughry, & O'Reilly, 2001). The target sequence is dependent on a previous context cue and varies depending on which stimulus was observed by the agent. For example, if the agent saw a 1 then the target sequence will be A-X. Instead, if the agent saw a 2 then the target sequence will be B-Y. In order to successfully learn the 1-2-AX-CPT task, the agent must maintain the outer loop of context, which is the task stimuli, and the inner loop of context, which is the sequence of letters. The context cue, which lets the agent know whether the target is A-X or B-Y, was provided to the agent randomly with equal probability at the beginning of each trial.

Methods

We implemented a multilayer hierarchical model of working memory that extends the capability of the WMtk. Our model uses a novel approach to internal working memory updates that relies on the SARSA TD learning algorithm. This differs from the approach provided by the WMtk originally, which uses the value function as a means for action selection and working memory updates. The utility of our model was determined by testing it on a set of learning tasks. First, we created a 2-layer hierarchical maze task for proof of concept. We then tested our model on the AX-CPT task, which is a common working memory task in the cognitive sciences that

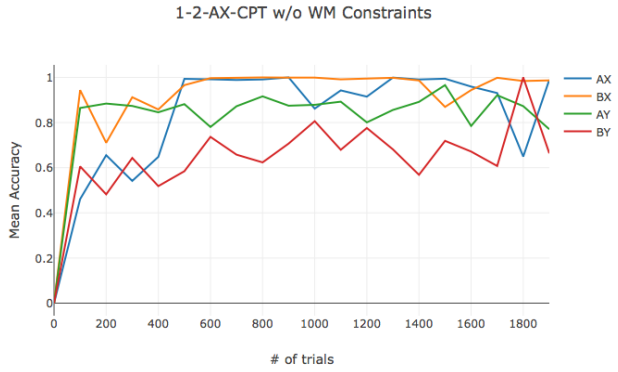

Figure (1) 1-2-AX-CPT task without working memory constraints. The graph depicts the mean accuracy of the 1-2-AX-CPT task over a 100 trial window. The graph shows the mean accuracy for all the A-X, B-X, A-Y and B-Y presented during each trial window.

tests one's ability to maintain relevant features over time. Finally, to fully observe the multilayer hierarchical learning of context features, we implemented the 1-2-AX-CPT task. The 1-2-AX-CPT extends the AX-CPT task by adding an additional layer of context, thereby requiring a multilayer context to be learned in order to complete the task effectively. We then compared the accuracy of our model to that of the LSTM model. LSTM is the state of the art in sequence learning. The juxtaposition our model and the LSTM model, while performing the 1-2-AX-CPT task, is key toward displaying our model's overall utility. The source code for the AX-CPT task, 1-2-AX-CPT task, and 2-layer hierarchical maze task is available online at: `https://github.com/arthurw125/TD_learning`. Next, we will explain the computational models and tasks.

Multilayer Hierarchical Context Model

The WMtk uses the temporal difference (TD) learning algorithm which learns the correct action to take by learning the value function $V(s)$, an estimate of the sum of discounted future rewards, for all states (Sutton, 1988). The SARSA learning algorithm is another reinforcement learning algorithm. SARSA is similar to TD in the sense that they are both temporally extended and estimate a value function. The difference is that SARSA learns the state-action-value function called the Q function, $Q(s,a)$, instead of the state-value function (Sutton, 1988). We learn $Q(s,a)$ by making transitions from one state-action pair to the next, thus learning the value of all state-action pairs. Reward is given only to the state-action pair that transitioned to the goal state. We decided to use the Q function approach as opposed to the value function approach used by the WMtk for memory updates.

In order to use SARSA to model working memory, we modified the Q function to be a state-action-working memory triplet. The amount of states, actions, and working memory slots are determined by the particular task being observed. The value of $Q(s_t,a_t,wm_t)$ can be written as:

$$Q(s_t,a_t,wm_t) = r(s_{t+1}) + \gamma Q(s_{t+1},a_{t+1},wm_{t+1}), \quad (1)$$

where t is the time, s is the set of observable states, a is the action taken, γ is the discount factor, α is the learning rate and wm is the set of working memory slots. The error is computed by taking the difference between the expected and true value of $Q(s_t,a_t,wm_t)$. The Q function is then updated using the following:

$$\begin{aligned}\delta(s_t,a_t,wm_t) =& [r(s_{t+1}) + \gamma Q(s_{t+1},a_{t+1},wm_{t+1})] \\ & - Q(s_t,a_t,wm_t)\end{aligned} \quad (2)$$

$$Q(s_t,a_t,wm_t) = Q(s_t,a_t,wm_t) + \alpha\delta(s_t,a_t,wm_t) \quad (3)$$

This process is repeated over several episodes until the Q function is learned for all state-action-working memory triplets. With this new configuration of the Q function, we can now begin to apply it as a means towards updating working memory slots within our model. The exact contents of what is being held in working memory is available to our model and can be recalled if needed. This feature of our model is contrasted with LSTM's mechanism, which stores its memory information in the weights of the recurrent neural network, thus making it difficult to see how memory content is being stored and maintained.

For our model to learn multiple layers of context, we must provide as many working memory slots as there are contextual layers within the task. We encode the external features and their internal representations for a task using HRRs. The HRR size must be an order of magnitude greater than the number of input permutations (NIP) of the task. This NIP can be computed by multiplying together all the states, actions, and working memory for a given task. Our model then learns the hierarchy of a task by computing the Q function $Q(s,a,wm)$ at each time step to determine whether to gate in new working memory representation or maintain old working memory representations. Also, external action selection is determined at this step. This is achieved by computing all the state-action-working memory combinations within the Q function and choosing the combination with the highest value. A weight vector is updated after each state transition and provides an input to output mapping of the Q function.

2-layer Hierarchical Maze Task

To prove that our model was able to tackle tasks that were hierarchical in nature, we created the multilayer hierarchical maze task. The 2-layer hierarchical maze task requires the agent to learn the location of the goal state within a 1D maze environment. The agent is dropped in the maze at a random location. The agent is then flashed the color red or green followed by blue or purple at the beginning of the task. The agent then must navigate the maze to find the goal state within the maze. Based on what combination of colors was flashed initially, the goal will be either in the middle or at the front of the maze. The objective of the agent is to find the goal

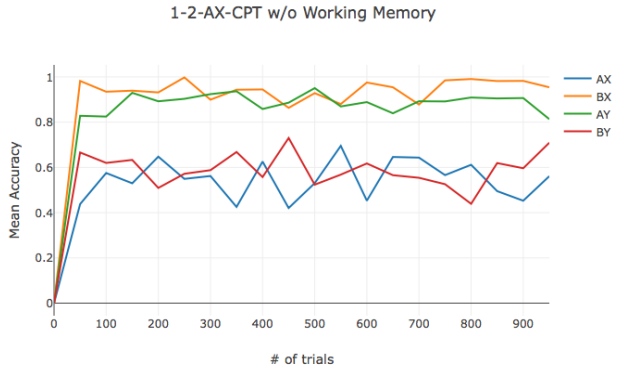

Figure (2) The 1-2-AX-CPT task with out working memory made available to the system. The graph depicts the mean accuracy of the 1-2-AX-CPT task over a 100 trial window. The graph shows the mean accuracy for all the AX, BX, AY and BY presented during each trial window.

based on the set of initial cues seen. This task is partially observable because the agent does not have access to cues after initial sightings. This is why the agent must use its working memory system to actively hold on to the color cue at both levels of hierarchy throughout the task. The purpose of the task is to show the agent's ability to maintain viable concepts in its working memory while discarding irrelevant information. Also, this task seeks to show how multilevel context reasoning can have an effect on the working memory system.

Our computational model for the 2-layer hierarchical maze task requires HRR representations for states, actions, color cues, and working memory slots. We then precompute the convolutions of each representation and store the output of vectors in an array. We initialize a weight vector and set the bias equal to 1. We then set the number of action and color cues to 2. We randomly place the agent in our maze environment by choosing a random number between 0 and the number of states. This value is then mapped to a location in a grid maze environment. A color signal is then flashed to agent, representing either red or green. If the color is red followed by blue then the goal is in the middle of the maze environment. Furthermore, if the color is green followed by purple then the goal will be located in the far-left corner of the maze environment. We provided 2 working memory slots to our model so that the agent has the ability to maintain both the inner and outer cues being relayed from the environment.

A NIP of 8100 was computed for this task. This was evaluated by multiplying all the encoded states, actions, and working memory features. We noticed that in order to achieve stability in our model, we needed to provide an HRR size that was an order of magnitude greater than the NIP of the task. This is why we chose the HRR and weight vector length for this task to be $n = 64000$. Also, the learning parameters for this task are as follows: number of episodes=100000; learning rate=0.1; epsilon-soft=0.01; discount factor=0.9.

AX-CPT Task

The implementation of the AX-CPT task required that we use the modified SARSA algorithm from equations 1, 2 and 3 for action selection and HRRs for concept encoding. We encoded HRRs for 1 working memory slot, 4 different external cues, and 2 actions. The HRRs were pre-convolved and stored in an array for later use. We created a weight vector from a new HRR, and fixed bias weight of 1. The possible working memory contents that could be used by the agent consisted of seeing nothing or the outer signal. Using SARSA along with a softmax function, the outer working memory is chosen along with an action of 1 or 0 (left or right button). The error is computed and the weight vector is updated. This process is repeated until the Q function is learned for the task.

We computed the NIP of this task to be 54. In order to achieve stable learning, the HRR length for our encoded features had to be an order of magnitude larger than the NIP. Due to this constraint, the HRR and weight vector length for this task was $n = 128$. Also, the learning parameters for this task were as follows: number of trials=1000; learning rate=0.1; discount factor=0.9; lambda=0.8; temp=0.1; and epsilon-soft=0.01.

1-2-AX-CPT Task

The implementation of the 1-2-AX-CPT task required that we use the modified SARSA algorithm from equations 1, 2 and 3 for action selection and internal memory updates. We encoded HRRs for 2 different working memory slots, 6 different external cues, and 2 action HRRs. The HRRs were pre-convolved and stored in an array for later use. We created a weight vector from a new HRR, and fixed bias weight of 1. The possible working memory contents that could be used by the agent consist of seeing nothing or the outer signal for the outer context stimulus. For the other working memory slot, the possible working memory contents are nothing or the inner signal. Using the modified SARSA algorithm along with a softmax function, the outer and inner memory slots are chosen, as displayed in Figure 3, along with an action of 1 or 0 (left or right button). The error is computed and the weight vector is then updated.

A NIP of 486 was computed for this task. As in the previous models, we observed that in order to achieve stability in our model we needed to provide an HRR size that was an order of magnitude greater than the NIP of the task. This is why we chose the HRR and weight vector length for this task to be $n = 1024$. Choosing a learning rate of 0.4 allowed our model to learn the task at a faster rate, causing the number of trials to equal only 2000, while still being low enough to allow for convergence. Also, we elected to use an epsilon soft policy set at 0.01 which allowed for exploration of suboptimal states. This allowed our model's loss function to avoid getting caught in its local minima. A full list of the learning parameters for this task were as follows: number of trials=2000; learning rate=0.4; discount factor=0.9; lambda=0.8; temp=0.1; and epsilon-soft=0.01.

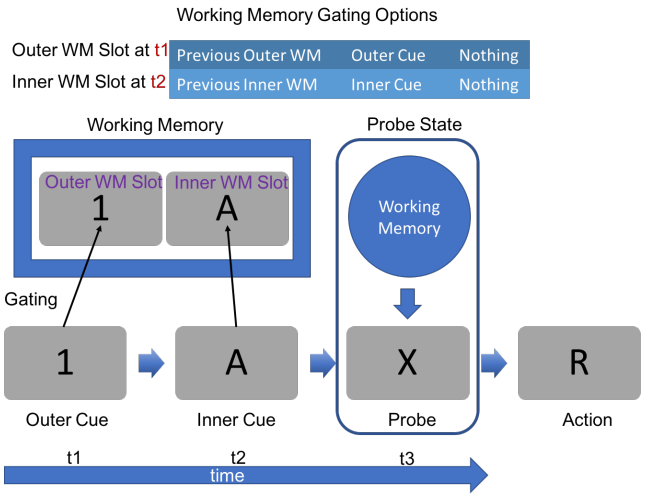

Figure (3) Diagram of our working memory model while solving the 1-2-AX-CPT task. The gating policy is learned by our modified Q function and is responsible for making internal working memory updates. At time step t1, the agent is presented with an outer cue of "1". The gating policy then determines what should be gated into the outer working memory slot. The options available to gate in at time t1 is as follows: gate in the current outer cue of "1", maintain what was previously in the outer working memory slot, or flush the outer working memory slot. At time step t2, the agent is presented with an inner cue of "A". The gating policy then determines what should be gated into the inner working memory slot. The options available to gate in at time t2 is as follows: gate in the current inner cue of "A", maintain what was previously in the inner working memory slot, or flush the inner working memory slot. At time step t3, the Probe State consists of convolving working memory with probe "X". Finally, the action "R" is selected and evaluated for correctness.

I would like to point out that without working memory, an agent within a multilayer hierarchical task is unable to learn the outer loop context of a task, resulting in performance that is equivalent to random chance. In Figure 2, we see that for the A-X and B-Y presentations the model struggles to perform above random chance. Without working memory our model is still able to learn the B-X and A-Y presentations. This learning occurred because the target sequence for both B-X and A-Y did not change during context switches. In order for our model to achieve adequate performance, it was required that we constrain the possible working memory options made available to the model. The graph in Figure 1 displays how learning can become erratic when constraints are taken off of the working memory slots. This meant that we had to limit the options for each working memory slot to only have the ability to gate in current external stimuli or gate in nothing. Without these constraints in place the model would become unstable and unlearn a given task over time.

We used Keras with a TensorFlow backend to implement the LSTM version of the 1-2-AX-CPT task for comparison

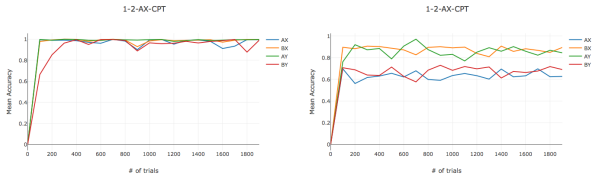

Figure (4) Mean accuracy of the 1-2-AX-CPT task using the multilayer hierarchical context model, left, and the 1-2-AX-CPT task using a simple one layer context model, right, presented over a 100 trial window. AX, BX, AY and BY trials are plotted independently.

with our own working memory model. We randomly generated test data to be fed into the neural network for learning. The LSTM model contained 3 hidden layers, each with 32 nodes. The output layer used softmax for action selection of either left button press or right button press. The parameters for the compile function in Keras were as follows: loss=categorical crossentropy; optimizer=rmsprop; and metrics=accuracy. We used the default setting for all other parameters. We then generated 10,000 sequences as training data and 100 sequences as testing data. The model was fit using a batch size=64 and epochs=20.

Results

Adding additional layers of working memory slots allowed our model to retain information in a hierarchical manner without the need for the continuous presence of stimuli. This allowed our model to solve the 2-layer hierarchical maze task in which the cue was only flashed once and the system had to determine whether or not to remember the initial cue. The model was able to learn the correct Q function for the given task despite multiple distractors. Also, the 2-layer hierarchical maze task has a large state space, so we noticed that the working memory encoding required large HRRs in order to get adequate learning. The HRR size needed to increase from 32000 for a 1-layer hierarchical task to 64000. If the HRR was less than 64000 we observed that the Q function for the task was unable to properly learn the correct values.

The performance metric we used in order to validate our model's ability to learn the 1-2-AX-CPT task was the mean accuracy per 100 trials. The mean accuracy was gathered by calculating the mean percentage of correct responses to the A-X, B-X, A-Y and B-Y letter sequences independently. We then graphed the mean correct action and saw that the percentages of correct actions maintained levels above 95% accuracy and was stable thereafter (see Figure 4). Small residual error remains due to the epsilon-soft policy employed, indicating our model can successfully solve the 1-2-A-X-CPT task. Comparatively, the simple one layer model generated suboptimal accuracy as conveyed in Figure 4. Also, behavioral data showed that human subjects were able to perform the 1-2-A-X-CPT task at a 95% mean accuracy rate (Nee & Brown, 2013). Once we set epsilon-soft=0 our model was

able to exceed human performance and reach 100% accuracy resulting in optimal behavior. We validated our model's ability to solve the A-X-CPT task by logging the mean percentage of correct actions per 100 episodes. Similarly, we logged the agent's response to the A-X, B-X, A-Y, and B-Y presentations. We observed our model was able to achieve performance accuracy of 100% and stabilize for the remaining episodes. Our model's ability to solve these cognitive tasks makes it amenable for human and animal performance data comparison.

The LSTM model was able to successfully learn the 1-2-A-X-CPT task. The LSTM model was able to obtain 100% accuracy rates while learning the task. This makes sense, being that the LSTM model uses a more informative supervised learning signal, making it easier to learn a given task. In contrast, our model only relies on reinforcement error signals to adjust the weights for learning. Also, our hierarchical model is a more biologically plausible simulation of the interaction between the prefrontal cortex and the mesolimbic dopamine system. Despite the previously stated constraints, our model was still able to obtain 100% accuracy rates while learning the 1-2-A-X-CPT task, just like the LSTM model, once we set epsilon-soft=0.

Discussion

The ability to model the brain gives us the ability to run simulations of certain cognitive phenomenon in which we lack clear understanding. Our working memory models are based on the interactions between the prefrontal cortex and the mesolimbic dopamine system. Understanding the role that the dopamine system plays in relation to cognition is one aspect that our model focuses on. To model dopamine function, we use learning algorithms based on temporal differences in estimated reward. In addition, we used HRRs which allowed for us to encode concepts dynamically, as compared to previous models within the WMtk. Our model extended the WMtk by allowing for multilayer of hierarchical context to be learned. Adding multiple layers of working memory slots gives our model the ability to handle more complex problems and tasks. Our model's capacity to learn both the A-X-CPT and 1-2-A-X-CPT validates our model's ability to learn partially observable tasks with hierarchical context reasoning. Unlike the LSTM model, our working memory model allows for the transparent viewing of the explicit storage of working memory contents in the PFC.

In the future, we would like to test the model's ability to match performance for Parkinson's patients or others with known dopamine dysfunction to test the appropriateness of SARSA for modeling the dopamine system.

References

Braver, T. S., Rush, B. K., Satpute, A. B., Racine, C. A., & Barch, D. M. (2005). Context processing and context maintenance in healthy aging and early stage dementia of the Alzheimer's type. *Psychology and Aging*, *20*(1), 33–46. doi: 10.1037/0882-7974.20.1.33

Busch, M. A., Skubic, M., Keller, J. M., & Stone, K. E. (2007, April). A robot in a water maze: Learning a spatial memory task. In *Proceedings 2007 ieee international conference on robotics and automation* (p. 1727-1732). doi: 10.1109/ROBOT.2007.363572

Dubois, G. M., & Phillips, J. L. (2017). Working Memory Concept Encoding Using Holographic Reduced Representations..

Erdemir, E., Frankel, C. B., Kawamura, K., Gordon, S. M., Thornton, S., & Ulutas, B. (2008, Sept). Towards a cognitive robot that uses internal rehearsal to learn affordance relations. In *2008 ieee/rsj international conference on intelligent robots and systems* (p. 2016-2021). doi: 10.1109/IROS.2008.4650745

Frank, M. J., Loughry, B., & O'Reilly, R. C. (2001). Interactions between frontal cortex and basal ganglia in working memory: A computational model. *Cognitive, Affective, & Behavioral Neuroscience*, *1*(2), 137–160. doi: 10.3758/CABN.1.2.137

Hochreiter, S., & Schmidhuber, J. U. (1997). Long Short-Term Memory. *Neural Computation*, *9*(8), 1735–1780. doi: 10.1162/neco.1997.9.8.1735

Kawamura, K., Gordon, S. M., Ratanaswasd, P., Erdemir, E., & Hall, J. F. (2008). Implementation of cognitive control for a humanoid robot. *International Journal of Humanoid Robotics*, *05*(04), 547-586. doi: 10.1142/S0219843608001558

Nee, D. E., & Brown, J. W. (2013, sep). Dissociable frontal-striatal and frontal-parietal networks involved in updating hierarchical contexts in working memory. *Cerebral Cortex*, *23*(9), 2146–2158. doi: 10.1093/cercor/bhs194

O'Reilly, R., & Munakata, Y. (2000). *Computational explorations in cognitive neuroscience*. A Bradford Book.

Phillips, J. L., & Noelle, D. C. (2005). A Biologically Inspired Working Memory Framework for Robots *.

Plate, T. A. (1995). Holographic Reduced Representations. *IEEE TRANSACTIONS ON NEURAL NETWORKS*, *6*(3).

Schultz, W., Dayan, P., & Montague, P. R. (1997). A neural substrate of prediction and reward. *Science*, *275*(June 1994), 1593–1599. doi: 10.1126/science.275.5306.1593

Sutton, R. S. (1988). Learning to Predict by the Methods of Temporal Differences. *Machine Learning*, *3*, 9–44.

Wang, X., Tugcu, M., Hunter, J. E., & Wilkes, D. M. (2009). Exploration of configural representation in landmark learning using working memory toolkit. *Pattern Recognition Letters*, *30*(1), 66 - 79. doi: https://doi.org/10.1016/j.patrec.2008.09.002