ARTICLE **Communicated by Terrence Sejnowski**

Flexible Working Memory Through Selective Gating and Attentional Tagging

Wouter Kruijne
w.kruijne@gmail.com
Faculty of Behavior and Movement Sciences, Vrije Universiteit Amsterdam, 1081 BT Amsterdam, Noord Holland, The Netherlands

Sander M. Bohte
S.M.Bohte@cwi.nl
Machine Learning Group, Centrum voor Wiskunde & Informatica, 1098 XG Amsterdam, Noord Holland, The Netherlands; Swammerdam Institute of Life Sciences, University of Amsterdam, 1098 XH Amsterdam, Noord Holland, The Netherlands; and Department of Computer Science, Rijksuniversiteit Groningen, 9747 AG Groningen, The Netherlands

Pieter R. Roelfsema
p.roelfsema@nin.knaw.nl
Department of Vision & Cognition, Netherlands Institute for Neuroscience, 1105BA Amsterdam, Noord Holland, The Netherlands; Department of Integrative Neurophysiology, Center for Neurogenomics and Cognitive Research, Vrije Universiteit Amsterdam, 1981 HV Amsterdam, Noord Holland, The Netherlands; and Department of Computer Science, Rijksuniversiteit Groningen, 9747 AG Groningen, The Netherlands

Christian N. L. Olivers
c.n.l.olivers@vu.nl
Faculty of Behavior and Movement Sciences, Vrije Universiteit Amsterdam, Amsterdam, Noord Holland, The Netherlands, Department of Psychiatry, Academic Medical Center, Amsterdam, The Netherlands

Working memory is essential: it serves to guide intelligent behavior of humans and nonhuman primates when task-relevant stimuli are no longer present to the senses. Moreover, complex tasks often require that multiple working memory representations can be flexibly and independently maintained, prioritized, and updated according to changing task demands. Thus far, neural network models of working memory have been unable to offer an integrative account of how such control mechanisms can be acquired in a biologically plausible manner. Here, we present WorkMATe, a neural network architecture that models cognitive control over working memory content and learns the appropriate control

Neural Computation **33, 1–40** (2021)
https://doi.org/10.1162/neco_a_01339

operations needed to solve complex working memory tasks. Key components of the model include a gated memory circuit that is controlled by internal actions, encoding sensory information through untrained connections, and a neural circuit that matches sensory inputs to memory content. The network is trained by means of a biologically plausible reinforcement learning rule that relies on attentional feedback and reward prediction errors to guide synaptic updates. We demonstrate that the model successfully acquires policies to solve classical working memory tasks, such as delayed recognition and delayed pro-saccade/anti-saccade tasks. In addition, the model solves much more complex tasks, including the hierarchical 12-AX task or the ABAB ordered recognition task, both of which demand an agent to independently store and updated multiple items separately in memory. Furthermore, the control strategies that the model acquires for these tasks subsequently generalize to new task contexts with novel stimuli, thus bringing symbolic production rule qualities to a neural network architecture. As such, WorkMATe provides a new solution for the neural implementation of flexible memory control.

1 Introduction

Complex behavior requires flexible memory mechanisms for dealing with information that is no longer present to the senses but remains relevant to current task goals. For example, before we decide it is safe to change lanes on a highway, we sequentially accumulate evidence in memory from various mirrors and the road ahead of us. Importantly, such complex behavior requires memory operations beyond mere storage. Not every object that we observe on the highway needs to be memorized, while often it is a specific combination of information (e.g., multiple cars and signs) that determines whether it is safe to switch lanes. As any novice driver has experienced, learning to properly apply these operations of selecting, maintaining, and managing the correct information in memory can take quite some effort. Yet after sufficient practice, we learn to apply these skills and abstract the essence across multiple environments, regardless of the specifics of the road or cars around us. This example illustrates the core functions that define working memory (WM), and that – in the words of O'Reilly and Frank (2006) – make WM. First, WM is *flexible* in that control processes determine what information is stored, when it is updated, and how it is applied during task performance. Second, the rules that govern these control operations for a given task setting are *trainable* and can be acquired with practice. Third, after training, these rules then *generalize* to the same task setting with different stimuli. It is this combination of flexibility, trainability, and generalizability that makes WM a cornerstone of cognition, not only in humans but also in nonhuman primates (Warden & Miller, 2007, 2010; Naya & Suzuki, 2011).

Here we present a neural network model of WM that integrates these core components.

Before we describe the WorkMATe model, we briefly explain how it extends previous models that either focused on the generic storage of arbitrary sensory stimuli in memory or on the learning of content-specific memory operations.

1.1 Models of Storage and Matching. A number of previous neural network models explain how the brain can temporarily maintain information (Brunel & Wang, 2001; Amari, 1977; Mongillo, Barak, & Tsodyks, 2008; Barak & Tsodyks, 2007; Fiebig & Lansner, 2017) and how different items can be maintained separately (Oberauer & Lin, 2017; Raffone & Wolters, 2001; Jensen & Lisman, 2005; Schneegans & Bays, 2017). Given their emphasis on storage, one of the most commonly modeled WM tasks is the delayed recognition task, in which the observer responds according to whether an observed stimulus matches a memorized stimulus.

Delayed recognition tasks do not require an agent to act on the specific content of information in memory. Rather, the agent produces a response based on the presence or absence of sufficient similarity between two successively presented stimuli, which in principle could be anything. Experimental work has revealed that both human and nonhuman primates can almost effortlessly determine such matches, even for stimuli that have never been seen before (Downing & Dodds, 2004; Warden & Miller, 2010, 2007; Siegel, Warden, & Miller, 2009), and studies have demonstrated neurons in frontal as well as parietal cortices whose activity depends on the match between sensory input and memory content (Miller, Erickson, & Desimone, 1996; Freedman, Riesenhuber, Poggio, & Miller, 2003; Rawley & Constantinidis, 2010). Taken together, these findings suggest that the computations governing matching tasks—determining the similarity or degree of match—are relatively independent of stimulus content.

Most models for matching, recognition, and recall tasks therefore implement a content-independent computation of a match signal. Ludueña and Gros (2013) demonstrated that a relatively simple, self-organizing neural network can learn to detect coactivation in neuronal pools representing similar information with nonoverlapping codes, allowing for a match signal between sensory and memory information to emerge upon presentation. Match signals also emerge in models of associative memory that assume one-shot Hebbian learning of arbitrary information in the hippocampus. In these models, the ease of subsequent context-driven retrieval provides an index of stimulus-memory similarity, which is used to simulate recall probabilities and response times (Howard & Kahana, 2002; Lohnas, Polyn, & Kahana, 2015; Raaijmakers & Shiffrin, 1981; Howard & Eichenbaum, 2013; Norman & O'Reilly, 2003). Meyer and Rust (2018) have shown that repetition suppression in the inferotemporal cortex after a repeated presentation of a stimulus predicts recognition performance for arbitrary stimuli in the

macaque (see also Engel & Wang, 2011; Sugase-Miyamoto, Liu, Wiener, Optican, & Richmond, 2008). The same idea is prevalent in models of visual search, where a match signal is computed between an item in memory and stimuli present in the to-be-searched scene, which is subsequently used to optimally guide attention (Zelinsky, 2008; Rao, Zelinsky, Hayhoe, & Ballard, 2002; Hamker, 2005).

In these models, match signals are automatically computed as an emergent consequence of the interaction between perception and memory, adding to the utility of WM without the need for training on specific stimulus content first. In contrast, more complex tasks call for additional WM operations, decisions, and motor actions depending on specific content. In the lane-changing example, an empty rear-view mirror may indicate that overtaking is safe unless the side mirror says otherwise. Generic match models typically do not explain how memory content is controlled, how control policies can be acquired through training, and how memory content in combination with sensory information leads to action selection. Such trainable, flexible, action-oriented models of WM will be discussed next.

1.2 Models of Memory Operations. A rather different class of models has focused on how WM can be trained to solve tasks in which multiple different stimuli map onto different responses—that is, how the cognitive system learns which of a number of available actions, including memory operations, is appropriate given particular (combinations of) stimuli. Training neural network models to solve tasks means that as the network processes examples, weights are updated to establish a desirable mapping between an input and output stream. In a reinforcement learning setting, a desired, optimal mapping yields a policy that maximizes reward and minimizes punishment. For multilayer neural networks, this becomes a problem of structural credit assignment, where the learning algorithm needs to determine to what extent a connection weight contributed to the outcome. For memory tasks, there is an additional temporal credit assignment problem, as the outcome of certain actions (e.g., storing an item into memory) will only later in the trial lead to success or failure. An ongoing issue in deep learning is how these credit assignment problems might be solved in a biologically plausible manner (Lillicrap, Cownden, Tweed, & Akerman, 2016; Richards & Lillicrap, 2019; Whittington & Bogacz, 2017; Scellier & Bengio, 2018; Marblestone, Wayne, & Kording, 2016).

One biologically plausible solution to temporal and structural credit assignment in WM tasks is provided by the AuGMEnT algorithm (Rombouts, Bohte, & Roelfsema, 2015; Rombouts, Roelfsema, & Bohte, 2012; Rombouts, Bohte, Martinez-Trujillo, & Roelfsema, 2015), which in turn is based on the AGREL model for perceptual learning (Roelfsema, van Ooyen, & Watanabe, 2010; Roelfsema & Ooyen, 2005; van Ooyen & Roelfsema, 2003). These models demonstrate that attentional feedback can play a critical role in solving credit assignment (Roelfsema & Holtmaat, 2018). The architecture used by

AuGMEnT is a multilayer neural network with a recurrent memory layer to maintain information. The output of the neural network is the expected reward value associated with each action. Upon selection of an action, the attentional feedback mechanism tags synapses that contributed to this action. When an action does not yield the expected reward, a reward prediction error (RPE) signal is broadcast across the network, which drives weight changes in tagged synapses. Through these mechanisms, AuGMEnT implements a rudimentary but trainable WM architecture. This architecture can learn to solve a variety of memory tasks where sequences of stimuli need to be integrated over time to yield a correct response (Rombouts, Bohte, & Roelfsema, 2015; Rombouts et al., 2012). However, at the same time, AuGMEnT lacks the operations that define the flexibility of primate WM: its store accumulates relevant information but does not allow, for example, items to be separately updated, selectively forgotten, or only to be encoded under certain conditions.

A highly popular neural network architecture that does incorporate such flexible control mechanisms is the long- short-term memory (LSTM) architecture (Hochreiter & Schmidhuber, 1997). This architecture introduces a gated memory store, implemented through gating units that open or close dependent on activity in the rest of the network. These gates allow an agent to control which information is allowed entry into memory, how new information is integrated, and which information is read out at any given time. LSTM networks and similar architectures are now commonplace in modern deep learning systems, which is a testament to their power (Gers, Schmidhuber, & Cummins, 1999; Gers, Schraudolph, & Schmidhuber, 2002; Monner & Reggia, 2012; Cho, van Merrienboer, Bahdanau, & Bengio, 2014; Costa, Assael, Shillingford, de Freitas, & Vogels, 2017; Graves & Schmidhuber, 2005; Graves et al., 2016). However, while LSTM architectures allow for flexible control over memory content, they were not developed with biological plausibility in mind: typical implementations rely on rather implausible learning rules from a biological perspective (Graves & Schmidhuber, 2005; Hochreiter & Schmidhuber, 1997). LSTMs can be trained using reinforcement learning methods (Bakker, 2002, 2007), but the complexity of the recurrent architecture renders training implausibly inefficient when compared to animal learning (requiring millions of trials to learn a relatively straightforward T-maze task).

Probably the most strongly established biologically inspired model of flexible WM control so far is the prefrontal cortex-basal ganglia working memory model (PBWM; O'Reilly & Frank, 2006; Hazy, Frank, & O'Reilly, 2006, 2007). PBWM allows for flexible memory control in a manner inspired by LSTM, but was designed with a strong focus on biological plausibility. PBWM only gates the entry of sensory stimuli into its WM store in an all-or-none fashion. Specifically, the basal ganglia determine whether items are allowed to enter WM on the basis of selecting internal gating actions. The model can learn complex hierarchical tasks (such as 12-AX,

described below), which require selective updating and maintenance of relevant items in WM while preventing the storage of distractor stimuli. However, as Todd, Niv, and Cohen (2009) noted, the exact functionality of PBWM is considerably obscured by the fact that it is a rather complex model with a highly interwoven architecture of a range of neural subsystems and several parallel learning algorithms, both supervised and unsupervised (O'Reilly, Frank, Hazy, & Watz, 2007; O'Reilly, 1996b, 1996a). Todd et al. (2009) presented a simplified PBWM model that distills only a core feature of PBWM, which is the use of internal gating actions to control memory content. This model replaces all biologically inspired neural subcomponents with a more abstract tabular representation of all possible input and memory states. States are then mapped to external motor and internal gating actions, the value of which is learned through a standard reinforcement learning algorithm that uses eligibility traces. The simplified PBWM model thus discards most of the biological realism of PBWM, but demonstrates its core functionality, the control over memory content through internal gating actions, that can be acquired using reinforcement learning alone.

Thus, these trainable, action-oriented models (AuGMEnT, LSTM, and PBWM) demonstrate working memory functions that go beyond mere storage and matching. In both LSTM and PBWM, memory control is flexible, as multiple items can be encoded, maintained, and updated separately, and there are mechanisms that prevent interference from task-irrelevant stimuli. LSTM and AuGMEnT solve tasks by constructing memory representations tailored to the task at hand: sensory information is encoded in a manner that links them to relevant actions in order to solve the task at hand. By focusing on tasks beyond mere storage, action-oriented models can explain how memory content can be utilized to solve a task. They provide control operations to update specific content and learn to apply them based on reinforcement. Yet these models do not easily cope with arbitrary stimuli that the agent has never observed before, and thus they lack the symbolic production-rule quality of WM operations. For this, the models would need the generic storage approach that matching-oriented models utilize, and it remains untested whether generalized matching signals can be integrated in this type of model.

1.3 WorkMATe: Generalizable, Flexible, Trainable WM. As laid out above, existing neural network models of flexible memory vary according to their focus of functionality (storage versus action). Here, we present WorkMATe (working memory through attentional tagging) a neural network architecture that integrates the core components of these models to arrive at a model of WM that is trainable, flexible, and generalizable. The model utilizes a new, gated memory circuit inspired by PBWM and LSTM to maintain multiple items separately in WM. We include a straightforward neuronal circuit for a generic matching process that compares the

memory content to incoming new stimuli. These structures are embedded in a multilayer neural network that is trained using the simple and biologically plausible reinforcement learning rule of AuGMEnT. We demonstrate how the resulting neural network architecture solves complex, hierarchical tasks with multiple stimuli that have different roles depending on context and that it can rapidly generalize an acquired task policy to novel stimuli that it has never encountered before.

2 Materials and Methods

We first describe the architecture of WorkMATe and how it compares the memory representations to sensory stimuli, as well as how its learning rule resolves the credit assignment problem by combining reinforcement learning with an attentional feedback mechanism. We then illustrate the virtues of WorkMATe in four general versions of popular WM tasks. First, we model a basic delayed recognition task with changing stimulus sets to illustrate how the model indeed generalizes to novel stimuli. Second, we illustrate how the model tackles hierarchical problem solving with the classic memory-juggling 12-AX task, where the agent is presented with a stream of symbols and must learn the rule. Third, the challenges of both tasks are combined by training the model on a sequential recognition task introduced by Warden and Miller (2007, 2010), where an agent has to store multiple, sequentially presented items and match them to subsequent test stimuli, in the same order. Here again we assess both flexibility and generalization to new stimuli. Finally, we turn to the delayed pro-saccade/anti-saccade task (Everling & Fischer, 1998; Munoz & Everling, 2004; Hallett, 1978; Brown, Vilis, & Everling, 2007; Gottlieb & Goldberg, 1999), because it allows for a direct comparison between the present architecture and the previous gateless AuGMEnT model (Rombouts, Bohte, & Roelfsema, 2015).

We present a model architecture (see Figure 1A) that achieves good performance in all four memory tasks. The parameter values and other network specifics were kept the same in all simulations. An overview of these parameters is given in Table 1. We describe the details of these computations, followed by a discussion of our simulations. Code used to implement the architecture and run the simulations is available: https://osf.io/jrkdq/?view_only=e2251230b9bf415a9da837ecba3a7d64.

2.1 Input Representations and Feedforward Sweep. The model is a neural network that receives input x at every time step t. Input is composed of sensory representations x_s and a representation of time x_τ. Sensory representations are, in all simulations, defined as binary patterns with activity levels [1,0] that uniquely identify each stimulus. The time representation is inspired by "*time* cells" as identified in multiple cortical and subcortical areas (Howard & Eichenbaum, 2013; Mello, Soares, & Paton, 2015; Naya & Suzuki, 2011; Tsao et al., 2018; Paton & Buonomano, 2018). These

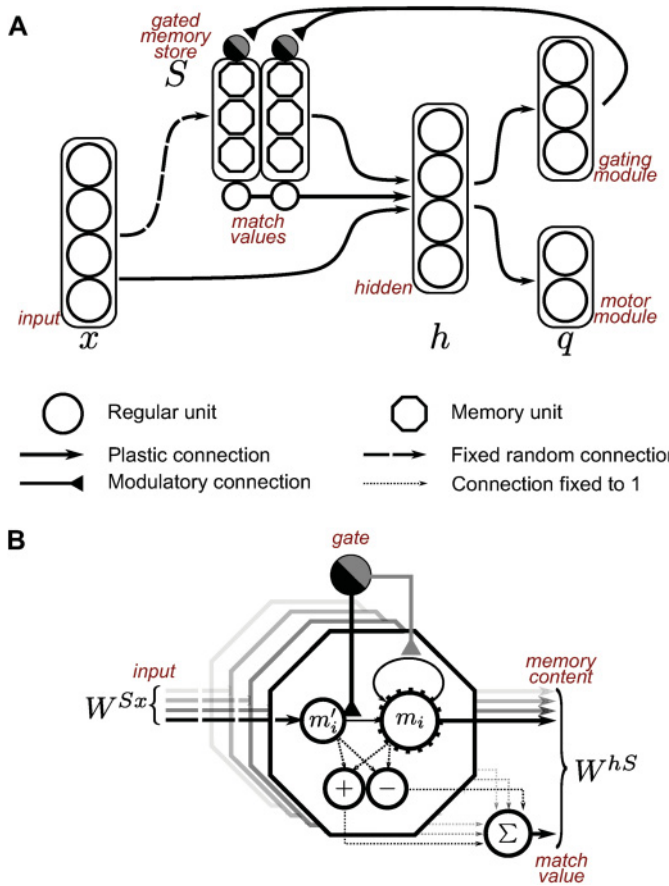

Figure 1: (A) The network architecture used in all simulations: a standard multilayer network, complemented by a gated store composed of two independent memory blocks. The input layer and memory store both project to the hidden layer, which in turn projects to two output modules. There, activity encodes Q-values that drive action selection. (B) Memory unit within a block, with a closed gate: the memory content is maintained via self-recurrent connections. Additionally, a match value is computed between sensory and memory information by comparing a projection of the sensory information (m_i') to memory content (m_i). The comparison is performed by two units that respond to positive and negative disparities between the two values. Their output is summed across memory units, yielding one match value for each block. The closed gate inhibits the connection between $m_i' \rightarrow m_i$ so that the original memory is maintained. Only when a gating action is selected, the recurrent projection is inhibited and $m_i' \rightarrow m_i$ is opened so that memory content is updated. Figure 2 illustrates network activity in a task context, and Table 1 lists the number of units in each layer.

Table 1: Parameters Used in All Simulations, Unless Stated Otherwise.

Symbol	Name	Value
β	Learning rate	0.15
γ	Temporal discounting factor	0.9
λ	Eligibility trace decay rate	0.8
ϵ	Exploration rate	0.025
Symbol	Name	Size
x	Total input units	17
x_s	Sensory input units	7
x_τ	Time input units	10
h	Hidden units	15
S	Memory store (blocks)	2
m_i	Units per memory block	14
q_{int}	Output q-units (internal actions)	3
q_{ext}	Output q-units (external actions)	2 or 3 (task dependent)

cells encode time by their delayed response profiles, each peaking at different times relative to the onset of a trial. For most tasks, the contribution of the time cells is minimal: the correct policy depends on the sequentially presented stimuli rather than their exact timing. However, in the ABAB-ordered recognition task, the moment at which a state is presented is critical for the correct response. For such tasks, we assume that the network can learn to make use of an underlying sense of time, without having to learn such a representation anew for new tasks at hand. In other words, a sense of time is considered as a part of perception and not dependent on working memory control. In our model units, activity profiles of time cells are defined as symmetrically increasing and decreasing activity around each unit's unique peak time, at the levels [0.125, 0.25, 0.5, 1.0]. An example sequence of inputs from time and sensory units is depicted in Figure 2C.

The network projects the input representation x to two different layers. One is a regular hidden layer h in which units are activated via the projection weight matrix W^{hx}. The other layer is the memory store S, which is composed of two equally sized blocks m_1 and m_2. Sensory information is encoded into one of these stores by means of the projection W^{Sx}. By separating the two stores, WorkMATe is able to selectively update part of its memory content with new information while leaving another part of its memory unaffected. In the current implementation, a stimulus that is gated into memory wholly replaces any previously stored content. Note that other than selective encoding, the memory blocks together act as a single memory layer that projects to the hidden layer via a single set of plastic connections W^{hX}.

During the initial feedforward sweep of activity, the projection $W^{Sx} \cdot x = S' = \{m'_1, m'_2\}$ serves to compute the match value between the projected

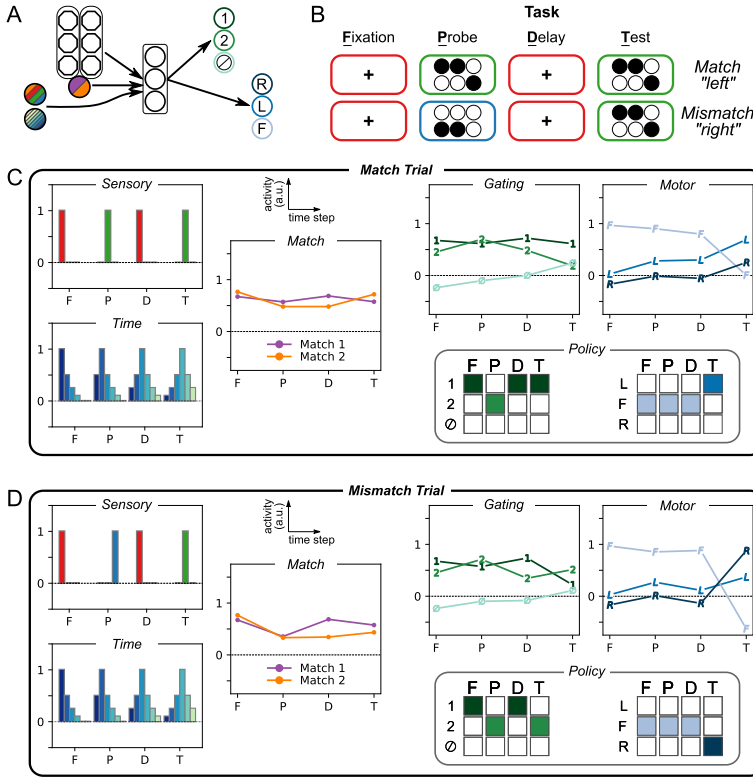

Figure 2: Illustrative model performance on a match and a mismatch trial after training. (A) Model architecture (as in Figure 1). Colors in the input and output layers correspond to the graphs in panels C and D. (B) An example match trial (top) and example mismatch trial (bottom). After fixation, a probe stimulus (a six-bit pattern) is presented that must be maintained in memory. After a delay, the test stimulus is presented. The agent has learned to hold fixation until the test, and then respond "left" in case of a match and "right" in case of a mismatch. (C) Input, match and output activity (in arbitrary units) for each time step in an example match trial. F-P-D-T are four different time steps (cf. panel B). Sensory units have unique patterns for each stimulus, though these patterns might partially overlap. The time cells (only six depicted) each peak at a different moment in time. Colors and height of the bar indicate different time cells and their activity, respectively. Match values reflect the overlap between input and memory content. The resulting gating and motor q-values determine behavior through winner-take-all selection. The resulting policy is depicted as grids with a filled square for each selected action. A filled square in the gating network denotes the updating of the activity of a memory cell. (D) Same as panel C, but for a mismatch trial. Note the different match values, which drive the eventual critical decision. The policy for both trial types is identical up to the test stimulus. See the text for more detail.

sensory representation m_i' and the contents of each memory block m_i. These match values are denoted as x_{m_1}, x_{m_2}. As noted above, there are a number of hypotheses on how this match value might be computed (Howard & Kahana, 2002; Meyer & Rust, 2018; Ludueña & Gros, 2013; Norman & O'Reilly, 2003). Here, we refrained from a specific modeling effort and instead computed the sum of absolute differences between the two representations, a common match metric (Zelinsky, 2012; Choo & Eliasmith, 2010; Stewart, Tang, & Eliasmith, 2011):

$$x_{m_i} = \frac{\sum \lfloor m_i - m_i' \rfloor + \lfloor m_i' - m_i \rfloor}{n}. \tag{2.1}$$

Here, n refers to the number of nodes in a block. This computation could be readily implemented by a set of accessory units that are activated by input projections and memory units, and respond to disparities between memory and sensory information (see Figure 1B). The summed activation in these units is a measure of dissimilarity, and one minus this value is used as a match signal.

The activity of the match nodes is combined with the activity of the memory content $S^* = \{S, x_m\}$ and projects to the hidden layer h. This layer integrates information from the input layer and memory stores via:

$$h = f(h_a) = f(W^{hx} \cdot x + W^{hS^*} \cdot S^* + b_h), \tag{2.2}$$

where b is a bias input vector and f is a standard sigmoid transfer function.

The hidden layer h projects to the output layer q through the weights W^{qh}:

$$q = \{q_{\text{int}}, q_{\text{ext}}\} = W^{qh} \cdot h + b_q. \tag{2.3}$$

These output values q will, after training, approximate the Q-values of each of the possible response options. The Q-value is the sum of the expected immediate and temporally discounted future rewards for the remainder of the trial that the agent can acquire by selecting that action. The output layer q is divided into two modules: one for *external* and one for *internal* actions. External actions reflect the motor response options of the agent, which in our simulations reflect either holding or releasing a lever or fixating left, right, or in the center of the screen. The internal actions determine memory gating. Based on the action selected in this layer, the currently presented stimulus is either memorized in block 1, in block 2, or ignored. On most time steps, the agent will selects internal and external actions associated with the units with the highest activation {**argmax**(q_{int}), **argmax**(q_{ext})}. On rare exploration time steps, determined by exploration rate ϵ, the agent will

select a random action, determined by a Boltzmann controller operating over the q-values within each module.

2.2 Storage and Gating. The memory layer S in WorkMATe is functionally similar to that used in PBWM. Separate memory representations are maintained via self-recurrent projections in the memory store. This is a strong abstraction of the presumed neurophysiological mechanisms of WM maintenance in the primate brain, as there is no consensus in the literature as to whether items in WM are functionally organized into slots (Zhang & Luck, 2008; Cowan, 2010), continuous resources (Bays & Husain, 2008; Van den Berg, Awh, & Ma, 2014; Ma, Husain, & Bays, 2014), hierarchically organized feature bundles (Brady & Alvarez, 2011), or through interactions with long-term memory representations (Orhan, Sims, Jacobs, & Knill, 2014). Here, we remain largely agnostic regarding the precise representation, but choose a mechanism where items in memory can be maintained separately, can be updated separately, and can be selectively ignored to prevent interference (O'Reilly & Frank, 2006). We will show that this approach allows us to investigate how complex cognitive control over the content of WM can be acquired via reinforcement learning.

After feedforward processing is completed and the Q-values in the output layer have been computed, the agent selects a gating action from $\{g_1, g_2, g_\emptyset\}$ in order to either gate the current sensory representation into block m_1, m_2 or to prevent the stimulus from entering the memory store altogether. Note that unlike in PBWM, a memory representation m_i is not a direct copy of sensory information. Rather, it is a compressed representation of the input representation, encoded via the weights W^{Sx}. This allows for generalization of learned task rules to novel stimuli.

Importantly, unlike the other, trained projections in the model, W^{Sx} remains fixed throughout each model run at the connection strengths it obtains through random initialization. As a result, memory representations of a stimulus are not tuned to the task at hand and will differ depending on whether they are encoded in block 1 or block 2. Previous work (Barak, Sussillo, Romo, Tsodyks, & Abbott, 2013; Saxe et al., 2011; Bouchacourt & Buschman, 2019) has demonstrated that untrained random projections can be used for memory encoding in a useful manner as long as dissociable memory representations can be formed. This is not to say that memory encoding in the brain is necessarily random and untrained, but we will use this architecture to illustrate that without additional tuning, the model can successfully encode stimuli in a generic manner and will explore whether learned policies generalize to novel stimulus sets.

2.3 Learning. Learning in the model follows the AuGMEnT-algorithm (Rombouts, Bohte, & Roelfsema, 2015), which was derived from the AGREL learning rule (Roelfsema, van Ooyen, & Watanabe, 2010). At every time step, the model predicts the Q-value of each of its possible actions. These

values are represented in the motor and the gating module in the network's output layer. Based on these values, the gating module selects an internal action and the motor module an external action, in parallel. The sum of the two Q-values associated with the selected actions, $q_{\text{int}}(t) + q_{\text{ext}}(t)$, reflects the total Q-value Q_t, that is, the network's estimate of the sum of discounted rewards predicted for the remainder of the trial. Note that there is no a priori constraint on how these two values are weighted, though in all the tasks simulated here, we found the Q-values in internal and external action modules to converge to comparable values, with each module accounting for approximately half of the total Q-value associated with the selected pair of actions.

The selected actions form a binary vector z, which is 1 for the units reflecting the selected actions and 0 otherwise. Once actions have been selected, an attentional feedback signal that passes through the system through attentional feedback connections originates from these units. This recurrent signal is used to tag synapses that contributed to the selected actions. These synaptic tags correspond to eligibility traces in traditional $SARSA(\lambda)$ reinforcement learning. The value of these tags gradually decays at each time step with a rate $\alpha = 1 - \lambda\gamma$, where γ is a temporal discounting factor (discussed below) and λ corresponds to common usage to indicate the persistence of an eligibility trace. The update of a tag depends on the contributions of a synapse to a selected action. Formally, this means that in each plastic connection in the weight matrices W^{Sx}, W^{hx}, W^{hS^*}, W^{qh}, each Tag_{ji} between presynaptic unit i and postsynaptic unit j is updated according to:

for the connections $h \rightarrow$ q:

$$\Delta Tag_{ji}^{qh} = -\alpha Tag_{ji}^{qh} + h_i \cdot z_j, \tag{2.4}$$

and for the connections $x \rightarrow h$ and $S \rightarrow h$:

$$\Delta Tag_{ji}^{hx} = -\alpha Tag_{ji}^{hx} + x \cdot \sigma'(h_j) \cdot w'_j, \tag{2.5}$$

$$\Delta Tag_{ji}^{hS} = -\alpha Tag_{ji}^{hS} + S \cdot \sigma'(h_j) \cdot w'_j, \tag{2.6}$$

with:

$$w'_j = \sum_k w_{kj} \cdot z_k. \tag{2.7}$$

Here, the term h_j refers to the output of hidden unit j, and σ' is the derivative of the sigmoid transfer function. The term w'_j indicates the amount of recurrent feedback from the action vector z onto the hidden layer nodes. This feedback is determined by the weight between the hidden nodes and the selected actions where $z_k = 1$ if action k is selected and

$z_j = 0$ for all nonselected actions j. Feedback connections are updated via the same learning rule as the feedforward connections. Therefore, the feedforward and feedback connections remain or become reciprocal, which has been observed in neurophysiology (Mao et al., 2011).

Synaptic connections are updated when the synaptic tags interact with a global reward-prediction error (RPE) signal. This signal, $\delta(t)$, is modeled after striatal dopamine and reflects the signed difference between the expected and obtained reward. This is expressed in the SARSA temporal difference rule:

$$\delta(t) = r(t) + \gamma Q(t) - Q(t-1). \tag{2.8}$$

That is, the model values the previous actions on the basis of the obtained reward $r(t)$ plus the amount of expected future reward $Q(t)$ multiplied by a temporal discounting factor $\gamma \in [0, 1]$ and contrasts this valuation with the previously expected value $Q(t-1)$. The RPE then triggers a global, neuromodulatory signal that spreads uniformly the network, and interacts with the synaptic tags to modify weights, that is:

$$\Delta W(t) = \beta \delta(t) Tag(t), \tag{2.9}$$

where β is the learning rate. Note that the two forces that determine weight updates are the RPE and the synaptic tags. The RPE signal assures that once the model accurately predicts rewards, the resulting $\delta(t) = 0$ and the weights remain unchanged, which allows the model to converge on an on-policy solution. The synaptic tags, on the other hand, solve the credit assignment problem by means of attention-gated feedback: units in the hidden layer whose activity had a larger influence on the Q-value of chosen actions receive stronger feedback and form stronger tags, whereas units that did not contribute to the selected action will not have weight updates. Previous work has established that this learning rule offers a biologically feasible approximation of error, backpropagation (Rombouts, Bohte, & Roelfsema, 2015).

In all simulations, the model was trained using the same, general principles that are in line with typical animal learning. Changes in the environment and the reward that was delivered depended on the external actions of the agent, whereas internal actions that pertain to WM updates were never directly rewarded. Trials were aborted without reward delivery whenever the model selected an incorrect motor response. Reward could be obtained twice in a trial. First, all tasks required the agent to perform a default action throughout the trial (such as maintaining gaze at a central fixation point or holding a response lever) until a memory-informed decision had to be made. We encouraged the initial selection of this action by offering a small shaping reward ($r = 0.2$) for selecting this action at the first time step. At

the end of a trial, if the correct decision was made in response to a critical stimulus, a large reward ($r = 1.5$) was delivered. In our model assessments, trials were considered correct only when both rewards were obtained.

Although not all inputs and computations were strictly necessary or useful in every task, the network architecture, parameter values and the representation of inputs were kept constant across simulations; Across tasks, we modified only the external action module to represent the valid motor responses for the different tasks.

3 Results

3.1 Task 1: Delayed Recognition. Arguably one of the most central and at the same time straightforward WM tasks is the delayed recognition (DR) task. Here, an agent is asked to compare two stimuli separated by a retention delay, and make a response based on whether they are the same or not. Here, we show that the random, untrained encoding projections in WorkMATe not only suffice for such a comparison task, but that the solution also generalizes to stimuli that the agent has not observed before. We trained the agent on a simple DR task, where it was sequentially presented with a fixation cross, a probe stimulus, another fixation cross, and a test stimulus that would either match the probe or not (see Figures 2A and 2B). Stimuli consisted of unique binary patterns of six values (see Figure 2B for two example stimuli). One additional seventh input was used to signal the presence of the fixation dot. The agent had to withhold a response until the test stimulus appeared, and it then had to make one of two choices to indicate whether the test stimulus matched the probe (we used a leftward/rightward saccade for match/mismatch). We modeled a total of 750 networks with randomly initialized weights. During initial training, the probe and test stimuli were chosen from a set of three unique stimuli (set 1). Once performance had converged (more than 85% correct trials), the stimulus set was replaced by a set of three novel stimuli (set 2) This process was repeated until performance had converged for six sets of stimuli.

In these and all other simulations, we report convergence rates based on all trials including those with exploratory actions.

Figures 2C and 2D illustrate how an example trained network solves a given match and mismatch trial. The left-most bar charts illustrate the network input, consisting of sensory and time units. Both trials have the same test stimulus (green bar) but differ in their probe (blue bar in D). These stimuli are each coded as a unique, partially overlapping six-bit pattern. Each of the time cell units (left bottom bar chart) peaks at a unique time point, and in concert, they convey a drifting representation of time since the trial started. The activity in the match nodes (orange and purple curves) conveys the result of comparing the content of each memory block to currently presented stimulus: Match 1 and Match 2 for the comparison with the content in memory block 1 and memory block 2, respectively. The agent's policy is

to store the probe stimulus in block 2 and to maintain this item throughout the trial so that match signal at the test stimulus can drive the final match versus mismatch decision (the difference in Match 2 activity at the test stimulus in panels C and D).

The Q-values computed in the output modules are depicted on the right of Figures 2C and 2D. The actions with the highest value are selected and give rise to the policy, depicted under both graphs. Individual values in these modules do not allow for a straightforward interpretation: only the sum of the values of selected actions is used to drive learning and will approximate the true Q-value. In practice, however, we found that the two modules somewhat evenly contributed to this total estimate, as illustrated in this example network.

Note that the policy acquired by this agent was the same for match and mismatch conditions and would readily apply to the same task setup with novel stimuli. To examine whether the policy generalized to novel stimuli, we assessed the number of trials that an agent required to converge after each switch to a new set. The results in Figure 3 illustrate that agents were able to generalize across stimulus sets. Convergence on the first set was relatively slow (Figure 3A): the median number of trials needed for convergence was approximately 12,700, with 95% of the agents converging within 4379 to 46,724 trials. After the first switch, convergence occurred much faster after a median of 1066 trials (95% within 212 to 5288, trials). With each subsequent switch, the agents displayed further generalization, and the median number of trials needed for convergence on set 6 was only 343.5 (95% within 90 to 1994 trials. We note that 85 trials is the absolute minimum number of trials before any agent could reach our criterion of 85% accuracy.

We next assessed performance in the first 100 to 500 trials with each new set to explore how fast the agents learned the task with novel stimuli (Figure 3). Initial performance on set 1 (after 100 trials) was near chance level: approximately 1% correct in a task that required four consecutively correct actions to be selected out of three options. Performance gradually increased and reached 18.5% accuracy within 500 trials. Following the first switch (to set 2), performance did not drop back to chance: rather, agents immediately performed 55.8% correct on the first 100 trials and were 66.3% correct after 500 trials. On each subsequent set switch, immediate performance with never-before-seen stimuli kept increasing, with performance at 70.3% for the final set. On the final two sets, criterion performance (85%) was acquired within 500 trials. These results suggest that agents were indeed able to generalize the acquired policy to novel contexts, although each set switch still required some additional learning.

We suspected that one important reason that the model failed to immediately generalize to new sets might have been that agents broke fixation for novel stimuli. Note that a completely novel input pattern makes use of connections that have not been used before in the task, which could trigger

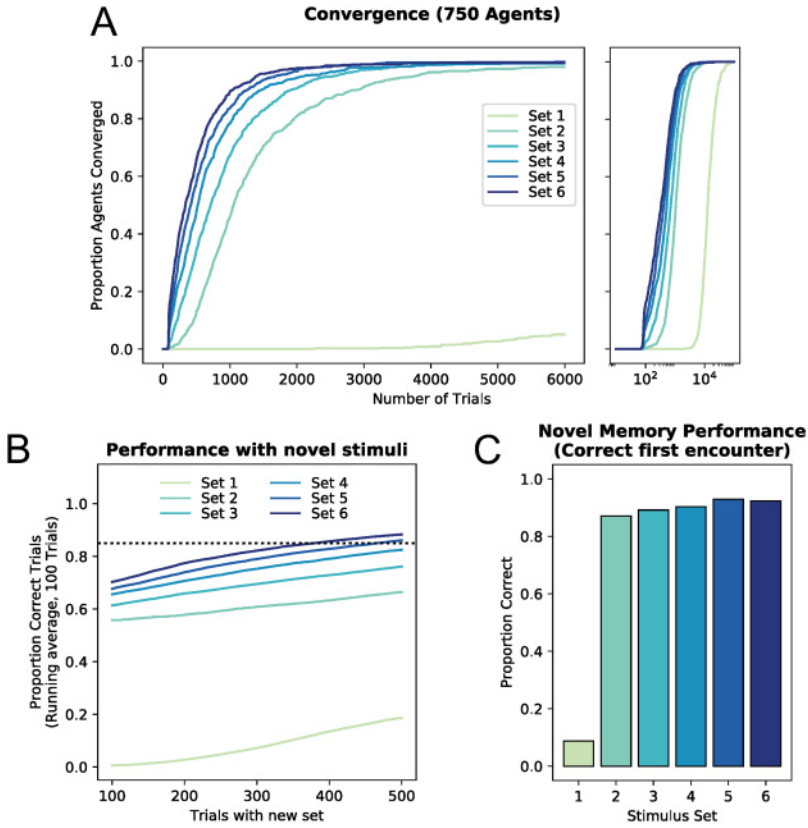

Figure 3: Performance and training on the delayed recognition task with novel stimuli. (A) Convergence rates across 750 agents (left: performance on first 6000 trials with each new stimulus set; right: convergence on all sets, with log timescale). On the first set, convergence is relatively slow, but on subsequent sets, agents learn much faster. The convergence rates keep increasing with each new set. (B) Performance with new stimuli immediately after a switch increases with each switch, indicating that the agents generalize the task to new stimulus sets. (C) Accuracy for the first encounter with a novel test stimulus, on the first trial in which the model maintained fixation until the test stimulus was presented. Note that accuracy is 87.1% on set 2, after the first stimulus switch. The agents then further generalize the rule across contexts, because accuracy is 90% or higher for all subsequent set switches.

erroneous saccades due to their random initialization. To account for such errors, we also assessed the accuracy of agents on the first trial in which they encountered a novel probe and maintained fixation until the test stimulus. We observed an average accuracy of 87.1% across agents on their first

encounter with a novel stimulus from set 2. This accuracy score also increased for subsequent sets, with an average accuracy of approximately 92.6% correct for the first encounters with stimuli from sets 5 and 6. Thus, the vast majority of errors in novel stimulus sets were caused by fixation breaks, and the model actually did learn the matching task in a manner that allows almost immediate generalization to new stimulus sets: the vast majority of errors in later sets were caused by fixation breaks.

3.2 Task 2: 12-AX. We next examined the performance of WorkMATe on the 12-AX task, a task that was used to illustrate the ability of PBWM to flexibly update WM content. The 12-AX task is a hierarchical task with two contexts: 1 and 2. In the task, letters and numbers are sequentially presented, and each requires a go or a no-go response. Whenever a 1 has been presented as the last digit, the 1 context applies. In this context, an A followed by an X should elicit a go response to the X, whereas every other stimulus requires a no-go response. When a 2 is presented, the second context applies: now only a B immediately followed by a Y should elicit a go response. Agents must separately maintain and update both the context (1 or 2), and the most recently presented stimulus, in order to make the correct go response to the imperative stimuli X or Y.

Human participants can do this hierarchical task after verbal instruction, but to acquire the rules that determine the correct response solely through trial and error learning poses a challenge. PBWM learned this task using a complex combination of reinforcement learning, supervised learning, and unsupervised learning techniques (O'Reilly et al., 2007; O'Reilly, 1996a; Aizawa & Wurtz, 1998), but Todd, Niv, and Cohen (2009) showed that agents can also learn this task using a simpler $SARSA(\lambda)$ reinforcement learning scheme. To our knowledge, no data have been published on humans or other primates learning a task of this complexity through reinforcement learning alone.

Here, we used a trial-based version of the task, where on every trial, a sequence of symbols with unpredictable length is presented, which ends with an X or a Y. During this sequence, the agent had to respond as outlined above. Given the complexity of the task, we trained the agents through curriculum learning (Bengio, Louradour, Collobert, & Weston, 2009; Zaremba & Sutskever, 2014; Graves et al., 2016), a training scheme in which trials were organized into levels, which gradually increased in difficulty. Once an agent showed sufficient performance on a level, training for the next level began. Example sequences at different difficulty levels are shown in Figure 4A. Key to curriculum learning is that trial types from previous, easier levels are also presented in order to prevent unlearning of the simpler cases. In our curriculum, 50% of the trials were always of the highest difficulty level, and the other 50% were simpler cases drawn from one of the previous levels with equal probability for all previous levels. The difficulty

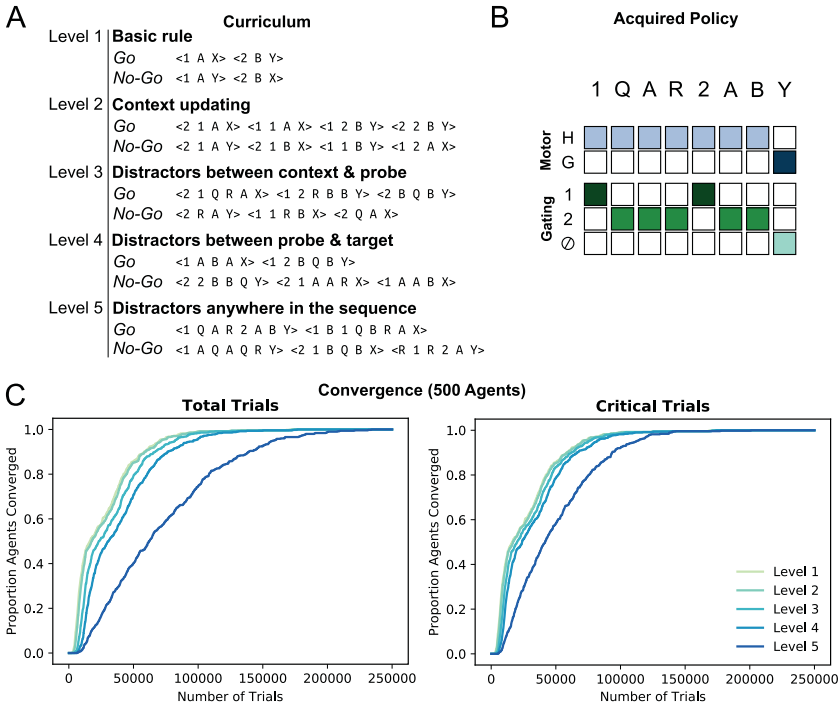

Figure 4: Training on trial-based 12AX. (A) The curriculum used to train the agent, with example trial sequences to illustrate the difficulty levels. As soon as the agent performs correctly on 85% of the trials, a higher difficulty level is introduced and presented on 50% of the trials (critical trials), with the other 50% sampled from the lower levels. (B) Policy on an example trial (see Figure 2E), acquired by an illustrative model agent converging on the highest difficulty level. The agent correctly updates memory content on each stimulus, but is only rewarded on the basis of its final motor action in response to the target symbol (X/Y). This agent stored the task context (1/2) in the memory block 1 and stored the last-seen stimulus, target or distractor, in block 2. (C) Cumulative histogram from 500 agents depicting the number of trials needed for convergence on each difficulty level. Training on higher difficulty levels does not start until lower levels have been learned. The graph on the right depicts convergence rates considering only the trials drawn from the highest difficulty.

was increased when performance on the last 100 trials was over 85% correct.

This trial-based curriculum not only facilitated training, but also had another benefit over previous approaches to train 12-AX (O'Reilly & Frank, 2006; Todd et al., 2009; Martinolli, Gerstner, & Gilra, 2017). In previous implementations, the imperative X/Y stimulus always occurred at one of a

few critical moments after the context rule, whereas here we intermixed sequences of very different lengths. Without this variation, we found that models could meet the convergence criterion on the basis of timing alone, without actually fully acquiring the task rules. In the current curriculum learning scheme, the agents truly solved the task, applying the appropriate storage policies to all difficulty levels and trial lengths.

All 500 agents converged and were able to accurately perform the task at the highest difficulty level. The policy acquired by one of these agents is depicted in Figure 4B, which illustrates an example trial at the highest difficulty. Throughout the sequence, the agent selected the hold action, while it updated each last-presented stimulus, encoding these into block 2. However, stimuli denoting the rule context (1/2) were encoded into memory slot 1 and updated only when the context changed. Once presented with the imperative stimulus (Y, in this case), this gating policy allowed the agent to use the memory of the current context (2) and the previous stimuli (B) to decide to yield a correct go response.

Convergence rates for this task are depicted in Figure 4C. Despite the complexity of this task, all agents reached criterion performance, within a median of approximately 62,000 trials (95% range 11,566, to 180,988). A large proportion of these trials were repetitions of easier levels, and the number of critical (final level) trials before convergence was lower, with a median of approximately 42,000 trials (95% 8700 to 121,208). Thus, the model was able to acquire the rules of a complex, hierarchical task that requires flexible gating of items into and out of WM, based only on relatively sparse rewards that were given only at the end of correctly performed trials.

The gating mechanisms of WorkMATe that allow it to solve the 12-AX problem are derived from the mechanisms proposed for PBWM (O'Reilly & Frank, 2006), and like the simplified PBWM model by Todd et al. (2009), WorkMATe demonstrates that a control policy for 12-AX can be acquired solely via reinforcement learning. Nevertheless, WorkMATe strongly differs from this simplified PBWM model in one critical way: whereas simplified PBWM uses a tabular architecture with a unique row for each combination of external and internal states, WorkMATe is a neural network that has to rely on distributed overlapping stimulus representations, as well as an imperfect compressed representation of stimuli in working memory. We set up a simulation to explore how this difference, together with more subtle differences between the two models, might affect learning. To this end, we compared 250 instances of WorkMATe to 250 instances of the simplified PBWM model and trained both groups of models on the first four lessons of the trial-based 12-AX task. We then assessed the number of critical trials needed for either model type to learn each consecutive level.

The results, plotted in Figure 5, illustrate that the neural network architecture initially puts WorkMATe at a disadvantage with respect to the tabular architecture. As WorkMATe has to rely on compressed, overlapping representations, it first needs to learn to dissociate the relevant stimuli

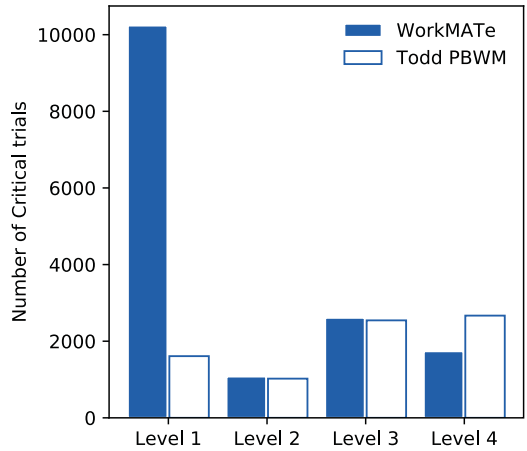

Figure 5: Comparing WorkMATe to the simplified PBWM model of Todd et al. (2009). The median number of critical trials needed to converge computed on 250 agents of each type is plotted. At the first level, the tabular model of the simplified PBWM model is clearly at an advantage. However, once WorkMATe has learned to dissociate the relevant stimuli, both models learn equally quickly.

before it can map these stimuli to a usable policy, whereas the simplified PBWM model effectively already does this upon initialization. However, after this has been learned at the first difficulty level, both models show similar convergence speeds for the higher difficulty levels. That is, once the neural network is able to dissociate the relevant stimuli, WorkMATe performs comparably to a symbolic architecture where stimuli are dissociated by definition. Note that although the tabular, one-on-one mapping between states and actions might benefit learning initially in the 12-AX setup used here, it is also precisely this architecture that prohibits models like PBWM from generalizing the acquired policies to new contexts with novel stimuli, as WorkMATe was shown to do in section 3.1. The next section shows similar generalization capabilities in a more complex, hierarchical task setting.

3.3 Task 3: ABAB Ordered Recognition. In a series of elegant studies, Miller and colleagues (Warden & Miller, 2007, 2010; Siegel et al., 2009; Rigotti et al., 2013), reported data from macaques trained in tasks in which multiple visual stimuli needed to be maintained in WM. For example, in the ordered recognition task, the monkey was trained to remember two sequentially presented visual stimuli (A and B), and to report whether the stimuli were later presented again, and in the same order. On match trials, the same objects were repeated (ABAB), and the monkey responded after a match to both objects; on the fourth stimulus in the sequence. There were mismatch trials in which the first or the second stimulus was replaced by

a third stimulus C (ABAC or ABCB), as well as mismatch trials with the same stimuli (A and B), but in reverse order (ABBA). In case of a mismatch, the monkey waited until the A and B were shown in the correct order as the fifth and sixth stimuli (e.g., ABACAB) and thus responded to the sixth stimulus. In each recording session, three novel visual stimuli were used to form the sequences, where each of these stimuli could take on the role of A, B, or C on any trial.

This ordered recognition task requires selective updating and read-out of memories in a way that shares features with the 12-AX and DR tasks from the previous sections. As in the 12-AX task, two stimuli need to be maintained and updated separately, and the task goes beyond simply memorizing two items: the order of stimuli also needs to be stored and determines the correct action sequence. As with the DR task, monkeys reached reasonable accuracies, even though novel stimuli were presented in each session, implying that they could generalize their policy to new stimulus sets.

We tested WorkMATe on this ordered recognition task. We trained 750 model agents, randomly selecting stimuli from the same set as we had used for the DR simulation described above. Half of the trials were match sequences, and the other half consisted of the three possible mismatch sequences, in equal proportion. Criterion performance was defined as an accuracy of at least 85% on the last 100 trials, with an added requirement of at least 75% accuracy on the last 100 trials in each of the four conditions. In the static training regime, we kept the three selected stimuli identical for an agent throughout a training run. In the dynamic regime, the three stimuli were replaced by three new randomly selected stimuli after 3000 trials. This meant that each of the three stimuli took on the role of A, B, or C approximately 1000 times before they were replaced by a new set.

The convergence rates for the static regime are plotted as solid lines in Figure 6A. The agents learned the full task after a median number of approximately 106,000 trials (95% of the agents between 25,880 and 856,868 trials). Under the static regime, we found that learning the overall task was primarily hindered by the condition Mismatch 1 (ABCB).

Convergence on this condition typically took much longer (median: 86,390) than on the other conditions (medians: 3128, 24,076 and 28,858 trials for Match, Swap, and Mismatch 2, respectively). The increase in complexity under the dynamic regime caused a total training time that was five to six times longer (see Figure 6A, dashed lines) than in the static regime, with convergence after a median of about 641,000 trials (95% of the models converged within 139,907 to 3,797,200 trials). Interestingly, compared to the static regime, initial convergence was comparatively quick on each of the mismatch conditions, within a median of about 13,000 trials (75% correct). The reason for this is that many agents initially learned to withhold their response until the end of the trial but did not learn to store or update the appropriate stimuli in WM. Although all mismatch conditions initially converged rather quickly, we noticed that during training, increases in Match

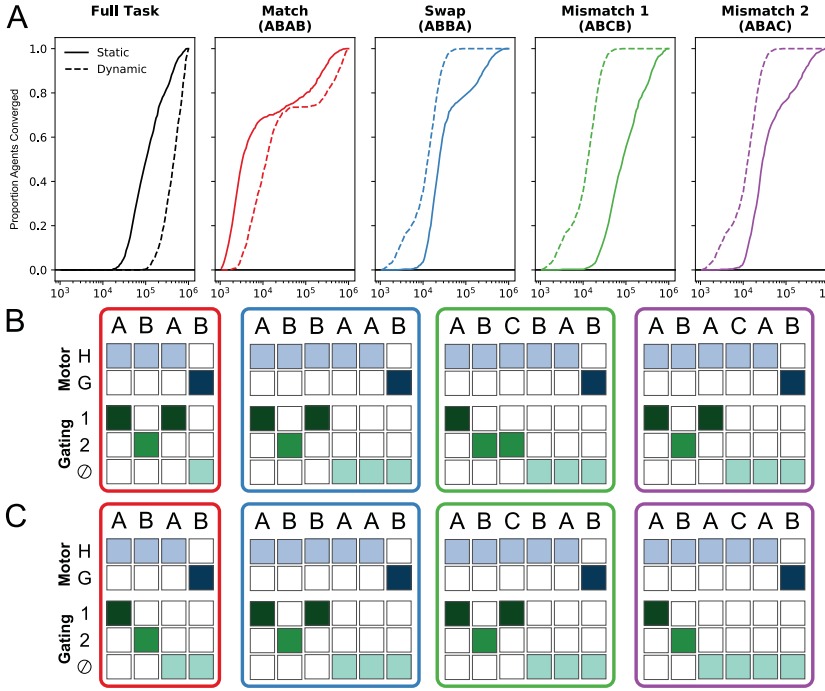

Figure 6: The ABAB ordered recognition task. (A) Convergence of 500 agents on the full task (black traces) and on different conditions separately (colored traces) under two training regimes: static (same stimuli used throughout training) or dynamic (new stimulus sets after each 3000 trials). In both regimes, the task is typically learned in approximately 10^5 trials, but convergence varies across conditions. Note the logarithmic time axis. (B) The memorized mismatch. (C) The memorized storage time policy. Both policies reflect generic, common solutions found among converged agents and are discussed in the main text. Both are plotted following the scheme of Figure 2C and 2D.

condition performance were often paired with decreases in performance on the Mismatch 1 condition.

We qualitatively investigated the policies of converged agents to explore why Mismatch 1 posed such a challenge for the model. Note that on trials from the other conditions (Match, Swap, and Mismatch 2, which together make up 83.3% of all trials), the correct response can be determined based on relatively simple inferences: The agent merely has to learn to encode the second stimulus (B) and maintain it for two time steps, and utilize its time cell input to identify the fourth and sixth stimulus presentations. Then, if the stimulus at $t = 4$ matches the stimulus that was encoded at $t = 2$, a go-response is needed; otherwise, it is to be held until $t = 6$. The Mismatch

1 condition, however, demands complex memory management. The agent must store both the initially presented A and B, detect the mismatch at $t = 3$, and somehow convey this mismatch in a manner that prevents responses to the matching stimulus (B) at $t = 4$. However, in the present architecture, WorkMATe has no way to encode this mismatch, so the agent is not capable of such metacognition.

Nevertheless, agents typically found a solution that fell into one of two classes. In both solutions, the first two stimuli (A/B) were separately encoded in the two memory blocks. The first solution, which we call the memorized mismatch strategy (see Figure 6B), essentially followed the following rule: if the stimulus at $t = 3$ does not match either stimulus in memory, and the trial must therefore be of the Mismatch 1 condition, the agent replaced the B stimulus in memory with the "new" stimulus C. As a result, stimulus B at $t = 4$ no longer matched any stimulus in memory, which led the agent to withhold a response. A second solution, the memorized storage time strategy, made use of the fact that time cell activity at the moment of encoding is incorporated in the memory representation in a manner that the network could learn to interpret. In this strategy, the key step was that if the stimulus at $t = 3$ did not match stimulus A, the mismatching stimulus was overwritten in memory by the new stimulus. At $t = 4$, the correct decision could then be made only by responding if the presented stimulus matched stimulus B in one memory store, and if the other memory store still contained temporal information from the first time step.

To conclude, these simulations demonstrate that WorkMATe can acquire complex control over WM content in order to appropriately solve complex hierarchical tasks with dynamically switching stimulus contexts—again, solely on the basis of reinforcement signals.

3.4 Task 4: Pro-/Anti-Saccade Task. To compare WorkMATe to its gateless predecessor, AuGMEnT (Rombouts, Bohte, & Roelfsema, 2015), we simulated agents learning the delayed pro-/anti-saccade task, a classic task in both human and nonhuman primate memory research, and on which AuGMEnT was also trained and evaluated. The task (see Figure 7A) requires an agent to maintain fixation at a central fixation point. The agent should encode the location of a peripheral probe and memorize it during a delay. Trials with a black fixation point are pro-saccade trials, and when the fixation point disappears, the agent makes a saccadic eye movement to the remembered location of the probe. On anti-saccade trials, the fixation point is white, and now the agent has to make an eye movement in a direction opposite to the remembered cue location, after the memory delay.

We trained 500 instances of our network and all learned the task (more than 85% correct) within 100,000 trials (see Figure 7B, solid line). The median number of trials was approximately 15,000 (95% 6,835 to 56,155 trials). This convergence rate is faster than that of monkeys, which typically learn such a task only after several months of daily training with about 1000

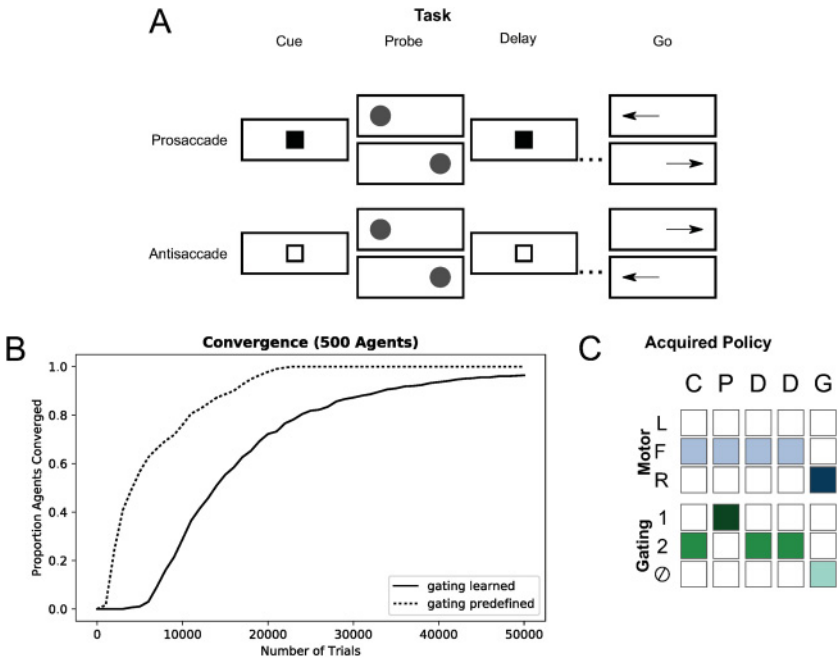

Figure 7: (A) Illustration of the four conditions in the pro-saccade/anti-saccade task. The agent has to memorize the location of the probe and make a pro- or anti-saccade after a delay, dependent on the trial type indicated by the cue (white or black fixation point). The agent thus has to integrate the information throughout the trial and make an "exclusive or" decision upon presentation of the go signal. Of note, the gating policy in this trial, depicted in panel C, is applicable in each of the four conditions in this task. (B) Convergence rates for 2×500 simulated agents of two different types. The solid line depicts convergence with WorkMATe. The dotted line depicts performance with a modified version of the model, where the gating policy is not learned but correctly predefined and fixed beforehand. (C) Policy (see Figure 2E) of an example agent after convergence, during an antisaccade trial with a "left" probe. This gating policy applies to all trial conditions.

trials per session. However, training took approximately three to four times longer than with the original AuGMEnT architecture. Several differences between AuGMEnT and WorkMATe could account for this. For example, the parameters governing Q-learning were not optimized for WorkMATe but adopted from AuGMEnT to facilitate comparison. The most critical difference between models, however, is that the gated memory store, the core of the WorkMATe model, was overly flexible for this task. The gateless AuGMEnT architecture encoded all relevant stimuli into its memory so that an

accumulation of relevant information was available at the go signal. The WorkMATe architecture first had to acquire an appropriate gating policy (see Figure 7C), to make sure that the correct decision can be made based the fixation color and probe location on the go display when no information is available anymore. Notably, the gating policy can be the same for all conditions: if cue and probe are separately available in memory, a correct decision can be made.

To examine if the added complexity of learning a gating policy could account for the difference in learning speeds between WorkMATe and AuGMEnT, we trained a new set of "gateless" agents on this task. These agents were identical to WorkMATe, except that the gating actions were, from the start, predefined to match those depicted in Figure 7C. With this setup, the complexity was comparable to that of the AuGMEnT architecture. Indeed, convergence rates for these gateless agents (median number of trials, about 5000; 95% 2076 to 20,334 trials) were very similar to those for AuGMEnT and were approximately three times faster than those with gated WorkMATe (see Figure 7B).

These simulations highlight the strengths and weaknesses of gateless and gated memory architectures. Simpler, gateless models that project all stimuli to memory suffice for tasks like pro-/anti-saccade task. These tasks do not require selective updating of memory representations, and they do not contain distractor stimuli that interfere with the memory representation. On the other hand, gating is essential for tasks in which access to WM needs to be controlled in a rule-based fashion. In both the ABAB ordered recognition task and the 12-AX task, a stimulus's access to memory is contingent on other items that are presented in the history of the trial. We envisage that both types of WM, gated and ungated, might exist in the brain, so that the advantages of both strategies can be exploited when useful.

3.5 Model Stability. Our simulations demonstrate that the WorkMATe model is able to learn accurate performance across a range of popular WM tasks. Across these simulations, we have kept the model architecture and parameters constant: that is, we used only the minimal number of memory blocks (two) and the same learning parameters in each task. In this section, we explore how sensitive WorkMATe's performance is to these choices.

First, we explored to what extent learning is affected by the number of memory blocks. For this, we used the DR task where the agent only has to memorize one stimulus. Since that task can be solved with only one memory block, it makes it suitable to study the effect of additional, effectively redundant blocks. We trained models with one to four memory blocks ($500 \times 4 = 2000$ models in total) and trained these models on three stimulus sets, switching twice to a new set after convergence.

In a task like DR, the effect of adding redundant blocks has advantages and disadvantages for the WorkMATe algorithm. On the one hand, having more blocks to encode stimuli increases the number of policies that suffice

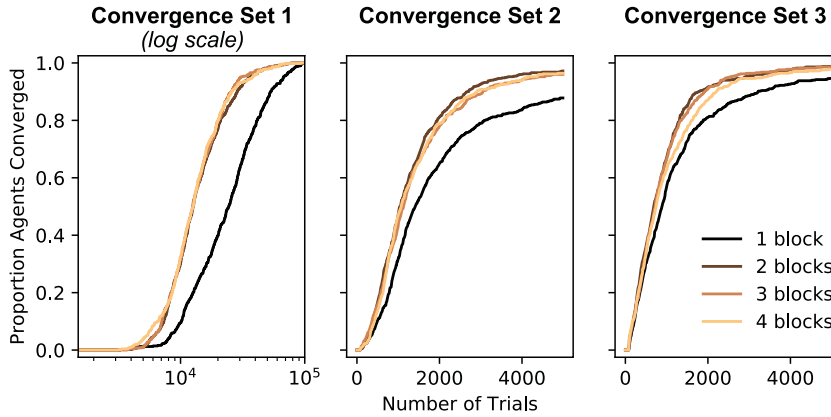

Figure 8: Convergence of 500 agents with different numbers of memory blocks, trained on the DR task with three different stimulus sets (see section 3.1). Convergence on the first set is plotted on a logarithmic x-axis (left-most graph). Performance is worse when only one memory block is used but similar across any higher number of blocks.

to solve the task. In the DR-example in Figure 2C and 2D, for example, the agent had learned to encode the probe in one memory store and to encode the fixation stimulus in the other store alongside it. Since the fixation stimulus is irrelevant for the task, ignoring it would also have sufficed. On the other hand, a larger number of blocks causes a small disadvantage during exploration, as the chance of choosing an optimal gating action decreases with the higher number of options. This interplay is reflected in the results in Figure 8, which shows that models with a redundant number of blocks all have similar performance. Similar to the results in Figure 3, performance on the first set took the longest (median number of trials, 12,501 to 12,659), but was faster on subsequent sets (1055 to 1137 and 700 to 756 trials for sets 2 and 3, respectively). With only one memory block, it took almost twice as long to converge on the initial set (median number of trials: 25,101), but once a policy was acquired that could be used on subsequent sets, performance gradually became similar to that with a redundant number of blocks (median number of trials 1418 and 892 on sets 2 and 3, respectively). These results suggest that the WorkMATe algorithm readily generalizes to larger networks with a higher number of blocks, without obvious disadvantages to its performance. Nevertheless, there are neurocognitive reasons to keep the number of memory blocks low, which we will turn to in section 4.

Next, we explored to what extent the learning parameters affected model performance. The values used for these parameters were kept constant in all simulations and were chosen to be consistent with the original AuGMEnT model. These parameters include β, which scales the magnitude of synaptic

weight updates, and the SARSA learning parameter λ, which, together with temporal discounting parameter γ, determines the decay of synaptic tags through the relation $\alpha = 1 - \lambda\gamma$. In order to explore to what extent WorkMATe's performance depends on the exact values of these parameters, we ran a grid-search exploration with different values of λ and β.

For this grid search, we used versions of the tasks defined for the simulations above. For the DR task, we used only three stimulus sets. For the ABAB ordered recognition task we only ran the "Static" learning regime (solid lines in Figure 6A). For 12-AX, we again used curriculum learning, and count only the critical trials at the highest difficulty level (see Figure 4C). The pro-/anti-saccade task ran as-is (see the solid line in Figure 7B). We assessed all combinations of $\beta = [0.05, 0.10, 0.15, \ldots, 1.0]$ and $\lambda = [0.1, 0.2, 0.3, \ldots, 0.9]$, and for each parameter combination, we ran 100 model instances. The simulations were ran on the Peregrine High Performance cluster of the University of Groningen. Per task, each model instance was allotted the same amount of wall clock time. Assuming comparable performance across all cores, this implies a similar maximum number of iterations (trials) held for these tasks. The maximum number of iterations in each task was approximately 1,870,000 in the delayed recognition task, approximately 1,700,000 critical trials in the 12-AX task, approximately 790,000 trials for ABAB ordered recognition, and approximately 500,000 in the pro-/anti-saccade task. The number of iterations reported in Figure 9 are the median number of iterations computed across all runs in which convergence was reached. In general, we found that model runs with a high β had relatively low convergence rates, an effect that was particularly pronounced for the ABAB task. To yield better insight into model stability for this task, we ran additional simulations where we varied β at a more finely-grained scale $\beta = [0.025, 0.05, 0.075, \ldots, 1.0]$.

The results are depicted in Figure 9. Across all tasks, a similar pattern was found: performance was rather robust across a range of values for λ, and more sensitive the precise value of β. With regard to β, the results suggest that learning rates that are too high are detrimental for WorkMATe. Values for the learning rate that are too high are generally harmful for convergence in neural networks, and for WorkMATe, this might have been extra detrimental due to the all-or-none gating policy in the model. As large weight changes could lead to sudden changes in the gating policy, this effectively alters the model's input state space. Large learning rates can therefore hinder convergence by rendering previously learned state-action pairings irrelevant. Although these sudden changes also occur with lower β values, they are less frequent, so that the models can adapt.

The effects of the parameter λ seem to similarly reflect adverse effects of large weight changes. Note that high weight changes are caused by high values of β, high tag values, and large prediction errors. High values of λ lead to slower weight decay and therefore result in relatively high tag values. Variations in λ have the largest effect in the 12-AX task and the

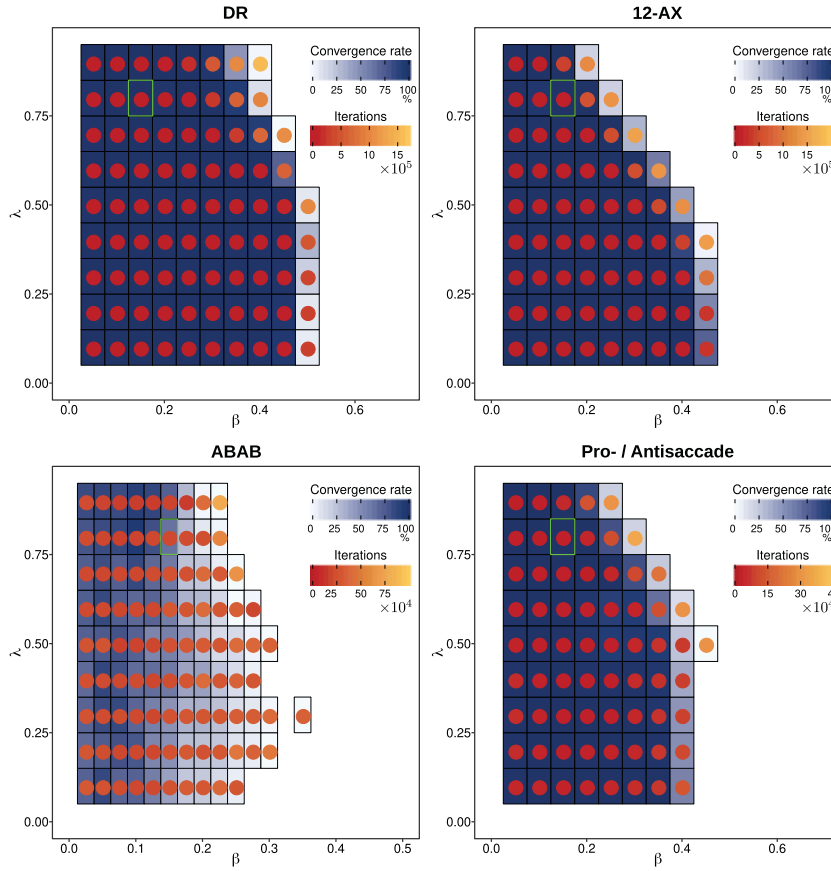

Figure 9: Model stability across the four different tasks. Each tile represents a λ, β-combination. The blue shading of the tiles indicates the convergence rate, and the red shading of the dots reflects the median number of iterations needed for convergence. Note that the color axes are different for each task, and the x-axis is different for the ABAB ordered recognition task (bottom left). The tile with the green outline indicates the single parameter combination that was used in all simulations in the preceding sections. It can be seen that model performance is largely independent of values for λ and that lower β values were generally associated with faster convergence.

pro-/anti-saccade task. A feature shared among these tasks is that the moment of reward delivery is variable, which makes it difficult for an agent to predict when exactly a reward is due, even when it behaves according to policy. As a result, high values of λ impair convergence in these tasks specifically.

Of note, the influence of β and λ on learning was similar to that observed with previous models (Rombouts et al., 2015; Todd et al., 2009). We conclude that there are large regions of the parameter space with successful and consistent performance in all four tasks. Within these regions, the performance of WorkMATe is robust and stable.

4 Discussion

We have presented WorkMATe, a neural network model that learns to flexibly control its WM content in a biologically plausible fashion by means of reinforcement. The model solves relatively basic WM tasks like delayed recognition and delayed pro/anti-saccade tasks, but also more complex tasks such as the hierarchical 12-AX task and the ABAB ordered recognition task. Furthermore, we show that the agent can learn gating policies that are largely independent of the stimulus content and apply these policies successfully to solve tasks with stimuli that were not encountered before. Thus, WorkMATe exhibits a number of crucial properties of WM: trainability, flexibility, and generalizability.

The terms *working memory* and *short-term memory* have often been used interchangeably in the cognitive sciences, even though the term *working memory* was popularized to place additional emphasis on the capability of the brain to flexibly regulate and update memory content given task demands (Baddeley, 2003). Many previous models of WM (Mongillo et al., 2008; Schneegans & Bays, 2017; Fiebig & Lansner, 2017) focus on storage of items and their retrieval. In our study, the focus was on learning to use and update memory content according to potentially complex task requirements. This approach highlights challenges that the brain is faced with beyond mere issues of capacity and fidelity: decisions to store and retrieve are cognitive operations that need to be learned in order to solve a task, and the organization of memory content should support learning these operations.

Previous models that we described in section 1 as action-oriented models have addressed this problem at a different level of abstraction and thereby have highlighted different aspects of these computational challenges. AuGMEnT models have used a basic neural network architecture to illustrate a biologically plausible implementation of reinforcement learning principles that can be applied to different tasks and different architectures. LSTM models have demonstrated the computational advantages of memory architectures with separately trained control nodes but have typically not considered biological plausibility. PBWM has shown how such gating can be implemented by the neural circuitry and activity patterns found in structures in the basal ganglia, and the subsequent simplification by Todd et al. (2009) showed that PBWMs, core functionality can be expressed in a traditional reinforcement learning setup. WorkMATe builds upon all these preceding models and offers a computationally tractable gated architecture

that efficiently, yet in a biologically plausible fashion, learns to solve a range of complex working memory tasks.

In addition to integrating views from these predecessors, WorkMATe addresses a key problem faced by action-oriented models, which is that the control operations acquired to solve a task should generalize to a new context with novel stimuli. The neural circuitry in the memory store in WorkMATe can store arbitrary representations and has a built-in capacity to compute the degree of match between the representations in memory and incoming sensory information. Using such circuitry, inspired by storage-oriented models, we found that it is unnecessary to first learn specific memory representations and that, instead, a fixed, random projection for encoding suffices. The properties of such an encoding scheme have been explored before (Barak et al., 2013; Saxe et al., 2011), indicating that this is a functionally rich approach that can be applied to a range of memory tasks. Our simulations with the pro-/anti-saccade task demonstrate that such random feedforward encoding suffices for at least some tasks where the relevant features are given as feedforward inputs to the model. It seems likely however, that it will be insufficient for other tasks, in which the memoranda require specific and nonlinear combinations of inputs. Recently, Bouchacourt and Buschman (2019) proposed a working memory storage architecture that was defined by two separate layers of neurons: a structured, sensory layer with pools for separate items, which projected to a shared unstructured layer via random recurrent connections, with balanced excitation and inhibition for each neuron as the only constraint. The resulting architecture could also store arbitrary representations and gave rise to capacity limits and forgetting dynamics that are also observed in humans. Future work might explore how WorkMATe might also benefit from a more sophisticated memory maintenance architecture, be it a multilayer subsystem or one with recurrent connections to the sensory inputs, while still allowing for the generic, built-in matching computations.

Because the WorkMATe architecture largely separates memory content from gating and updating operations, the models acquire policies that implement a type of symbolic memory control: in many of our simulations, the acquired gating policy can be interpreted as a set of production rules that are applicable to all stimuli. Previous studies have noted that the gap between traditional artificial neural network architectures and symbolic systems is one of the great challenges to be overcome by artificial intelligence (Reggia, Monner, & Sylvester, 2014). Previous neural network models that attempt to implement a similar approach to memory control have relied on predetermined, hand-coded sequences of memory operations, hard-coded into the model (Sylvester, Reggia, Weems, & Bunting, 2013; Sylvester & Reggia, 2016; Eliasmith, 2005; but see Graves et al., 2016). Here we show, for the first time, that such control over WM can be acquired in a neural system by means of a biologically plausible reinforcement learning rule.

WorkMATe makes several simplifying assumptions that touch on contended topics in WM research and require further discussion. First, all our simulations made use of two, independently maintained memory blocks to store content, which proved sufficient for these tasks. There is an ongoing debate regarding the storage capacity limits of WM and to what extent these speak to the functional organization of items in memory. Two opposing views are slot-based models (Zhang & Luck, 2008), which state that storage is limited by a discrete number of slots in memory, and resource-based models, which propose that there is no limit on the number of items that can be stored, but the total fidelity is limited by a certain amount of resources (Van den Berg & Ma, 2018; Van den Berg et al., 2014; Bays & Husain, 2008). Although WorkMATes memory circuit at a glance most closely aligns with slot-based architectures, it should not be taken as direct evidence in support of such models. While that debate focuses on memory capacity and fidelity, our memory blocks served a different functional purpose: separate, independent memory blocks directly allow for independent matching, gating, and updating of memoranda. It is conceivable that similar control functions could be implemented within a resource-based architecture, though this would require additional assumptions on how memory items can be independently addressed and updated (see Stewart et al., 2011, for one possible approach). Conversely, the present work does not touch on the capacity and fidelity of working memory. We have simulated tasks that require at most two items in memory, which is well within the capacity limits of human working memory (Vogel & Machizawa, 2004; Cowan, 2010; Oberauer & Hein, 2012). Even though we have demonstrated that WorkMATe's control functions can in principle be scaled up to control more blocks, it seems that a more complete model of working memory should also consider how memoranda deteriorate under interference and decay.

A second simplifying assumption that we have made here is that matches between sensory and memory representations are computed automatically and in parallel. Whether multiple objects in WM can be matched simultaneously by a single percept is a topic of debate in cognitive psychology (Sternberg, 1966; Banks & Fariello, 1974; Olivers, Peters, Houtkamp, & Roelfsema, 2011; Wolfe, 2012; Konecky, Smith, & Olson, 2017). The tasks that we have chosen to focus on here unfold at relatively slow speeds, which would allow for serial comparisons. Previous research has shown that at high speeds, matching multiple memory targets comes at a cost (Houtkamp & Roelfsema, 2009). A serial comparison circuit might introduce additional control operations to determine which representation should be prioritized for matching. Here, we refrained from simulating such additional operations. Related to this, some models such as LSTM can also gate WM *output*, in addition to the input. These might come into play in task-switching setups, where multiple goals need to be maintained but only one should drive behavior (Monsell, 2003; Alport, Styles, & Hsieh, 1994; Chatham, Frank, &

Badre, 2014; Myers et al., 2015; Myers, Stokes, & Nobre, 2017; Rushworth, Passingham, & Nobre, 2002), and in sequential visual search tasks where multiple items may be held in WM but only one drives attentional selection (Houtkamp & Roelfsema, 2006; Soto, Humphreys, & Heinke, 2006; Olivers et al., 2011; Ort, Fahrenfort, & Olivers, 2017; de Vries, Van Driel, & Olivers, 2017; de Vries, Van Driel, Karacaoglu, & Olivers, 2018; de Vries, Van Driel, & Olivers, 2019). Recordings in macaque PFC suggest that sequential search tasks, which require such prioritization, of one memory item over another, are characterized by elevated cortical representation of the prioritized stimulus in preparation of search (Warden & Miller, 2007, 2010; Siegel et al., 2009). Future extensions of WorkMATe might investigate tasks that could benefit from such output gating operations and whether they can be learned through plasticity rules related to those studied here.

Interestingly, not every task benefited from a gated memory. Notably, training on the pro-/anti-saccade task actually took three to four times longer with the gated model than with a model without these gates. This is important, as it shows that for certain tasks, it may indeed be beneficial to merely accumulate relevant information into memory and learn a policy that relies on these accumulated representations. These types of memory tasks are actually more akin to perceptual decision-making tasks, which require an agent to aggregate information until a threshold is reached that triggers a decision (Shadlen & Newsome, 2001; Gold & Shadlen, 2007), rather than to flexibly store, update, and maintain memory representations. This qualitative dissociation between different types of tasks might warrant a model that comprises separate routes to a decision: one relying on the automatic integration of relevant information and one describing a more controlled process that stores and updates information as variables to be used in a task (Collins & Frank, 2018; see Masse, Yang, Song, Wang, & Freedman, 2019, for a similar conclusion derived from modeling work with a very different approach). We may therefore use models like WorkMATe to predict more precisely which tasks will rely on flexible, controlled memory and which tasks an organism should be able to solve without the necessity for flexible control structures.

5 Conclusion

We have presented a neural network model of primate WM that is able to learn the correct set of internal and external actions based on a biologically plausible neuronal plasticity rule. The network can be trained to execute complex hierarchical memory tasks and generalize these policies across stimulus sets that were never seen before. We believe this to be an important step toward unraveling the enigmatic processes that make WM work: that is, be used as an active, flexible system with capabilities beyond the mere short-term storage of information.

References

Aizawa, H., & Wurtz, R. H. (1998). Reversible inactivation of monkey superior colliculus. I: Curvature of saccadic trajectory. *Journal of Neurophysiology*, *79*(4), 2082–2096.

Allport, A., Styles, E. A., & Hsieh, S. (1994). Shifting intentional set: Exploring the dynamic control of tasks. In Carlo Umiltà & Morris Moscovitch (Eds.), *Attention and performance XV*. Cambridge, MA: MIT Press.

Amari, S.-I. (1977). Dynamics of pattern formation in lateral-inhibition type neural fields. *Biological Cybernetics*, *27*(2), 77–87.

Baddeley, A. (2003). Working memory: Looking back and looking forward. *Nature Reviews Neuroscience*, *4*(10), 829.

Bakker, B. (2002). Reinforcement learning with long short-term memory. In *Proceeding of the 14th International Conference on Neural Information Processing Systems: Natural and Synthetic* (pp. 1475–1482). Cambridge, MA: MIT Press.

Bakker, B. (2007). Reinforcement learning by backpropagation through an LSTM model/critic. In *Proceedings of the IEEE International Symposium on Approximate Dynamic Programming and Reinforcement Learning* (pp. 127–134). Piscataway, NJ: IEEE.

Banks, W. P., & Fariello, G. R. (1974). Memory load and latency in recognition of pictures. *Memory and Cognition*, *2*(1), 144–148.

Barak, O., Sussillo, D., Romo, R., Tsodyks, M., & Abbott, L. F. (2013). From fixed points to chaos: Three models of delayed discrimination. *Progress in Neurobiology*, *103*, 214–222.

Barak, O., & Tsodyks, M. (2007). Persistent activity in neural networks with dynamic synapses. *PLOS Computational Biology*, *3*(2), e35.

Bays, P. M., & Husain, M. (2008). Dynamic shifts of limited working memory resources in human vision. *Science*, *321*(5890), 851–854.

Bengio, Y., Louradour, J., Collobert, R., & Weston, J. (2009). Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning* (pp. 41–48). New York: ACM.

Bouchacourt, F., & Buschman, T. J. (2019). A flexible model of working memory. *Neuron*, *103*(1), 147–160.

Brady, T. F., & Alvarez, G. A. (2011). Hierarchical encoding in visual working memory: Ensemble statistics bias memory for individual items. *Psychological Science*, *22*(3), 384–392.

Brown, M. R. G., Vilis, T., & Everling, S. (2007). Frontoparietal activation with preparation for antisaccades. *Journal of Neurophysiology*, *98*(3), 1751–1762.

Brunel, N., & Wang, X.-J. (2001). Effects of neuromodulation in a cortical network model of object working memory dominated by recurrent inhibition. *Journal of Computational Neuroscience*, *11*(1), 63–85.

Chatham, C. H., Frank, M. J., & Badre, D. (2014). Corticostriatal output gating during selection from working memory. *Neuron*, *81*(4), 930–942.

Cho, K., van Merrienboer, B., Bahdanau, D., & Bengio, Y. (2014). *On the properties of neural machine translation: Encoder-decoder approaches*. arXiv:1409.1259.

Choo, F.-X., & Eliasmith, C. (2010). A spiking neuron model of serial-order recall. In S. Ohlsson & R. Cattrambone (Eds.), *Proceedings of the 32nd Annual Conference of the Cognitive Science Society* (p. 218893). Cognitive Science Society.

Collins, A. G. E., & Frank, M. J. (2018). Within- and across-trial dynamics of human EEG reveal cooperative interplay between reinforcement learning and working memory. In *Proceedings of the National Academy of Sciences*, *115*(10), 2502–2507.

Costa, R., Assael, I. A., Shillingford, B., de Freitas, N., & Vogels, T. (2017). Cortical microcircuits as gated-recurrent neural networks. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, & R. Garnett (Eds.), *Advances in neural information processing systems, 30* (pp. 272–283). Red Hook, NY: Curran.

Cowan, N. (2010). The magical mystery four: How is working memory capacity limited, and why? *Current Directions in Psychological Science*, *19*(1), 51–57.

de Vries, I. E., Van Driel, J., Karacaoglu, M., & Olivers, C. N. L. (2018). Priority switches in visual working memory are supported by frontal delta and posterior alpha interactions. *Cerebral Cortex*, *28*, 4090–4104.

de Vries, I. E. J., Van Driel, J., & Olivers, C. N. L. (2017). Posterior αEEG dynamics dissociate current from future goals in working memory-guided visual search. *Journal of Neuroscience*, *37*(6), 1591–1603.

de Vries, I. E. J., Van Driel, J., & Olivers, C. N. L. (2019). Decoding the status of working memory representations in preparation of visual selection. *NeuroImage*, *191*, 549–559.

Downing, P., & Dodds, C. (2004). Competition in visual working memory for control of search. *Visual Cognition*, *11*(6), 689–703.

Eliasmith, C. (2005). A unified approach to building and controlling spiking attractor networks. *Neural Computation*, *17*(6), 1276–1314.

Engel, T. A., & Wang, X.-J. (2011). Same or different? A neural circuit mechanism of similarity-based pattern match decision making. *Journal of Neuroscience*, *31*(19), 6982–6996.

Everling, S., & Fischer, B. (1998). The antisaccade: A review of basic research and clinical studies. *Neuropsychologia*, *36*(9), 885–899.

Fiebig, F., & Lansner, A. (2017). A spiking working memory model based on Hebbian short-term potentiation. *Journal of Neuroscience*, *37*(1), 83–96.

Freedman, D. J., Riesenhuber, M., Poggio, T., & Miller, E. K. (2003). A comparison of primate prefrontal and inferior temporal cortices during visual categorization. *Journal of Neuroscience*, *23*(12), 5235–5246.

Gers, F. A., Schmidhuber, J., & Cummins, F. (1999). Learning to forget: Continual prediction with LSTM. In *Proceedings of the Ninth International Conference on Artificial Neural Networks* (pp. 850–855).

Gers, F. A., Schraudolph, N. N., & Schmidhuber, J. (2002). Learning precise timing with LSTM recurrent networks. *Journal of Machine Learning Research*, *3*, 115–143.

Gold, J. I., & Shadlen, M. N. (2007). The neural basis of decision making. *Annual Review of Neuroscience*, *30*, 535–574.

Gottlieb, J., & Goldberg, M. E. (1999). Activity of neurons in the lateral intraparietal area of the monkey during an antisaccade task. *Nature Neuroscience*, *2*(10), 906.

Graves, A., & Schmidhuber, J. (2005). Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Networks*, *18*(5–6), 602–610.

Graves, A., Wayne, G., Reynolds, M., Harley, T., Danihelka, I., Grabska-Barwińska, A., . . . Hassabis, D. (2016). Hybrid computing using a neural network with dynamic external memory. *Nature*, *538*(7626), 471–476.

Hallett, P. E. (1978). Primary and secondary saccades to goals defined by instructions. *Vision Research*, *18*(10), 1279–1296.

Hamker, F. H. (2005). The reentry hypothesis: The putative interaction of the frontal eye field, ventrolateral prefrontal cortex, and areas V4, IT for attention and eye movement. *Cerebral Cortex*, *15*(4), 431–447.

Hazy, T. E., Frank, M. J., & O'Reilly, R. C. (2006). Banishing the homunculus: Making working memory work. *Neuroscience*, *139*(1), 105–118.

Hazy, T. E., Frank, M. J., & O'Reilly, R. C. (2007). Towards an executive without a homunculus: Computational models of the prefrontal cortex/basal ganglia system. *Philosophical Transactions of the Royal Society B: Biological Sciences*, *362*(1485), 1601–1613.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, *9*(8), 1735–1780.

Houtkamp, R., & Roelfsema, P. R. (2006). The effect of items in working memory on the deployment of attention and the eyes during visual search. *Journal of Experimental Psychology: Human Perception and Performance*, *32*(2), 423–442.

Houtkamp, R., & Roelfsema, P. R. (2009). Matching of visual input to only one item at any one time. *Psychological Research PRPF*, *73*(3), 317–326.

Howard, M. W., & Eichenbaum, H. (2013). The hippocampus, time, and memory across scales. *Journal of Experimental Psychology. General*, *142*(4), 1211–1230.

Howard, M. W., & Kahana, M. J. (2002). A distributed representation of temporal context. *Journal of Mathematical Psychology*, *46*(3), 269–299.

Jensen, O., & Lisman, J. E. (2005). Hippocampal sequence-encoding driven by a cortical multi-item working memory buffer. *Trends in Neurosciences*, *28*(2), 67–72.

Konecky, R. O., Smith, M. A., & Olson, C. R. (2017). Monkey prefrontal neurons during Sternberg task performance: Full contents of working memory or most recent item? *Journal of Neurophysiology*, *117*(6), 2269–2281.

Lillicrap, T. P., Cownden, D., Tweed, D. B., & Akerman, C. J. (2016). Random synaptic feedback weights support error backpropagation for deep learning. *Nature Communications*, *7*.

Lohnas, L. J., Polyn, S. M., & Kahana, M. J. (2015). Expanding the scope of memory search: Modeling intralist and interlist effects in free recall. *Psychological Review*, *122*(2), 337–363.

Ludueña, G. A., & Gros, C. (2013). A self-organized neural comparator. *Neural Computation*, *25*(4), 1006–1028.

Ma, W. J., Husain, M., & Bays, P. M. (2014). Changing concepts of working memory. *Nature Neuroscience*, *17*(3), 347–356.

Mao, T., Kusefoglu, D., Hooks, B. M., Huber, D., Petreanu, L., & Svoboda, K. (2011). Long-range neuronal circuits underlying the interaction between sensory and motor cortex. *Neuron*, *72*(1), 111–123.

Marblestone, A. H., Wayne, G., & Kording, K. P. (2016). Toward an integration of deep learning and neuroscience. *Frontiers in Computational Neuroscience*, 10.

Martinolli, M., Gerstner, W., & Gilra, A. (2017). *Multi-timescale memory dynamics in a reinforcement learning network with attention-gated memory.* arXiv:1712/10062 [cs, q-bio, stat].

Masse, N. Y., Yang, G. R., Song, H. F., Wang, X.-J., & Freedman, D. J. (2019). Circuit mechanisms for the maintenance and manipulation of information in working memory. *Nature Neuroscience*, *22*(7), 1159–1167.

Mello, G. B. M., Soares, S., & Paton, J. J. (2015). A scalable population code for time in the striatum. *Current Biology*, *25*(9), 1113–1122.

Meyer, T., & Rust, N. C. (2018). Single-exposure visual memory judgments are reflected in inferotemporal cortex. *eLife*, *7*.

Miller, E. K., Erickson, C. A., & Desimone, R. (1996). Neural mechanisms of visual working memory in prefrontal cortex of the macaque. *Journal of Neuroscience*, *16*, 5154–5167.

Mongillo, G., Barak, O., & Tsodyks, M. (2008). Synaptic theory of working memory. *Science*, *319*(5869), 1543–1546.

Monner, D., & Reggia, J. A. (2012). A generalized LSTM-like training algorithm for second-order recurrent neural networks. *Neural Networks*, *25*(1), 70–83.

Monsell, S. (2003). Task switching. *Trends in Cognitive Sciences*, *7*(3), 134–140.

Munoz, D. P., & Everling, S. (2004). Look away: The anti-saccade task and the voluntary control of eye movement. *Nature Reviews Neuroscience*, *5*(3), 218–228.

Myers, N. E., Rohenkohl, G., Wyart, V., Woolrich, M. W., Nobre, A. C., & Stokes, M. G. (2015). Testing sensory evidence against mnemonic templates. *eLife*, *4*, e09000.

Myers, N. E., Stokes, M. G., & Nobre, A. C. (2017). Prioritizing information during working memory: Beyond sustained internal attention. *Trends in Cognitive Sciences*, *21*(6), 449–461.

Naya, Y., & Suzuki, W. A. (2011). Integrating what and when across the primate medial temporal lobe. *Science*, *333*(6043), 773–776.

Norman, K. A., & O'Reilly, R. C. (2003). Modeling hippocampal and neocortical contributions to recognition memory: A complementary-learning-systems approach. *Psychological Review*, *110*(4), 611.

Oberauer, K., & Hein, L. (2012). Attention to information in working memory. *Current Directions in Psychological Science*, *21*(3), 164–169.

Oberauer, K., & Lin, H.-Y. (2017). An interference model of visual working memory. *Psychological Review*, *124*(1), 21–59.

Olivers, C. N. L., Peters, J., Houtkamp, R., & Roelfsema, P. R. (2011). Different states in visual working memory: When it guides attention and when it does not. *Trends in Cognitive Sciences*, *15*(7), 327–334.

O'Reilly, R. C. (1996a). Biologically plausible error-driven learning using local activation differences: The generalized recirculation algorithm. *Neural Computation*, *8*(5), 895–938.

O'Reilly, R. C. (1996b). *The Leabra model of neural interactions and learning in the neocortex*. PhD diss., Carnegie Mellon University, Pittsburgh, PA.

O'Reilly, R. C., & Frank, M. J. (2006). Making working memory work: A computational model of learning in the prefrontal cortex and basal ganglia. *Neural Computation*, *18*(2), 283–328.

O'Reilly, R. C., Frank, M. J., Hazy, T. E., & Watz, B. (2007). PVLV: The primary value and learned value Pavlovian learning algorithm. *Behavioral Neuroscience*, *121*(1), 31.

Orhan, A. E., Sims, C. R., Jacobs, R. A., & Knill, D. C. (2014). The adaptive nature of visual working memory. *Current Directions in Psychological Science*, *23*(3), 164–170.

Ort, E., Fahrenfort, J. J., & Olivers, C. N. L. (2017). Lack of free choice reveals the cost of having to search for more than one object. *Psychological Science*, *28*(8), 1137–1147.

Paton, J. J., & Buonomano, D. V. (2018). The neural basis of timing: Distributed mechanisms for diverse functions. *Neuron*, *98*(4), 687–705.

Raaijmakers, J. G., & Shiffrin, R. M. (1981). Search of associative memory. *Psychological Review*, *88*(2), 93–134.

Raffone, A., & Wolters, G. (2001). A cortical mechanism for binding in visual working memory. *Journal of Cognitive Neuroscience*, *13*(6), 766–785.

Rao, R. P. N., Zelinsky, G. J., Hayhoe, M. M., & Ballard, D. H. (2002). Eye movements in iconic visual search. *Vision Research*, *42*(11), 1447–1463.

Rawley, J. B., & Constantinidis, C. (2010). Effects of task and coordinate frame of attention in area 7a of the primate posterior parietal cortex. *Journal of Vision*, *10*(1), 12.

Reggia, J. A., Monner, D., & Sylvester, J. (2014). The computational explanatory gap. *Journal of Consciousness Studies*, *21*(9–10), 153–178.

Richards, B. A., & Lillicrap, T. P. (2019). Dendritic solutions to the credit assignment problem. *Current Opinion in Neurobiology*, *54*, 28–36.

Rigotti, M., Barak, O., Warden, M. R., Wang, X.-J., Daw, N. D., Miller, E. K., & Fusi, S. (2013). The importance of mixed selectivity in complex cognitive tasks. *Nature*, *497*(7451), 585–590.

Roelfsema, P. R., & Holtmaat, A. (2018). Control of synaptic plasticity in deep cortical networks. *Nature Reviews Neuroscience*, *19*(3), 166–180.

Roelfsema, P. R., & Ooyen, A. v. (2005). Attention-gated reinforcement learning of internal representations for classification. *Neural Computation*, *17*(10), 2176–2214.

Roelfsema, P. R., van Ooyen, A., & Watanabe, T. (2010). Perceptual learning rules based on reinforcers and attention. *Trends in Cognitive Sciences*, *14*(2), 64–71.

Rombouts, J. O., Bohte, S. M., Martinez-Trujillo, J., & Roelfsema, P. R. (2015). A learning rule that explains how rewards teach attention. *Visual Cognition*, *23*(1–2), 179–205.

Rombouts, J. O., Bohte, S. M., & Roelfsema, P. R. (2015). How attention can create synaptic tags for the learning of working memories in sequential tasks. *PLOS Comput. Biol.*, *11*(3), e1004060.

Rombouts, J. O., Roelfsema, P., & Bohte, S. M. (2012). Neurally plausible reinforcement learning of working memory tasks. In F. Pereira, C. J. C. Burges, L. Bottou, & K. Q. Weinberger (Eds.), *Advances in neural information processing systems, 25* (pp. 1880–1888). Red Hook, NY: Curran.

Rushworth, M. F. S., Passingham, R. E., & Nobre, A. C. (2002). Components of switching intentional set. *Journal of Cognitive Neuroscience*, *14*(8), 1139–1150.

Saxe, A., Koh, P. W., Chen, Z., Bhand, M., Suresh, B., & Ng, A. Y. (2011). On random weights and unsupervised feature learning. In *Proceedings of the 28th International Conference on Machine Learning* (pp. 1089–1096). Madison, WI: Omnipress.

Scellier, B., & Bengio, Y. (2018). Equivalence of equilibrium propagation and recurrent backpropagation. *Neural Computation*, *31*(2), 312–329.
Schneegans, S., & Bays, P. M. (2017). Neural architecture for feature binding in visual working memory. *Journal of Neuroscience*, *37*(14), 3913–3925.
Shadlen, M. N., & Newsome, W. T. (2001). Neural basis of a perceptual decision in the parietal cortex (area lip) of the rhesus monkey. *Journal of Neurophysiology*, *86*(4), 1916–1936.
Siegel, M., Warden, M. R., & Miller, E. K. (2009). Phase-dependent neuronal coding of objects in short-term memory. In *Proceedings of the National Academy of Sciences*, *106*(50), 21,341–21,346.
Soto, D., Humphreys, G. W., & Heinke, D. (2006). Working memory can guide popout search. *Vision Research*, *46*(6), 1010–1018.
Sternberg, S. (1966). High-speed scanning in human memory. *Science*, *153*(3736), 652–654.
Stewart, T. C., Tang, Y., & Eliasmith, C. (2011). A biologically realistic cleanup memory: Autoassociation in spiking neurons. *Cognitive Systems Research*, *12*(2), 84–92.
Sugase-Miyamoto, Y., Liu, Z., Wiener, M. C., Optican, L. M., & Richmond, B. J. (2008). Short-term memory trace in rapidly adapting synapses of inferior temporal cortex. *PLOS Computational Biology*, *4*(5), e1000073.
Sylvester, J., & Reggia, J. (2016). Engineering neural systems for high-level problem solving. *Neural Networks*, *79*, 37–52.
Sylvester, J. C., Reggia, J. A., Weems, S. A., & Bunting, M. F. (2013). Controlling working memory with learned instructions. *Neural Networks*, *41*, 23–38.
Todd, M. T., Niv, Y., & Cohen, J. D. (2009). Learning to use working memory in partially observable environments through dopaminergic reinforcement. In D. Koller, C. Schuurmans, Y. Bengio, & L. Bottou (Eds.), *Advances in neural information processing systems, 21* (pp. 1689–1696). Red Hook, NY: Curran.
Tsao, A., Sugar, J., Lu, L., Wang, C., Knierim, J. J., Moser, M.-B., & Moser, E. I. (2018). Integrating time from experience in the lateral entorhinal cortex. *Nature*, *561*(7721), 57–62.
Van den Berg, R., Awh, E., & Ma, W. J. (2014). Factorial comparison of working memory models. *Psychological Review*, *121*(1), 124.
Van den Berg, R., & Ma, W. J. (2018). A resource-rational theory of set size effects in human visual working memory. *eLife*, *7*, e34963.
van Ooyen, A., & Roelfsema, P. R. (2003). A biologically plausible implementation of error-backpropagation for classification tasks. In *Supplementary Proceedings of the International Conference on Artificial Neural Networks* (pp. 442–444). Berlin: Springer.
Vogel, E. K., & Machizawa, M. G. (2004). Neural activity predicts individual differences in visual working memory capacity. *Nature*, *428*(6984), 748–751.
Warden, M. R., & Miller, E. K. (2007). The representation of multiple objects in prefrontal neuronal delay activity. *Cerebral Cortex*, *17*(suppl. 1), i41–i50.
Warden, M. R., & Miller, E. K. (2010). Task-dependent changes in short-term memory in the prefrontal cortex. *Journal of Neuroscience*, *30*(47), 15801–15810.
Whittington, J. C. R., & Bogacz, R. (2017). An approximation of the error backpropagation algorithm in a predictive coding network with local Hebbian synaptic plasticity. *Neural Computation*, *29*(5), 1229–1262.

Wolfe, J. M. (2012). Saved by a log: How do humans perform hybrid visual and memory search? *Psychological Science*, *23*(7), 698–703.

Zaremba, W., & Sutskever, I. (2014). *Learning to execute*. arXiv:1410.4615.

Zelinsky, G. J. (2008). A theory of eye movements during target acquisition. *Psychological Review*, *115*(4), 787–835.

Zelinsky, G. J. (2012). TAM: Explaining off-object fixations and central fixation tendencies as effects of population averaging during search. *Visual Cognition*, *20*(4–5), 515–545.

Zhang, W., & Luck, S. J. (2008). Discrete fixed-resolution representations in visual working memory. *Nature*, *453*(7192), 233–235.

Received February 3, 2020; accepted August 5, 2020.