Indirection and symbol-like processing in the prefrontal cortex and basal ganglia

Trenton Kriete[a,1], David C. Noelle[b], Jonathan D. Cohen[c], and Randall C. O'Reilly[a,1]

[a]Department of Psychology and Neuroscience, University of Colorado Boulder, Boulder, CO 80309; [b]Cognitive and Information Sciences, University of California, Merced, CA 95340; and [c]Princeton Neuroscience Institute and Department of Psychology, Princeton University, Princeton, NJ 08544

Edited by John R. Anderson, Carnegie Mellon University, Pittsburgh, PA, and approved August 16, 2013 (received for review March 7, 2013)

The ability to flexibly, rapidly, and accurately perform novel tasks is a hallmark of human behavior. In our everyday lives we are often faced with arbitrary instructions that we must understand and follow, and we are able to do so with remarkable ease. It has frequently been argued that this ability relies on symbol processing, which depends critically on the ability to represent variables and bind them to arbitrary values. Whereas symbol processing is a fundamental feature of all computer systems, it remains a mystery whether and how this ability is carried out by the brain. Here, we provide an example of how the structure and functioning of the prefrontal cortex/basal ganglia working memory system can support variable binding, through a form of indirection (akin to a pointer in computer science). We show how indirection enables the system to flexibly generalize its behavior substantially beyond its direct experience (i.e., systematicity). We argue that this provides a biologically plausible mechanism that approximates a key component of symbol processing, exhibiting both the flexibility, but also some of the limitations, that are associated with this ability in humans.

generativity | generalization | computational model

One of the most impressive aspects of human cognition is also one of its most enduring mysteries: how it can respond in appropriate ways to novel circumstances. In our everyday lives, we are constantly confronted with the need to make sense of and respond appropriately to new situations. Almost always, the individual constituents of these situations (e.g., the people, places, and/or actions involved) are things with which we have had prior experience, and it is the particular combination that is new. A person may appear in a new context or carry out an action we have never before witnessed them perform, or a word may be used in a novel way within a sentence. Nevertheless, we are able to makes sense of and respond appropriately to such circumstances, drawn from a nearly infinite array of possible combinations, despite having had experience with only a limited number of them. It has frequently been argued that this flexibility, or systematicity, relies on symbol processing, that is, the ability to represent information in the form of abstract variables that can be bound to arbitrary values, as is possible in a symbol system. For example, in trying to understand a sentence, if the constituent parts can be represented as variables, then any possible word can be assigned, or "bound," to each (e.g., in the sentence "I want to desk you," "desk" can be understood as the verb). Such variable binding provides tremendous flexibility and is fundamental to the power of computer systems. However, whether and how this ability is implemented in the brain remains one of the great mysteries of neuroscience. Historically, this ability has been used as a key argument by those advocating for symbolic cognitive models, over "associationist" neural network models (1, 2). In response, some have argued that human symbol processing ability is limited at best and that many behaviors that might be construed as evidence of such an ability can be explained by general-purpose learning algorithms that can infer statistical regularities of the environment (3–5).

Here, we propose a set of neural mechanisms, involving the prefrontal cortex (PFC) and basal ganglia (BG), that support a form of variable binding through the use of indirection (corresponding to the use of a pointer, in computer science terms). We demonstrate that these mechanisms can exhibit the kind of systematicity in processing novel combinations of stimuli of which people are capable, typically attributed to a symbol processing system. However, it does so using standard neural network mechanisms for both learning and processing. As a consequence, its mechanisms for indirection and variable binding are limited. It can only assign pointers to memory locations with which it has had some previous experience, and those locations can only represent information that has been learned to be represented. Also, neural pointers cannot be nested at arbitrary levels of complexity or depth. In this respect, these neural representations fall short of qualifying as symbols in the most general sense. Accordingly, the processing capabilities of this system fall short of the more powerful capabilities of fully general symbol processing systems found in most computers. However, human systematicity has its limits (4, 5). These limits may be explained by the structure of the PFC/BG system, learning based on standard neural network mechanisms, and the distributed representation of information. In other words, although human behavior exhibits some of the characteristics of a classic symbol processing system, the mechanisms upon which this relies may not implement symbols in the usual sense. By understanding the limitations of the mechanisms our behavior is built upon we may shed a light on the limitations of human behavior itself.

In the following, we describe a model in which neurons in one part of the PFC (area A) encode and maintain a pattern of neural activity that represents the location (or address) of information maintained in another part of the PFC (area B). Furthermore, representations in area A can regulate the use of information in area B by way of the BG: Representations in area A project to, and are decoded by, a region of the BG associated with area B, which in turn can regulate efferent projections from area B to more posterior neocortical areas responsible for controlling behavior. Thus, area A of the PFC encodes a pointer to area B and, by way of the BG, can regulate the influence of information stored in area B on behavior. With reasonable assumptions about the connectivity between the PFC and BG, area A can point to a wide range of areas (i.e., not just B), providing considerable flexibility in processing.

This use of indirection to implement variable binding extends a more traditional view of neural representation, in which a given population of neurons is assumed to encode a particular type of information, and different patterns of activity over that population correspond to different possible contents of each type. For example, in the case of sentence processing, there might be separate populations for encoding the various constituents, or

Author contributions: T.K., D.C.N., J.D.C., and R.C.O. designed research; T.K. and D.C.N. performed research; R.C.O. contributed new reagents/analytic tools; T.K. analyzed data; and T.K., D.C.N., J.D.C., and R.C.O. wrote the paper.

The authors declare no conflict of interest.

This article is a PNAS Direct Submission.

[1]To whom correspondence may be addressed. E-mail: Trenton.Kriete@Colorado.edu or randy.oreilly@colorado.edu.

This article contains supporting information online at www.pnas.org/lookup/suppl/doi:10.1073/pnas.1303547110/-/DCSupplemental.

roles, within a sentence (e.g., agent, verb, or patient). The pattern of activity within each of these populations would then represent the current value, or filler, of that role. With a sufficient number of different populations (e.g., to represent the different possible roles), each with a sufficient number of neurons (e.g., to represent the possible fillers of each role), rich combinations of information can be expressed (e.g., sentences can be formed). This traditional strategy can support the learning of compositional representations when the range of training experiences spans the space of possible combinations of constituents (6), but three fundamental problems arise as the space of possible representations gets very large. First, this approach risks extensive duplication of resources. A second related, but more important, problem is that this scheme cannot benefit from generalization resulting from similarity of representation without the inclusion of additional mechanisms. This is because the representation learned for fillers in one role are specific to the neural population used to represent that role and isolated from the representation of the same fillers in other roles. For example, any similarity that boys and girls share in the role of agent will not generalize to their roles as patient, because agents and patients are represented over separate populations of neurons. Finally, learning from a limited number of experiences can shape neural representations based on accidental correlations (or anticorrelations) in the sampled set of experiences (7, 8). For example, if "girl" had never been experienced in the role of agent, then the population of neurons for the agent role would have no opportunity to learn to represent the "girl" filler in that role. Because of these problems, some researchers have proposed that compositional representation schemes in the brain are not learned but arise from specialized neural binding circuits that combine role and filler representations using a fixed method, such as the tensor product (9) or circular convolution (10). The resulting representations can seem identical to traditional ones involving the association of roles to isolated populations of neurons, but many of the problems involving generalization are avoided by making the binding operation nonadaptive and insensitive to experience. Avoiding the learning of compositional representations, in this way, makes it difficult to account for ways in which human behavior departs from pure systematicity (4, 5), however. Perhaps more importantly, although fixed binding operations of this kind have been formally implemented in simulated neural circuits (11), there is little anatomical or physiological evidence for such specialized circuitry in the brain.

Introducing a mechanism for indirection can avert the problems described above while still allowing compositional representations to be learned from experience. For example, this can be used to separate the representation of roles and fillers. Role populations can be used to represent pointers (e.g., in the PFC area A of our example above), which in turn can be assigned to reference fillers represented elsewhere (e.g., area B). This allows each filler to be represented only once rather than separately for each role. Furthermore, any role that points to that filler can exploit the similarity that filler shares with other fillers represented in that population, allowing what is learned about the relationship between fillers in one role to generalize to others. This separation between form and content is central to the classic symbol processing model (1). Using the model described below, we demonstrate that under certain architectural assumptions—that are consistent with the known anatomy of the PFC and BG systems—the relationship between representations of pointers and their referents (e.g., between roles and fillers) can be learned. However, although this mechanism exhibits considerable flexibility, its performance is constrained by its learning experiences—a feature that seems consistent with human performance.

We begin by providing a brief review of the neurobiological mechanisms that we propose support variable binding in the brain, followed by a computational model that embodies these properties. We then use the model to simulate a simple sentence processing task that requires variable binding. We demonstrate that, through trial-and-error learning, the model can learn to bind words (fillers) to their appropriate roles in the sentence and, critically, can do so for words that it has never seen in a particular role. We evaluate this by testing the model on several benchmark tests of generalization and by comparing it to others, including neural network architectures that have been used in previous work to model variable binding.

PFC, BG, and Indirection

Our model focuses on functional specializations within the BG that we have previously proposed implement a dynamic, adaptive gating mechanism for regulating the updating, maintenance, and output of information in the PFC (12–14) (Figs. 1 *A* and *B* and 2 *A* and *B*). Importantly, the model assumes that separate pathways through the BG can independently regulate different subregions of the PFC. This is consistent with the anatomy of both the PFC and BG. Within the PFC, relatively isolated stripe-like patches of neurons have been observed, each of which exhibits dense within-stripe interconnectivity and sparse (largely inhibitory) between-stripe connectivity (15, 16). Furthermore, these stripes project to distinct regions within the BG which, in turn, project back to different stripes within the PFC (17). In previous modeling work, we have shown that this anatomy and physiology can support a system of control in which gating signals from the BG regulate when a given stripe within the PFC will either (*i*) update to encode new information, (*ii*) continue to robustly maintain information in an active state (through sustained neural firing), or (*iii*) output the information it is currently encoding to drive information processing elsewhere in the brain (18). That is, it provides a separation of function that allows different signals to control when and where information is encoded or used (BG) compared with the signals that encode the information content itself (PFC) (14). Furthermore, we have shown that if the output of one PFC stripe controls the BG gating signal that regulates a different stripe, this architecture can support a system of hierarchically nested control (19). This, in turn, can be used to separate the representation of variables and their values, upon which a mechanism for indirection can be built.

Consider the sentence processing example introduced above (and illustrated in Fig. 1). Imagine there are different PFC stripes dedicated to representing the different roles of a sentence (e.g., agent, verb, and patient). In the simplest case, the pattern of activity in a given role-specific stripe would represent the current filler for that role. However, now consider that the pattern of activity within each role-specific stripe, instead of representing any particular filler, represents the address of another PFC stripe that represents that filler (Figs. 1 and 2*C*). There could then be a large number of different such filler stripes, organized in useful ways (e.g., according to semantic relationships among the fillers). The BG system can then be used to update the address information in the role-specific stripes as new sentences are presented and new fillers need to be assigned to a given role, whereas the role-specific stripe, when queried, can signal the BG to trigger the output of information from the filler stripe to which it currently points, permitting a read-out of the current filler for that role. Below, we show not only that such a system can self-organize through learning, but also that, with only a modicum of such learning, it can accurately process a wide range of role-filler combinations to which it has never been exposed.

Methods

Generalization Tests. To test these ideas, we used a simple sentence encoding and decoding task. Each sentence was composed of three roles: an agent, verb, and patient. Each role could be filled with words drawn from a set of 10 words, and each word could be used in any role, resulting in 1,000 possible sentences. The network was trained on only a small subset (20%) of the 1,000 possible sentences and then tested on sentences it had not previously seen, to evaluate its ability to generalize. On each trial, the network was presented

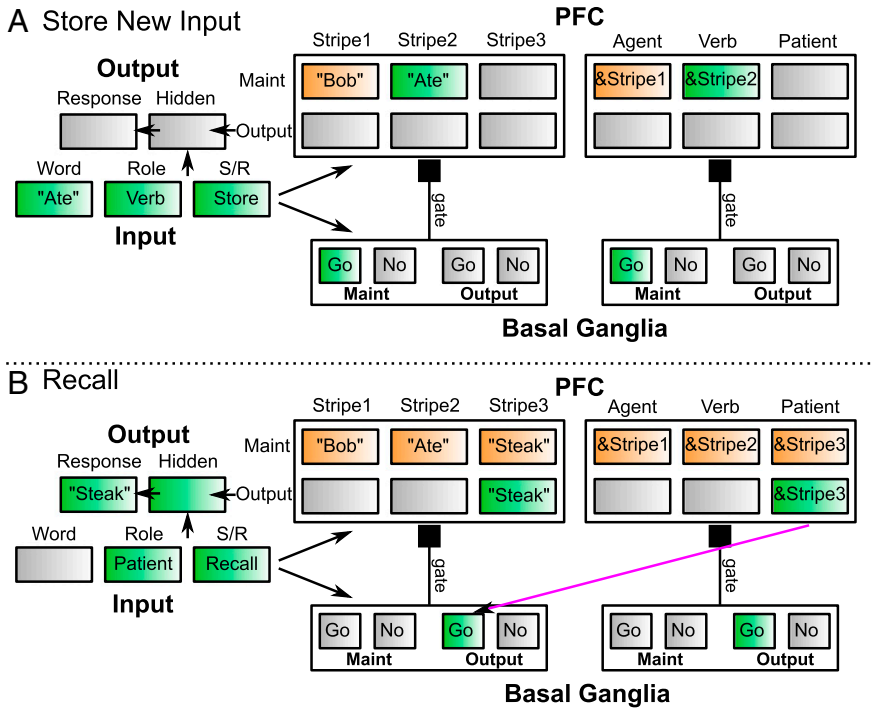

Fig. 1. Simple sentence encoding task demonstrating indirection in the PFC/BG working memory system. Three-word sentences are encoded one word at time along with a sentential role. After encoding, the network is probed for each word using the associated roles. Green indicates currently active inputs; orange represents actively maintained information. (*A*) One step of the encoding process for the sentence "Bob ate steak." "Ate" is presented along with its current role (verb) and the instruction to store, or encode, this information. In this example, the word "ate" is stored in Stripe2 of PFC filler stripes (*Left*). The identity/location of Stripe2 is subsequently stored in the verb stripe of PFC role stripes (*Right*). This process repeats for each of the other two words in the sentence. (*B*) One step of the recall process. A role (Patient in the example) and the instruction Recall are presented as input. This drives output gating of the address information stored that role stripe (highlighted by purple arrow), driving the BG units corresponding to that address to output gate the corresponding filler stripe, thus outputting the contents of that stripe (Steak).

with a sentence, word by word (Fig. 1*A*), requiring the maintenance of each word (filler) and its assigned role. The network was then tested by presenting each of the three roles as input, one at a time, to which it had to respond by generating the corresponding filler (Fig. 1*B*). The large majority (80%) of these were sentences it had not seen before, involving not only novel combinations of role-filler pairs, but also fillers in novel roles (i.e., novel role-filler pairs). The network's ability to generalize was indexed by its ability to process these and other types of novel sentences.

To further characterize generalization in the model, we used three variants of the basic learning and testing procedures described above. These tested for standard generalization, spurious correlations, and full combinatoric generalization, as described below. In addition to the full indirection model, we trained and tested two other variants as well as a standard neural network architecture that has been used extensively to model sequential processes, including language. A strict performance criterion of 95% correct was used to determine the amount of experience each network received during the training phase, requiring that every network performed nearly perfectly on the training sentences. Each network was trained and tested using each of the protocols listed below.

Standard generalization. After training on a random selection of 200 out of the 1,000 different sentences, 100 novel sentences were selected randomly from the remaining 800 to be used as the testing set. These sentences had not been experienced by the network during training. However, the training set was constrained so as to ensure that every individual word was presented as a filler in each of the different roles during training. This generalization protocol tested the ability of the network to encode and decode arbitrary combinations of role-filler pairs, but not its ability to process novel role-filler pairs.

Spurious anticorrelations. An anticorrelation occurs whenever one member of a pair of words is never seen in the same sentence as the other during training. The anticorrelation is spurious if there is no valid reason why the two words should not appear together. Such anticorrelations are pervasive in natural environments, in which there are large numbers of possible combinations of stimuli, all of which are plausible but few of which have been experienced. Learning such anticorrelations can be maladaptive, because the anticorrelated pairs could occur in the future. Many artificial neural network learning mechanisms show susceptibility to such spurious anticorrelations, which interferes with their ability to generalize to novel combinations of stimuli (7, 8). To evaluate the response to such anticorrelations, sentences in the training set were selected so that certain words never occurred together during learning (e.g., the words "knife" and "tire" were never part of the same sentence). Then, at test, networks were evaluated on their ability to process sentences that contained the anticorrelated words.

Full combinatorial generalization. This tested the network's ability to generalize not only to novel combinations of role-filler pairs, but also to novel role-filler pairs themselves. To do so, 2 words out of the 10 possible were selected and never used in the role of patient in sentences during training. Networks were then evaluated on their ability to process sentences containing those words as patients in the test set. This is an extremely difficult task for systems that learn from experience about the structure of the world.

Models. Fig. 2 shows the networks that we simulated to determine which architectural and processing features were critical to performance on each of the three generalization tests (*Supporting Information*). This includes three progressively elaborated variants of the PFC/BG working memory (PBWM) model (13, 14, 20). The first variant (PBWM Maintenance Only, Fig. 2*A*) was the simplest version of the PBWM model, in which each of the three words in a sentence was encoded and maintained in a separate PFC stripe. The identity of the PFC stripes that stored each word was determined by a learning process in the gating system (BG). The filler structure was learned randomly but included an additional assumption that if a PFC stripe recently gated in new information (e.g., a new word), then it was unlikely to do so again immediately. The specific "role" inputs were provided to the BG, allowing the network to learn a policy that specialized different PFC stripes to the different sentential roles. A response was generated by projections from the PFC and a sentential role input layer to a hidden (association) layer that, in turn, activated a representation of the filler for the specified role over the output units. The sentential role input layer was used during encoding to tell the network what the current role of a word should be (agent, patient, etc.) and, during recall, to identity the role of the filler to be retrieved (e.g., "What was the agent in the sentence you just saw?"). A total of 15 PFC/BG stripes were used in the maintenance-only version. This number was determined by use of a grid search procedure, optimizing the performance of the network across tasks. The other PBWM networks also used 15 stripes to store the "filler" information based on the grid search performed in the maintenance-only version. In other words, we optimized the number of stripes based on the performance of the simplest of the PBWM networks—to give that network the greatest advantage—and used that number for the other PBWM variants.

In the second variant (PBWM Output Gating, Fig. 2*B*), an output gating mechanism was added to the network that selectively decided when information maintained in the PFC stripes should be output, to drive responding. This allowed multiple items to be actively maintained in working memory while allowing only one to be selected to influence the response. A total of 15 PFC/BG stripes were used in the output gating

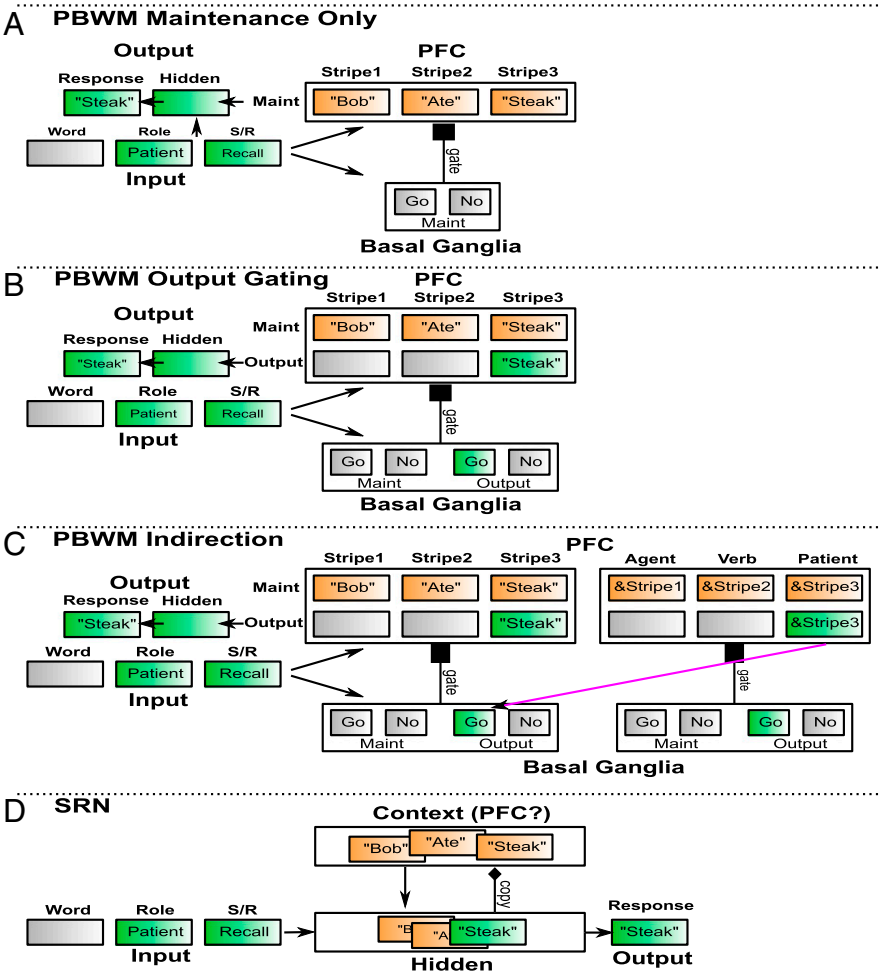

Fig. 2. Model architectures. Green indicates currently active inputs; orange represents actively maintained information. The inputs to all networks include the sentential role, the content word (filler), and a signal indicating whether this is an encoding (Store) or retrieval (Recall) trial. Please note that the actual number of PFC stripes used is greater than what is presented here for reasons of clarity. (*A*) PBWM maintenance-only network that implements the key components of the PBWM architecture. (*B*) PBWM output gating network, which separates active maintenance of representations in PFC from the driving a response. (*C*) Full PBWM indirection network. A word is presented as input and is gated into the filler-specific network on the left; its stripe address is propagated to gating units for the role-specific network on the right, which then gate that information into a role-specific PFC stripe. At recall, presentation of the role, together with the recall instruction, activates the output gating unit for that role-specific stripe. That stripe is storing the address of the corresponding filler-specific stripe, which activates the output gating unit for that stripe in the filler-specific network. (*D*) The simple recurrent network.

network (based on the same optimization procedure used for the maintenance-only version of the model).

The third variant (PBWM Indirection, Fig. 2*C*) implemented the full model. This possessed two complete and distinct PBWM networks as well as output gating. One PBWM network (filler-specific) learned to store each word in a set of different filler stripes. Each filler stripe learned to represent multiple fillers, and each filler was represented across multiple stripes. In the results discussed below, the PBWM "role-specific" network contained three sets of three PFC/BG stripes (nine in total), each set dedicated to a different role. Nine stripes were sufficient for satisfactory performance in the indirection network and, unlike the filler-specific network, this parameter was not optimized across models, because it is unique to the indirection network. The "pointer" that specified the location of the appropriate content in the filler-specific portion of the network was maintained until an output gating signal allowed a particular stripe's contents to influence the response. Output gating of the filler-specific network was tightly coupled with the role-specific PBWM network. For each word, the role-specific network learned to store the location of the stripe in the filler-specific network that represented that word. At test, presentation of a given role as input to the network generated an ouptut gating signal in the BG for the stripe corresponding to that role in the role-specific network. The address information stored in the role-specific stripe then generated, in the BG, an output gating signal for the stripe in the filler-specific network corresponding to that address. Output gating of the filler stripe then drove a response, via the hidden layer, corresponding to the word in the probed role. Note that the sentential role information was always explicitly provided in the input, instead of requiring the network to learn to recognize the role based on syntactic or other cues.

Finally, we tested a simple recurrent network (SRN) (Fig. 2*D*). SRNs have been used successfully to model a diverse set of complex behavioral phenomena associated with sequence processing, including early language acquisition, patterns of implicit learning, and routine task performance (21–23). Importantly, SRNs have been shown to exhibit behavior that seems componential and structured, using distributed representations without any explicit componential or hierarchical structure (23). This makes SRNs a useful comparison with our networks, which include structural elements to support componential and hierarchical relationships.

Results

The generalization results for each network are shown in Fig. 3. The SRN struggled to generalize successfully on all three tasks, indicating that it could not represent each of the sentence elements in a way that supported generalization to novel sentences. Based on prior models, it may require training on a larger portion of all possible sentences (23–25).

The PBWM maintenance-only network performed well on the standard generalization task. This is because it was able to store each word as the filler of the appropriate role in dedicated, independently updatable PFC stripes for each role. As long as it experienced each word in each role, it could process sentences involving novel combinations of these pairs. This was confirmed by examination of the network, which revealed that the PFC stripes learned to encode words for a specific sentential role (e.g., all of the verbs). This replicates findings from both other PBWM models and other models of representation learning in the PFC (26). However, this network failed to generalize in both the anticorrelation and full combinatorial generalization tasks, for reasons that are clarified by examination of the other two PBWM networks.

The PBWM output gating network did well on both the standard and anticorrelation generalization tasks, but not on the full combinatorial task. Its performance on the anticorrelation task sheds light on why the basic PBWM model performed poorly

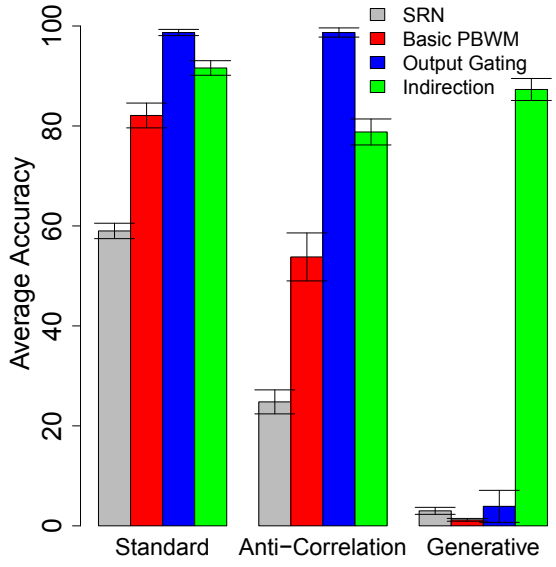

Fig. 3. Results for the four networks tested in the three generalization tasks (error bars represent SEM). The results are grouped by task: standard, anticorrelation, and generative.

on this task. Including an output gating mechanism restricted the influence of the items maintained across the various stripes of the PFC, encouraging a componential item-by-item influence on downstream cortical processing (e.g., in the hidden to response output pathway). Thus, the hidden layer only had to process a single, relevant, active representation from the output PFC stripes, instead of the entire set of active information across all of the PFC stripes, as in both the SRN and basic PBWM. This protected it from learning anticorrelations. For example, it would selectively allow the verb item to be processed by the response pathway, without any "contamination" from the maintained agent and patient role items. This is consistent with and replicates previous findings concerning the generalization benefits of output gating within the PBWM framework (27). However, the output gating network failed on the full combinatorial task because it had not been exposed to (many of) the role-filler pairings in the test set, and therefore did not learn filler representations for those roles.

Finally, the PBWM indirection network was able to generalize effectively in all of the tasks. This was due to its ability to encode fillers in roles as pointers to filler-specific stripes, rather than directly within the role-specific stripes themselves. So long as the network had experienced encoding a given stripe address where a filler was being maintained, it could then process anything stored at that location, allowing it to generalize to novel role-filler pairs. The network learned these joint content and location representations across the two sets of interconnected PFC/BG networks.

Discussion

We have described a unique neural network architecture that implements arbitrary variable binding using biologically plausible mechanisms and demonstrated its ability to generalize, from limited experience, to a rich combinatorial domain. Using a simple sentence processing task, we demonstrated the model's ability to generalize not only to novel combinations of role-filler pairs, but also to novel assignments of fillers to roles. This level of generalization was accomplished only by the full PBWM indirection mode, and could not be achieved by models without a mechanism for indirection.

The design of the architecture of the PBWM indirection network was motivated in part by the computational demands of full combinatorial generalization that require a mechanism for variable binding. However, it was also inspired by, and is consistent with, a growing body of neuroscientific evidence concerning the anatomy and physiology of the PFC and BG. There is long-standing evidence of an asymmetric "spiral" pattern of projections between the PFC and BG, in which the PFC projects to areas of the BG that project to a nearby but distinct area of the PFC (17). Such evidence continues to accrue, suggesting that projections from the PFC to striatum exhibit a rostral-to-caudal bias (28, 29); similar spiraling patterns of connectivity are seen within the striatum proper (30). Gradients of function have also been proposed within the PFC. For example, some neuroimaging studies have suggested a dorsal–ventral organization according to content (31–35). This has also been suggested by previous modeling work using the PBWM, in which more dorsal areas have been proposed to represent higher-level information (e.g., dimensional or category information and task specifications), whereas more ventral areas represent more specific information (e.g., featural) (18, 35). Another proposal suggests that there is an anterior–posterior functional hierarchy in the frontal cortex (36, 37), in which anterior areas represent more abstract information (e.g., high-level goals), whereas more posterior areas represent more concrete information (e.g., motor plans). The separation of subnetworks within the PFC and the pattern of projections between the PFC and BG in the PBWM indirection network are broadly consistent with these proposals. Importantly, it provides a more specific characterization of the function of distinct PFC regions that it should be possible to test empirically.

A critical finding from the current simulations is that a neural network architecture can learn to implement indirection and thus achieve the capacity for variable binding required for full combinatorial generalization. This required a particular predetermined macroarchitecture (e.g., the pattern of projections between the PFC and BG) and set of physiological functions (e.g., the learning algorithm, input and output gating). However, the microarchitecture of the model (e.g., the representations within the filler subnetwork) and its information-processing functions (e.g., the conditions under which individual stripes were gated) were not prespecified. These were learned through experience. It should be noted, however, that the pointer representations were specified as a localist map of the possible filler stripe locations. Although there is evidence of topographic connections throughout the cortex, it seems plausible that these could be learned through an extended developmental process. Thus, the model can be characterized as a hybrid, in which general characteristics of anatomy and physiology are predetermined (presumably by genetics) but the functional characteristics are learned by exposure to the environment. This included the capacity for indirection, and thereby variable binding. This was made possible by the separate, but anatomically related, sets of PFC/BG subnetworks, allowing the system to segregate function (role) from content (filler). As a consequence, the role-specific stripes needed only to concern themselves with learning, at a high level, where to look for content when the time was appropriate. This greatly reduced the representational burden of these stripes, relieving them of the burden of encoding all possible fillers that may ever be needed in a particular role.

Although the model we described focused on a simple sentence processing task, the principles of function can be readily generalized. The role-specific subnetwork can be thought of as representing elements of task context and the filler-specific network as representing stimuli and/or actions that are appropriate in that particular context. Thus, the model could also be used to simulate not only the interpretation of novel sequences of inputs, but also the production of novel, appropriately structured sequences of actions. In this regard, it can be considered as providing the functionality necessary not only for generalization but also generativity (the ability to generate meaningful novel behavior). In this respect, it offers a middle ground in the long-standing

debate between symbolic and subsymbolic models of cognition (1, 38). Advocates of symbolic models have long pointed to the kind of combinatorial generalization we have tested here and, critically, capacity for generativity as support for the claim that human cognitive function relies on symbol processing and that this cannot be accomplished by associationist (neural network) architectures. In contrast, advocates of subsymbolic models have argued that the evident limitations on human symbolic reasoning (39, 40) suggest that actual everyday human cognition is better characterized by subsymbolic processing. Our model suggests that both approaches may be partially correct: The architecture of the brain supports a form of indirection and thereby variable binding, but it is limited. In particular, it relies on extensive learning, and the representations it develops are embedded, and distributed—the representations our indirection model learned were distributed across multiple stripes, which worked together to represent the information. This characterization also differs significantly from other biologically based attempts to account for symbolic processing abilities. These accounts build in automatic low-level variable binding mechanisms (11, 41–45), which currently lack an explanation for how these systems can be learned from experience and at the extreme may suggest that this symbolic capacity is present even at the lowest levels of processing, without any learning necessary. Furthermore, we expect that the subsymbolic foundation of our indirection model, and its grounding in neural learning mechanisms, will render it more robust and powerful than purely symbolic models, which are often brittle.

The computational architecture we have described could be used to explore the development of specific systems of content (filler) and pointer (role) representations in a range of different cognitive processing domains. This holds promise for generating predictions about the stages of cognitive flexibility and systematicity as a function of learning experience that can be compared with available human developmental data to further test and inform the model.

ACKNOWLEDGMENTS. This work was supported by the Intelligence Advanced Research Projects Activity (IARPA) via Department of the Interior (DOI) Contract D10PC20021. The US government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA, DOI, or the US government.

1. Fodor JA, Pylyshyn ZW (1988) Connectionism and cognitive architecture: A critical analysis. *Cognition* 28(1–2):3–71.
2. Marcus GF (1998) Rethinking eliminative connectionism. *Cognit Psychol* 37(3): 243–282.
3. van Gelder T (1990) Compositionality: A connectionist variation on a classical theme. *Cogn Sci* 14:355–384.
4. Johnson K (2004) On the systematicity of language and thought. *J Philos* 101:111–139.
5. van Gelder T, Niklasson LF (1994) *Classicism and Cognitive Architecture* (Lawrence Erlbaum, Hillsdale, NJ), pp 905–909.
6. Plaut DC, McClelland JL, Seidenberg MS, Patterson K (1996) Understanding normal and impaired word reading: computational principles in quasi-regular domains. *Psychol Rev* 103(1):56–115.
7. Noelle DC, Cottrell GW (1996) *In Search of Articulated Attractors*, ed Cottrell GW (Lawrence Erlbaum, Mahwah, NJ), pp 329–334.
8. Noelle DC, Zimdars AL (1999) *Methods for Learning Articulated Attractors over Internal Representations*, eds Hahn M, Stoness SC (Lawrence Erlbaum, Mahwah, NJ), pp 480–485.
9. Smolensky P (1990) Tensor product variable binding and the representation of symbolic structures in connectionist networks. *Artif Intell* 46:159–216.
10. Plate TA (2008) Holographic reduced representations. *IEEE Trans Neural Networks* 6:623–641.
11. Stewart TC, Bekolay T, Eliasmith C (2011) Neural representations of compositional structures: Representing and manipulating vector spaces with spiking neurons. *Connect Sci* 23:145–153.
12. Frank MJ, Loughry B, O'Reilly RC (2001) Interactions between frontal cortex and basal ganglia in working memory: A computational model. *Cogn Affect Behav Neurosci* 1(2):137–160.
13. O'Reilly RC, Frank MJ (2006) Making working memory work: A computational model of learning in the prefrontal cortex and basal ganglia. *Neural Comput* 18(2):283–328.
14. O'Reilly RC (2006) Biologically based computational models of high-level cognition. *Science* 314(5796):91–94.
15. Levitt JB, Lewis DA, Yoshioka T, Lund JS (1993) Topography of pyramidal neuron intrinsic connections in macaque monkey prefrontal cortex (areas 9 and 46). *J Comp Neurol* 338(3):360–376.
16. Pucak ML, Levitt JB, Lund JS, Lewis DA (1996) Patterns of intrinsic and associational circuitry in monkey prefrontal cortex. *J Comp Neurol* 376(4):614–630.
17. Alexander GE, DeLong MR, Strick PL (1986) Parallel organization of functionally segregated circuits linking basal ganglia and cortex. *Annu Rev Neurosci* 9:357–381.
18. O'Reilly RC, Noelle DC, Braver TS, Cohen JD (2002) Prefrontal cortex and dynamic categorization tasks: Representational organization and neuromodulatory control. *Cerebral Cortex* 12:246–257.
19. Reynolds JR, O'Reilly RC (2009) Developing PFC representations using reinforcement learning. *Cognition* 113(3):281–292.
20. Hazy TE, Frank MJ, O'Reilly RC (2006) Banishing the homunculus: Making working memory work. *Neuroscience* 139(1):105–118.
21. Hare M, Elman JL (1995) Learning and morphological change. *Cognition* 56(1):61–98.
22. Cleeremans A, McClelland JL (1991) Learning the structure of event sequences. *J Exp Psychol Gen* 120(3):235–253.
23. Botvinick M, Plaut DC (2004) Doing without schema hierarchies: A recurrent connectionist approach to normal and impaired routine sequential action. *Psychol Rev* 111(2):395–429.
24. O'Reilly RC (2001) Generalization in interactive networks: The benefits of inhibitory competition and Hebbian learning. *Neural Comput* 13(6):1199–1241.
25. Brousse O (1993) Generativity and systematicity in neural network combinatorial learning (Department of Computer Science, Univ of Colorado, Boulder, CO), Technical Report CU-CS-676-93.
26. Rougier NP, Noelle DC, Braver TS, Cohen JD, O'Reilly RC (2005) Prefrontal cortex and flexible cognitive control: Rules without symbols. *Proc Natl Acad Sci USA* 102(20): 7338–7343.
27. Kriete T, Noelle DC (2011) Generalisation benefits of output gating in a model of prefrontal cortex. *Connect Sci* 23:119–129.
28. Haber SN, Kim KS, Mailly P, Calzavara R (2006) Reward-related cortical inputs define a large striatal region in primates that interface with associative cortical connections, providing a substrate for incentive-based learning. *J Neurosci* 26(32):8368–8376.
29. Verstynen TD, Badre D, Jarbo K, Schneider W (2012) Microstructural organizational patterns in the human corticostriatal system. *J Neurophysiol* 107(11):2984–2995.
30. Haber SN, Fudge JL, McFarland NR (2000) Striatonigrostriatal pathways in primates form an ascending spiral from the shell to the dorsolateral striatum. *J Neurosci* 20(6): 2369–2382.
31. Funahashi S, Chafee MV, Goldman-Rakic PS (1993) Prefrontal neuronal activity in rhesus monkeys performing a delayed anti-saccade task. *Nature* 365(6448):753–756.
32. Smith EE, Jonides J (1999) Storage and executive processes in the frontal lobes. *Science* 283(5408):1657–1661.
33. Romanski LM (2004) Domain specificity in the primate prefrontal cortex. *Cogn Affect Behav Neurosci* 4(4):421–429.
34. Petrides M (2005) Lateral prefrontal cortex: architectonic and functional organization. *Philos Trans R Soc Lond B Biol Sci* 360(1456):781–795.
35. O'Reilly RC (2010) The *What* and *How* of prefrontal cortical organization. *Trends Neurosci* 33(8):355–361.
36. Koechlin E, Ody C, Kouneiher F (2003) The architecture of cognitive control in the human prefrontal cortex. *Science* 302(5648):1181–1185.
37. Badre D, D'Esposito M (2007) Functional magnetic resonance imaging evidence for a hierarchical organization of the prefrontal cortex. *J Cogn Neurosci* 19(12): 2082–2099.
38. McClelland JL, et al. (2010) Letting structure emerge: Connectionist and dynamical systems approaches to cognition. *Trends Cogn Sci* 14(8):348–356.
39. Johnson-Laird PN (2001) Mental models and deduction. *Trends Cogn Sci* 5(10): 434–442.
40. Todd PM, Gigerenzer G (2000) Précis of simple heuristics that make us smart. *Behav Brain Sci* 23(5):727–741, discussion 742–780.
41. Hummel JE, Biederman I (1992) Dynamic binding in a neural network for shape recognition. *Psychol Rev* 99(3):480–517.
42. David S, Touretzky GEH (1988) A distributed connectionist production system. *Cogn Sci* 12:423–466.
43. Lebiere C, Anderson JR (1993) *A Connectionist Implementation of the ACT-R Production System* (Lawrence Earlbaum, Hillsdale, NJ).
44. Stocco A, Lebiere C, Anderson JR (2010) Conditional routing of information to the cortex: A model of the basal ganglia's role in cognitive coordination. *Psychol Rev* 117(2):541–574.
45. Hayworth KJ (2012) Dynamically partitionable autoassociative networks as a solution to the neural binding problem. *Front Comput Neurosci* 6:73, 10.3389/fncom.2012.00073.

Supporting Information

Kriete et al. 10.1073/pnas.1303547110

Overview of Indirection and Comparison Models

In *Supporting Information* the reader is provided with additional information to help explain, in greater detail, the structure and layout of the models presented in the main text. It is important to note that the same training and testing environments were used across all of the network architectures as described later in this document. More precisely, the networks had access to the same pieces of information. However, due to meaningful architectural differences, the exact manner that this information was provided necessarily differed. For instance, the simple recurrent network (SRN) receives all pertinent information via the network's "hidden" layer. Whereas the prefrontal cortex/basal ganglia working memory (PBWM) models allowed for segregation of inputs such that the prefrontal cortex (PFC) only received information that it may need to maintain, the modeled basal ganglia (BG) only received inputs that were informative with regard to the updating of the contents of the PFC. The precise connectivity scheme is contained within the description of the networks below.

Environment and Task. In this section the inputs and information that are available to all networks will be described. To perform the sentential role task, three pieces of information are presented to the networks. These are the word ("tire", "knife," etc), the role (agent, patient, or verb), and whether this is an encoding or retrieval trial (store vs. recall). The networks are first presented with three words for storage and immediately probed for retention of the proper word–role binding. For instance, during an encoding trial the network could be presented with activity vectors that correspond to the following: the word is "knife," it is the agent of the sentence, and this is storage trial. For correct performance on a storage trial, the network simply needs to repeat the word it was presented ("knife" in this case). To complete the sentence, the next two words are presented sequentially along with their roles. During a retrieval scenario, the network would be probed with the role and the fact that this is a retrieval trial in the same sequential manner. For correct performance the network must respond with the correct word based on the sentential role for the previously encoded minisentence. There are three roles and 10 different words, resulting in 1,000 unique minisentences. The training environment always consisted of 20% of this total, or 200 minisentences. Every minisentence consisted of three words, and the network was scored on whether it both encoded as well as retrieved each word correctly. The networks were trained until they reached the performance criteria of no more than 5% error on encoding and retrieval of all words per 200-sentence block.

PBWM. The PBWM architecture uses a specific form of biologically inspired information processing. Information processing is separated into online/active maintenance of information in the PFC, updating/gating of this information via the BG, and stimulus response mapping from the actively maintained information in the PFC to an overt response through the hidden layer. This separation is true in all variants of PBWM, including PBWM plus output gating, as well as the full indirection model, both described in the following sections. Given this organization, the information (inputs) that originates from an external environment is highly processed and, in general, only influences the parts of the PBWM system that require the respective inputs to function properly. Specifically, the inputs that need to be actively maintained over an extended period will only project to the PFC, and those that are informative of the precise timing required for updating of the PFC will selectively influence the BG. This connectivity scheme is a simplification made for clarity and computational efficiency but does not represent any specific theoretical claim with regard to the specificity of environmental inputs' influence on areas of the human brain.

The original PBWM model network uses a set of PFC/BG stripes that learn, through experience, proper credit assignment for proper PFC stripe updating (1–3). This updating policy is based upon contextual information available as both environmental inputs and the current information maintained within the PFC itself. The PFC actively maintains the "content" (e.g., words) and is used to generate an appropriate response when required. The representations for the content terms are stored in a distributed activity pattern across the units in the various PFC stripes. The words are initially presented as a purely localist input layer (one unit per word) but are stored (and used) as highly distributed patterns across 25 units contained within each PFC stripe. The response is generated by projections from the PFC to a standard hidden layer, which is used to model simple stimulus to response mappings in the task. (Fig. 1*A*). In this original PBWM, the hidden layer also receives input that conveys the sentential role (e.g., patient). This information is used, along with the pattern of activity across the PFC, to choose what response is appropriate. It is the case that PBWM nearly always uses multiple stripes when learning to store any single piece of information. Although the exact number of stripes used is an adjustable parameter, three is the standard and is used in most PBWM networks past and present. The advantage is evident during the learning, where using multiple stripes to encode information benefits the Pavlov (PVLV) gating learning process. Essentially, by allowing more stripes to encode the relevant information, the search process is parallelized to some degree, allowing a modest degree of noise or blurring to occur when determining which stripes are encoding which piece of information. Not only do we view using multiple stripes to encode information as more biologically realistic than only one per item, without more than one PFC stripe used to encode information the process would be extremely fragile, relying on perfect recall of the perfect stripe to store and output the information for every piece of information the organism may store and use. Having multiple stripes also allows for redundancy in the learning of gating policies that we have found very useful in practical application. The BG layers [which are used to learn the proper PFC gating policy via a biologically based reinforcement learning paradigm (4)] receive inputs about the "task" as well as if the current trial is a storage or retrieval trial. Using the sentential working memory task from the main text, the BG receives the sentential role (e.g., agent, patient, or verb) as well as the store/recall information and the PFC receives the "word" inputs. Importantly, the PFC/BG stripe structure encourages the information to be initially encoded in a componential manner (one item per PFC stripe). Structuring the problem space in this manner can greatly simplify the task of learning the appropriate storage policy. The benefit is achieved by allowing each PFC stripe to learn to specialize what information it is going to store rather than being a general purpose learning mechanism.

PBWM with Output Gating. A key issue with the standard PBWM model is that items being actively maintained in PFC stripes can immediately influence other cortical areas. For instance, as soon as the model updates a PFC stripe with an item that is relevant for future task performance, the actively maintained representation

immediately influences response pathways. This is because once cells in the modeled PFC are firing they immediately influence the hidden layer, which is used to drive an actual response. Human behavior does not seem to follow this pattern, however. Instead, it is apparent we have the ability to maintain multiple items and goals active within the PFC, only allowing items to influence responses when appropriate. Interestingly, the ability to withhold the influence of a specific PFC stripe until it is relevant for task processing is very similar to the process of preventing a stripe from updating its contents until the proper time. In other words, the process of updating PFC with new contents that will be needed at a later time can be thought of as input gating and the process by which information is prevented from influencing other cortical areas until ready is response gating or output gating. This is isomorphic in function, just different in the direction of the information flow in the networks. Given this observation, we have instantiated the output gating process using the exact same mechanistic framework as input gating. Under this account the BG system learns, via the midbrain dopamine signal, a useful output gating policy for the task at hand given contextual cues. To simulate this process in the PBWM framework, an additional set of PFC/BG stripes is included. The network now has two different BG/PFC gating policies that it must learn from experience. First, and identical to the original PBWM network, it must learn when a PFC stripe should be updated with new information that is available in the environment. Additionally, the network must now learn when it is appropriate to let the contents of specific PFC stripes influence the response pathways. Although a slightly more difficult learning task, the addition of the output gating PFC stripes has been shown to both speed learning and improve generalization performance in previous simulations (5). The structure of this network is shown in Fig. 1*B*. The manner in which the inputs connect to the model is very similar to the original PBWM model. The differences of note are a direct consequence of the fact that there are now two sets of PFC/BG stripes, one for maintenance only and one used for response gating. First, both modeled BG layers receive the task-relevant information (sentential role) as well as whether the model is in store or recall mode. The second difference involves the original PFC (maintenance only) and the new output gating mechanism. The maintenance-only portion of the PFC can no longer directly influence the response of the network; instead, this information is relayed via the output gating PFC stripes. This prevents any unwanted or unnecessary information from influencing the network's response until it this the appropriate time. The third connectivity difference is that the hidden processing layer no longer receives any task information, and instead is only influenced by the response/output gating stripes of the PFC.

PBWM Indirection. As discussed in the main text, the indirection version of the PBWM framework links together two PBWM with output gating networks (Fig. 1*C*). The two networks can be thought of as encoding task-relevant and content-specific information, respectively. The task-based PBWM network is nearly an exact copy of the standard PBWM with output gating network just described. The modeled BG layers still receive the task-relevant information including the sentential role as well as the store vs. recall inputs. The main difference in the task-based portion of the PBWM network and the standard output-gating version is that the PFC now encodes the location of a PFC stripe from the content-based PBWM network instead of the actual content. For simplicity, each task-relevant PFC stripe contains a localist map of the possible content PFC stripes. These maps receive topographically precise convergent connections from the content-based PBWM system. In other words, each task PFC stripe is provided exact information with regard to which content-based PFC stripe was just used. This removes the need for the complicated learning of these representations; however, it is possible that just such representations have been developed with a large amount of diverse prior experience.

The content-based PBWM portion of the indirection framework is slightly different from previously explained versions. Specifically, both the PFC and the maintenance portion of the BG receive information about the content (words). Previously, the BG would receive task-relevant information, but this is not the case in the indirection model. This connectivity scheme, plus a bias to not immediately "overwrite" a recently updated stripe, results in stripes that specialize to a subset of the words experienced by the network. In general, no stripe needs to encode every possible word because this responsibility is spread across the multitude of stripes, reducing the representational burden. Importantly, the content-based PFC stripes communicate with the task-based stripes via the topographic connections described previously. When the network stores a new word in one of the content-based PFC stripes, the identity of the updated content-stripe(s) is passed through this connection to the task-based PFC. During a response trial, the task-based PFC releases this location (of the relevant content-based PFC stripe), driving the gating of the proper content-based PFC stripe. In our network, this information is sent directly to the output gating portion of the content-based BG. This results in the proper content-based PFC stripe output gating the word that it is currently maintaining. This information is next processed via the hidden layer of the network to drive the final response. It is important to note that the connection from the task-based PFC (output) to the content-based output BG has a strong topographic component. Specifically, each of the task PFC stripes essentially contains a map of possible locations of the content PFC stripes, and this map is used to directly connect the correct representation of the location within the task-based PFC to the actual (and correct) output stripe. This removes the need of the network to learn this mapping through experience; however, it is plausible that precisely this kind of functional connectivity would naturally arise from the proper previous experience and is another important area of future investigation.

The indirection network learns two different strategies for storing information, one for the role (agent, patient, or verb) and a different strategy for the filler (the actual words). For the role portion of the PBWM with indirection network, the topography described above encourages the network to encode the agent, verbs, and patients each in their own set of nonoverlapping PFC stripes. The topography was included to simplify the learning process and to enable us to cleanly explore the benefits of using indirection. However, this is not the case for the filler portion of the network where the words themselves are stored. The filler section of the network learns to use a distributed pattern of PFC stripes to encode the various words in the vocabulary. Furthermore, the pattern of activity within each PFC stripe is also distributed in this portion of the network.

Neural Plasticity in PBWM. PBWM is a powerful and complex model of human learning and behavior. As such, it can be difficult to discern what is changing in these networks (what is learned) and what is set up by the modelers themselves. In the PBWM networks presented here, every simulated neural pathway (or "connection" in neural network terms) is plastic and learned, except for one set of connections described below. There are three main types of connections used in PBWM networks that can be roughly broken up as cortical connections. These are the connections that project into the PFC and the simulated "posterior" areas (e.g., hidden layers) of the network, those that project into the BG gating system, and those that are used by the PVLV critic portion of the system. The PVLV system is described in detail below in *PVLV Equations*. The connections into the BG gating system are adjusted simply according to the reward signal provided by the PVLV system and is described in detail in refs. 1 and 3). The

other cortical connections, including those from the environmental inputs into the PFC or hidden layer, the connections between the PFC and the hidden layer, and the connections between the response layer and the hidden layer, are adjusted according to Leabra's conditional principal components analysis (CPCA) learning algorithm. The standard Leabra framework is described and reviewed in ref. 6;for in-depth details of the PBWM framework see refs. 1–4. However, there is one set of connections in the indirection version of PBWM presented in this paper that is not currently learned. This is the pattern of connections from the filler portion of the network to the PFC stripes in the role portion of the indirection model that conveys which PFC stripe(s) recently updated their contents. This is essentially the "address" of what specific PFC stripes just encoded new information. However, it is important to note that the network must learn to store or gate this information at the appropriate time to succeed. We remain optimistic that a pattern of connectivity similar to what was specified in the network will emerge naturally with learning, given a much richer training environment, but it was not a focus of this work to investigate this important issue.

SRNs. SRNs have been used to successfully model a diverse set of complex behavioral phenomena, including early language acquisition, patterns of implicit learning, and instructed task performance, to name just a few (7–9). Importantly, SRNs have been shown to exhibit componential and structured behavior (9). This makes SRNs an intriguing candidate to use in our task. In the SRN framework, the ability to use temporally extended information (such as in a working memory task) is provided by including a context layer to a standard back-propagation network. The context layer is used as an additional input to the hidden layer. The hidden layer is used to compute stimulus response mappings, given current inputs. In this way the SRN is the same as standard neural network approaches. The SRN differs from those approaches by allowing the context layer to be a copy of the previous activity pattern of the hidden layer, providing information about what the network has computed at an earlier time. By providing temporally distal information via the context layer, the network can learn, with experience, to solve tasks that rely on processing over multiple time steps. The structure of the SRN used to model our task is shown in Fig. 1*D*. The simple structure of the SRN only provides us with one option for the inputs: All inputs must be routed directly to the hidden layer and the network must learn to use them appropriately. The SRN was capable of completely learning the task but incapable of any generalization.

Leabra: Implementational Details

Leabra (10, 11) is a biologically based computational modeling framework that has been used to explain the neural basis of cognition in a wide range of different domains. Leabra incorporates many biologically inspired mechanisms, including a biophysical neural activation function, complex network dynamics using bidirectional excitatory connections and fast pooled inhibition, both a correlational (Hebbian) learning and an error-driven mechanism for synaptic plasticity, and dopamine-modulated reinforcement learning. In each of the PBWM-based computational models presented here the Leabra framework was used. To develop and run the simulations, Emergent (12), the latest software available supporting the Leabra equations, was used for all of the results generated. The SRN network results were generated using the standard back-propagation learning algorithm. In general, the SRN learned the task faster and more reliably using a standard back-propagation compared with using the Leabra learning algorithms. For a recent review of the Leabra framework see ref. 6, and for in-depth details of the PBWM framework see refs. 1–4. Both are described in some detail below as well.

The model was implemented using the Leabra framework, which is described in detail in refs. 1 and 11, and summarized here. These same parameters and equations have been used to simulate over 40 different models in ref. 11 and a number of other research models. Thus, the model can be viewed as an instantiation of a systematic modeling framework using standardized mechanisms, instead of constructing new mechanisms for each model.

Point Neuron Activation Function. Leabra uses a point neuron activation function that models the electrophysiological properties of real neurons while simplifying their geometry to a single point. This function is nearly as simple computationally as the standard sigmoidal activation function, but the more biologically based implementation makes it considerably easier to model inhibitory competition, as described below. Further, using this function enables cognitive models to be more easily related to more physiologically detailed simulations, thereby facilitating bridge building between biology and cognition.

The membrane potential V_m is updated as a function of ionic conductances g with reversal (driving) potentials E as follows:

$$\Delta V_m(t) = \tau \sum_c g_c(t)\overline{g_c}(E_c - V_m(t)) \qquad \textbf{[S1]}$$

with three channels (c) corresponding to e, excitatory input; l, leak current; and I, inhibitory input. Following electrophysiological convention, the overall conductance is decomposed into a time-varying component $g_c(t)$ computed as a function of the dynamic state of the network and a constant $\overline{g_c}$ that controls the relative influence of the different conductances. The equilibrium potential can be written in a simplified form by setting the excitatory driving potential (E_e) to 1 and the leak and inhibitory driving potentials (E_l and E_i) to 0:

$$V_m^{\infty} = \frac{g_e\overline{g_e}}{g_e\overline{g_e} + g_l\overline{g_l} + g_i\overline{g_i}}, \qquad \textbf{[S2]}$$

which shows that the neuron is computing a balance between excitation and the opposing forces of leak and inhibition. This equilibrium form of the equation can be understood in terms of a Bayesian decision-making framework (11).

The excitatory net input/conductance $g_e(t)$ or η_j is computed as the proportion of open excitatory channels as a function of sending activations times the weight values:

$$\eta_j = g_e(t) = \langle x_i w_{ij} \rangle = \frac{1}{n}\sum_i x_i w_{ij}. \qquad \textbf{[S3]}$$

The inhibitory conductance is computed via the k-Winners-Take-All (kWTA) function described in the next section, and leak is a constant.

Activation communicated to other cells (y_j) is a thresholded (Θ) sigmoidal function of the membrane potential with gain parameter γ:

$$y_j(t) = \frac{1}{\left(1 + \frac{1}{\gamma[V_m(t) - \Theta]_+}\right)}, \qquad \textbf{[S4]}$$

where $[x]_+$ is a threshold function that returns 0 if $x < 0$ and x if $X > 0$. Note that if it returns 0, we assume $y_j(t) = 0$, to avoid dividing by 0. As it is, this function has a very sharp threshold, which interferes with graded learning mechanisms (e.g., gradient descent). To produce a less discontinuous deterministic function with a softer threshold, the function is convolved with a Gaussian

noise kernel ($\mu = 0$, $\sigma = .005$), which reflects the intrinsic processing noise of biological neurons:

$$y_j^*(x) = \int_{-\infty}^{\infty} \frac{1}{\sqrt{2\pi}\sigma} e^{-z^2/(2\sigma^2)} y_j(z-x)dz, \quad \textbf{[S5]}$$

where x represents the $[V_m(t)-\Theta]_+$ value, and $y_j^*(x)$ is the noise-convolved activation for that value. In the simulation, this function is implemented using a numerical lookup table.

kWTA Inhibition. Leabra uses a kWTA function to achieve inhibitory competition among units within a layer (area). The kWTA function computes a uniform level of inhibitory current for all units in the layer, such that the $k+1$th most excited unit within a layer is generally below its firing threshold, whereas the kth is typically above threshold. Activation dynamics similar to those produced by the kWTA function have been shown to result from simulated inhibitory interneurons that project both feed-forward and feedback inhibition (11). Thus, although the kWTA function is somewhat biologically implausible in its implementation (e.g., requiring global information about activation states and using sorting mechanisms), it provides a computationally effective approximation of biologically plausible inhibitory dynamics.

kWTA is computed via a uniform level of inhibitory current for all units in the layer as follows:

$$g_i = g_{k+1}^\Theta + q\left(g_k^\Theta - g_{k+1}^\Theta\right), \quad \textbf{[S6]}$$

where $0 < q < 1$ (0.25 default used here) is a parameter for setting the inhibition between the upper bound of g_k^Θ and the lower bound of g_{k+1}^Θ. These boundary inhibition values are computed as a function of the level of inhibition necessary to keep a unit right at threshold:

$$g_i^\Theta = \frac{g_e^* \overline{g_e}(E_e - \Theta) + g_l \overline{g_l}(E_l - \Theta)}{\Theta - E_i}, \quad \textbf{[S7]}$$

where g_e^* is the excitatory net input without the bias weight contribution—this allows the bias weights to override the kWTA constraint.

In the basic version of the kWTA function, which is relatively rigid about the kWTA constraint and is therefore used for output layers, g_k^Θ and g_{k+1}^Θ are set to the threshold inhibition value for the kth and $k+1$th most excited units, respectively. Thus, the inhibition is placed exactly to allow k units to be above threshold, and the remainder below threshold. For this version, the q parameter is almost always 0.25, allowing the kth unit to be sufficiently above the inhibitory threshold.

In the average-based kWTA version, g_k^Θ is the average g_i^Θ value for the top k most excited units, and g_{k+1}^Θ is the average of g_i^Θ for the remaining $n-k$ units. This version allows for more flexibility in the actual number of units active depending on the nature of the activation distribution in the layer and the value of the q parameter (which is typically 0.6), and is therefore used for hidden layers.

PVLV Equations. A biologically rigorous reinforcement system is used to learn how to update the contents of the PFC. The PVLV (Pavlov) system is described in detail in ref. 4. The PVLV value layers use standard Leabra activation and kWTA dynamics as described above, with the following modifications. They have a three-unit distributed representation of the scalar values they encode, where the units have preferred values of (0, 0.5, and 1). The overall value represented by the layer is the weighted average of the unit's activation times its preferred value, and this decoded average is displayed visually in the first unit in the layer. The activation function of these units is a "noisy" linear function [i.e., without the $x/(x+1)$ nonlinearity, to produce a linear value representation, but still convolved with Gaussian noise to soften the threshold, as for the standard units, Eq. **S5**], with gain $\gamma = 220$, noise variance $\sigma = .01$, and a lower threshold $\Theta = .17$. The k for kWTA (average based) is 1, and the q value is 0.9 (instead of the default of 0.6). These values were obtained by optimizing the match for value represented with varying frequencies of 0–1 reinforcement (e.g., the value should be close to 0.4 when the layer is trained with 40% 1 values and 60% 0 values). Note that having different units for different values, instead of the typical use of a single unit with linear activations, allows much more complex mappings to be learned. For example, units representing high values can have patterns of weights completely different from those encoding low values, whereas a single unit is constrained by virtue of having one set of weights to have a monotonic mapping onto scalar values.

Learning Rules. The PVe layer does not learn and is always just clamped to reflect any received reward value (r). By default we use a value of 0 to reflect negative feedback, 0.5 for no feedback, and 1 for positive feedback (the scale is arbitrary). The PVi layer units (y_j) are trained at every point in time to produce an expectation for the amount of reward that will be received at that time. In the minus phase of a given trial, the units settle to a distributed value representation based on sensory inputs. This results in unit activations y_j^- and an overall weighted average value across these units denoted PV_i. In the plus phase, the unit activations (y_j^+) are clamped to represent the actual reward r (also known as PV_e). The weights (w_{ij}) into each PVi unit from sending units with plus-phase activations x_i^+ are updated using the delta rule between the two phases of PVi unit activation states:

$$\Delta w_{ij} = \epsilon\left(y_j^+ - y_j^-\right)x_i^+. \quad \textbf{[S8]}$$

This is equivalent to saying that the US/reward drives a pattern of activation over the PVi units, which then learn to activate this pattern based on sensory inputs.

The LVe and LVi layers learn in much the same way as the PVi layer (Eq. **S8**), except that the PV system filters the training of the LV values, such that they only learn from actual reward outcomes (or when reward is expected by the PV system, but is not delivered), and not when no rewards are present or expected. This condition is

$$PV_{filter} = PV_i < \theta_{min} \vee PV_e < \theta_{min} \vee PV_i > \theta_{max} \vee PV_e > \theta_{max} \quad \textbf{[S9]}$$

$$\Delta w_i = \begin{cases} \epsilon\left(y_j^+ - y_j^-\right)x_i^+ & \text{if } PV_{filter} \\ 0 & \text{otherwise} \end{cases}, \quad \textbf{[S10]}$$

where θ_{min} is a lower threshold (0.2 by default) below which negative feedback is indicated and θ_{max} is an upper threshold (0.8) above which positive feedback is indicated (otherwise, no feedback is indicated). Biologically, this filtering requires that the LV systems be driven directly by primary rewards (which is reasonable and required by the basic learning rule anyway) and that they learn from DA dips driven by high PVi expectations of reward that are not met. The only difference between the LVe and LVi systems is the learning rate ε, which is 0.05 for LVe and 0.001 for LVi. Thus, the inhibitory LVi system serves as a slowly integrating inhibitory cancellation mechanism for the rapidly adapting excitatory LVe system.

The four PV,LV distributed value representations drive the dopamine layer [ventral tegmental area/substantia nigra pars compacta (VTA/SNc)] activations in terms of the difference

between the excitatory and inhibitory terms for each. Thus, there is a PV delta and an LV delta:

$$\delta_{pv} = PV_e - PV_i \quad \textbf{[S11]}$$

$$\delta_{lv} = LV_e - LV_i. \quad \textbf{[S12]}$$

With the differences in learning rate between LVe (fast) and LVi (slow), the LV delta signal reflects recent deviations from expectations and not the raw expectations themselves, just as the PV delta reflects deviations from expectations about primary reward values. This is essential for learning to converge and stabilize when the network has mastered the task (as the results presented in the main text show). We also impose a minimum value on the LVi term of 0.1, so that there is always some expectation—this ensures that low LVe learned values result in negative deltas.

These two delta signals need to be combined to provide an overall DA delta value, as reflected in the firing of the VTA and SNc units. One sensible way of doing so is to have the PV system dominate at the time of primary rewards, whereas the LV system dominates otherwise, using the same PV-based filtering as holds in the LV learning rule (Eq. **S10**):

$$\delta = \begin{cases} \delta_{pv} & \text{if } PV_{filter} \\ \delta_{lv} & \text{otherwise} \end{cases}. \quad \textbf{[S13]}$$

It turns out that a slight variation of this where the LV always contributes works slightly better, and is what is used in this paper:

$$\delta = \delta_{lv} + \begin{cases} \delta_{pv} & \text{if } PV_{filter} \\ 0 & \text{otherwise} \end{cases}. \quad \textbf{[S14]}$$

Synaptic Depression of LV Weights. The weights into the LV units are subject to synaptic depression, which makes them sensitive to changes in stimulus inputs, and not to static, persistent activations (14). Each incoming weight has an effective weight value w^* that is subject to depression and recovery changes as follows:

$$\Delta w_i^* = R\left(w_i - w_i^*\right) - D x_i w_i, \quad \textbf{[S15]}$$

where R is the recovery parameter, D is the depression parameter, and w_i is the asymptotic weight value. For simplicity, we compute these changes at the end of every trial instead of in an online manner, using $R = 1$ and $D = 1$, which produces discrete one-trial depression and recovery.

1. O'Reilly RC, Frank MJ (2006) Making working memory work: A computational model of learning in the prefrontal cortex and basal ganglia. *Neural Comput* 18(2):283–328.
2. O'Reilly RC (2006) Biologically based computational models of high-level cognition. *Science* 314(5796):91–94.
3. Hazy TE, Frank MJ, O'Reilly RC (2006) Banishing the homunculus: Making working memory work. *Neuroscience* 139(1):105–118.
4. Hazy TE, Frank MJ, O'Reilly RC (2010) Neural mechanisms of acquired phasic dopamine responses in learning. *Neurosci Biobehav Rev* 34(5):701–720.
5. Kriete T, Noelle DC (2011) Generalisation benefits of output gating in a model of prefrontal cortex. *Connect Sci* 23:119–129.
6. O'Reilly R, Hazy T, Herd S, *Oxford Handbook of Cognitive Science*, ed Chapman S (Oxford Univ Press, Oxford).
7. Hare M, Elman JL (1995) Learning and morphological change. *Cognition* 56(1):61–98.
8. Cleeremans A, McClelland JL (1991) Learning the structure of event sequences. *J Exp Psychol Gen* 120(3):235–253.
9. Botvinick M, Plaut DC (2004) Doing without schema hierarchies: A recurrent connectionist approach to normal and impaired routine sequential action. *Psychol Rev* 111(2):395–429.
10. O'Reilly RC (1996) Biologically plausible error-driven learning using local activation differences: The generalized recirculation algorithm. *Neural Comput* 8:895–938.
11. O'Reilly RC, Munakata Y (2000) *Computational Explorations in Cognitive Neuroscience: Understanding the Mind by Simulating the Brain* (MIT Press, Cambridge, MA).
12. Aisa B, Mingus B, O'Reilly R (2008) The emergent neural modeling system. *Neural Netw* 21(8):1146–1152.
13. O'Reilly RC (2001) Generalization in interactive networks: The benefits of inhibitory competition and Hebbian learning. *Neural Comput* 13(6):1199–1241.
14. Abbott LF, Varela JA, Sen K, Nelson SB (1997) Synaptic depression and cortical gain control. *Science* 275(5297):220–224.

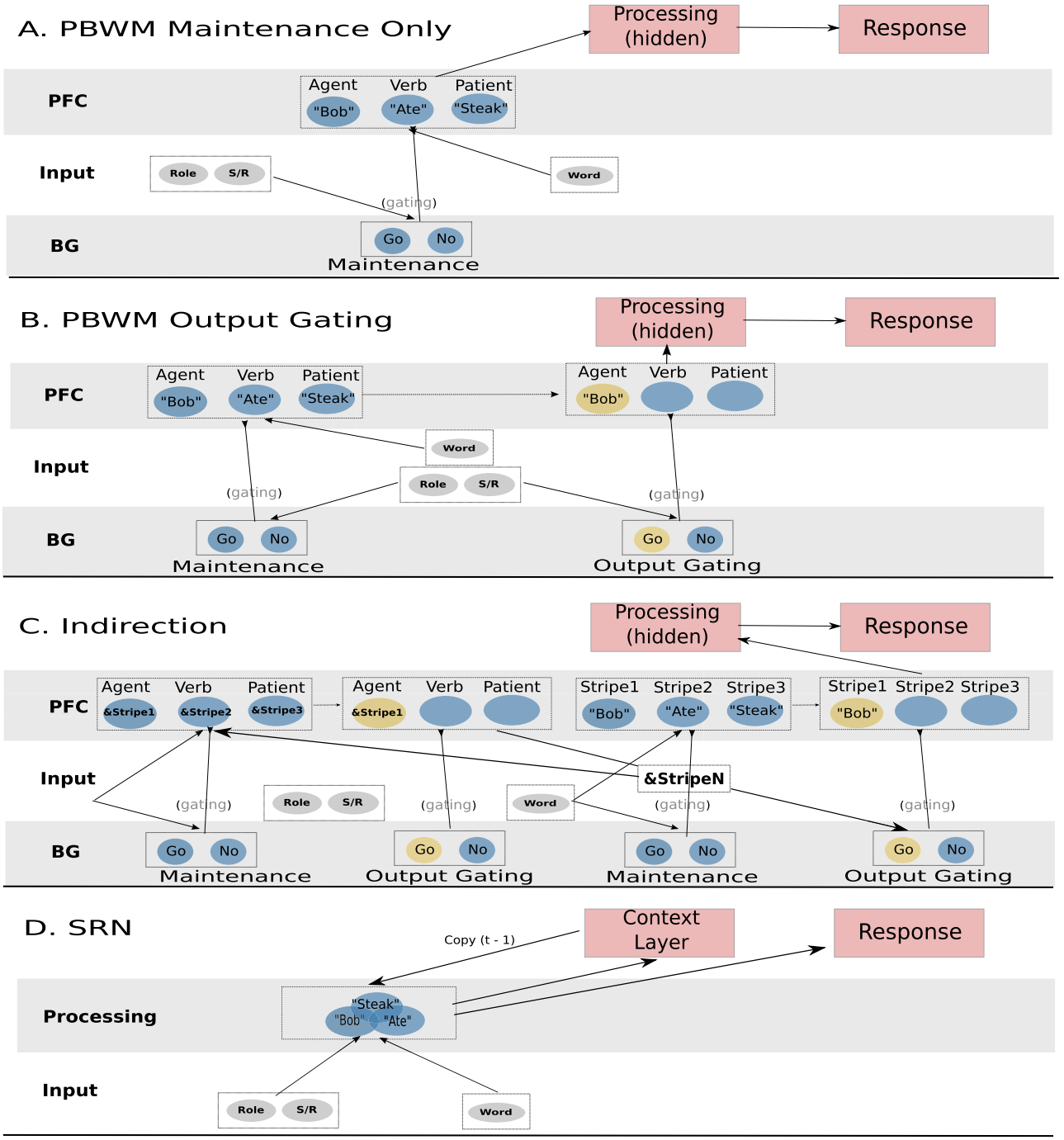

Fig. S1. Model architectures. (*A*) Standard PBWM uses a set of inputs, BG gating layers, PFC maintenance layers, a processing hidden layer, and a response layer that determines the answer the network provides on each trial. The inputs include the sentential role, the content word, and a signal indicating whether this is an encoding (store) or retrieval (recall) trial. (*B*) PBWM with output gating adds an additional set of BG/PFC stripes to determine what information should be allowed to influence downstream processing within the hidden layer and subsequent response. (*C*) The PBWM indirection model has two PBWM networks, encoding item content and role separately. When an item is gated into the content network, the stripe address information propagates to the role network, which gates that into role-specific PFC stripes. At recall, the stripe address in the role network drives output gating of the corresponding stripe in the content network. (*D*) The SRN consists of a standard three-layer back-propagation network, with a context layer that is a complete and direct copy of the activity that the hidden processing layer produced on the previous time step ($t - 1$). The copied information is fed back into the hidden layer, allowing the network to solve problems that involve the use of different pieces of information across multiple time steps.