A modeling framework for adaptive lifelong learning with transfer and savings through gating in the prefrontal cortex

Ben Tsuda[a,b,c,1], Kay M. Tye[d], Hava T. Siegelmann[e], and Terrence J. Sejnowski[a,f,g,1]

[a]Computational Neurobiology Laboratory, Salk Institute for Biological Studies, La Jolla, CA 92037; [b]Neurosciences Graduate Program, University of California San Diego, La Jolla, CA 92093; [c]Medical Scientist Training Program, University of California San Diego, La Jolla, CA 92093; [d]Systems Neuroscience Laboratory, Salk Institute for Biological Studies, La Jolla, CA 92037; [e]Biologically Inspired Neural & Dynamical Systems Laboratory, School of Computer Science, University of Massachusetts Amherst, Amherst, MA, 01003; [f]Institute for Neural Computation, University of California San Diego, La Jolla, CA 92093; and [g]Division of Biological Sciences, University of California San Diego, La Jolla, CA 92093

Contributed by Terrence J. Sejnowski, September 5, 2020 (sent for review May 14, 2020; reviewed by Matthew M. Botvinick and Robert A. Jacobs)

The prefrontal cortex encodes and stores numerous, often disparate, schemas and flexibly switches between them. Recent research on artificial neural networks trained by reinforcement learning has made it possible to model fundamental processes underlying schema encoding and storage. Yet how the brain is able to create new schemas while preserving and utilizing old schemas remains unclear. Here we propose a simple neural network framework that incorporates hierarchical gating to model the prefrontal cortex's ability to flexibly encode and use multiple disparate schemas. We show how gating naturally leads to transfer learning and robust memory savings. We then show how neuropsychological impairments observed in patients with prefrontal damage are mimicked by lesions of our network. Our architecture, which we call DynaMoE, provides a fundamental framework for how the prefrontal cortex may handle the abundance of schemas necessary to navigate the real world.

neural networks | gating | prefrontal cortex | lifelong learning | reinforcement learning

Humans and animals have evolved the ability to flexibly and dynamically adapt their behavior to suit the relevant task at hand (1). During a soccer match, at one end of the pitch, a player attempts to stop the ball from entering the net. A few moments later at the opposite end of the pitch, the same player now tries to put the ball precisely into the net. To an uninitiated viewer, such apparently contradictory behaviors in nearly identical settings may seem puzzling, yet the ease with which the player switches between these behaviors (keep ball away from net or put ball into net) highlights the ease with which we adapt our behavior to the ever-changing contexts (near own net or opposing team's net) we experience in the world. A bulk of evidence from observations of humans with prefrontal cortical lesions, neuroimaging studies, and animal experiments has indicated the importance of the prefrontal cortex (PFC) and connected regions in encoding, storing, and utilizing such context-dependent behavioral strategies, often referred to as mental schemas (2–6). Yet how the prefrontal and related areas are able to translate series of experiences in the world into coherent mental schemas, which can then be used to navigate the world, remains unknown.

Research in reinforcement learning has helped provide some insight into how the PFC may transform experiences into operational schemas (7–9). In reinforcement learning paradigms, an agent learns through trial and error, taking actions in the world and receiving feedback (10). Recent work has demonstrated how recurrent neural networks (RNNs) trained by trial-by-trial reinforcement learning can result in powerful function approximators that mimic the complex behavior of animals in experimental studies (9).

Although reinforcement learning has provided invaluable insight into mechanisms the PFC may use, it remains unclear how the PFC is able to encode multiple schemas, building on each other, without interference, and persisting so they may be accessed again in the future. The majority of models capable of solving multistrategy problems require specially curated training regimens, most often by interleaving examples of different problem types (11). Models learn successfully due to the balanced presentation of examples in training; if the training regimen is altered—for example, problem types appear in sequence rather than interleaved, as often happens in the world—the unbalanced models fail miserably (12).

Some techniques have been proposed to help models learn and remember more robustly, yet none have established how these processes may occur together in the brain. For example, continual learning techniques (e.g., refs. 13 and 14) propose selective protection of weights. Yet such techniques heavily bias networks toward internal structures that favor earlier, older experiences over newer ones and are potentially not biologically realistic (11). Other models either require explicit storage of past episodes for constant reference (15, 16), or an "oracle" to indicate when tasks are "new" (17, 18).

Experimental studies have suggested that areas within the PFC and related regions may adopt a gating-like mechanism to control the flow of information in the brain in order to support complex behaviors involving multiple schemas (19–21). Many forms of prefrontal gating have been proposed in the literature to date, including gating of sensory information (22–24), gating mechanisms to support working memory (25), and gating of

Significance

The prefrontal cortex (PFC) enables humans' ability to flexibly adapt to new environments and circumstances. Disruption of this ability is often a hallmark of prefrontal disease. Neural network models have provided tools to study how the PFC stores and uses information, yet the mechanisms underlying how the PFC is able to adapt and learn about new situations without disrupting preexisting knowledge remain unknown. We use a neural network architecture to show how hierarchical gating can naturally support adaptive learning while preserving memories from prior experience. Furthermore, we show how damage to our network model recapitulates disorders of the human PFC.

Author contributions: B.T., K.M.T., H.T.S., and T.J.S. designed research; B.T. performed research; and B.T., K.M.T., H.T.S., and T.J.S. wrote the paper.

Reviewers: M.M.B., DeepMind; and R.A.J., University of Rochester.

The authors declare no competing interest.

[1] To whom correspondence may be addressed. Email: terry@salk.edu or btsuda@salk.edu.

This article contains supporting information online at https://www.pnas.org/lookup/suppl/doi:10.1073/pnas.2009591117/-/DCSupplemental.

First published November 5, 2020.

task-relevant activity (6, 20, 21). Building off experimental findings, computational models incorporating gating have been developed for action sequences (26) and, particularly, for working memory (9, 27–31).

Furthermore, experimental and modeling work has suggested functional divisions within the PFC and neighboring areas (5, 32–35). Clinical and neuroimaging observations from patients with prefrontal lesions have strongly linked dorsolateral PFC (dlPFC) to set-shifting (5, 33, 34). Additionally, some clinical and experimental findings have indicated ventromedial PFC (vmPFC) and anterior cingulate cortex (ACC) involvement in set-shifting (5, 20, 35), while others have not (5, 34). Notably, although multiple models for working memory mechanisms have been proposed, there is a lack of models designed to investigate schema encoding and shifting, linked to functions distributed across dlPFC, vmPFC, orbitofrontal cortex (OFC), and ACC among other areas.

We hypothesized that, in addition to gating mechanisms supporting maintenance of working memory, the presence of structural gating could support manipulation and adaptation of multiple task-specific schemas in the PFC. Such a network architecture could learn multiple schemas through reinforcement and adapt to new environments without “oracle” supervision, while remaining robust against catastrophic forgetting. To investigate this, we developed a neural network framework for the PFC that mirrors the mixture of experts (MoE) class of models (36) (Fig. 1*A*). MoE networks are used widely across machine learning applications (37, 38) and have been shown to support transfer learning in combination with reinforcement learning (39–42) and Bayesian nonparametrics (17, 43–47).

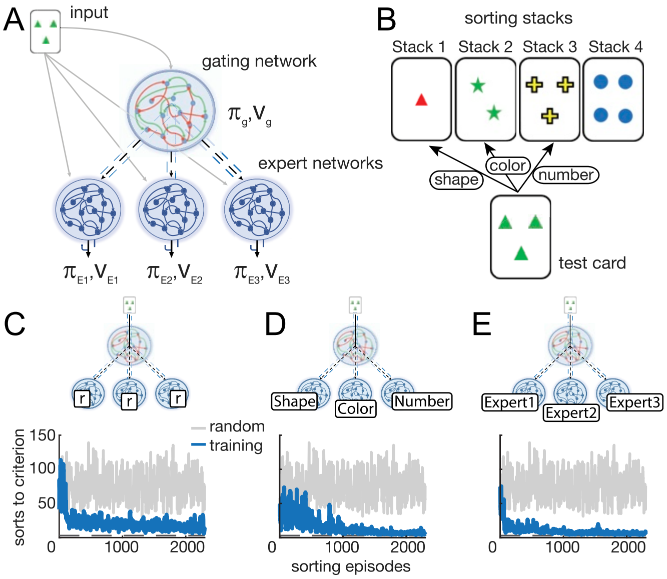

Fig. 1. DynaMoE network structure and the WCST. (*A*) The DynaMoE network is in the MoE family of networks. A gating network takes input and outputs a decision of which expert network to use (π_g) and a value estimate (v_g). The chosen expert network (e.g., E1) takes input and outputs an action to take (π_{E1})—for the WCST, in which stack to place the current card—and a value estimate (v_{E1}). (*B*) The WCST. The subject must sort the presented test card to one of four stacks by matching the relevant sort rule. The subject continues to attempt sorting test cards until achieving the termination criterion (correctly sorting a given number of cards consecutively). (*C*–*E*) MoE networks on the classic WCST. (*C*) MoE network with three experts achieves good performance quickly and slowly improves further over time. (*D*) MoE network with pretrained experts on the sort rules also learns quickly, reaching near-perfect performance faster. (*E*) DynaMoE network pretrained sequentially on the sort rules learns rapidly and reaches near-perfect performance fastest. In all plots, blue traces are from networks during training, and gray traces are random behavior for reference. Dark gray dashed line in line plots shows the minimum sorts to criterion.

Using our network model, we demonstrate how structural gating naturally leads to transfer learning as new scenarios are encountered and schemas are encoded. Furthermore, we show how our network adaptively learns and, due to its architecture, demonstrates robust memory savings for past experiences. We implemented lesions to our model to study how functional components may become disrupted in disease and found that the lesions recapitulated specific neuropsychological impairments observed in patients. Our framework provides a basis for how the PFC and related areas may encode, store, and access multiple schemas.

Results

To demonstrate the properties of our framework, we chose the Wisconsin Card Sorting Task (WCST), a neuropsychological assessment of PFC function commonly used in clinic (2, 5, 48, 49). In the WCST, a subject is required to sequentially sort cards according to one of three possible sorting rules: shape, color, or number (Fig. 1*B*; see *Materials and Methods* for full description). The sorting rule is not explicitly given, but rather must be discovered through trial and error. After each attempted card sort, the subject receives feedback as to whether the sort was correct or incorrect. After a set number of correct card sorts in a row, the sort rule is changed without signal, requiring the subject to adapt behavior accordingly (5, 49). Performance can be measured by the number of attempted card sorts until the episode termination criterion is achieved (“sorts to criterion”; three correct sorts in a row in our simulations), with fewer attempted sorts representing superior performance.

The WCST requires the PFC’s abilities to encode, store, and access multiple schemas. The task requires a recognition of “rule scenarios” (a form of “set learning”) and flexible adaptation through reinforcement signals to shift with changing rules. Patients with prefrontal damage often have difficulty with this task, with some stereotypically making perseveration errors, indicating an inability to switch rules when given reinforcement (4, 5).

Although many models are able to solve the classic WCST (Fig. 1 *C*–*E* and *SI Appendix*, Fig. S6), we sought to use the WCST to help uncover the mechanisms by which the PFC is able to learn and remember multiple schemas in the absence of curated training or supervision. The framework we develop can be generalized to many similar tasks.

The Model: Dynamic Mixture of Experts. Our neural network architecture, which we call Dynamic Mixture of Experts (DynaMoE), combines RNNs used previously to model the function of PFC (9) with the MoE architecture (36), and introduces two features that enable flexible lifelong learning of disparate schemas: a progressive learning process and repeated focal retuning. Our MoE design uses two specialized networks: a gating network that receives input from the external environment and outputs a decision of which expert network to use and expert networks that take external input and output an action decision—the card stack to sort the current card in the WCST (Fig. 1*A*). To capture the complex dynamics of the PFC, we modeled both the gating network and expert networks as RNNs (long short-term memory networks [LSTMs] in our implementation). While other architectures have been used in MoE networks, recent work by Wang et al. (9) demonstrated the ability of RNNs to reliably store biologically realistic function approximators when trained by reinforcement learning that mimic animal behaviors.

Using this network architecture, we first introduce a progressive learning process (Fig. 2*B*). Our neural network begins as a gating network with a single expert network (Fig. 2*A*). As it gathers experience in the world, it learns in series of

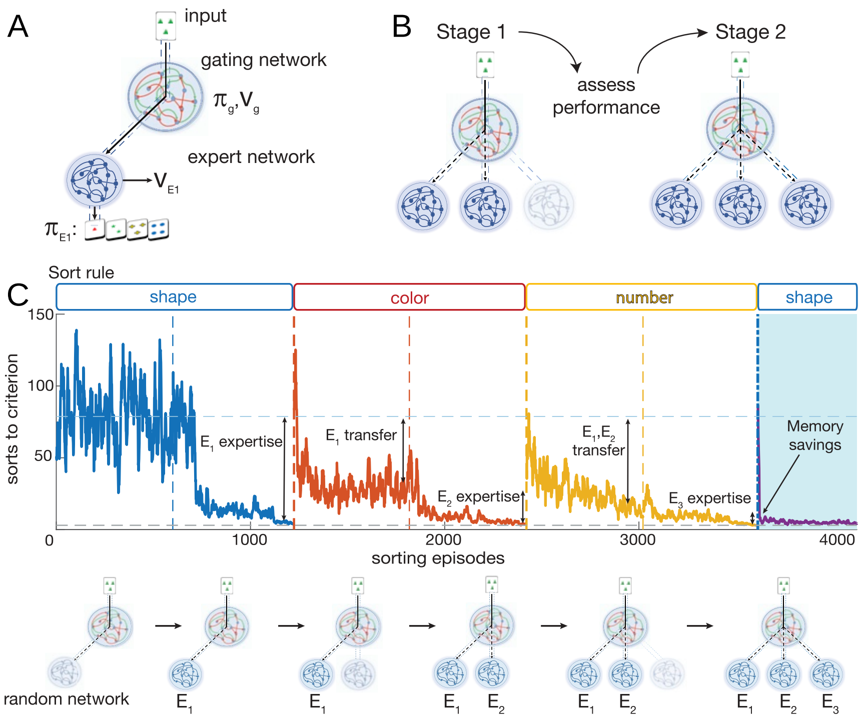

Fig. 2. Training of a DynaMoE network. (*A*) DynaMoE begins with a gating network and a single expert network, E_1. Both the gating and expert networks train by reinforcement learning, outputing a predicted value (v_g and v_{E1}) and policy (π_g and π_{E1}). (*B*) DynaMoE's two-step learning process. In stage 1, the gating network retunes to attempt to solve the task at hand with current experts; if performance is unsatisfactory, the network adds an additional expert in stage 2 which preferentially trains on tasks that could not be solved by other experts. (*C*) A sample training trajectory of a DynaMoE network presented with sequential periods of sorting rules in the WCST. A randomly initialized DynaMoE begins in the shape sorting scenario. First, the gating network is tuned alone. In the second step of learning, the first expert network, E_1, is trained (second half of the blue curve). The sort rule is switched to color (red curve), and the same two-step training process is repeated, followed by the number sort rule (yellow). The improved performance between the first and second stages of training in each sort rule scenario results from expert training. The improved performance from gate retuning results from transfer learning from past experts and increased network capacity. The purple curve shows how DynaMoE rapidly "remembers" past experience due to robust memory savings. The schematic below the graph shows the progression of the DynaMoE network as it experiences the scenarios. Each stage of training above was done for 625 sorting episodes to display convergent learning behavior.

two-step tunings. When the neural network experiences a scenario (e.g., a series of card sorts in the WCST), it first tunes its gating network to attempt to solve the problem by optimally delegating to expert networks, much as a traditional MoE model would. If some combination of expert actions results in satisfactory performance, no further learning is necessary. If, however, the experiences are sufficiently novel such that no combination of the current expert networks' outputs can solve the task fully, the network then brings online a latent untrained expert (Fig. 2 *B* and *C*). The new expert is trained along with the gating network, resulting in a new expert that handles those problems that could not be solved with previous experts. Importantly, this training procedure is agnostic to the order of training scenarios presented and does not require any supervision. Instead, given only the desired performance criteria (e.g., level of accuracy) and limit of training duration per step (how long to try to solve with current experts), our neural network dynamically tunes and grows to fit the needs of any scenario it encounters (Fig. 2*C*). The learning curves in Fig. 2*C* reveal two prominent features. First, the speed of learning is successively faster for the second (55.8% improvement in performance on color sorting with only gate retuning; "E_1 transfer" in Fig. 2*C*) and third (77.5% improvement in performance on number sorting with only gate retuning; "E_1, E_2 transfer" in Fig. 2*C*) scenarios, which is a form of transfer learning. Second, after learning all three scenarios, relearning the first scenario (shape sorting) was rapid, a form of memory savings.

The second feature is repeated retuning of the gating network. Training standard neural networks on new tasks leads to overwriting of network parameters, resulting in catastrophic forgetting (12) (*SI Appendix*, Fig. S1). By decomposing a single network into a hierarchy of gating and expert networks, we are able to separate the memory of the neural network into the "decision strategy" (gate), which maps between inputs and experts, and the "action strategies" (experts), which map from input to actions. The hierarchical separation enables repurposing expertise through combinatorial use of previously acquired experts and a natural means to confine memory overwriting to a small portion of the neural network that can be easily recovered through repeated retuning. Experimental evidence suggests areas of the brain similarly support different levels of plasticity, with regions higher in hierarchical structures exhibiting increased plasticity (50). The resulting network exhibits memory savings (51) that remain robust to new learning and lead to rapid "remembering" rather than relearning from scratch (compare purple curve in blue shaded region in Fig. 2*C* and *SI Appendix*, Fig. S1).

We found that the implementation of these two features in a hierarchical MoE composed of RNNs results in an architecture that organically learns by reinforcement relative to past experiences and preserves memory savings of past experiences, reminiscent of PFC. Importantly, when presented with the classic interleaved WCST, our network (Fig. 1*E*) learns just as fast or faster than standard RNNs and traditional MoE networks (Fig. 1 *C* and *D* and *SI Appendix*, Fig. S6). We next sought to understand how our dynamic architecture enabled the observed transfer and savings.

Transfer Learning: DynaMoE Seeded with Pretrained Experts. To probe how the DynaMoE network implements transfer learning, we first created an easily interpretable scenario in which two expert networks were separately pretrained on specific rule sets of the WCST, one on shape sorting (E_{shape}) and another on color sorting (E_{color}) (Fig. 3*A*). We then seeded a DynaMoE network with the two pretrained experts and a randomly initialized untrained expert, and introduced it to the third rule, number sorting, and studied its behavior.

Reflexively, one may speculate that a DynaMoE network with a shape sorting expert, a color sorting expert, and a random network would perform no better on number sorting than with only a random network, since number sorting is seemingly independent of shape or color. Somewhat surprisingly, we found this was not the case. After tuning the gating network, the DynaMoE network with pretrained experts performed drastically better than without them, nearly reaching perfect performance (Fig. 3*B*). We found that the gating network learned to identify cards for which the shape or color sort matched the correct number sort, and allocate them to the corresponding expert. For example, a card with one blue triangle would be sorted to stack 1 in both the shape (triangle) and number (one) scenarios ("shape-match-number"). Similarly, some cards, for example the card with one red circle, would be sorted to stack 1 in both the color (red) and number (one) scenarios ("color-match-number"). The gating network learned to map these cards to E_{shape} and E_{color} to perform correct card sorts in the number rule (Fig. 3*C*). Only cards for which the number sort did not match the shape or color sort ("no-match-number") were unsolvable with either E_{shape} or E_{color}; for these cards, the gating network used a mixture of the shape, color, and untrained expert networks (Fig. 3 *C*, *Right*), since no expert could reliably sort these cards correctly. The network had learned to exploit a hidden intrinsic symmetry between the features in the task to enhance performance.

Consequently, when the new expert network was brought online and trained (Fig. 3*D*), the gating network allocated a large proportion of "no-match-number" cards to the new expert (E_3) (Fig. 3 *F*, *Right*). E_3's expertise thus became number sorting cards that do not match shape or color sorts. Interestingly, this demonstrates a form of transfer learning. The gating network learned to use existent experts to find partial solutions for new problems, leaving unsolvable parts of the problem to be disproportionately allocated to the new expert in the second step of training. New experts thus learn relative to old experts, building on prior knowledge and experience.

In practice, the expertise of E_3 varied between number sorting predominantly "no-match-number" cards and all cards. This likely reflects a trade-off between the complexity of mapping functions the gating and expert networks must learn. In the number sorting scenario, the gating network can learn to map each card type to the appropriate expert or the simpler function of mapping all cards to E_3; E_3, in turn, learns to number sort only "no-match-number" cards or all cards. This highlights a trade-off that occurs in biological systems like the brain. We may be able to solve a new problem by piecing together numerous tiny bits of completely disparate strategies, but as complexity of the mapping function increases, at some point, it becomes more efficient to simply learn a separate strategy for the new problem, allocating dedicated memory for it.

We found that, in the first stage of training, tuning of the gating network consistently led to a mapping function that allocated the vast majority of "shape-match-number" cards to E_{shape} and "color-match-number" cards to E_{color} (Fig. 3*C*). "No-match-number" cards were allocated between all three experts. After the second stage of training in which both the gating network

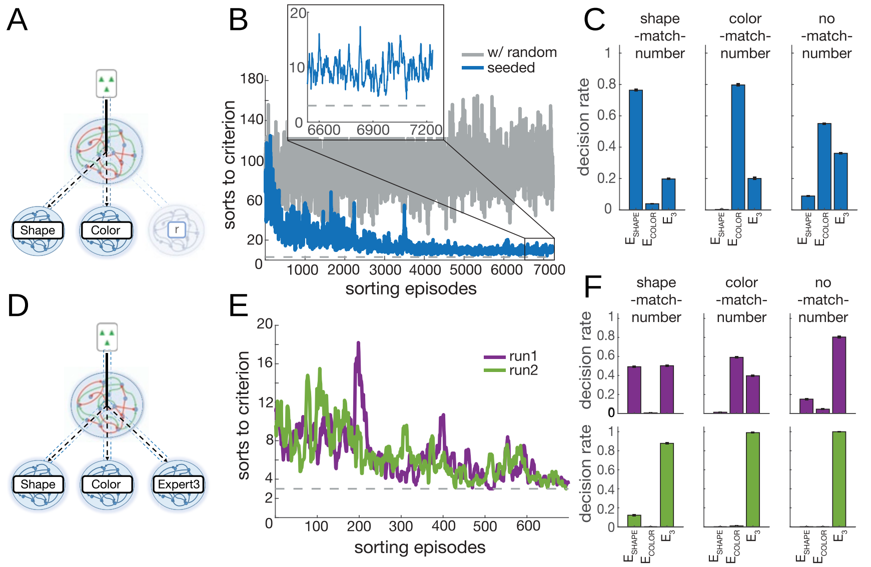

Fig. 3. Transfer learning with a seeded DynaMoE network. (*A*) A DynaMoE network seeded with pretrained shape and color experts and a randomly initialized untrained network. (*B*) The DynaMoE network from *A* achieves near-perfect performance in number sorting when only the gating network is trained (blue) in contrast to a network with only an untrained expert (gray). *Inset* shows that performance of the seeded network does not reach the minimum sorts to criterion (gray dash) without training the third expert network. (*C*) The proportion of cards allocated to each expert network after the training in *B* in three different subsets of the number sort rule: shape-match-number (*Left*), color-match-number (*Center*), no-match-number (*Right*). (*D*) Seeded DynaMoE network with trained Expert3 network. (*E*) Performance (measured by decrease in sorts to criterion) of two example training runs from the same initial network (*A*–*C*) that result in different end behavior (see *F*). The performances of both runs improve from DynaMoE with an untrained expert (*B, Inset*) and are indistinguishable from each other. (*F*) (*Top*) Proportion of experts used in same subsets of number sort rule as in C [shape-match-number (*Left*), color-match-number (*Center*), no-match-number (*Right*)] for an example run (run 1). A varying decision rate for experts is used depending on the scenario subset. (*Bottom*) Proportion of experts used for a second example run (run 2). The new expert (E_3) is used regardless of subset of number sort rule. See *SI Appendix*, Fig. S2 for all 10 runs. Error bars are SD over 1,000 test episodes after training. Absence of bar indicates zero selections of the given expert during testing.

and E_3 are trained, we found that the "no-match-number" cards were almost entirely allocated to E_3, as expected (Fig. 3 *F*, *Right*). We found that usage of experts for "shape-match-number" and "color-match-number" cards varied across different training runs (Fig. 3*F* and *SI Appendix*, Fig. S2). To see how often training led to different expert network decision rates, we ran the second stage of training 10 times from the same initial network that had gone through the first stage of training. Usage of the relevant pretrained expert (e.g., E_{shape} for "shape-match-number" cards) ranged from as much as 65% to as low as 1%, representing end behavior in which E_{shape} and E_{color} continued to be used or in which E_3 was used almost exclusively (run1 and run2, respectively, in Fig. 3*F*). The nonrelevant expert (e.g., E_{color} for "shape-match-number" cards) was rarely ever used (0 to 5%). This shows that, while DynaMoE networks support pure transfer, the degree of transfer learning implemented depends on network capacity, learning efficiency, and the stochastic nature of learning. All networks achieved the same near-perfect performance stop criteria within similar numbers of sorting episodes (Fig. 3*E*; see *Materials and Methods*).

Transfer Learning: Organic Case. To probe how a DynaMoE network naturally implements the transfer learning described in the previous section, we trained 10 DynaMoE networks independently from scratch through sequential experiences of the different rules of the WCST. Each network began with an untrained gating network and expert network (E_1). The DynaMoE networks were then trained on shape followed by color and then number sorting, adding a new expert in each new sort rule scenario (see *Materials and Methods*).

As expected from the result in the previous section, we found that the expert networks were not pure rule sorters but rather had learned an expertise in a mixture of rule types relative to the other experts. For each sort rule scenario, one expert network was used preferentially (E_1 for the first rule experienced, E_2 for the second, etc.), which we refer to as the "dominant expert network" for that sort rule scenario. To quantify the degree of transfer learning utilized, we measured the usage of all three expert networks in the different sorting scenarios. For each sort rule scenario, the gating network was retuned until "expert performance" was once again attained. We then measured the usage of each of the nondominant expert networks with respect to usage of the dominant expert network. Although the magnitude of relative usage varied between independent runs, a consistent pattern emerged. In the shape sort scenario—the first scenario encountered with only E_1—E_2 and E_3 were used very little or never (Fig. 4 *A*, *Left*). For the second scenario encountered—color sort scenario—E_1 was used a small amount, and E_3 was never used (Fig. 4 *A*, *Center*). Finally, for the third scenario—number sort scenario—E_1 and E_2 were used a small but significant portion of the time (Fig. 4 *A*, *Right*).

This trend of increased usage of experts that were present during the learning of a rule compared to experts added afterward strongly indicates transfer learning as the DynaMoE network encountered new scenarios (Fig. 4*B*). Newly added experts predominantly trained on examples that the other experts could not solve. Thus, when the gating network was retuned to solve a scenario later, it continued to use the previously added experts. In contrast, if an expert was added after the learning of a scenario, all of the knowledge to solve the scenario was already contained in the existent experts, so the expert added after learning was rarely used. This shows that new experts were trained relative to knowledge contained by existent experts. Furthermore, while the aggregated expert use percentages clearly show the presence of transfer learning, they mute the degree of transfer learning adopted by some individual networks (*SI Appendix*, Fig. S3).

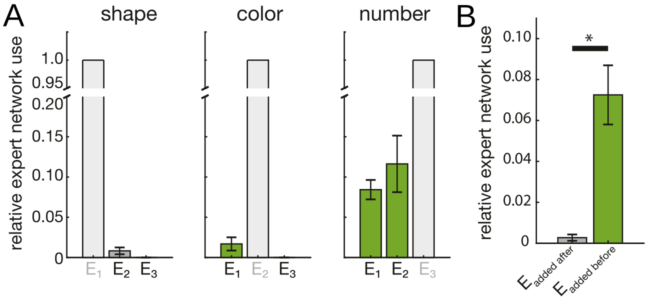

Fig. 4. Transfer learning in an unseeded DynaMoE network. (*A*) The relative use of each expert network in each sort rule [shape (*Left*), color (*Center*), number (*Right*)] normalized to the dominant expert for the sort rule from 10 independent DynaMoE networks trained in a sequential training regimen (see *Materials and Methods*). The greyed out expert network label with lightest gray bar of value 1 indicates the dominant expert network for each sort rule. Darker gray bars indicate usage of experts that were not present during initial training of the given sort rule (e.g., E_2 and E_3 for the shape rule). Green bars indicate experts that were present during initial training of the given sort rule (e.g., E_1 and E_2 for the number rule). Absence of a bar indicates the given expert was never used. Error bar is SEM over 10 independent runs (*SI Appendix*, Fig. S3). (*B*) Aggregated bar plot from *A* grouped by whether the expert was added before or after initial training on the rule. Use of experts present during initial training of a rule indicates transfer learning (green bar), while use of experts not present during initial training indicates nontransfer usage (gray bar). The usage of experts added before was significantly higher (*$P < 0.01$; $P = 1.16e-05$, Student's *t* test) than that of experts added after initial training on a rule. Error bar is SEM.

Robust Memory Savings. A critical feature of the PFC is the ability to retain knowledge from past experiences in the form of learned connectivity patterns (52). Many neural network models suffer from castastrophic forgetting (12), overwriting information from previous experiences. Put in terms of network parameters, when such networks retune weights to solve new problems, they move to an optimal point in weight space for the current task which can be far away from the optimal space for previous tasks.

In contrast, DynaMoE networks, like the PFC, maintain near-optimal configuration for previously experienced scenarios, exhibiting "memory savings" (51). The hierarchical architecture of DynaMoE networks confines memory loss to a small flexible portion of the network: the gating network. If a scenario has been encountered before, retuning the gating network to optimal configuration is rapid, requiring only a small number of reinforcement episodes (Fig. 5 *A* and *B* and *SI Appendix*, Fig. S5). Retuning the gating network requires much less movement in weight space compared to standard RNNs, since tuning is confined to only the gating network. This is in stark contrast to standard neural networks which can require complete retraining (Fig. 5*C* and *D* and *SI Appendix*, Fig. S4).

To measure the memory savings of DynaMoE networks, we sequentially trained networks with identical presentations of, first, shape, then color, then number sorting scenarios (see *Materials and Methods*). We then tested how many sorting episodes of reinforcement were required for the network to regain expertise in the first sorting rule it experienced (shape). As Fig. 5 *A* and *B* show, DynaMoE networks required 78% fewer episodes to regain expertise than standard RNNs ($p = 2.49e-11$, Student's *t* test). The number of episodes required to remember was drastically fewer than when they first learned the rule, whereas standard RNNs improved only slightly compared to when they first learned the rule (*SI Appendix*, Figs. S1, S5, and S6). While standard RNNs nearly completely overwrote the information learned through initial training, DynaMoE networks preserved their memory and only required brief reinforcement for the gating network to remember how to allocate cards to experts.

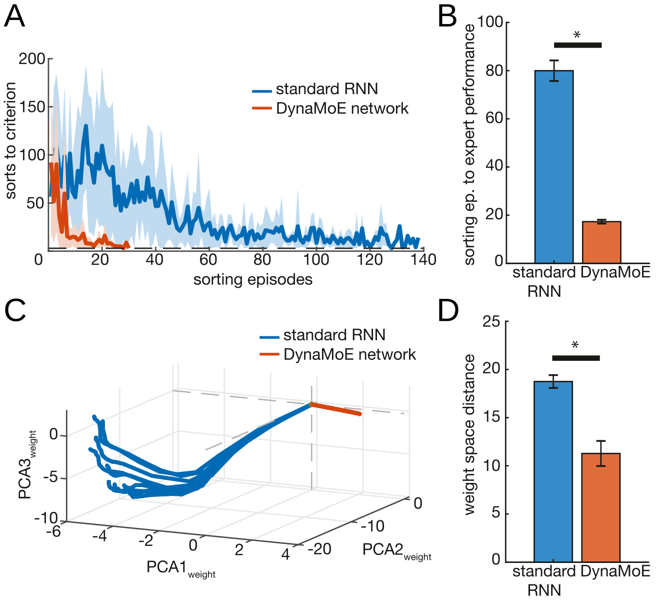

Fig. 5. Robust memory savings of DynaMoE. (*A*) Example of performance over sorting episodes of retraining of standard RNN (blue) and DynaMoE network (orange) on a previously encountered task. Shading indicates SD over 10 independent retraining runs. (*B*) Average number of sorting episodes required until expert performance for standard RNN and DynaMoE networks over 10 independently trained networks of each type. DynaMoE networks require 78% fewer sorting episodes (abbreviated in barplot as sorting ep.) to remember (*$P < 0.01$; $P = 2.49e - 11$, Student's *t* test). (*C*) Visualization of top three principal components of weight space for 10 relearning/remembering trajectories of an example standard RNN and a DynaMoE network trained sequentially. (*D*) Euclidean distance between networks before and after remembering previously learned rule in full weight space (average of 10 independently trained networks of each type). DynaMoE network moves 40% less in weight space compared to the standard RNN (*$P < 0.01$; $P = 7.43e - 05$, Student's *t* test). Error bars are SEM.

To measure the weight changes required to regain optimal performance, we measured the distance in weight space each of the networks traversed when remembering the shape sort rule after sequential training. DynaMoE networks traversed 40% less distance in weight space to reach optimal performance compared to standard RNNs ($p = 7.43e - 05$, Student's *t* test; Fig. 5 *C* and *D* and *SI Appendix*, Fig. S4). Even after sequential training, DynaMoE networks remain relatively close in weight space to the optimal performance configurations on all previously experienced tasks. In contrast, standard RNNs moved far from their initial optimal point in weight space for the shape scenario, resulting in movement of nearly equal distance when relearning the shape scenario as when initially learned (*SI Appendix*, Fig. S4).

Lesions of DynaMoE Cause PFC Lesion-like Impairments. The DynaMoE framework provides an opportunity to understand how disruptions to specific functional aspects of the PFC and related areas can lead to different neuropsychological impairments observed in clinical cases. Numerous clinical and neuroimaging studies have indicated regional specialization within the PFC, yet evidence from human studies is invariably messy, involving overlapping brain regions and varying degrees of impairment in different aspects of tasks (2, 5, 53). Our framework enables targeted disruption of specific functional components of our network that may help clarify the underlying organization of the human PFC. The WCST has served as a standard clinical assessment to evaluate PFC impairment (5), making it an ideal task with which to analyze functional consequences of various lesion types.

To assess how lesions of our network architecture could result in behavioral impairments, we damaged specific regions of the gating network of our architecture. Importantly, in our lesion studies, the expert networks were unperturbed, leaving available the action strategies to perfectly perform the task. This characteristic is often seen in patients with prefrontal damage: Although they have difficulty with the full WCST, if explicitly told which sort rule to use, patients are often fully capable (5, 54). We first trained DynaMoE networks on each rule type, and then the classic interleaved WCST (*SI Appendix*, Fig. S6*E*; see *Materials and Methods*). We then lesioned the gating network and performed testing on the classic WCST to assess changes in performance and behavior.

Lesions were targeted to five different regions within the gating network (Fig. 6*A*): inputs to the network (red region 1 in Fig. 6*A*)—ablation of reward feedback (L1), action feedback (L2), or both (L3); internal network dynamics (region 2)—ablation of varying numbers of synaptic connections to the "forget" gate of the LSTM (L4$_{0.5}$ and L4$_{0.9}$ were 50% and 90% ablations of synaptic connections, respectively; see *SI Appendix*, Fig. S7 *A* and *B* for full range); output of the network (region 3; L5); and areas downstream of the network—ablation of synaptic connections to the units that determined which expert network to use (region 4; L6), to the unit that estimated value (region 5; L8), or both (L7; *SI Appendix*, Fig. S7 *C* and *D* for full range). Since lesions could potentially have differential effects depending on the specific disruptions incurred, we performed each lesion 10 times and measured the average effect.

We found that different lesion types resulted in different degrees of impairment, ranging from no change in error rate (e.g., L8$_{0.9}$, $p = 0.3664$, Student's *t* test) to 10.5-fold more errors

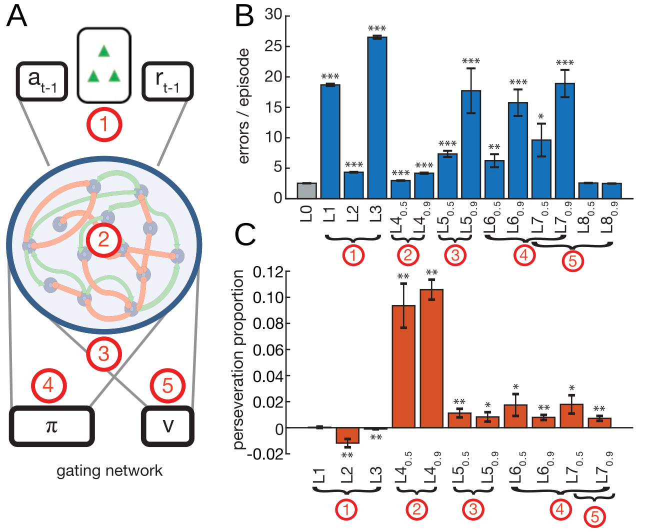

Fig. 6. Different lesion-induced error modes of DynaMoE gating networks. (*A*) Map of lesioned regions in DynaMoE's gating network. Three lesions of input (region 1) were done (L1 to L3), one lesion of the network dynamics (region 2; L4), one lesion of network output (region 3; L5), one lesion of decision determination (region 4; L6 and L7), one lesion of value determination (region 5; L7 and L8). (*B*) Average number of errors per episode for each lesion type. L1 is ablation of reward feedback from previous trial; L2 is ablation of action feedback from previous trial; L3 is simultaneous L1 and L2 lesions; L4$_{0.5}$ is ablation of 50% of connections to forget gate of network, and L4$_{0.9}$ is ablation of 90%; L5 is ablation of output units; L6 is ablation of connections to decision units (π); L8 is ablation of connections to value unit (v); L7 is simultaneous L6 and L8. Asterisk indicates significant difference from no lesion (L0) (Student's *t* test; *$P < 0.05$; **$P < 0.01$; ***$P < 0.001$) (*C*) Proportion of increase in errors that were perseveration errors for lesions that caused significant increase in errors. Asterisk indicates confidence interval (CI) excluding zero (*: 95% CI; **: 99% CI). All error bars are SEM.

(L3, $p = 2.1268e - 25$, Student's t test) than before the lesion (Fig. 6*B* and *SI Appendix*, Table S1). Since perseverative errors are a signature of some prefrontal lesions, particularly associated with dlPFC impairment, we measured the proportion of the increase in error rate that was due to perseverative errors ("perseveration proportion"; a negative value indicates a decrease in perseverative errors relative to no lesion). Fig. 6*C* shows the variability in perseveration proportion for the lesions that caused a significant increase in error rate. To ensure that the error profiles we observed were not an artifact due to the presence of ambiguous cards for which sorting by multiple rules could result in the same action decision, we also tested the lesioned networks with the card deck from the Modified WCST (MWCST) (5, 55), in which all ambiguous cards are removed, and found qualitatively similar results (*SI Appendix*, Fig. S8).

The neuropsychological impairment profiles defined by increase in total error and perseveration proportion reveal different lesion-specific error modes that mirror the different error modes observed from patients across the range of prefrontal lesions (Fig. 6 and *SI Appendix*, Table S1). Overall, lesions grouped qualitatively into three categories: lesions that caused often substantially increased total error rate (1.72× to 10.51×), a small proportion of which were perseverative errors (−1.18 to 1.79%) (regions 1, 3, and 4; L1 to L3, L5 to L7); lesions that caused a small but significant increase in total errors (1.18× to 1.65×), a large proportion of which were perseverative errors (9.36 to 10.59%) (region 2; L4); and lesions that caused no change in error rate (region 5; L8).

Our lesion results provide a roadmap with which to interpret and understand the variety of error modes observed in human patients with prefrontal damage due to trauma or disease. While the PFC as a whole has been definitively linked to set-shifting and cognitive flexibility, localization of functional components to specific subregions remains unclear. Lesions throughout the prefrontal areas have been associated with impairments observed in the WCST, ranging from no change in error rate to large increases in perseverative and nonperseverative error rates similar to the range of behavioral outcomes resulting from our lesions (5). Canonically, although with mixed evidence, impairment of the dlPFC is associated with increased error rate on the WCST, particularly perseveration errors. Our lesion study indicates this behavioral phenotype may be due to impairment of gating network dynamics, suggesting the dlPFC may contribute to a gating-like mechanism within the PFC. Interestingly, the lesions of components inside the gating network's recurrent connections that caused a specific increase in perseverations only weakly increased the total error rate. In contrast, lesions to input components led to a large increase in total error, while perseverations increased relatively less. These contrasting neuropsychological impairments highlight a double dissociation of neural components underlying perseveration errors and total errors, a characteristic also observed in patients.

Discussion

In this paper, we propose a framework for how the PFC may encode, store, and access multiple schemas from experiences in the world. Like the PFC, the DynaMoE neural network is agnostic to training regimen and does not require "oracle" supervision. We showed how the hierarchical architecture of DynaMoE naturally leads to progressive learning, building on past knowledge. We then demonstrated how DynaMoE networks reliably store memory savings for past experiences, requiring only brief gate retuning to remember. Finally, we showed how lesions to specific functional components of the DynaMoE network result in different error modes in the WCST, analogous to the error modes described for patients with different forms and severity of prefrontal damage.

The parallels seen between the DynaMoE network and the PFC and related areas encourages investigation into the extent to which these two systems recapitulate each other. Perhaps most poignantly, this comparison puts forth the hypothesis that the PFC may incorporate a gating system that is tuned to optimally combine knowledge from past experiences to handle problems as they are encountered. Some studies have provided evidence for such a functional architecture in the brain (56), and prefrontal cortical areas in particular (6, 19–21). In our model, each "unit" within our recurrent networks represents a population of neurons, and, as such, inputs to the network were represented by populations of neurons, each of which receives multiple inputs and responds with mixed selectivity as has been observed in PFC (57, 58). Our model also suggests that the neural representations of different contextual "rules" can distributed across multiple overlapping subpopulations. In this way, our model adopts a hierarchy of distributed representations in which overlapping neural subpopulations support distributed representations of both lower-level sensory information (inputs) and higher-level abstract information (context-dependent rules). Experimental investigations that compare the diverse neural population activities of DynaMoE networks to that in relevant prefrontal cortical areas will be fruitful in supporting or refuting our framework. Studies in nonhuman primates with a WCST analog (6) and related task (20) have reported specific neural activity patterns associated with set-shifting, suggesting the possibility of direct comparisons to our model; a topic we are currently exploring.

Our model's feature of adaptive growth by adding new expert networks represents the recruitment of neural subpopulations for new learning. Both the assessment of performance and recruitment of new subpopulation bring up interesting parallels in neurophysiological studies (Fig. 2*B*). Error-related negativity and positivity signals are well-known phenomena that are thought to relate to self-assessment of performance, preceding a switch to an alternative strategy (59). Research on neuromodulation, neuropeptides, and metaplasticity provides evidence for spatially and temporally regulated plasticity differences in neural subpopulations (60–64), potentially supporting a dynamic, regionally selective learning system like DynaMoE. Elaborating DynaMoE to explicitly model these processes will help untangle how these processes may interact to support behavior and learning.

The DynaMoE model suggests functional relationships underlying organization in the PFC and related areas. The gating network in the model is reflective of the set-shifting functions that are closely linked to activity in dlPFC. Our lesions studies support this association, showing a relative increase in perseveration errors with disruption of the gating network internal dynamics, similar to dlPFC lesions' association with increased perseveration errors in patients (33, 34) and increased dlPFC activity observed in patients during rule shifts (2, 5, 48). However, the gating network in our model likely incorporates elements that are distributed anatomically, including processing of error signals attributed to ACC (20, 65). Other elements of our model may correspond to adjunct prefrontal regions. For example, value tracking functions attributed to OFC—medial OFC in particular (66)—may correspond to the value units in both the gating and expert networks (67). The expert networks in our model correspond to functional elements spanning regions of PFC (perhaps downstream of dlPFC) to premotor cortex, culminating in an action decision sent to the motor system to execute in the environment. The different expert networks then represent different effector pathways originating from the gating network signal and extending to the premotor cortex. These "cognitive maps" representing different context-dependent behavioral strategies are characteristic of areas of OFC (68), perhaps lateral OFC in particular (66). Both the gating network and the effector expert networks also encapsulate cortical–thalamic–basal ganglia loops, routing and processing sensory information and reward information to and from areas of PFC and ACC, nuclei of the thalamus, and areas of the striatum and other basal ganglia regions (7). Further comparison between individual functional elements in our model (e.g., expert networks and OFC) will be helpful in studying the relationship between subregions of the PFC.

Our model also can be integrated with the anterior–posterior cascade model of the frontal regions as proposed by Koechlin and Summerfield (69). From this perspective, the gating network of our model corresponds to the more abstract contextual information in the anterior regions of lateral PFC, while the expert networks correspond to more immediate context sensory processing, taking external input and mapping to an output action. Our model layers into the anterior–posterior cascade scheme, providing a mechanism for encoding and flexible usage with transfer and savings.

Several computational models have provided strong support for a hierarchical organization in the PFC (31, 69–71). Combining our model with insights from other models will likely yield further insights into the architecture and functions of the prefrontal areas. Together with Koechlin and Summerfield's (69) cascade model, Botvinick's (71) model of Fuster's hierarchy suggests that added layers to a hierarchical structure lead to organizing principles that map to higher levels of abstraction and context. Likewise, in our model, the gating network sits above the expert networks, processing the higher-level context to choose which expert to use. Understanding how the processing of higher and lower levels of the task self-organize within the hierarchical structure, that is, how the division of labor between the levels is determined, is a fascinating area of future research we are pursuing. Integrating the Hierarchical Error Representation model's (70) hierarchical propagation of errors into DynaMoE is also a promising direction, particularly in understanding the roles and relationship of dlPFC and ACC.

Our model also relates closely to previous computational models investigating the emergence of hierarchical rule-like representations (30, 35). The model proposed by Donoso et al. (35) to study the medial (vmPFC-perigenual ACC, dorsal ACC, ventral striatum) and lateral (frontopolar cortex, middle lateral PFC) tracks' contribution to strategy inference shares features with DynaMoE in storing multiple strategies and allowing combinatorial use. Our model further posits that a part of PFC, likely within dlPFC and perhaps spanning to parts of ACC, supports nonlinear and dynamically evolving combinations of stored strategies, enabling more powerful transfer learning and reducing memory requirements. To gain a deeper understanding of how multischema inference is done in the prefrontal regions, further functional MRI studies elaborating on the paradigm used by Donoso et al. (35) will be important to compare features of their model to those of the DynaMoE model.

Our lesion studies motivate further investigation of functional specialization in the PFC through comparison of our framework and clinical, experimental, and neuroimaging studies (2, 5). Clinical and experimental studies have yielded unclear and sometimes contradictory findings due to the anatomical inseparability of PFC functions (5). Although studies have most clearly linked perseverative abnormalities to dlPFC, similar abnormalities can be observed in compulsive behaviors like alcohol consumption, which depend on strategies generated by the mPFC that predict individual behavioral patterns (72). Our model provides full access to the underlying structure, enabling targeted studies to use as a reference for interpreting human and animal studies. Further comparison of our framework with in-depth phenotypic analyses across various tasks may help us understand the functional organization of the PFC and the consequences of disruptions due to trauma and disease.

Our lesion analysis also motivates future studies on adaptation to lesions. In the present study, we focused on lesions after learning was complete, since most clinical case reports describe testing of patients after acute injury. In clinic, it is also important to understand how patients may cope and adapt after a lesion has occurred. The DynaMoE framework may be useful for studying the effects of lesions on learning and adaptation.

The DynaMoE framework also has interesting implications for areas of machine learning. It combines the advantages of prior models that leverage transfer learning in the MoE architecture with reinforcement learning (39–42) and Bayesian nonparametric MoE models (17, 43–47). DynaMoE's organic, unsupervised implementation of transfer may be useful for intractable problems that may be handled piecewise in a way that may be nonobvious to an "oracle" supervisor. By letting the model learn how to grow and structure itself, our framework puts the burden of optimally solving complex problems on the algorithm. This may significantly improve progress, by removing the need for careful curation of training data and training regimen.

The form of transfer learning demonstrated by our dynamic architecture—acquiring new knowledge (new expert) based on indirect knowledge of what other parts of the network (old experts) know—has not been reported before, to our knowledge. This form of transfer learning is reminiscent of "learning by analogy," a learning skill humans are very good at but machines continue to struggle with (73, 74). Through our framework, this dynamic form of transfer could be extended to much larger networks, utilizing a myriad of experts. Such a framework could be useful both as a model of the brain and for machine learning applications.

Finally, our framework provides a method for lifelong learning and memory. Major challenges persist in developing methods that do not get overloaded but also scale well to lifelong problems (75). Similar to "grow-when-required" algorithms, our network adds capacity when necessary. However, our network also leverages already acquired knowledge to help solve new problems, reducing demand for growth. This feature supports scalability, which both the brain and machine learning methods must support, given their limited resources. Elaborating and adapting DynaMoE to more complex tasks and incorporating other techniques such has simultaneous combinatorial use of experts will lead to exciting steps forward in lifelong learning.

Materials and Methods

Behavioral Task. To demonstrate our framework, we used the WCST. In this task, the subject is asked to sort cards. Each card has symbols with a shape type (triangle, star, cross, or circle), a color type (red, green, yellow, or blue), and a specific number of symbols (one, two, three, or four). During each episode, an unsignaled operating rule is chosen: either shape, color, or number. The subject must discover the rule by trial and error and then sort a given card into one of four stacks, according to the relevant rule. The first stack, stack 1, has one red triangle, stack 2 has two green stars, stack 3 has three yellow crosses, and stack 4 has four blue circles (Fig. 1*B*). After each attempted card sort, the subject is given feedback as to whether the sort was correct or incorrect. Once the subject has sorted a given number of cards correctly consecutively, the operating rule is switched without signal, and the subject must discover the new rule through trial and error. For all of our simulations, the operating rule was switched after three correct sorts in a row.

At the beginning of each episode, a deck of cards containing all 64 possible combinations of shape, color, and number was generated. Cards were randomly drawn from this deck and presented to the subject for sorting, removing each card from the deck after presentation. If all 64 cards from the deck were used before termination of the episode, the deck was regenerated, and new cards continued to be drawn in the same manner. An episode was terminated by meeting one of two termination criteria: 1) achieving the given number of correct sorts in a row (three for our simulations) or 2) reaching the maximum episode length, which we set to 200 card draws.

In our sequential scenario training simulations, a particular operating rule was kept constant for the duration of training in that period, either until a given number of sorting episodes was achieved or until performance passed a satisfactory threshold. In the next training period, a new operating rule was held constant, and training was repeated in the same manner. As a demonstration, a DynaMoE network was trained in a sequential training protocol with sequential blocks of 1,250 sorting episodes (one "sorting episode" $\approx$ 12 total WCST episodes across whole network) of each sort rule type (Fig. 2*C*). Each 1,250 sorting episode block was split into two 625 sorting episode subblocks; in the first subblock, the gating network was tuned, and, in the second, both the gating and new expert network were tuned. To evaluate the degree of transfer learning, a moving mean over every 100 sorting episodes was taken for the periods of isolated gate retuning (no expert training), and the minimum value was compared to the minimum value of the initial network before training any expert (baseline without transfer). When the shape sort rule was reintroduced, only the gating network was tuned. The 1,250 sorting episode block training protocol described above was also done with a standard RNN, for comparison (*SI Appendix*, Fig. S1). In all line plots of sorts to criterion over training, a moving mean over every 10 sorting episodes was calculated and plotted for readability.

Reinforcement Learning Training. To train our networks with reinforcement learning, we used the Advantage Actor-Critic algorithm of Mnih et al. (76), where a full description of the algorithm can be found. Briefly, the objective function for our neural network consists of the gradient of a policy term, an advantage value term, and an entropy regularization term,

$$\begin{aligned}\nabla\mathcal{L} &= \nabla\mathcal{L}_\pi + \nabla\mathcal{L}_v + \nabla\mathcal{L}_H \\ &= \frac{\partial log\pi(a_t|s_t;\theta)}{\partial\theta}\delta_t(s_t;\theta) + \beta_v\delta_t(s_t;\theta)\frac{\partial V}{\partial\theta} + \beta_H\frac{\partial H(\pi(a_t|s_t;\theta))}{\partial\theta},\end{aligned}$$

where π is the policy, a_t is the action taken at time t, s_t is the state at time t, θ is the network parameters, β_v, β_H are hyperparameters for scaling the contribution of the value and entropy terms, respectively, V is the value output of the network, and H is the entropy regularization term of the policy. δ_t is the advantage estimate, which represents the temporal difference error,

$$\delta_t(s_t;\theta) = R_t - V(s_t;\theta),$$

where R_t is the discounted reward,

$$R_t = \sum_{i=0}^{k-1}\gamma^i r_{t+i} + \gamma^k V(s_{t+k};\theta),$$

where k is the number of steps until the next end state. When $\gamma = 0$, $R_t = r_t$.

The advantage equation in this case is equivalent to a temporal difference error signal, enabling temporal difference reinforcement learning.

The parameters of the model were updated, during training, by gradient descent and back propagation through time after the completion of every three episodes. For all simulations, we used 12 asynchronous threads for training. In our plots, a single "sorting episode" was defined as the number of total WCST episodes completed while a single thread completed one episode, which was roughly equal to 12 episodes for the total network. We used the AdamOptimizer with a learning rate of 1e-03 to optimize weights. The objective function scaling hyperparameters β_v and β_H were both set to 0.05 for all our simulations.

For feedback as to whether each card sort was correct or incorrect, we gave a reward of +5 if correct and -5 if incorrect. For the WCST, a discount factor of $\gamma = 0$ was used, since each card sort was an independent event, based only on the relevant operating rule rather than any prior previous action sequence.

Similar to the implementation by Wang et al. (9), the input to the networks for each step was given as vector with the current card shape, color, and number, the action taken for the previous time step, a_{t-1}, and the reward given for previous card sort action, r_{t-1}.

Network Architecture. Both our standard RNN and DynaMoE network architectures were composed of LSTMs as implemented by Wang et al. (9) (for details, see *SI Appendix, Supporting Information Text*). In contrast to "vanilla" RNNs, LSTMs copy their state from each time step to the next by default and utilize a combination of built-in gates to forget, input new information, and output from the states. This RNN structure allows for robust learning and storage of function approximators for various tasks, as demonstrated by Wang et al. (9). The LSTM states and gates are described by the following equations:

$$\begin{aligned}f_t &= \sigma(W_{xf}x_t + W_{hf}h_{t-1} + b_f) \\ i_t &= \sigma(W_{xi}x_t + W_{hi}h_{t-1} + b_i) \\ o_t &= \sigma(W_{ox}x_t + W_{ho}h_{t-1} + b_o) \\ c_t &= f_t \circ c_{t-1} + i_t \circ tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \\ h_t &= o_t \circ tanh(c_t),\end{aligned}$$

where f_t, i_t, and o_t are the forget, input, and output gates, respectively, at time t; σ is the sigmoid activation function; W_{ij} denotes the weights from component i to component j; x_t is the external input at time t; h_t is the ouput of the LSTM at time t; c_t is the state of the LSTM cell at time t; b_f, b_i, and b_o are the biases of the forget, input, and output gates, respectively; b_c is the bias of the cell states; and $\circ$ denotes the Hadamard product.

For all our simulations described in the paper, we used a standard RNN of 105 units and a DynaMoE network with a 98-unit gating network and 19 unit experts. We chose these network sizes because they provided ample capacity to learn the WCST scenarios and shared the same number of total trainable network parameters (47,145), which enabled the direct comparisons between standard RNN and DynaMoE networks.

In DynaMoE networks, if the gating network could not solve a scenario using its current experts, a new expert was brought online. In this case, first, the gating network was retuned with the current experts and an additional randomly initialized expert of the same size. If performance did not achieve the desired performance criterion, the gating network and the new expert were then trained simultaneously. The gating network LSTM learned a function approximator mapping from inputs to experts, and the experts learned function approximators mapping from inputs to actions in their input domain of expertise, which was determined by the gating network's mapping function.

DynaMoE Seeded with Pretrained Experts Transfer Simulations. For our demonstration of DynaMoE networks' transfer learning property, we performed a simulation with pretrained experts. We trained one expert on only shape sorting until the expert network achieved near-perfect "expert performance," defined in this simulation as an average sorts to criterion of less than four in the last 100 episodes of a single asynchronous thread (minimum sorts to criterion is three). We repeated the same with a second expert network trained on only color sorting. We then created a DynaMoE network with these two pretrained expert networks and a third randomly initialized expert network, and trained the gating network only on the number sorting rule for 7,500 sorting episodes to ensure convergent decision behavior. "Expert performance" as defined above was not achieved during this stage of training (Fig. 3 *B, Inset*). Network weights were then fixed, and behavior and performance of the network was evaluated. To evaluate behavior of the network, 1,000 test episodes were performed in the number rule, and the proportion of decisions to use each expert network (the decision rate) was measured in subsets of the number rule described in *Results* ("shape-match-number," "color-match-number," and "no-match-number"; Fig. 3*C*). From this parent network, we then ran 10 independent training runs in parallel in which the gating network and the randomly initialized expert network were trained simultaneously on the number sorting rule until the "expert performance" criteria was achieved. To evaluate the decision rate of the gating network for each of the 10 independent training runs, 1,000 test episodes were performed, and the mean and SD of the decision rates were calculated in the same subsets of the number rule (Fig. 3*F* and *SI Appendix*, Fig. S2).

Organic Transfer Simulations. For our demonstration of the DynaMoE network's implementation of transfer learning without any pretraining, we independently trained 10 DynaMoE networks from scratch in the following manner. We began with a randomly initialized gating network with a single randomly initialized expert network. The gating network was then trained alone on the shape sort scenario of the WCST for 1,250 sorting episodes. The gate and single expert network were then trained simultaneously until "expert performance," defined, for this simulation, as an average sorts to criterion of less than four in the last 100 episodes of a single asynchronous thread. A new randomly initialized expert network was then added, and the gating network was trained for 7,500 sorting episodes in the color scenario to allow full convergence of decision behavior. The gate and new expert were then trained simultaneously until "expert performance" was achieved. This was repeated finally for the number scenario and a third expert network. To evaluate transfer, for each sort rule, we retuned the gating network until expert performance was achieved. The gating network was then tested for 1,000 episodes in the given sort scenario, and the relative expert network use was measured as described in *Additional Data Analysis* (Fig. 4 and *SI Appendix*, Fig. S3).

Robust Memory Savings Simulations. To demonstrate the DynaMoE network's robust memory savings, we independently trained 10 DynaMoE networks and standard RNNs with the same number of trainable parameters (47,145) in an identical presentation of scenarios. First, the randomly initialized networks trained on 1,250 sorting episodes of the shape sort scenario to ensure convergent performance. This was followed by 1,250 sorting episodes of the color sort scenario, followed by 1,250 sorting episodes of the number sort scenario (same as for networks in Fig. 2*C* and *SI Appendix*, Fig. S1). For the DynaMoE network, each block of 1,250 sorting episodes with a sort rule was broken into two subblocks of 625 sorting episodes; in the first 625 sorting episodes, the DynaMoE network did the first stage of training in which only the gating network is tuned, and, for the second 625, the second stage of training was done in which both the gating and new expert networks are tuned simultaneously. After this sequential scenario training, for each standard RNN and the DynaMoE network, we ran 10 independent retrainings on the first scenario encountered: the shape scenario. For the DynaMoE network, only the gating network was retuned. To measure how quickly the networks could recover performance in the previously learned rule, the networks were tuned until they reached a performance criteria of average sorts to criterion <10 cards for the last 10 episodes of a single asynchronous thread. The number of sorting episodes required to achieve this performance were measured, as well as the distance traveled in weight space during relearning/remembering the shape scenario (Fig. 5 and *SI Appendix*, Fig. S4). For additional methods on comparing network remembering to initial learning, see *SI Appendix*, *Supporting Information Text*.

Classic WCST Simulations with Untrained and Pretrained Networks. To simulate performance on the classic WCST in which the different sorting rule episodes are interleaved randomly, five different networks were created. The first network was a standard RNN with randomly initialized weights (*SI Appendix*, Fig. S6*A*). The second network was a standard RNN that was pretrained sequentially on, first, the shape rule, followed by the color rule, followed by the number rule (*SI Appendix*, Fig. S6*B*). For each rule type, the network was trained until "expert performance," defined as average sorts to criterion of less than four over the last 100 episodes of single asynchronous thread before switching rules. The third network was a DynaMoE network with three untrained expert networks with randomly initialized weights (Fig. 1*C* and *SI Appendix*, Fig. S6*C*). The fourth network was a DynaMoE network seeded with three pretrained expert networks—one pretrained on shape sorting, one on color sorting, and one on number sorting (Fig. 1*D* and *SI Appendix*, Fig. S6*D*). Each of these pretrained experts had been trained on the given rule until reaching "expert performance." The fifth network was a DynaMoE network that was pretrained sequentially on first the shape rule, followed by the color rule, followed by the number rule (Fig. 1*E* and *SI Appendix*, Fig. S6*E*). The network started with a gating network and a single expert network with randomly initialized weights. The gating and expert networks were trained simultaneously on the shape rule until "expert performance" was reached. The rule was then switched to the color rule, and a new expert network with random weights was added. The gating network was trained for a maximum of 250 sorting episodes, and then the new expert was brought online and trained until "expert performance." The same was then repeated for the number rule.

Each network was then trained on the classic WCST, in which rules are randomly interleaved (rules switch after every episode; see full description in *Behavioral Task*). The center column of *SI Appendix*, Fig. S6 shows performance of each network over 2,500 sorting episodes of training. Networks with pretraining (*SI Appendix*, Fig. S6 *B*, *D* and *E*) were also trained for 2,500 sorting episodes on the shape rule (the first rule experienced), to compare each network's ability to "remember" a previously learned rule.

Lesion Studies. To perform the lesions studies, we first trained a DynaMoE network identical to the network in *SI Appendix*, Fig. S6*E* as described above. We then implemented one of the following lesions: L0, no lesion; L1, ablation of the reward feedback input to the network; L2, ablation of the action feedback input; L3, both L1 and L2 simultaneously; L4, ablation of varying amounts of the synaptic connections of the "forget gate" component of the LSTM, ranging from 10 to 100% and denoted by the subscript (e.g., L4$_{0.9}$ has 90% of the synaptic connections ablated); L5,-ablation of varying amounts of output from the RNN; L6, ablation of synaptic connections to the units used to determine which expert network to use; L8, ablation of the synaptic connects to the unit used to estimate value; and L7, both L6 and L8 simultaneously. For two of the lesions types (L4, L7), we show the full severity spectrum, as an example, in *SI Appendix*, Fig. S7.

After implementing the lesion, we then tested the full DynaMoE network on the classic interleaved WCST. We ran 1,000 test episodes and then performed analysis on performance as described in *Results*. For each lesion type, we randomly ablated at each level of severity 10 times and analyzed average behavior, since lesions of specific connections or units within a given region may have differential effects. Errors per episode in Fig. 6 was the total number of errors in a sort rule episode. Perseveration proportion was calculated as the proportion of increase in total error due to change in perseveration errors. We defined perseveration errors as incorrect card sorts immediately following the inevitable error trial after a rule change, which would have been correct according to the previous rule (77). The inevitable error trial refers to the first trial which receives feedback that a sort according to the previously correct rule is now incorrect, signaling a rule change has occurred. We note that there are more complex methods for scoring perseverations on the WCST, largely due to the ambiguity introduced by cards that can sort to multiple stacks (34, 49, 77, 78). Practically, we categorized an error as a perseveration if it immediately followed an inevitable error trial, another perseveration error, or an unbroken streak of perseveration errors and correct sorts with ambiguity that included the previous sort rule. Using the same criteria, we also tested the effects of the lesions with the subset of cards excluding ambiguous cards, as in the Modified Wisconsin Card Sorting Task (5, 55) and found qualitatively similar results (*SI Appendix*, Fig. S8).

Additional Data Analysis. For additional data analysis methods, please see *SI Appendix*, *Supporting Information Text*.

Data Availability. The code used to generate the data in this work is available at https://github.com/tsudacode/DynaMoE.

ACKNOWLEDGMENTS. We thank Robert Kim, Yusi Chen, and Gal Mishne for helpful discussions and feedback on the manuscript. We thank Jorge Aldana for support with computing resources. This work was supported by the Kavli Institute for Brain and Mind at University of California San Diego, Office of Naval Research Grant N000141612829, NSF Grant 1735004, and Defense Advanced Research Projects Agency Grant W911NF1820.

1. S. Monsell, Task switching. *Trends Cognit. Sci.* **7**, 134–140 (2003).
2. B. R. Buchsbaum, S. Greer, W.-L. Chang, K. F. Berman, Meta-analysis of neuroimaging studies of the Wisconsin Card-Sorting Task and component processes. *Hum. Brain Mapp.* **25**, 35–45 (2005).
3. J. M. Fuster, The prefrontal cortex—An update: Time is of the essence. *Neuron* **30**, 319–333 (2001).
4. J. Gläscher, R. Adolphs, D. Tranel, Model-based lesion mapping of cognitive control using the Wisconsin Card Sorting Test. *Nat. Commun.* **10**, 20 (2019).
5. S. MacPherson, S. Sala, S. Cox, A. Girardi, M. Iveson, *Handbook of Frontal Lobe Assessment* (Oxford University Press, Oxford, United Kingdom, 2015).
6. F. A. Mansouri, K. Matsumoto, K. Tanaka, Prefrontal cell activities related to monkeys' success and failure in adapting to rule changes in a Wisconsin Card Sorting Test analog. *J. Neurosci.* **26**, 2745–2756 (2006).
7. T. J. Sejnowski, H. Poizner, G. Lynch, S. Gepshtein, R. J. Greenspan, Prospective optimization. *Proc. IEEE* **102**, 799–811. (2014).
8. H. F. Song, G. R. Yang, X.-J. Wang, Reward-based training of recurrent neural networks for cognitive and value-based tasks. *eLife* **6**, e21492 (2017).
9. J. X. Wang *et al.*, Prefrontal cortex as a meta-reinforcement learning system. *Nat. Neurosci.* **21**, 860–868 (2018).
10. R. S. Sutton, A. G. Barto, *Reinforcement Learning: An Introduction* (MIT Press, Cambridge, MA, ed. 2, 2018).
11. G. R. Yang, M. R. Joglekar, H. F. Song, W. T. Newsome, X.-J. Wang, Task representations in neural networks trained to perform many cognitive tasks. *Nat. Neurosci.* **22**, 297–306 (2019).
12. R. French, Catastrophic forgetting in connectionist networks. *Trends Cognit. Sci.* **3**, 128–135 (1999).
13. J. Kirkpatrick *et al.*, Overcoming catastrophic forgetting in neural networks. *Proc. Natl. Acad. Sci. U.S.A.* **114**, 3521–3526 (2017).
14. F. Zenke, B. Poole, S. Ganguli, "Continual learning with intelligent synapses" in *Proceedings of International Conference on Machine Learning (ICML)*, D. Precup, Y. W. Teh, Eds. (Proceedings of Machine Learning Research, 2017), vol. 70, pp. 3987–3995.

15. A. Chaudhry, M. Ranzato, M. Rohrbach, M. Elhoseiny, "Efficient lifelong learning with a-GEM" in *International Conference on Learning Representations (ICLR)* (International Conference on Learning Representations, 2019).
16. D. Lopez-Paz, M. Ranzato, "Gradient episodic memory for continual learning" in *Advances in Neural Information Processing Systems 30*, I. Guyon *et al.*, Eds. (Curran Associates, Inc. 2017), pp. 6467–6476.
17. R. Aljundi, P. Chakravarty, T. Tuytelaars, "Expert gate: Lifelong learning with a network of experts" in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (Institute of Electrical and Electronics Engineers, 2017), pp. 7120–7129.
18. A. A. Rusu *et al.*, Progressive neural networks. arXiv:1606.04671 (15 June 2016).
19. T. Gisiger, M. Boukadoum, Mechanisms gating the flow of information in the cortex: What they might look like and what their uses may be. *Front. Comput. Neurosci.* **5**, 1 (2011).
20. K. Johnston, H. M. Levin, M. J. Koval, S. Everling, Top-down control-signal dynamics in anterior cingulate and prefrontal cortex neurons following task switching. *Neuron* **53**, 453–462 (2007).
21. R. V. Rikhye, A. Gilra, M. M. Halassa, Thalamic regulation of switching between cortical representations enables cognitive flexibility. *Nat. Neurosci.* **21**, 1753–1763 (2018).
22. B. R. Postle, Delay-period activity in prefrontal cortex: One function is sensory gating. *J. Cognit. Neurosci.* **17**, 1679–1690 (2005).
23. T. Ott, A. Nieder, Dopamine and cognitive control in prefrontal cortex. *Trends Cognit. Sci.* **23**, 213–234 (2019).
24. C. M. V. Weele *et al.*, Dopamine enhances signal-to-noise ratio in cortical-brainstem encoding of aversive stimuli. *Nature* **563**, 397–401 (2018).
25. F. McNab, T. Klingberg, Prefrontal cortex and basal ganglia control access to working memory. *Nat. Neurosci.* **11**, 103–107 (2008).
26. T. M. Desrochers, C. H. Chatham, D. Badre, The necessity of rostrolateral prefrontal cortex for higher-level sequential behavior. *Neuron* **87**, 1357–1368 (2015).
27. T. S. Braver, J. D. Cohen, "On the control of control: The role of dopamine in regulating prefrontal function and working memory" in *Attention and Performance XVIII*, S. Monsell, J. Driver, Eds. (MIT Press, London, United Kingdom, 2000), pp. 713–737.
28. R. C. O'Reilly, M. J. Frank, Making working memory work: A computational model of learning in the prefrontal cortex and basal ganglia. *Neural Comput.* **18**, 283–328 (2006).
29. M. J. Frank, B. Loughry, R. C. O'Reilly, Interactions between frontal cortex and basal ganglia in working memory: A computational model. *Cognit. Affect Behav. Neurosci.* **1**, 137–160 (2001).
30. N. P. Rougier, D. C. Noelle, T. S. Braver, J. D. Cohen, R. C. O'Reilly, Prefrontal cortex and flexible cognitive control: Rules without symbols. *Proc. Natl. Acad. Sci. U.S.A.* **102**, 7338–7343 (2005).
31. M. J. Frank, D. Badre, Mechanisms of hierarchical reinforcement learning in corticostriatal circuits 1: Computational analysis. *Cerebr. Cortex* **22**, 509–526 (2012).
32. K. M. Tye, Neural circuit motifs in valence processing. *Neuron* **100**, 436–452 (2018).
33. D. T. Stuss, B. Levine, Adult clinical neuropsychology: Lessons from studies of the frontal lobes. *Annu. Rev. Psychol.* **53**, 401–433 (2002).
34. B. Milner, Effects of different brain lesions on card sorting. *Arch. Neurol.* **9**, 100–110 (1963).
35. M. Donoso, A. G. E. Collins, E. Koechlin, Foundations of human reasoning in the prefrontal cortex. *Science* **344**, 1481–1486 (2014).
36. R. A. Jacobs, M. I. Jordan, S. J. Nowlan, G. E. Hinton, Adaptive mixture of local experts. *Neural Comput.* **3**, 79–87 (1991).
37. S. Yuksel, J. Wilson, G. Paul, Twenty years of mixture of experts. *IEEE Trans. Neural Netw. Learn. Syst.* **23**, 1177–1193 (2012).
38. N. Shazeer *et al.*, "Outrageously large neural networks: The sparsely-gated mixture-of-experts layer" in *5th International Conference on Learning Representations, ICLR* (International Conference on Learning Representations, 2017).
39. S. P. Singh, "The efficient learning of multiple task sequences" in *Advances in Neural Information Processing Systems 4*, J. E. Moody, S. J. Hanson, R. P. Lippmann, Eds. (Morgan-Kaufmann, 1992), pp. 251–258.
40. S. P. Singh, Transfer of learning by composing solutions of elemental sequential tasks. *Mach. Learn.* **8**, 323–340 (1992).
41. M. S. Dobre, A. Lascarides, "Combining a mixture of experts with transfer learning in complex games" in *AAAI Spring Symposium Series* (AAAI Press, 2017).
42. M. Gimelfarb, S. Sanner, C.-G. Lee, "Reinforcement learning with multiple experts: A bayesian model combination approach" in *Advances in Neural Information Processing Systems 31*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, R. Garnett, Eds. (Curran Associates, Inc., 2018), pp. 9528–9538.
43. C. E. Rasmussen, Z. Ghahramani, "Infinite mixtures of Gaussian process experts" in *Advances in Neural Information Processing Systems 14*, T. G. Dietterich, S. Becker, Z. Ghahramani, Eds. (MIT Press, 2002), pp. 881–888.
44. S. R. Waterhouse, A. J. Robinson, "Pruning and growing hierachical mixtures of experts" in *Fourth International Conference on Artificial Neural Networks* (Institution of Engineering and Technology, 1995), pp. 341–346.
45. K. Saito, R. Nakano, "A constructive learning algorithm for an HME" in *Proceedings of International Conference on Neural Networks (ICNN'96)* (Institute of Electrical and Electronics Engineers, 1996), vol. 2, pp. 1268–1273.
46. J. Fritsch, M. Finke, A. Waibel, "Adaptively growing hierarchical mixtures of experts" in *Advances in Neural Information Processing Systems 9*, M. C. Mozer, M. I. Jordan, T. Petsche, Eds. (MIT Press, 1997), pp. 459–465.
47. M. Khamassi, L.-E. Martinet, A. Guillot, "Combining self-organizing maps with mixtures of experts: Application to an actor-critic model of reinforcement learning in the basal ganglia" in *From Animals to Animats 9*, S. Nolfi *et al.*, Eds. (Springer, Berlin, Germany, 2006), vol. 4095, pp. 394–405.
48. M. Mitrushina, K. B. Boone, J. Razani, L. F. D'Elia, *Handbook of Normative Data for Neuropsychological Assessment* (Oxford University Press, New York, NY, 2005).
49. D. A. Grant, E. Berg, A behavioral analysis of degree of reinforcement and ease of shifting to new responses in a Weigl-type card-sorting problem. *J. Exp. Psychol.* **38**, 404–411 (1948).
50. K. V. Haak, C. F. Beckmann, Plasticity versus stability across the human cortical visual connectome. *Nat. Commun.* **10**, 3174 (2019).
51. T. O. Nelson, Ebbinghaus's contribution to the measurement of retention: Savings during relearning. *J. Exp. Psychol. Learn. Mem. Cognit.* **11**, 472–479 (1985).
52. D. R. Euston, A. J. Gruber, B. L. McNaughton, The role of medial prefrontal cortex in memory and decision making. *Neuron* **76**, 1057–1070 (2012).
53. M. Dimitrov, M. Phipps, T. P. Zahn, J. Grafman, A thoroughly modern Gage. *Neurocase* **5**, 345–354 (1999).
54. K. Richard Ridderinkhof, M. M. Span, M. W. van der Molen, Perseverative behavior and adaptive control in older adults: Performance monitoring, rule induction, and set shifting. *Brain Cognit.* **49**, 382–401 (2002).
55. Hazel. E. Nelson, A modified card sorting test sensitive to frontal lobe defects. *Cortex* **12**, 313–324 (1976).
56. B. Yao, D. Walther, D. Beck, L. Fei-fei, "Hierarchical mixture of classification experts uncovers interactions between brain regions" in *Advances in Neural Information Processing Systems 22*, Y. Bengio, D. Schuurmans, J. D. Lafferty, C. K. I. Williams, A. Culotta, Eds. (Curran Associates, Inc., 2009), pp. 2178–2186.
57. M. Rigotti *et al.*, The importance of mixed selectivity in complex cognitive tasks. *Nature* **497**, 585–590 (2013).
58. V. Mante, D. Sussillo, K. V. Shenoy, W. T. Newsome, Context-dependent computation by recurrent dynamics in prefrontal cortex. *Nature* **503**, 78–84 (2013).
59. J. R. Wessel, Error awareness and the error-related negativity: Evaluating the first decade of evidence. *Front. Hum. Neurosci.* **6**, 88 (2012).
60. E. Marder, V. Thirumalai, Cellular, synaptic and network effects of neuromodulation. *Neural Network.* **15**, 479–493 (2002).
61. G. Leal, D. Comprido, C. B. Duarte, BDNF-induced local protein synthesis and synaptic plasticity. *Neuropharmacology* **76**, 639–656 (2014).
62. R. P. Roelfsema, A. Holtmaat, Control of synaptic plasticity in deep cortical networks. *Nat. Rev. Neurosci.* **19**, 166–180 (2018).
63. W. C. Abraham, Metaplasticity: Tuning synapses and networks for plasticity. *Nat. Rev. Neurosci.* **9**, 387 (2008).
64. R. Velez, J. Clune, Diffusion-based neuromodulation can eliminate catastrophic forgetting in simple neural networks. *PLoS One* **12**, e0187736 (2017).
65. C. Orr, R. Hester, Error-related anterior cingulate cortex activity and the prediction of conscious error awareness. *Front. Hum. Neurosci.* **6**, 177 (2012).
66. N. Lopatina *et al.*, Ensembles in medial and lateral orbitofrontal cortex construct cognitive maps emphasizing different features of the behavioral landscape. *Behav. Neurosci.* **131**, 201–212 (2017).
67. P. H. Rudebeck, E. A. Murray, The orbitofrontal oracle: Cortical mechanisms for the prediction and evaluation of specific behavioral outcomes. *Neuron* **84**, 1143–1156 (2014).
68. N. W. Schuck, M. Bo. Cai, R. C. Wilson, Y. Niv, Human orbitofrontal cortex represents a cognitive map of state space. *Neuron* **91**, 1402–1412 (2016).
69. E. Koechlin, C. Summerfield, An information theoretical approach to prefrontal executive function. *Trends Cognit. Sci.* **11**, 229–235 (2007).
70. W. H. Alexander, J. W. Brown, Hierarchical error representation: A computational model of anterior cingulate and dorsolateral prefrontal cortex. *Neural Comput.* **27**, 2354–2410 (2015).
71. M. M. Botvinick, Multilevel structure in behaviour and in the brain: A model of Fuster's hierarchy. *Philos. Trans. R. Soc. B* **362**, 1615–1626 (2007).
72. C. A. Siciliano *et al.*, A cortical-brainstem circuit predicts and governs compulsive alcohol drinking. *Science* **366**, 1008–1012 (2019).
73. F. Hill, A. Santoro, D. G. Barrett, A. S. Morcos, T. Lillicrap, "Learning to make analogies by contrasting abstract relational structure" in *International Conference on Learning Representations (ICLR)* (International Conference on Learning Representations, 2019).
74. Y.-F. Kao, R. Venkatachalam, Human and machine learning. *Comput. Econ.*, 10.1007/s10614-018-9803-z (2018).
75. G. I. Parisi, R. Kemker, J. L. Part, C. Kanan, S. Wermter, Continual lifelong learning with neural networks: A review. *Neural Network.* **113**, 54–71 (2019).
76. V. Mnih *et al.*, Asynchronous methods for deep reinforcement learning. *J. Mach. Learning Res.* **48**, 1928–1937 (2016).
77. K. W. Greve, Can perseverative responses on the Wisconsin Card Sorting Test be scored accurately? *Arch. Clin. Neuropsychol.* **8**, 511–517 (1993).
78. L. A. Flashman, M. D. Horner, D. Freides, Note on scoring perseveration on the Wisconsin Card Sorting Test. *Clin. Neuropsychol.* **5**, 190–194 (1991).