Transformer Mechanisms Mimic Frontostriatal Gating Operations When Trained on Human Working Memory Tasks

Aneri Soni*, Aaron Traylor*, Jack Merullo, Michael J. Frank†, and Ellie Pavlick†

Downloaded from direct.mit.edu/jocn/article-pdf/38/11/2128/2617974/jocn.a.2679.pdf by University of Wisconsin, Madison, Jacqueline Fulvio on 18 November 2026

Abstract

■ Working memory (WM) is thought to rely on sophisticated frontostriatal mechanisms for selective gating, supporting updating and readout of information to and from distinct memory "addresses" (neural populations). According to computational models, capacity limitations in WM arise due to challenges in "role addressability": learning to correctly bind items in memory to their respective roles and assigning credit to the corresponding gating operations. However, these conclusions are based on particular assumptions about biological neural networks, and it is unclear whether the principles generalize to other architectures. To address this question, we examine whether similar gating mechanisms emerge in Transformer neural network architectures, which have demonstrated success on tasks requiring executive function—the ability to represent, coordinate, and manage multiple subtasks—yet lack intentionally built-in gating mechanisms. We analyze the mechanisms that emerge within Transformers trained on human WM tasks explicitly designed to place demands on gating. We find that the Transformer's attention mechanism develops role-addressable input and output gating, but only when trained on task distributions that benefit from frontostriatal-like gating mechanisms. Moreover, these gating strategies support enhanced generalization and variable binding and increase the models' effective capacity to store and access multiple items in memory, resembling the constraints found in frontostriatal models. These results suggest that gating mechanisms serve a fundamental computational role in managing role addressability and binding and highlight opportunities for future research on computational similarities between modern artificial intelligence architectures and models of the human brain. ■

INTRODUCTION

Working memory (WM) involves the ability to store, maintain, and manipulate multiple items such that they can be readily used for ongoing task processing and to constrain behavior. WM capacity is, however, limited (Cowan, 2008). Theoretical accounts differ in the nature of this limitation (e.g., whether it reflects a limited number of discrete slots or a common pool of resources to store multiple items, thereby limiting their precision as WM load grows; Ma, Husain, & Bays, 2014). Various hybrid models have also been proposed, supported by evidence (Nassar, Helmers, & Frank, 2018; Swan & Wyble, 2014; van den Berg, Awh, & Ma, 2014). Another line of work suggests that capacity limits are further limited by difficulties in WM management: the need to filter what information should be maintained, to bind individual items to distinct "roles" so that they can be accessed appropriately, and to prioritize information already in memory for readout (Soni & Frank, 2024; Oberauer & Lin, 2017; Rose et al., 2016; Oberauer, 2013; Baier et al., 2010; Todd, Niv, & Cohen, 2009; McNab & Klingberg, 2008).

In biologically inspired computational models of frontostriatal WM circuitry, these operations are supported by selective gating mechanisms. Clusters of prefrontal neurons (or stripes) represent distinct addresses in memory that can be selectively accessed via gating actions triggered by the basal ganglia and thalamus (Soni & Frank, 2024; Calderon, Verguts, & Frank, 2022; Kriete, Noelle, Cohen, & O'Reilly, 2013; Dayan, 2012; Frank & Badre, 2012; O'Reilly & Frank, 2006). Gating mechanisms are responsible for multiple distinct aspects of WM management, including determining which items to store and retrieve, when, and from where (e.g., how to bind items to distinct roles that can be used for later retrieval). These gating mechanisms contain at least three important components (Soni & Frank, 2024; Frank & Badre, 2012; Todd et al., 2009; O'Reilly & Frank, 2006) supported by experimental evidence (Rac-Lubashevsky & Frank, 2021; Bhandari & Badre, 2018; Rac-Lubashevsky & Kessler, 2016; Rose et al., 2016; Chatham, Frank, & Badre, 2014; Badre & Frank, 2012; Baier et al., 2010; McNab & Klingberg, 2008). First, input-gating controls whether or not given information is stored in memory and, if stored, determines the "address" (population of neurons) to

Brown University
*These authors share co-first authorship.
†These authors share co-senior authorship and contributed equally to this work.

© 2026 Massachusetts Institute of Technology

Journal of Cognitive Neuroscience 38:11, pp. 2128–2140
https://doi.org/10.1162/JOCN.a.2679

which it should be written. Second, output gating determines when and what information to prioritize, or read out of memory, to inform a subsequent decision, such as to produce a response to a task (Rose et al., 2016; Chatham et al., 2014; Frank & Badre, 2012). Finally, WM is role-addressable, meaning that items can be bound to a learned task-dependent context (i.e., role) when stored and accessed in WM. For example, after being introduced to a few people, one might want to retrieve the name associated with a particular face when planning to address them. In a lab setting, this might require a participant to report the color that was associated with a particular orientation (Nassar et al., 2018; van den Berg et al., 2014). When learning effective gating policies, these models afford functions such as variable binding and indirection that support rapid generalization to new situations (Bhandari & Badre, 2018; Collins & Frank, 2013; Kriete et al., 2013; Frank & Badre, 2012; O'Reilly & Frank, 2006). They also account for a variety of data implicating a network linking frontal cortex with basal ganglia and thalamus in WM gating.

Capacity limits arise in such models because as the number of items to maintain increases, it becomes increasingly difficult to learn which gating operations are most effective, leading to interference and difficulties in credit assignment (Soni & Frank, 2024; Todd et al., 2009). These limitations can be partially mitigated by learning to chunk multiple items in WM; such that networks with chunking abilities performed better than those allocated a larger number of "slots" with which they could conceivably store information (Soni & Frank, 2024). These simulation experiments suggested that a key factor limiting WM capacity lies in the difficulties in learning to manage WM access through filtering and binding (Soni & Frank, 2024). An implication of this conclusion is that a part of the WM challenges lies in management rather than maintenance per se and thus that even computational architectures without any explicit limitations on maintenance may also require managing effective gating policies, or analogous processes thereof.

STUDYING WM IN TRANSFORMER NEURAL NETWORKS

The Transformer architecture (Vaswani et al., 2017) has recently become the dominant neural network model in artificial intelligence (AI). Unlike some earlier AI architectures, which were inspired (albeit loosely) from human processing of language (Hochreiter, 1997) or vision (LeCun, Bengio, & Hinton, 2015), Transformers have no mechanisms designed overtly to resemble the human brain. It thus remains an open question as to what, if any, similarities exist and whether there are opportunities for theory or insights from AI and neuroscience to mutually inform one another (Russin, Pavlick, & Frank, 2025; McGrath, Russin, Pavlick, & Feiman, 2024).

Recent work has explored how Transformer key value computations relate to memory systems in the brain, with a particular focus on episodic memory (Gershman, Fiete, & Irie, 2025; Whittington, Warren, & Behrens, 2021). WM, however, places distinct demands on a system: Rather than encoding and retrieving experiences with high fidelity, it requires active, selective gating of a small set of currently task-relevant items, updated in a Markovian fashion and protected from proactive interference by earlier overlapping content (see the Summary and Discussion section for elaboration).

In this work, we focus specifically on WM management via gating mechanisms to meet these specific demands. Although some Transformer variants have additional built-in structure for memory (Burtsev, Kuratov, Peganov, & Sapunov, 2020; Dai et al., 2019; Wang, Lee, & Chen, 2019), the architectures that currently dominate modern AI systems are "vanilla" (Touvron et al., 2023; Brown et al., 2020), lacking specialized components to support this type of control.

We thus investigate whether, and under what conditions, such structure can emerge as a result of training. We train vanilla Transformer models on a task from cognitive neuroscience that was designed to investigate selective gating and WM in humans (Rac-Lubashevsky & Frank, 2021; O'Reilly & Frank, 2006). We use recent techniques from mechanistic interpretability (Nanda & Bloom, 2022; Olah, 2022) to expose the mechanism that the Transformer uses to perform the task. We find that, as a result of training, the self-attention mechanism specializes in a way that resembles existing models of input–output gating (see the Key and Query Vectors Specialize for Input and Output Gating section), but that these mechanisms only arise when the training task places specific demands on gating that mimics biological networks (see the Emergence of Gating Policies Depends on Task Demands section). We further find that when such mechanisms do arise, they are predictive of better task performance and of generalization to changes in the input distribution and task demands (see the Emergence of Gating Predicts Task Performance section), improving effective WM capacity (see the Gating Policy Transfers to Increased Task Demands section). Our findings highlight the importance of considering the emergent mechanisms that result from training in addition to the innate architectural mechanisms when drawing comparisons between AI systems and human cognitive processes. In principle, Transformers are good candidates for learning such gating behavior. Transformers' native self-attention mechanism consists of attention heads that are arguably functionally analogous to frontostriatal stripes. The decomposition of these heads into distinct keys, queries, and values (see Supplemental Information Self-Attention in Transformers) means that the Transformer can, in principle, learn to differentiate reading and writing operations in a role- and context-dependent way across its multiple heads. However, whether Transformers will use their self-attention to implement such a mechanism is an open question, especially in cases when it is possible to fit the training data using more heuristic and less generalizable solutions.

Downloaded from direct.mit.edu/jocn/article-pdf/38/11/2128/2617974/jocn.a.2679.pdf by University of Wisconsin, Madison, Jacqueline Fulvio on 18 November 2026

The core theoretical claim we test is that computational constraints on learning role-addressable memory management are sufficient to give rise to gating-like mechanisms and that these constraints manifest similarly across biological and artificial systems. Human studies have shown that WM capacity is not limited by the number of items one can maintain but rather by their effective gating strategies in frontostriatal circuits (Baier et al., 2010; McNab & Klingberg, 2008; Vogel, McCollough, & Machizawa, 2005). Theoretical work has shown that capacity limits in these circuits do not stem from a limitation in the number of available neural populations but rather result from a credit assignment problem that arises when learning to manage multiple items in memory (Soni & Frank, 2024; Todd et al., 2009). Moreover, these limitations can be partially mitigated by learning to reuse effective gating policies (Soni & Frank, 2024).

Note that an implication of this view is that artificial systems facing these management demands should be pressured toward discovering analogous gating solutions even if they have no demands on memory maintenance or capacity. We test this prediction in vanilla Transformers, which have no explicit gating machinery and no meaningful memory capacity constraints (since all information is available in the context window), making them an ideal test case for isolating management and learning demands from storage demands. We derive three specific predictions. First, if gating emerges from management demands rather than architectural bias, it should only arise when the training task requires role-addressable storage and retrieval (and conversely, it should fail to emerge under control task conditions that remove these demands). Second, if the learned mechanisms are genuinely gating-like (rather than other heuristic solutions), they should support generalization to out-of-distribution inputs and transfer to tasks with higher WM demands. Third, echoing findings from frontostriatal models, increasing the number of items to be managed should produce capacity-like limitations even in a system with no storage constraints, and these limitations should be partially mitigated by prior learning of effective gating policies.

The results support all three predictions, suggesting that gating is a computational response to the demands of role-addressable WM management.

EXPERIMENTAL DESIGN

Tasks

Reference-back 2 Task

We use a variant of a task from cognitive neuroscience known as the "reference-back 2" task (Rac-Lubashevsky & Frank, 2021). This is one among a number of task designs inspired by frontostriatal neural networks (Soni & Frank, 2024; O'Reilly & Frank, 2006), which requires selective updating and accessing of information in a role-addressable manner. In the reference-back paradigm, symbols are viewed one at a time with associated roles, and the participant must determine whether the current symbol is the same or different as that stored in memory for a given role. For example, a sequence might contain letters (Role 1) and numbers (Role 2), each of which occurs along side an update instruction, which is either store or ignore. For each symbol in the sequence, the participant must do two things: (1) make a same/different judgment based on whether the current symbol matches the previously stored symbol for that role, and (2) if the update instruction is "store," update the symbol associated with the associated role. See Figure 1 for an example. We create a modified text-based version of the reference-back 2 task suitable for the Transformer. In our design, roles are denoted explicitly using special tokens (i.e., either Reg0 or Reg1 for the two-role version) indicating the role (or "register") to which the symbol should be bound. See Supplemental Information Textual Reference-Back-2 Task for more details about our task implementation.

Split-set Control

The computational advantages of role-addressable gating in frontostriatal networks (relative to other recurrent neural networks) are particularly evident when any symbol can be assigned to any register, so that the networks have to learn to assign them separable addresses (O'Reilly & Frank, 2006). To test the hypothesis that gating mechanisms only emerge in response to such task demands, we created a "split-set" control condition in which the registers are associated with disjoint sets of symbols. In this task, it is possible to succeed without role-addressable gating, since symbols never need to be decoupled from their associated registers. This condition also serves to demonstrate that attentional mechanisms within Transformer models do not, by definition, implement selective gating but only do so when incentivized by task demands.

Ignore-integrated and Ignore-separated Split-set Controls

In the same vein, we include a further simplified task variant that is designed to require selective input gating but not output gating. This condition, along with the split-set control, have been previously used to show that the advantage of frontostriatal gating networks relative to other recurrent networks is largely reduced in such scenarios (O'Reilly & Frank, 2006).

Specifically, in the "ignore separated" condition (referred to as split-set control earlier), each register sees a disjoint set of symbols. This separation also applies to the Ignore update instruction, such that the model could in principle learn to Ignore each of the symbols paired with the Ignore instruction and does not need to learn to differentiate Store and Ignore in a meaningful way. In contrast, in the "ignore integrated" condition, symbols under an Ignore instruction will be in distribution with

Downloaded from direct.mit.edu/jocn/article-pdf/38/11/2128/2617974/jocn.a.2679.pdf by University of Wisconsin, Madison, Jacqueline Fulvio on 18 November 2026

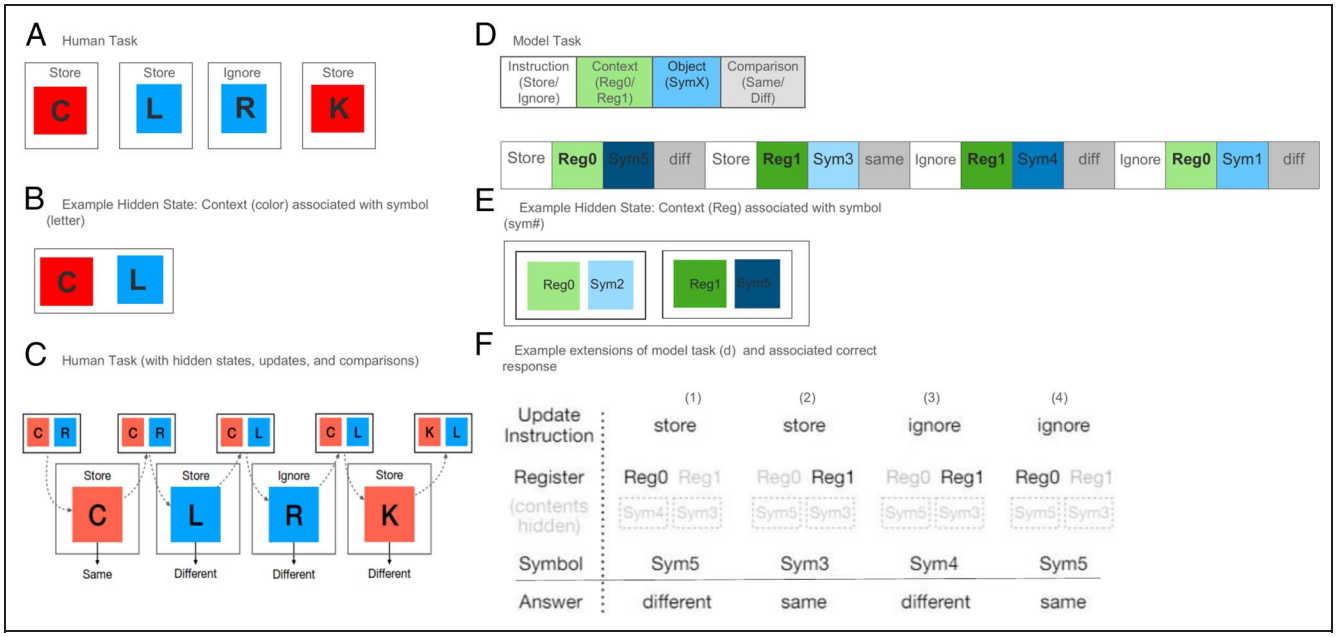


Figure 1. Task. The reference-back 2 task requires making same versus different judgments for each symbol in a sequence by comparing against a previously shown symbol. See text for description of the task. (A) Four example trials in the human task. In this example, the colors blue and red serve as contexts (roles). The letters are the associated information to be stored and retrieved from WM. An additional instruction indicates whether the content of memory should be stored (updated) or ignored. (B) The hidden state maintains each context and associated letter. (C) The arrows show how the letters are updated and compared with the hidden states. Note that even in an "ignore" trial, the task requires comparing the current item to that in memory for correct responding. (D) The model task is presented with the instruction, context (register), associated symbol and the comparison response. (E) Corresponding hidden state that the model should represent to associate a register (context/role) to symbol. (F) Four example trials where model has to compare hidden state symbols to incoming stimuli and update memory based on instruction. Model correctness is determined by if model response (same/different) matches expected value.

the chosen register, and hence, it will sometimes have to attend to those symbols (when they are paired with Store). Thus, in this case, the model must distinguish between a Store and an Ignore instruction and thus learn input gating but does not have to learn an output-gating policy because both registers have mutually exclusive symbols.

Models

Our focus was to understand and interpret attentional mechanisms of the models, and therefore, we used a minimal model that was able to learn the task. We train small, attention-only, decoder-only Transformer models from scratch on our task. Our models contain two decoder-only layers, each with two heads, and no multi-layer perceptrons or layer normalization, followed by a linear "unembed" layer. The models are trained on 100 k training data points for 60 epochs. See Supplemental Information Models and Training for additional details.

Metrics

Performance on Reference-back 2 Task

We evaluated how well the model performs on the task on which it was trained using standard accuracy on the same versus different prediction for each symbol in the sequence.

Input and Output Gating

The reference-back 2 task is assumed to benefit from gating mechanisms, but success on the task is not in and of itself diagnostic of having learned the gating mechanisms. To develop an intrinsic measure of the input- and output-gating mechanisms, we use path patching (Goldowsky-Dill, MacLeod, Sato, & Arora, 2023; Wang, Variengien, Conmy, Shlegeris, & Steinhardt, 2022), a generalization of causal mediation analysis (Pearl, 2001) that allows us to determine which components of a neural model (e.g., attention heads) work together in order to produce observed behavior on a task. Path patching involves designing a minimal pair of inputs, a "clean" input and a "corrupted" input, and then finding specific components of a model that fully account for the difference in the model's output between the two cases. By "patching in" only these components, the model can be made to behave as though it is seeing the corrupted input even when it is in fact seeing the clean input. See Supplemental Information Path Patching for more details.

We use path patching as a measure of input gating in the following way. For input gating, we design a minimal pair of inputs in which the corrupted copy includes Ignore in a

Downloaded from direct.mit.edu/jocn/article-pdf/38/11/2128/2617974/jocn.a.2679.pdf by University of Wisconsin, Madison, Jacqueline Fulvio on 18 November 2026

place where the clean copy contained a Store, or vice versa. We then use path patching to identify an incisive edit that can be made to the activations of the model in order to prevent the model from storing ("gating in") a given symbol. We report the accuracy of the input-gating mechanism as the percentage of inputs on which this edit to the weights changes the model's behavior in the expected way.

Analogously for output gating, we design minimal pairs that we expect to yield differences in an output-gating mechanism, assuming that one exists and is functioning correctly. Specifically, our clean and corrupted sequences differ in the register associated with one of the symbols, which should trigger a difference in what information is read ("gated out") in order to make a final same versus difference judgment. Again, we report the accuracy of the output-gating mechanism as the percentage of cases on which it is possible to make such an edit and produce the desired effect.

Our description of results in the Results section elaborates on both input- and output-gating metrics and their interpretation.

RESULTS

Key and Query Vectors Specialize for Input and Output Gating

We find that Transformers implement a role-addressable gating mechanism whereby key vectors control input gating and query vectors control output gating (see Supplemental Information Self-Attention in Transformers for summary of key, query, value attention, and the Summary and Discussion section for elaboration on how key and query vectors work together in the service of input and output gating). Specifically, after training, the key for Sym_i (e.g., at Tokens 2, 6, 10, and 14 in Figure 2A) contains information about the combination of the update instruction, register, and symbol for position i. A query's ability to address this position depends on whether the represented tuple contains a Store or an Ignore. Indeed, key vectors representing an Ignore tuple receive very little attention (0.4% of Layer 1 attention averaged over test set), whereas those representing a Store tuple receive the bulk of the attention (86.8%). We demonstrate the above narrative using path patching, as described in the Metrics section and further below. Figure 2 shows a summary of the path-patching experiments and results. Our task is simple and does not contain noise, so in all cases, the intervention (i.e., patching to the keys or queries) results in a 100% change in the model prediction in the expected direction. Thus, for compactness, Figure 2 depicts the conditions but does not include quantitative results.

First, to investigate input gating, we create clean sequences sampled from our test set and then corrupt these sequences by switching a Store within tuple i to an Ignore. We path-patch only the key vectors of i. We expect that, if the key controls input gating, patching these key vectors to those associated with an Ignore instruction should "block" attention to all of tuple i. An example attention pattern is in Figure 2A and B. We find that the model's attention shifts away from the tuple accordingly in 100% of patched instances. The presence of an Ignore or a Store within a tuple controls whether the key construction acts as an open input gate or a closed input gate.

Key construction also depends on the role of the represented content; within our task, the role corresponds to

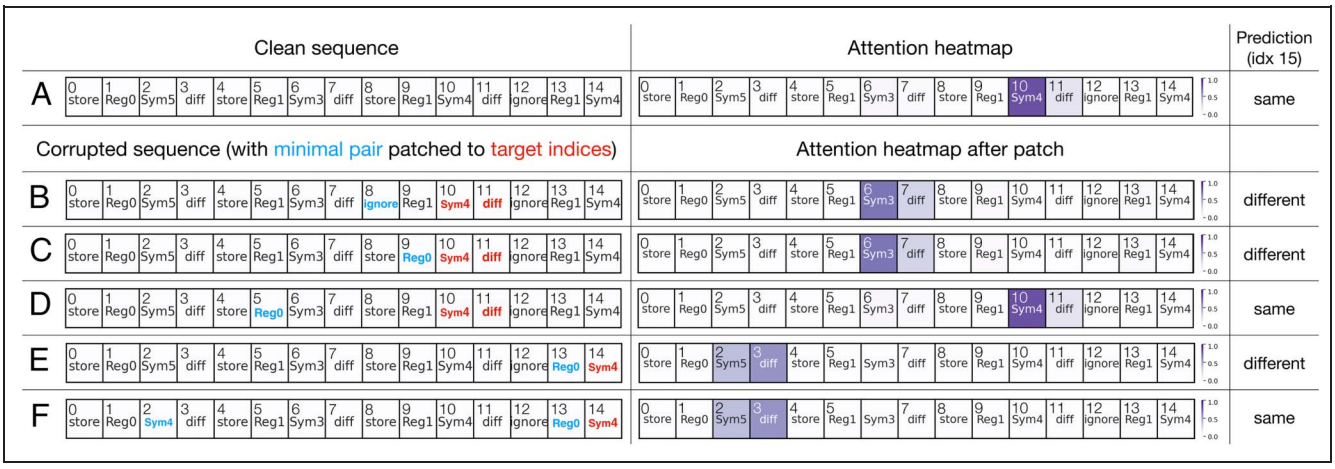


Figure 2. Path patching examples. Model behavior across different path patching conditions. Attention is visualized as a shade of purple, with deeper shade corresponding to higher attention to that token. We create "corrupted" minimal pairs in which changing a token in the input (light blue) either changes the correct label at index 15 (examples B, C, and E) or does not (D and F). We make small path-patching edits with the minimal pair to targeted network components (layer 1 keys for B, C, D, and F; queries for E and F, see text). In all test examples, making the small patch successfully alters the model's prediction to align with the "corrupted" example, as expected.

Downloaded from direct.mit.edu/jocn/article-pdf/38/11/2128/2617974/jocn.a.2679.pdf by University of Wisconsin, Madison, Jacqueline Fulvio on 18 November 2026

whether the Reg_i is Register 0 or Register 1. When making a same/different prediction, key vectors representing a tuple that matches the target register receive most of the model's attention (92.5% of total attention), whereas those that do not match are not attended to (3.3% of total attention). Again, we use path patching to causally determine whether key construction encodes roles, this time perturbing the target register rather than the update instruction (see Figure 2C). We hypothesized that this perturbation would be akin to altering the address in which an item is input gated. Indeed, the model's attention shifts away accordingly across every example in the test set. For example, in Panel C, when Reg1 is corrupted to Reg0 at Position 9, the model's attention switches to the earlier symbol (Sym3 at Position 6), because it is now the most recent item associated with Reg1 in the final query, leading to a "different" response. Note that the stored tuple must be modified; if the same corruption is made earlier (as in row d), attention does not shift (because Reg1 is later updated anyway). This behavior shows that the gating within self-attention is role-addressable; the registers within the task function as roles and are embedded within the key vectors as part of the representation.

Given that key vectors serve the role of addresses, query vectors, in turn, control which key vectors are accessed through the final Q*K dot product in attention. Query construction thus performs the role of output gating within Transformers. The query composition controls which addressable Symboli representations are attended to based on the identity of the target register. We again determine this through a set of path-patching experiments in which we perturb the target register (Figure 2E).

We find that patching to the query vector in such cases indeed causes the attention to shift from the original stored tuple (74.1% of attention) to the stored tuple that matches the edited register, resulting in a corresponding change in the final same/different judgment. Editing aspects of the target tuple other than the target register has minimal effect on the query construction. No edits to the query cause the model to attend to an Ignore tuple, further evidencing output-gating behavior—only content that has been made "addressable" can be accessed for a response. Furthermore, we find that the target instruction and symbol do not factor into the query composition—changing them through path patching to the query does not affect attention. This is notable because the model could employ other strategies for determining which tuples are eligible to be the stored tuple; for example, attending to all symbols to match if any of them are the same as the target symbol.

Emergence of Gating Policies Depends on Task Demands

Under which conditions do gating computations arise? One possibility is that they are a trivial consequence of the Transformer's architecture. After all, gating is sometimes expressed as a simple multiplicative operation, as in long short-term memory (Hochreiter, 1997). And indeed, the key, query, and value vectors underlying the attention heads are combined via matrix multiplications. However, this alone does not ensure that the networks learn effective, role-addressable gating policies. We thus investigate whether the training task demands induce such gating policies by running several control experiments (see the Tasks section), which simplify the task, thereby removing the role addressability of gating requirements. We thus hypothesized that since this task is much easier to learn, the Transformer will learn an overly memorized, brittle solution and will not develop effective input- and output-gating strategies.

As shown in Figure 3, networks trained on the split-set control task do not perform well at either the input- or output-gating subtasks (see the Metrics section for gating metrics). Moreover, models trained on the ignore-integrated task perform well at the Store versus Ignore input-gating tasks, but not the output-gating task, as expected, while models trained on the ignore-separated task perform poorly on both input and output gating (Figure 3B). These results strongly support the intuition that while Transformers have good inductive biases for learning role-addressable gating, it does not come "for free" and emerges only when demanded by the training task—the same contrast in demands that demonstrated advantages of frontostriatal gating (O'Reilly & Frank, 2006).

Emergence of Gating Predicts Task Performance

Is learning this gating policy useful for succeeding on the task? To answer this question, we first compare models with the same hyperparameters across different random seeds. We train 20 new models, each with a different random initialization, and measure both training loss and test set accuracy. Five of the models succeed 100% of the time, and the other 15 models succeed between 94% and 99.99% of the time, with a mean of 97.72% and a standard deviation of 2.03. We measure the intrinsic quality of the input- and output-gating subtasks using the path-patching metrics described in the Metrics section.

Figure 4 shows the five runs that reach 100% accuracy on the test data as well as five randomly selected runs that do not. Two trends stand out. First, models that score a perfect test accuracy appear to succeed at the gating subtasks more readily than models that do not. Of the former, three of five models reach 100% accuracy on both subtasks readily, plateauing less than halfway through training. In contrast, models that make errors in the test set also do not reach such immediate success at the subtasks (including the 10 not pictured in this graph); in fact, many categorically fail, scoring as low as 49% accuracy. These results do not indicate that this class of models' representations are useless for the task—they all score between 94% and 99.99%, well above chance performance.

Downloaded from direct.mit.edu/jocn/article-pdf/38/11/2128/2617974/jocn.a.2679.pdf by University of Wisconsin, Madison, Jacqueline Fulvio on 18 November 2026

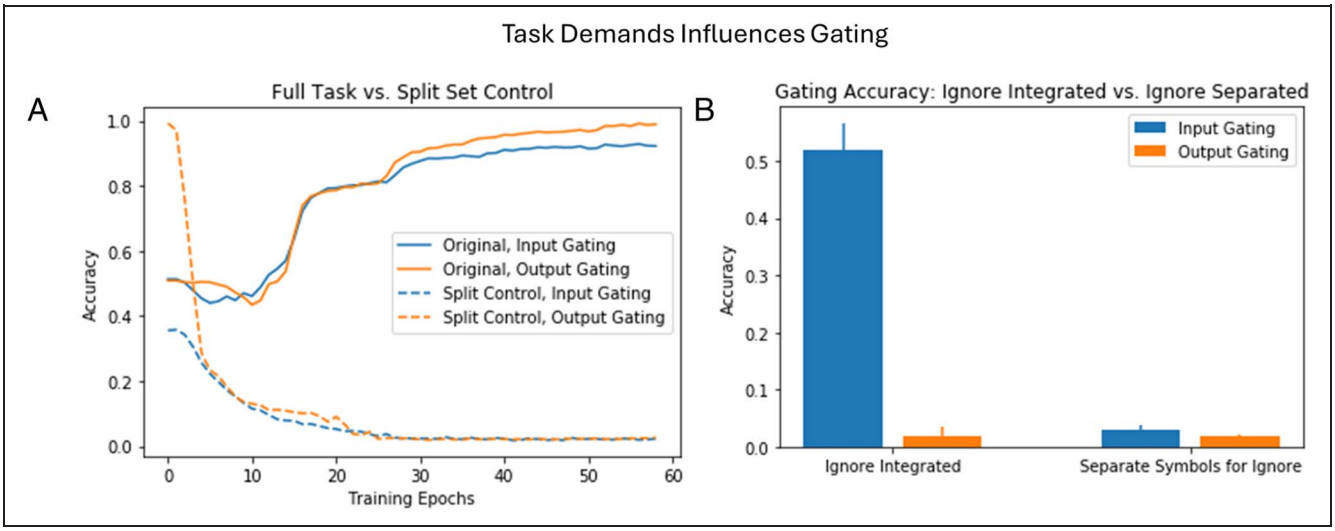


Figure 3. Split-set control. The split-set control (a.k.a. ignore segregated) uses a disjoint set of symbols for each register and a disjoint set of symbols for the ignore condition. This is the most simplified version of the task, as the network does not need to distinguish between store and ignore instructions and thus does not require input gating (it just needs to learn to always ignore some symbols, which is easily accomplished with standard weight-learning mechanisms). It also does not require output gating because it never has to assign the same symbol to different registers, so it does not have to flexibly perform variable binding. In the ignore integrated condition, the registers have disjoint symbol sets, but the ignore condition pulls from any of the register–symbol sets. This task should require input gating to distinguish store from ignore instructions but not necessarily how to differentially read out items from the registers (output gating). (A) Compared with the full task where both input and output gating are learned through training, there is no development of either gating strategy in the split-set control (ignore separated). (B) When ignore symbols are integrated with store symbols, the models must learn to respond appropriately to store/ignore instructions (input gating). Indeed, the network does learn the input-gating subtask for the ignore-integrated but not ignore-separated conditions. Neither condition shows any learning of output gating since each register has its own mutually exclusive symbol set.

The second trend is that many models across both classes have a sharp decline in training loss, which correlates with a similarly steep increase in accuracy on both subtasks. We interpret this phase transition as suddenly learning a gating mechanism. Models that do not exhibit phase transitions to the same degree take longer to fit the task and do not reach high subtask accuracy.

While the above results suggest that it is possible for a model to achieve fine accuracy in distribution without learning a gating policy, we hypothesize that models that do learn a more general gating strategy will better generalize to out-of-distribution examples. To assess this, we held out a subset of symbols from each register (randomized which symbol would be held out from which

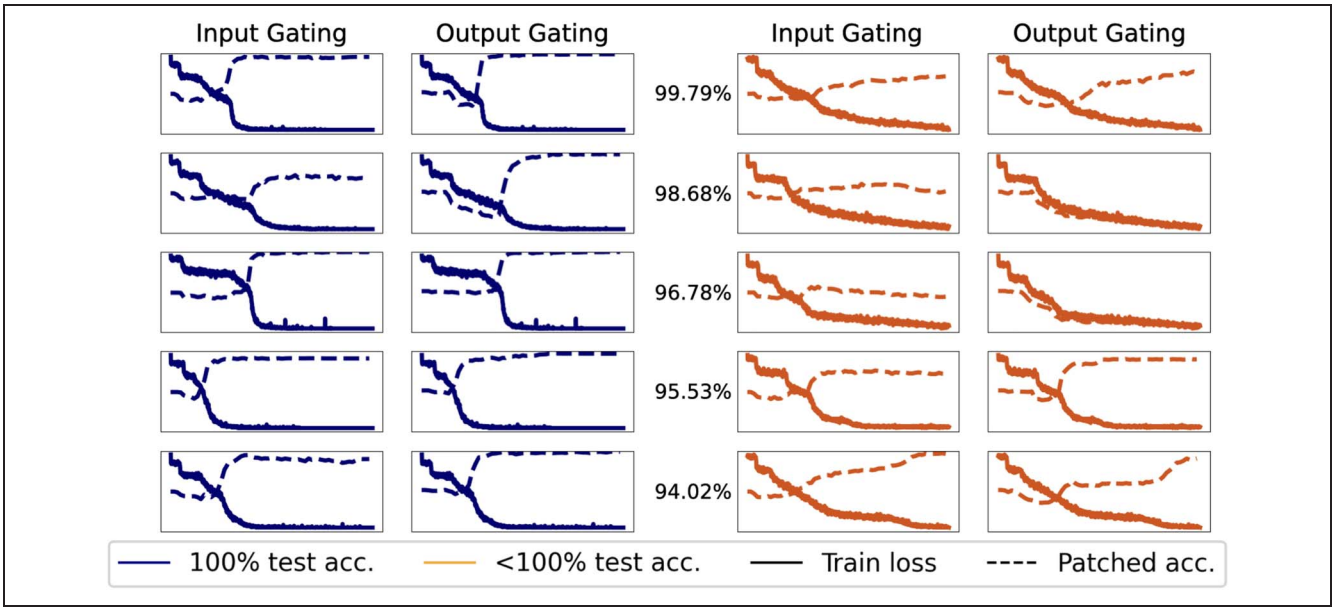


Figure 4. Model performance over training on patching subtasks. Each graph contains an individual model's training loss (solid line) and subtask accuracy (dashed line, between 0 and 1) over time; the line's color corresponds to whether the model reaches 100% accuracy on the general test set.

Downloaded from direct.mit.edu/jocn/article-pdf/38/11/2128/2617974/jocn.a.2679.pdf by University of Wisconsin, Madison, Jacqueline Fulvio on 18 November 2026

register). We tested these models on a challenge data set that included examples of in-distribution pairings (register–symbol pairings that were seen in training) and out-of-distribution pairings (register–symbol pairings that were not seen during training). When holding out 5% of the symbols, the models learn a robust input- and output-gating strategy and notably perform on average at 99.7% for out-of-distribution symbol–register pairings. In contrast, the models trained on the split-set controls do not learn robust gating strategies (despite performing perfectly at the trained task) and accordingly perform more poorly on the out-of-distribution examples: 77.2% (ignore integrated) and 88.6% (ignore separated; Table 1).[1]

Note that the split-set (ignore separated) task depends least on learning an effective gating strategy. These models learn the task very quickly and show no ability to perform on the gating subtask, and instead likely learn heuristic solutions. Interestingly, when tested on the challenge data set (which includes examples of both in-distribution and out-of-distribution register–symbol pairs), these models show a large disruption in their ability to perform on in-distribution sequences (80.8%). This insinuates that the strategy learned by these models is highly dependent on memorization, and by adding in new elements, the strategy fails. The out-of-distribution accuracy is slightly higher than the in-distribution accuracy, and future work could try to better understand the effects of perturbing a brittle model.

Gating Policy Transfers to Increased Task Demands

Thus far, the experiments have focused on tasks with two registers, mimicking that used in the reference-back 2 task (Rac-Lubashevsky & Frank, 2021). However, human WM has a capacity of about three to four items (Cowan, 2008), albeit with vigorous debates questioning whether this limit is discrete or continuous (Luck & Vogel, 2013; Wei, Wang, & Wang, 2012; Zhang & Luck, 2008). More recent models and data suggest a "chunking" hybrid between the two, whereby multiple memoranda can compete for shared continuous resources within discrete slots (Soni & Frank, 2024; Nassar et al., 2018). Chunking increases effective capacity, allowing more items to be remembered at the cost of precision of some of the items. Notably, in frontostriatal gating models, when the number of registers to manage was larger than two, networks given limited memory allocation but with chunking capabilities performed better than those that were allocated as many PFC populations as items to store (Soni & Frank, 2024). The reason for this seeming paradox is credit assignment: As the number of PFC populations (stripes) increases, the gating management problem becomes more challenging—the network has to learn to route each item to distinct populations and to also learn to read out from the corresponding population for a given probe. Moreover, these learning problems are interdependent.

These limitations are computational rather than anatomical and stem from the learning process. We hypothesize that they would also manifest in Transformers, even though Transformers have no inherent memory demands (since all information is available in the context window). Specifically because of the flexibility and expressivity of Transformers, we predicted that by increasing task demand, the network would learn the task but struggle to do so. We further predict that models that learn mechanistic solutions will generalize better to new tasks.

To test this hypothesis, we increased the number of registers from 2 to 3. First, we confirmed that asymptotic accuracy dropped to 95.4%. This result is qualitatively the same even when training for twice the number of epochs. This is nontrivial given that we are just adding one register. We predicted that the degree to which the Transformer solves the task is related to heuristics and memorization rather than gating in these cases. We predicted that if we first pretrained the model to effectively manage two registers, the network will be able to scaffold the learned gating strategy to learn the three-register task more robustly. We further predicted that this pretraining would only be useful if pretraining encouraged gating strategies. Our results in Figure 5 support both of these conclusions. Not only do pretrained networks on the original reference-back 2 task exhibit higher gating subtask accuracy (Figure 5), but networks that were pretrained with the original two-register task showed very rapid learning in the three-register task in the first few epochs. Moreover, this pretraining was far

Table 1. Accuracy for In-distribution and Out-of-distribution Symbol–Register Pairings in Challenge Data Set Experiments and Their Associated Performance on the Challenge Data Set

Experiment	*Out of Distribution*	*In Distribution*
5% Held out	99.7%	99.8%
60% Held out	95.3%	99.3%
Split-set (ignore integrated)	77.2%	94.6%
Split-set (ignore separated)	88.6%	80.8%

The challenge data set includes a mix of in-distribution and out-of-distribution symbol–register pairings. This table breaks down the accuracy based on whether the symbol–register pairing was in the training (in-distribution) or not seen during training (out-of-distribution).

Downloaded from direct.mit.edu/jocn/article-pdf/38/11/2128/2617974/jocn.a.26679.pdf by University of Wisconsin, Madison, Jacqueline Fulvio on 18 November 2026

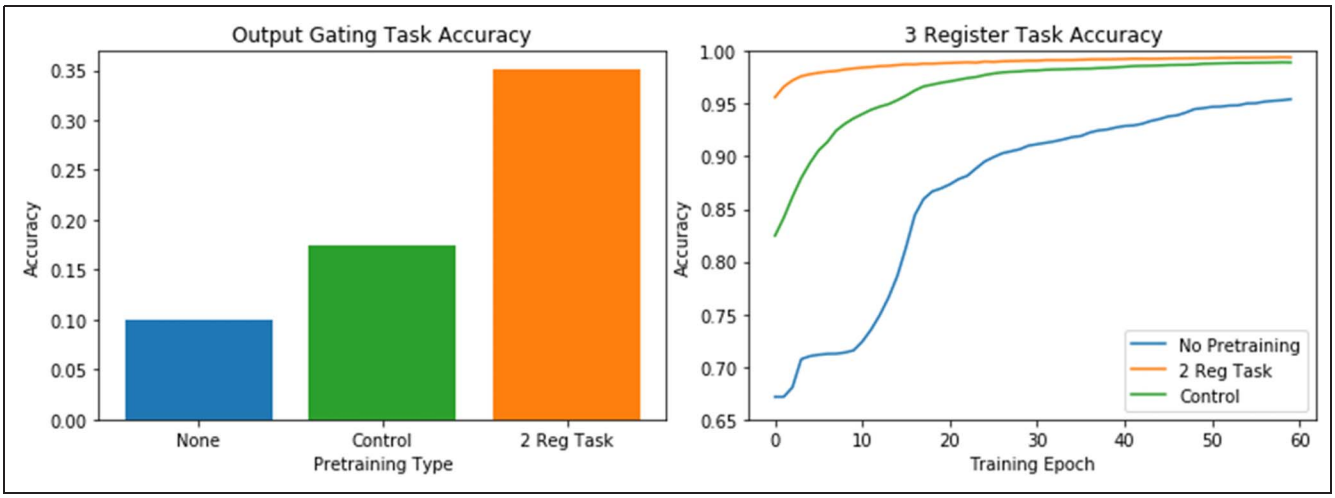


Figure 5. Three-register task and pretraining. (A) At the end of training on the three-register task, models with two-register task pretraining perform the best on output gating tasks, insinuating learning of a mechanistic solution. (B) Accuracy curves through training (after the pretraining) show a stark difference between pretraining on the normal two-register task and the control task. Models pretrained on the two-register task, generalize quickly (first few epochs) and with high accuracy (above 95%) on this new task. At the end of training, no pretraining models are below 95% accuracy—which is below the accuracy that two-register task pretraining models showed at the beginning of training. Control pretraining models show some generalization (due to learning of basic elements, e.g., symbols) but do not generalize to the same degree. Models pretrained with the two-register tasks, which should encourage a mechanistic solution, perform well at the new three-register task, showing a superior ability to generalize. Twenty-two random seeds were run for each experimental condition, and the results here are an average of those models. See Supplemental Information Input vs. Output Gating Accuracies, Pretraining for comparison on input- and output-gating task accuracies.

less effective when it did not encourage gating (the split symbol control from above.

SUMMARY AND DISCUSSION

In this work, we investigate Transformer models for emergence of a learned gating mechanism; a network component performing role-addressable gating, similar to that in WM of humans. We observe that the model learns a gating policy and find that task performance is correlated with gating ability. Our results show how learning gating mechanisms is one-way Transformers can excel at tasks that require executive function.

The Transformer models are capable of making use of key composition for input gating and query composition for output gating on the task. We find that making precise corruptions to specific architectural elements of the network causes the model's prediction to change from Same to Different or vice versa, indicating that those components are causally responsible for the gating mechanism. The architectural biases of attention within the vanilla Transformer model lend themselves well to representing role-addressable content. The learnable nature of keys, queries, and values allows the model to learn to create internal representations. These representations can be learned to signify roles and addresses, mimicking the variable binding and input-/output-gating mechanisms in biological neural networks (Collins & Frank, 2013; Kriete et al., 2013; Frank & Badre, 2012; O'Reilly & Frank, 2006).

Our finding that input and output gating were mediated by keys and queries might seem surprising, given that the two vectors always work in tandem in a Transformer. And indeed, the key-query dot product is required for attention to operate, and in that sense, both are always involved. But this is actually analogous to the relationship between input and output gating in biological systems: You cannot output gate something that was never input gated in the first place. The two operations are jointly necessary yet functionally dissociable—and our path-patching experiments directly support this dissociation. When we patch only the key vectors of a tuple, attention shifts away from that tuple, showing that keys control addressability. When we patch only the query vectors (switching the target register), attention shifts to a different stored tuple, showing that queries control which address is accessed. These are distinct interventions with distinct functional consequences, even within the same attention head.

When we trained more models on the task, we found that the models that perform best on the task correlate with the markers of gating we observed in our circuit analysis and that the learning trajectory shows a steep decrease in training loss and a steep rise in patched subtask accuracy simultaneously, suggesting that the model has learned a gating policy at that time. Both findings are analogous to those of Frank and Badre (2012), in which they find that networks that learned a hierarchical gating policy performed better at a hierarchical learning task and that humans that learn this policy also show a sharp decrease in loss when they discover it. There is still more work to be done to better understand the models that do not learn the gating policy: What types of solutions do they learn? How do we push models toward a mechanistic solution by hyperparameter tuning?

Downloaded from direct.mit.edu/jocn/article-pdf/38/11/2128/2617974/jocn.a.2679.pdf by University of Wisconsin, Madison, Jacqueline Fulvio on 18 November 2026

Understanding grokking phase planes in a similar manner to Liu et al. (2022) could be informative.

Nevertheless, we show that a critical factor controlling the learned gating policy is the task demands. We found that experimental conditions in which biological networks exhibit gating advantages were similarly needed to give rise to learned gating policies in Transformers. Conversely, when the task places less demands on role-addressable gating (our split symbol control conditions), the models (1) did not learn gating policies, (2) were less able to generalize to out-of-distribution pairs, and (3) were less able to rapidly acquire tasks with higher gating demands (three registers).

We show that models that learn and solve the tasks using gating mechanisms are better at generalization. Furthermore, we show how learning brittle and memorized solutions causes some models to falter on even in-distribution pairings when they are mixed with out-of-distribution examples. These results can be used to inform larger models that are needed for better and faster generalization. Future work can characterize what kinds of mistakes are common within the mechanistic and heuristic models to further characterize these models and promote better generalizability.

Ultimately, finding connections between emergent behavior of Transformer models and human WM serves to benefit both computational cognitive neuroscience and AI. Although Transformer models themselves are limited in their biological plausibility, in this setting they learned behavior mimicking the functionality of WM, and their application within computational models of the brain should be further explored. From the perspective of AI, understanding the strengths and limitations of Transformer models on executive function tasks may inform model analysis across the many diverse settings in which these models are applied.

Related Work

Our findings complement recent work relating Transformer key-value computations to hippocampal episodic memory (Gershman et al., 2025; Whittington et al., 2021). While episodic memory is characterized by high capacity, it is vulnerable to proactive interference that is overcome by WM (e.g., Kane & Engle, 2000). WM, by contrast, operates under a fundamentally different set of demands. It has a famously limited capacity but is specifically adapted for tasks requiring flexible, active management of currently task-relevant information—deciding what to store, when and where to store it, and, crucially, how to access it without suffering interference from earlier items that overlap in content or role. In WM, the system must learn to update stored content in a Markovian fashion: What matters is the current state of each register, not the full history of what has been stored there. This places particular demands on selective gating—both for filtering what enters memory (input gating) and for selectively retrieving the right item at the right time (output gating). The emergence of gating policies only under task conditions that demand role-addressable storage and retrieval—and their failure to emerge under simplified conditions—speaks to challenges specific to WM management, beyond what a standard key-value retrieval mechanism needs to solve. Together, these lines of work suggest that Transformer attention may provide a general substrate for multiple forms of memory, with the specific mechanisms that emerge shaped by the demands of the task on which the system is trained.

Memory-augmented neural networks such as Neural Turing Machines (Graves, Wayne, & Danihelka, 2014) and Differentiable Neural Computers (Graves et al., 2016) feature explicitly designed read/write gating mechanisms, and like the hippocampal key-value memory frameworks discussed above, their large external memory matrices and content-based addressing make them more analogous to episodic memory systems than to WM—optimized for high-capacity storage and retrieval rather than for flexible Markovian updating of a small set of actively maintained items. Our work addresses a complementary and distinct question: Can gating emerge in a system with no such machinery, purely from the demands of the task? Vanilla Transformers are well suited for this test precisely because they lack explicit gating and, for the sequence lengths used here, face no meaningful memory capacity constraints. This allows us to isolate the contribution of management and credit assignment demands from memory storage demands per se—directly testing the theoretical claim from the prefrontal cortex basal ganglia working memory (PBWM) literature that capacity limits in WM arise not from storage limitations but from the difficulty of learning role-addressable gating policies (Soni & Frank, 2024; Todd et al., 2009). The emergence of gating-like mechanisms in Transformers only under task conditions that demand role-addressable storage and retrieval—and their failure to emerge otherwise—suggests that such mechanisms reflect a fundamental computational solution to the challenges of WM management rather than merely a useful engineering choice.

Limitations

While this article draws parallels between the Transformer architecture and the brain, it is important to emphasize that there are some significant differences between how Transformers and humans solve tasks. In particular, because Transformers can attend to any part of the sequence when creating a representation, they are not limited by memory constraints. Transformers can solve tasks that would push the limits of human WM (but it should be noted that Transformers require disproportionately large amounts of training data to do so). Nevertheless, we hypothesized that Transformers might still learn effective gating policies that mimic those in frontostriatal networks. Moreover, as briefly reviewed in the

Downloaded from direct.mit.edu/jocn/article-pdf/38/11/2128/2617974/jocn.a.26679.pdf by University of Wisconsin, Madison, Jacqueline Fulvio on 18 November 2026

Introduction, WM capacity in humans and biological computational models is not limited primarily by the memory demand per se, but rather by the difficulty of the credit assignment problem for learning how to manage role-addressable storage and access of multiple items in memory. Thus it is nevertheless significant that we find that Transformers also had to confront a credit assignment problem, in that it was inefficient at learning the three-register task unless it had been pretrained to learn a gating policy within the two-register task—despite an effectively unlimited memory capacity.

The present work focuses on a single task—the reference-back 2 paradigm—chosen because it was specifically designed to place demands on role-addressable gating motivated by similar work in PBWM and has well-characterized behavioral and neural correlates in humans. Whether similar gating mechanisms would emerge in Transformers trained on other WM paradigms, such as *n*-back, delayed match-to-sample, or AX-CPT (AX-Continuous Performance Test) variants, remains an open question. We note that the original PBWM work demonstrated advantages of gating mechanisms across a family of such tasks (O'Reilly & Frank, 2006), and we expect the same principles would generalize, but systematic demonstration of this is beyond the scope of the current article. More broadly, we suspect that the most robust and flexible gating policies would emerge from training on a curriculum spanning a diverse range of tasks that collectively demand WM management—a direction we leave for future work (but see previous work on how such curriculum learning can induce generalizable strategies that mimic aspects of human behavior that likely relate to WM; Russin et al., 2025). Future work could also assess how other factors such as sequence length and task structure affect the emergence of gating in networks. While we did test these models on novel symbol–register pairings, future work could work to test the models on explicitly novel symbols. The present findings are intended as a proof of concept, establishing that gating can emerge in a system with no explicit gating machinery, that its emergence depends critically on task demands, and that when it does emerge it supports better generalization and transfer to novel task conditions.

Corresponding author: Aneri Soni, Department of Neuroscience, Brown University, 111 Thayer St., Providence, RI 02912, United States, e-mail: aneri_soni@alumni.brown.edu.

Data Availability Statement

Supplemental material can be accessed on this article's homepage: https://doi.org/10.1162/JOCN.a.2679.

Author Contributions

Aneri Soni: Conceptualization; Data curation; Formal analysis; Investigation; Methodology; Software; Validation; Visualization; Writing—Original draft; Writing—Review & editing. Aaron Traylor: Conceptualization; Data curation; Formal analysis; Investigation; Methodology; Software; Visualization; Writing—Original draft; Writing—Review & editing. Jack Merullo: Formal analysis; Methodology; Writing—Review & editing. Michael J. Frank: Conceptualization; Funding acquisition; Methodology; Project administration; Resources; Supervision; Writing—Review & editing. Ellie Pavlick: Conceptualization; Funding acquisition; Methodology; Project administration; Resources; Supervision; Writing—Review & editing.

Declaration of AI and AI-assisted Technologies

The authors declare that (AI) was not used for any aspect of the research described here, nor in manuscript preparation.

Funding Information

This work was supported by the National Institute of General Medical Sciences Centers of Biomedical Research Excellence (grant no. #5P20GM103645-10). It was also supported by the Office of Naval Research (https://dx.doi.org/10.13039/100000006, grant no. N00014-23-1-2792). E. P. was supported in part by a Young Faculty Award from the Defense Advanced Research Projects Agency of United States, grant #D24AP00261. M. J. F. was supported in part by Office of Naval Research (MURI award N00014-23-1-2792 to M. J. F.). Moreover, this work was conducted using computational resources and services at the Center for Computation and Visualization, Brown University, which is supported by the National Institutes of Health (NIH) (grant no. S10OD025181). E. P. is a paid consultant for Google DeepMind. The content of this article does not necessarily reflect the views of the US Government or of Google, and no official endorsement of this work should be inferred.

Diversity in Citation Practices

Retrospective analysis of the citations in every article published in this journal from 2010 to 2021 reveals a persistent pattern of gender imbalance: Although the proportions of authorship teams (categorized by estimated gender identification of first author/last author) publishing in the *Journal of Cognitive Neuroscience (JoCN)* during this period were M(an)/M = .407, W(oman)/M = .32, M/W = .115, and W/W = .159, the comparable proportions for the articles that these authorship teams cited were M/M = .549, W/M = .257, M/W = .109, and W/W = .085 (Postle and Fulvio, *JoCN*, 34:1, pp. 1–3). Consequently, *JoCN* encourages all authors to consider gender balance explicitly when selecting which articles to cite and gives them the opportunity to report their article's gender citation balance.

Downloaded from direct.mit.edu/jocn/article-pdf/38/11/2128/2617974/jocn.a.2679.pdf by University of Wisconsin, Madison, Jacqueline Fulvio on 18 November 2026

Note

1. One concern is that in the 5% held out, each register is trained on 49 symbols while in the split-set (ignore integrated), each register is trained on 25 symbols. While all other parameters are held constant, this difference might be big enough to account for the large difference in generalization. To account for this, we ran another set of simulations holding out 60% of the symbols (each register is trained on 35 symbols). These models robustly learn a gating policy. Even with a drastic increase in the number of held out symbols from 5% to 60%, the out-of-distribution accuracy is still very high compared with either of the split-set controls.

REFERENCES

Badre, D., & Frank, M. J. (2012). Mechanisms of hierarchical reinforcement learning in cortico-striatal circuits 2: Evidence from fMRI. *Cerebral Cortex*, *22*, 527–536. https://doi.org/10.1093/cercor/bhr117, PubMed: 21693491

Bahdanau, D., Cho, K., & Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. *arXiv*. https://doi.org/10.48550/arXiv.1409.0473

Baier, B., Karnath, H. O., Dieterich, M., Birklein, F., Heinze, C., & Müller, N. G. (2010). Keeping memory clear and stable—The contribution of 18 human basal ganglia and prefrontal cortex to working memory. *Journal of Neuroscience*, *30*, 9788–9792. https://doi.org/10.1523/JNEUROSCI.1513-10.2010, PubMed: 20660261

Bhandari, A., & Badre, D. (2018). Learning and transfer of working memory gating policies. *Cognition*, *172*, 89–100. https://doi.org/10.1016/j.cognition.2017.12.001, PubMed: 29245108

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., et al. (2020). Language models are few-shot learners. *Advances in Neural Information Processing Systems*, *33*, 1877–1901. https://doi.org/10.48550/arXiv.2005.14165

Burtsev, M. S., Kuratov, Y., Peganov, A., & Sapunov, G. V. (2020). Memory transformer. *arXiv*. https://doi.org/10.48550/arXiv.2006.11527

Calderon, C. B., Verguts, T., & Frank, M. J. (2022). Thunderstruck: The acdc model of flexible sequences and rhythms in recurrent neural circuits. *PLoS Computational Biology*, *18*, e1009854. https://doi.org/10.1371/journal.pcbi.1009854, PubMed: 35108283

Chatham, C. H., Frank, M. J., & Badre, D. (2014). Corticostriatal output gating during selection from working memory. *Neuron*, *81*, 930–942. https://doi.org/10.1016/j.neuron.2014.01.002, PubMed: 24559680

Collins, A. G., & Frank, M. J. (2013). Cognitive control over learning: Creating, clustering, and generalizing task-set structure. *Psychological Review*, *120*, 190–229. https://doi.org/10.1037/a0030852, PubMed: 23356780

Cowan, N. (2008). What are the differences between long-term, short-term, and working memory? *Progress in Brain Research*, *169*, 323–338. https://doi.org/10.1016/S0079-6123(07)00020-9, PubMed: 18394484

Dai, Z., Yang, Z., Yang, Y., Carbonell, J., Le, Q. V., & Salakhutdinov, R. (2019). Transformer-xl: Attentive language models beyond a fixed-length context. *arXiv*. https://doi.org/10.18653/v1/P19-1285

Dayan, P. (2012). How to set the switches on this thing. *Current Opinion in Neurobiology*, *22*, 1068–1074. https://doi.org/10.1016/j.conb.2012.05.011, PubMed: 22704797

Frank, M. J., & Badre, D. (2012). Mechanisms of hierarchical reinforcement learning in corticostriatal circuits 1: Computational analysis. *Cerebral Cortex*, *22*, 509–526. https://doi.org/10.1093/cercor/bhr114, PubMed: 21693490

Gershman, S. J., Fiete, I., & Irie, K. (2025). Key-value memory in the brain. *Neuron*, *113*, 1694–1707. https://doi.org/10.1016/j.neuron.2025.02.029, PubMed: 40147436

Goldowsky-Dill, N., MacLeod, C., Sato, L., & Arora, A. (2023). Localizing model behavior with path patching. *arXiv*. https://doi.org/10.48550/arXiv.2304.05969

Graves, A., Wayne, G., & Danihelka, I. (2014). Neural turing machines. *arXiv*. https://doi.org/10.48550/arXiv.1410.5401

Graves, A., Wayne, G., Reynolds, M., Harley, T., Danihelka, I., Grabska-Barwińska, A., et al. (2016). Hybrid computing using a neural network with dynamic external memory. *Nature*, *538*, 471–476. https://doi.org/10.1038/nature20101, PubMed: 27732574

Hochreiter, S. (1997). Long short-term memory. *Neural Computation*, *9*, 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735, PubMed: 9377276

Kane, M. J., & Engle, R. W. (2000). Working-memory capacity, proactive interference, and divided attention: Limits on long-term memory retrieval. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, *26*, 336–358. https://doi.org/10.1037/0278-7393.26.2.336, PubMed: 10764100

Kriete, T., Noelle, D. C., Cohen, J. D., & O'Reilly, R. C. (2013). Indirection and symbol-like processing in the prefrontal cortex and basal ganglia. *Proceedings of the National Academy of Sciences, U.S.A.*, *110*, 16390–16395. https://doi.org/10.1073/pnas.1303547110, PubMed: 24062434

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, *521*, 436–444. https://doi.org/10.1038/nature14539, PubMed: 26017442

Liu, Z., Kitouni, O., Nolte, N., Michaud, E. J., Tegmark, M., & Williams, M. (2022). Towards understanding grokking: An effective theory of representation learning. In *NeurIPs*. https://doi.org/10.48550/arXiv.2209.11143

Luck, S. J., & Vogel, E. K. (2013). Visual working memory capacity: From psychophysics and neurobiology to individual differences. *Trends in Cognitive Sciences*, *17*, 391–400. https://doi.org/10.1016/j.tics.2013.06.006, PubMed: 23850263

Ma, W. J., Husain, M., & Bays, P. M. (2014). Changing concepts of working memory. *Nature Neuroscince*, *17*, 347–356. https://doi.org/10.1038/nn.3655, PubMed: 24569831

McGrath, S. W., Russin, J., Pavlick, E., & Feiman, R. (2024). How can deep neural networks inform theory in psychological science? *Current Directions in Psychological Science*, *33*, 325–333. https://doi.org/10.1177/09637214241268098, PubMed: 39949337

McNab, F., & Klingberg, T. (2008). Prefrontal cortex and basal ganglia control access to working memory. *Nature Neuroscience*, *11*, 103–107. https://doi.org/10.1038/nn2024, PubMed: 18066057

Nanda, N., & Bloom, J. (2022). *Transformer lens*. GitHub. https://github.com/neelnanda-io/TransformerLens

Nassar, M. R., Helmers, J. C., & Frank, M. J. (2018). Chunking as a rational strategy for lossy data compression in visual working memory. *Psychological Review*, *125*, 486–511. https://doi.org/10.1037/rev0000101, PubMed: 29952621

O'Reilly, R. C., & Frank, M. J. (2006). Making working memory work: A computational model of learning in the prefrontal cortex and basal ganglia. *Neural Computation*, *18*, 283–328. https://doi.org/10.1162/089976606775093909, PubMed: 16378516

Oberauer, K. (2013). The focus of attention in working memory—From metaphors to mechanisms. *Frontiers in Human Neuroscience*, *7*, 673. https://doi.org/10.3389/fnhum.2013.00673, PubMed: 24146644

Downloaded from direct.mit.edu/jocn/article-pdf/38/11/2128/2617974/jocn.a.2679.pdf by University of Wisconsin, Madison, Jacqueline Fulvio on 18 November 2026

Oberauer, K., & Lin, H.-Y. (2017). An interference model of visual working memory. *Psychological Review*, *124*, 21–59. https://doi.org/10.1037/rev0000044, PubMed: 27869455

Olah, C. (2022). Mechanistic interpretability, variables, and the importance of interpretable bases. https://www.transformer-circuits.pub/2022/mech-interp-essay

Pearl, J. (2001). Direct and indirect effects. In *Proceedings of the Seventeenth Conference on Uncertainty in Artificial Intelligence* (pp. 411–420). Morgan Kaufmann.

Rac-Lubashevsky, R., & Frank, M. J. (2021). Analogous computations in working memory input, output and motor gating: Electrophysiological and computational modeling evidence. *PLoS Computational Biology*, *17*, e1008971. https://doi.org/10.1371/journal.pcbi.1008971, PubMed: 34097689

Rac-Lubashevsky, R., & Kessler, Y. (2016). Dissociating working memory updating and automatic updating: The reference-back paradigm. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, *42*, 951–969. https://doi.org/10.1037/xlm0000223, PubMed: 26618910

Rose, N. S., LaRocque, J. J., Riggall, A. C., Gosseries, O., Starrett, M. J., Meyering, E. E., et al. (2016). Reactivation of latent working memories with transcranial magnetic stimulation. *Science*, *354*, 1136–1139. https://doi.org/10.1126/science.aah7011, PubMed: 27934762

Russin, J., Pavlick, E., & Frank, M. J. (2025). Parallel trade-offs in human cognition and neural networks: The dynamic interplay between in-context and in-weight learning. *Proceedings of the National Academy of Sciences, U.S.A.*, *122*, e2510270122. https://doi.org/10.1073/pnas.2510270122, PubMed: 40875812

Soni, A., & Frank, M. J. (2024). Adaptive chunking improves effective working memory capacity in a prefrontal cortex and basal ganglia circuit. *eLife*, *13*, RP97894. https://doi.org/10.7554/eLife.97894, PubMed: 40019141

Swan, G., & Wyble, B. (2014). The binding pool: A model of shared neural resources for distinct items in visual working memory. *Attention, Perception, & Psychophysics*, *76*, 2136–2157. https://doi.org/10.3758/s13414-014-0633-3, PubMed: 24634029

Todd, M. T., Niv, Y., & Cohen, J. D. (2009). Learning to use working memory in partially observable environments through dopaminergic reinforcement. In *Advances in neural information processing systems 21—Proceedings of the 2008 Conference*. Princeton University.

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., et al. (2023). Llama: Open and efficient foundation language models. *arXiv*. https://doi.org/10.48550/arXiv.2302.13971

van den Berg, R., Awh, E., & Ma, W. J. (2014). Factorial comparison of working memory models. *Psychological Review*, *121*, 124–149. https://doi.org/10.1037/a0035234, PubMed: 24490791

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., et al. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*, *30*. https://doi.org/10.48550/arXiv.1706.03762

Vogel, E. K., McCollough, A. W., & Machizawa, M. G. (2005). Neural measures reveal individual differences in controlling access to working memory. *Nature*, *438*, 500–503. https://doi.org/10.1038/nature04171, PubMed: 16306992

Wang, Y.-S., Lee, H.-Y., & Chen, Y.-N. (2019). Tree transformer: Integrating tree structures into self-attention. *arXiv*. https://doi.org/10.48550/arXiv.1909.06639

Wang, K., Variengien, A., Conmy, A., Shlegeris, B., & Steinhardt, J. (2022). Interpretability in the wild: A circuit for indirect object identification in gpt-2 small. *arXiv*. https://doi.org/10.48550/arXiv.2211.00593

Wei, Z., Wang, X. J., & Wang, D. H. (2012). From distributed resources to limited slots in multiple-item working memory: A spiking network model with normalization. *Journal of Neuroscience*, *32*, 11228–11240. https://doi.org/10.1523/JNEUROSCI.0735-12.2012, PubMed: 22895707

Whittington, J. C., Warren, J., & Behrens, T. E. (2021). Relating transformers to models and neural representations of the hippocampal formation. *arXiv*. https://doi.org/10.48550/arXiv.2112.04035

Zhang, W., & Luck, S. J. (2008). Discrete fixed-resolution representations in visual working memory. *Nature*, *453*, 233–235. https://doi.org/10.1038/nature06860

Downloaded from direct.mit.edu/jocn/article-pdf/38/11/2128/2617974/jocn.a.2679.pdf by University of Wisconsin, Madison, Jacqueline Fulvio on 18 November 2026