MEMORY PRESSURE IN RECURRENT MODELS: BEHAVIOR, MECHANISMS AND REPRESENTATIONS

Anonymous authors
Paper under double-blind review

ABSTRACT

While you read this abstract, try to remember how many points fine-tuning adds to cued recall. This instruction is already changing how your brain registers this abstract, and arguably is a major difference between recurrent models (RMs) with finite state and Transformers, which can attend back to any context. Consequently, an RM must commit to its finite memory what it thinks will be relevant in the future and compress information on the go. This intuition behind state and data dependent memory drove recent architectural designs like Gated DeltaNet and RWKV-7. However, we find that language-trained RMs do not actually behave this way. In a task we call cued key–value recall, a cue that names the future query helps mostly through exact matching: the models barely process its meaning, and telling them to ignore a record still helps them recall it. Fine-tuning adds 89 points to RWKV-7's cued recall (8% to 97% at 256 biographies), yet what the models learn remains largely token matching. Trained from scratch on a synthetic version of the task, RWKV-7 behaves rationally: uncued recall is limited by memory capacity, cued recall by length generalization, and a smaller state starts relying on the cue sooner. But the same limit that hurts recall can help learning: a finite state must compress, and we find that this *inference-time compression pressure* can actually push a model towards learning better features and concepts. When trained only to recall given pixels of a digit, an RM learns to draw the whole digit, while a Transformer trained on the same data and loss simply copies from its inputs. Finally, we show that these results compile into a phase diagram where task load and memory load both contribute to inference-time pressure, creating better-generalizing features. Now, do you remember how many points fine-tuning added? Perhaps the cue helped, even though you do not have quadratic attention.

1 INTRODUCTION

Imagine driving to work while dozens of cars pass by (Fig. 1a). When you arrive, a co-worker asks how many of them were red. If you are a human, you almost certainly cannot answer. However, if your co-worker had asked the question before you left, the task would likely not be too hard for humans as well.

This example demonstrates that humans have evolved to save the most important information by letting their goals shape what they perceive and store. Attending to one aspect of a scene can hide even a salient unexpected event (Simons & Chabris, 1999). Simply telling people what to remember changes what they store: items followed by an instruction to remember are recalled better than items followed by an instruction to forget (Hall et al., 2021), and people selectively remember the items they are told are worth more (Castel et al., 2002; Knowlton & Castel, 2022).

A recurrent model (RM)[1] is in the human's position (Fig. 1b). It must compress all useful incoming information into a finite state, and thus it must decide what to keep before it knows what will be asked. A Transformer (Vaswani et al., 2023) given a video of the drive has no such problem, since self-attention always allows it to look back at any part of the context. This is arguably the single biggest difference between RMs and Transformers.

[1]We use the term *recurrent model* (RM) for any sequence model with a fixed-size state: RNNs, LSTMs, GRUs, Linear Transformers, Mamba, DeltaNet, Gated DeltaNet, RWKV and their relatives.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

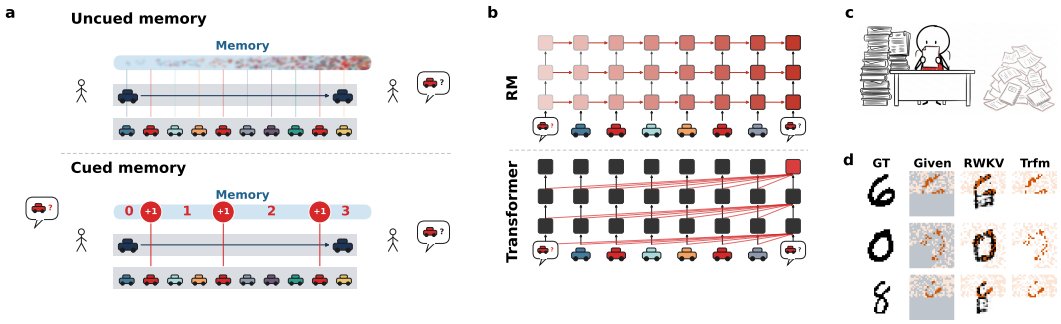


Figure 1: A finite memory system must decide what to keep before the question arrives, and the same pressure can help learning. a. A driver passes a stream of cars on the way to work. An uncued driver probably can't answer how many red cars they saw, but a driver who already know the question can easily perform the task. **b.** An RM carries a fixed-size state from token to token, so what it keeps is decided as each car arrives. A Transformer, however, can always attend back to every token and easily retrieve what it needs. **c.** A person reading 100 papers likely cannot remember even a single abstract perfectly. However, they end up with a highly compressed world model on the topic. **d.** We find *inference-time compression pressure* in action when training sequence models to simply recall the pixels values they are given. Observed pixels are in orange and hidden pixels in gray. RWKV-7, an RM, fills in the hidden part of the digit, while a Transformer (Trfm) reproduces only the given pixels. The RWKV has learned a compressed concept while the Transformer simply recalls.

This difference drives much of RM architecture design. Many recent RMs are built precisely so that the state can choose what to keep: input-dependent gates and decays decide what to forget (Peng et al., 2023; Gu & Dao, 2024; De et al., 2024; Yang et al., 2025a; Peng et al., 2025; Team et al., 2025), the delta rule overwrites stale associations (Yang et al., 2025b), and test-time learning trains the state as a memory on the fly (Sun et al., 2025; Behrouz et al., 2024; 2025). In other words, how to selectively save and forget is at the core of these architectures' design. However, outside the minds of their designers, these architectures are evaluated like any other Transformer: trained on standard data and scored on scaling curves, throughput and benchmarks.

This work. This leaves a basic question open. Does a language-trained RM use its state the way its mechanism intends? Simply told what will matter, does it change what it saves to memory? We test this with a task we call "cued key–value recall": a typical key–value recall task, but where the key is partially or completely revealed by a cue given before the list. We find that models respond to exact matches but barely process the meaning of the cue when deciding what to save. Trained from scratch on a synthetic version of the task, the same architectures indeed behave rationally: they still save by exact match, but use the cue far more effectively than language-trained RMs, relying on it exactly when their state can no longer hold everything. Finally, limited memory does not only hurt. After reading a hundred papers, you have forgotten almost every detail, yet you have likely built a far richer set of features and a compressed model of the field (Fig. 1c). Cognitive scientists have similarly argued that limits on processing can help a learner find structure (Hertwig & Todd, 2003; Cochran et al., 1999). We find the same in neural networks: just as compression pressure during training is thought to drive representation learning, compression pressure at inference time pushes an RM toward better features and concepts. **Overall, our contributions are:**

1. **Pretrained RMs mostly lack sophisticated memory control (Sec. 3, Fig. 2, 3):** In a cued recall task, only an exact-match cue improves recall. Cues whose meaning must be semantically processed do nothing, and telling a model to ignore a record gains similarly to telling it to remember it. Fine-tuning strengthens both cued and uncued recall, but the exact match property mostly remains.
2. **Synthetic experiments showing how different architectures modulate their memory (Sec. 3.4, Fig. 4):** We find that model architectures train quite rationally: RWKV-7 learns to rely on the cue when its state can no longer hold every record, and uncued recall is limited by memory capacity while cued recall is limited by length generalization. Mechanistically, we find that the cue is held in the first layer's state and used by the layers above to decide what to store.
3. **Inference-time compression pressure can improve feature and concept learning (Sec. 4, Fig. 5, 6):** We show that a limited inference-time memory can force the model to learn more features. Trained only to recall the pixels it is given, an RM learns features that complete unseen

digits and characters and support classification and concept readout, while a Transformer trained on the same data and loss simply copies.

4. **A phase diagram of inference-time compression pressure (Sec. 4.4, Fig. 7):** We draw a phase diagram which shows how the inference-time compression pressure is clearly modulated by memory size and by how much the cue reveals about the query. The pressure helps in a middle regime: a moderate memory paired with an uninformative cue.

2 PRELIMINARIES

We discuss related work in App. A. One can write a fixed-state sequence model as:

$$s_t = F_\theta(s_{t-1}, x_t), \qquad y_t = G_\theta(s_t, x_t), \tag{1}$$

where the dimensionality of s_t does not grow with t. For a multi-layer model, s_t collects the states of all layers and each layer updates its state from its previous state and the output of the layer below. A full-attention Transformer can only be written in this form if the "state" contains the whole key–value cache, which grows linearly with the context. RWKV carries per-layer matrix states; Mamba uses selective state-space dynamics; Gated DeltaNet combines decay with a delta-rule update (Gu & Dao, 2024; Peng et al., 2025; Yang et al., 2025a).

It is worth pointing out that at least two layers are required to do *state* (and thus arbitrary history) dependent saving in the linear RMs we study, since a layer's gates are computed from its input rather than from its state. A lower layer can, however, pass a stored cue up to the gates of the layer above, which is how the trained model implements cued saving (More in Sec. 3.4). Nonlinear RNNs such as LSTMs compute their gates from the previous state and can do this within one layer.

3 RECURRENT MODELS MOSTLY RESPOND TO EXACT MATCH CUES

3.1 CUED KEY–VALUE RECALL

We design a simple "cued key–value recall" task (Fig. 2a) where the context is a list of N short fictional biographies, followed by a question about one attribute of one person. Each biography consists of a person's name, birth date and place, nationality, residence, phone number and email. Unlike typical key–value recall (Graves et al., 2014; Ba et al., 2016) (also sometimes called needle in a haystack (Kamradt, 2023)), the model gets a cue *before* the list arrives. The cue narrows down which person the question will be about: it can give the person's exact name, an exact value from their profile, their initials, a coarse feature such as their birth decade, or the attribute that will be asked. The cue can also tell the model to ignore a person rather than remember them.

We generally measure recall accuracy across different numbers of biographies. We also often report the cue effect in points: recall with the cue minus recall without it, on the same list. Note that we explore both benign cues, which point toward the person asked about (remember A, or ignore another person B), and adversarial cues, which point away (remember B, or ignore A); a rational memory policy gains from the first and loses from the second. For further details, see App. B.2, C.2.

3.2 PRETRAINED RWKV-7 MOSTLY RESPONDS TO EXACT-MATCH CUES

First of all, and perhaps most surprisingly, among the many language-pretrained models we test, only RWKV-7 reliably responds to any cue! At 1.3–1.5B parameters and 16 biographies, the cue effects for DeltaNet (Yang et al., 2025b), Gated DeltaNet (Yang et al., 2025a), GLA (Yang et al., 2024), RetNet (Sun et al., 2023), two Gated DeltaNet–attention hybrids (Wang et al., 2026) and Pythia (Biderman et al., 2023) are all within a few points of zero (Fig. 2f; App. C.1). Note that this is not because the models are at ceiling: their uncued recall has already fallen to 5–10%, and even closer to their own recall limit (1–8 biographies) the cue helps by at most about 13 points (Fig. 9b, d).

For RWKV-7, the one model that listens, an exact-name cue starts helping when recall starts falling (Fig. 2b, c). As we scale the model, the cue effect starts later, since recall stays at ceiling longer, and the benefit grows: within the G1i line it peaks at 18 points for 2.9B and 42 for 7.2B (64 biographies), and at 52 for 13.3B (128 biographies). However, we find that the model isn't really processing the cue semantically (Fig. 2d, G1i 13.3B). Only cues with an exact matching string elicit any cue effect:

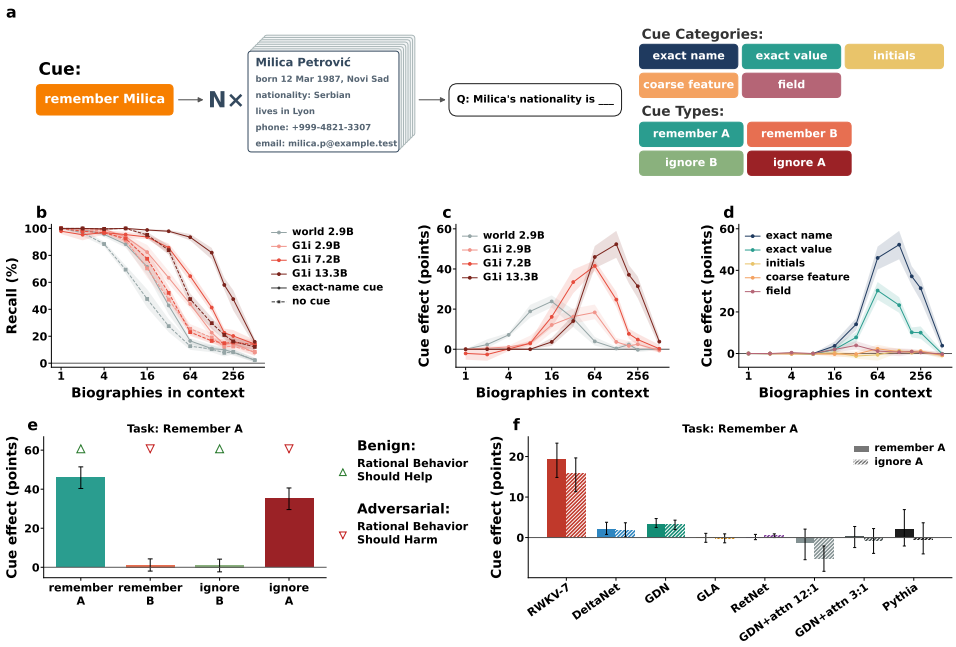


Figure 2: Pretrained RMs mostly respond only to exact match cues. a. The cued key–value recall task. Similar to a biography based key–value recall task, but a cue narrows down who the question will be about. The cue can be more or less precise, but could also be adversarial. (b) Recall accuracy with an exact-name cue and without for four pretrained RWKV-7 checkpoints (World 2.9B and G1i 2.9B, 7.2B, 13.3B) across different biography sizes. (c) The isolated cue effect grows with scale and peaks at longer contexts in larger models. (d) For RWKV G1i 13.3B, only cues that repeat exact text from the record help: the exact name or an exact field value. Initials, coarse features and the queried field have no effect. (e) When the task is to remember A, telling the model to ignore A still helps recall of A by 35 points, most of the 46-point benefit of telling it to remember A, where a rational policy would be harmed rather than helped. Cues about another person have no effect. (f) Remember-A and ignore-A cue effects on different models. Only RWKV-7 shows an effect, and negation barely reduces it. Shading and error bars: 95% bootstrap intervals over eight content banks.

the full name, or an exact value from another field of the record (up to 30 points). Any other cue, e.g., the person's initials, a coarse feature such as their birth decade, or the field that will be asked, does not help recall. Furthermore, as seen in Fig. 2e, negative cues, which should harm recall, in fact help almost as much as positive ones: telling the model to ignore A still raises recall of A by 35 points, against 46 for telling it to remember A.

Overall, pretrained RMs mostly ignore the cue, and RWKV-7, the one that uses it, does so by exact matching rather than by understanding it.

3.3 Fine-tuned RMs follow the cue, but still mostly by matching its tokens

Disappointed by the models' response to the cue, we next try to fine-tune them for this behavior. We fine-tune each model on 12,000 biography episodes, all with benign cues and correct answers (one training seed per model). We find that for RWKV-7, fine-tuning greatly improves both uncued and cued recall: at 64 biographies uncued recall rises from 13% to 84%, and cued recall stays at 97% even at 256 biographies, up from 8% before fine-tuning, an 89-point gain (Fig. 3a). Gated DeltaNet instead improves mostly when cued: its uncued recall stays at 12–17%, but cued recall reaches 84–93% up to 256 biographies. GLA and RetNet do not acquire such behavior (App. C.3).

However, semantic processing of the cue still *mostly* remains disappointing (Fig. 3b, c). All models still show a positive effect of an adversarial cue, although smaller than that of the benign one: telling a model to ignore A still raises recall of A by 58 (Gated DeltaNet), 42 (DeltaNet) and 9 (RWKV-7) points (Fig. 3c). Most gains are also still limited to exact matches. *There is some hope, however: RWKV-7 does start responding to vague cues* such as initials (20 points for World 2.9B, 36 for G1i 13.3B) and coarse features (9 and 13), while Gated DeltaNet is strictly limited to the exact name

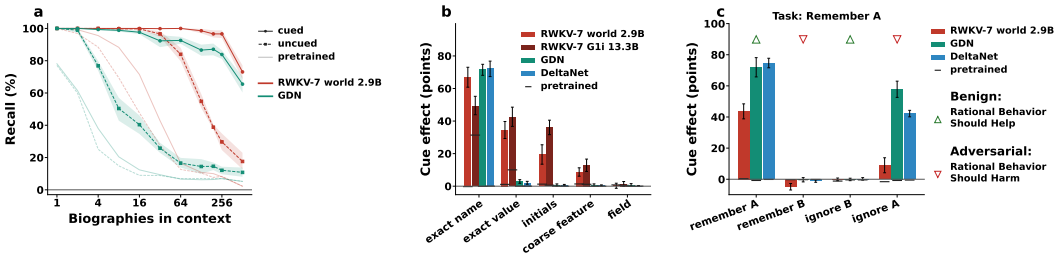


Figure 3: Fine-tuning makes the cue work, but how much of its meaning is used depends on the architecture. Models are fine-tuned on benign cues only, including vague ones (initials, coarse features, the queried field) and negative ones ("ignore B"); faint lines and black ticks show the same models before fine-tuning. **a.** With an exact-name cue, RWKV-7 greatly improves both uncued and cued recall, while GDN improves mostly on cued recall. **b.** Cue effects by cue type at 256 biographies. GDN and DeltaNet still use only the exact name, whereas RWKV-7 also uses exact values, initials and, more weakly, coarse features. No model uses the field cue. **c.** Cue effects at 128 biographies when the question is about A. "Remember A" helps every model, but so does the adversarial "ignore A" (strongly for GDN and DeltaNet), while the benign "ignore B" does nothing, despite explicit training on such negative cues. Shading and error bars: 95% bootstrap intervals over eight content banks.

(Fig. 3b). No model learns to use a cue naming the field that will be asked, even though such cues made up a sixth of the training data. Note that we do not measure whether the fine-tuned models change what they write to state, or implement the cue through another query-time strategy.

Overall, fine-tuning shows that the capacity to follow the cue was there all along, but what is learned stays largely token-based. Whether a recurrent architecture discovers the policy when trained from scratch, and which details of the architecture decide that, is what we test next in synthetic episodes where every factor can be controlled.

3.4 Trained from scratch, RWKV-7 learns a rational cue policy

Since RWKV-7 was the most promising model so far, we now ask whether it can learn the cue policy from scratch, in a synthetic version of the task where nothing but the cue tells the model what to keep. Each episode is a cue key, then N random key–value records, then a query (Fig. 4a); since keys and values are random tokens, no record is more salient than another.

And indeed, RWKV-7 learns it. A small four-layer RWKV-7 (16,384 state floats) trained up to $N = 1024$ recalls 95% of cued records at $N = 1024$ but only 7.5% of uncued ones (Fig. 4c). A Transformer, on the other hand, recalls everything with or without the cue within its training length ($N \leq 1024$), so the cue buys it nothing (App. D.1). Training on longer lists, uncued recall always collapses along the same curve, set by memory capacity, while cued recall extends with the training length, set by length generalization (Fig. 4b).

The policy is also rational: the cue starts to matter once the state is full, so the smaller the state, the sooner this happens (Fig. 4d). At $N = 128$, the cue is worth 0.77 for hidden size $d = 128$, 0.28 for $d = 256$ and almost nothing for $d = 512$, whose state still holds most records. Interestingly, a model whose state holds its whole training range may never learn the cue at all: $d = 512$ trained up to $N = 512$ shows a cue effect of at most 0.04, but reaches 0.79 when trained up to $N = 1024$ (App. D.3; one seed per cell).

Finally, the cue lives in the layer-1 state (Fig. 4e). Transplanting only the layer-1 state after the cue into an uncued run restores cued recall, while layers 2–4, token shifts or a wrong-key state do nothing. Layer 1 holds the cue, and the layers above use it to decide what to store (App. D.4). Other RMs are less reliable: with a matched state, Gated DeltaNet learns the policy only when warm-started from shorter training or trained three times longer, and DeltaNet in two of three seeds (App. D.2).

Overall, trained on data that rewards it, RWKV-7 saves what the cue names exactly when its state can no longer hold everything.

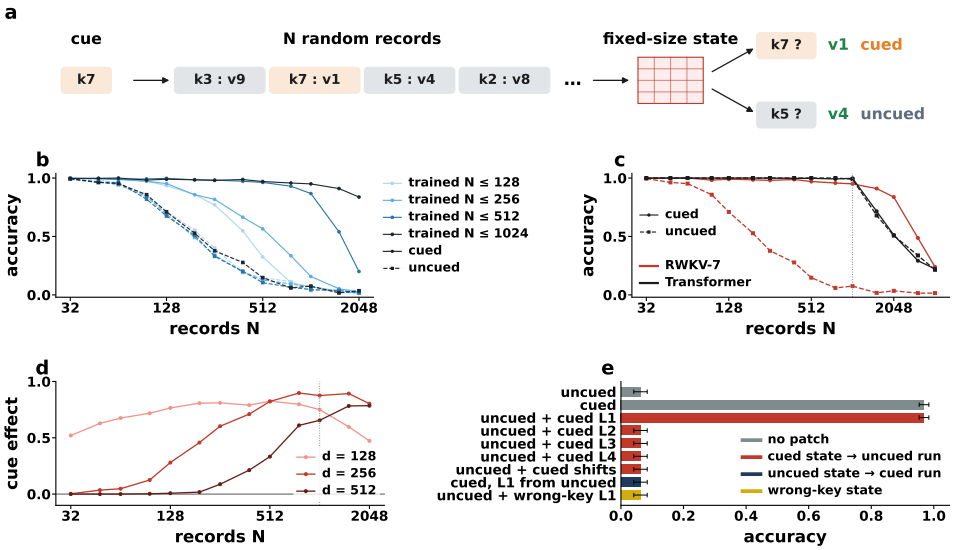


Figure 4: Trained from scratch, RWKV-7 uses the cue rationally. a. Synthetic cued key–value recall: a cue names one key, then N random key–value records follow, and the query asks for the value of either the cued key or another key. **b.** RWKV-7 trained up to different lengths N. Uncued recall collapses onto one curve for every training length, set by memory capacity, while cued recall extends with the training length, set by length generalization. **c.** A Transformer recalls everything with or without the cue within its training length ($N \leq 1024$), so the cue buys it nothing, while RWKV-7 needs the cue once the records exceed its state (dotted line: training length, $N = 1024$). **d.** The smaller the state (hidden size d), the sooner the cue starts to matter. **e.** Transplanting only the layer-1 state after the cue into an uncued run restores cued recall, while layers 2–4, token shifts or a wrong-key state do not; conversely, a cued run with an uncued layer-1 state loses it. Error bars: standard error over 128 documents. One training seed per model.

4 Inference-time pressure can improve learned features

4.1 From known relevance to unknown future use

Cued recall assumes that the future relevance is partly given during inference. But what should a bounded model do when the future task is not known? We argue the best it can do is to learn the shared structure underying the task to store the incoming evidence efficiently.

This is directly related to the common statement "intelligence is compression". This statement is often framed as a pressure during pretraining, but here we argue that an *inference-time compression pressure*, exists for recurrent models.

We therefore ask the following: *can inference-time pressure change the learned solution for the same task/learning rule?*

4.2 Inference-time pressure forces a generalizable summary state

We devise a task (Fig. 5a) where *random pixels* from a binarized MNIST (Lecun et al., 1998) digit are given as a list of (x, y, value) tuples in random order. During training, we only ask for pixels that are already in the context. Thus literal key–value recall is a perfect solution, since missing pixels never contribute to the loss.

We then test the model on pixels which are *not* specified by the context. First, quantitatively, we find that both models recall observed pixels (balanced accuracy 0.97 for RWKV-7 and 1.00 for a parameter-matched Transformer with Canon layers), but only RWKV-7 completes the missing ones: 0.86, against 0.50 for the Transformer, which simply predicts background (Fig. 5b). This completion uses the episode itself: replacing the context with an ink-matched image of a different digit drops RWKV-7 to 0.64. Qualitatively, the reconstructions in Fig. 5c clearly show that the Transformer barely predicts ink outside of what the context specifies, while RWKV-7 completes the digit. The MNIST-only state even transfers to EMNIST (Cohen et al., 2017) letters, reaching 0.80 on missing

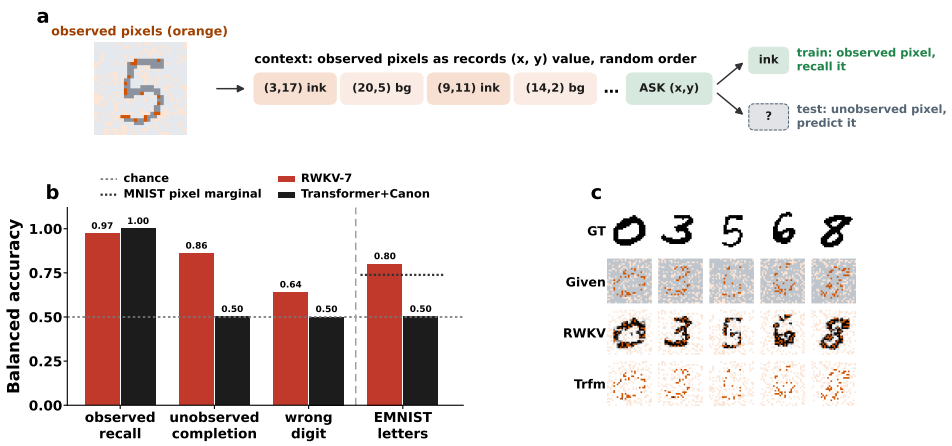


Figure 5: Inference-time pressure forces models to learn generalizable summary states. a. Partial pixels of an MNIST digit are given as $(x, y, value)$ records in random order. During training *we only query for observed pixels* so copying solves it, if the model has enough memory. **b.** Both the Transformer and RWKV-7 manage to recall seen pixels, but only RWKV-7 completes unobserved ones, and this also transfers to EMNIST letters. The Transformer stays at chance level. **c.** RWKV-7 can roughly reconstruct missing pixels from random conditionings while the Transformer returns only the given pixels; more examples in App. E.1.

pixels against 0.74 for the MNIST pixel-frequency baseline and 0.50 for the Transformer (Fig. 5b; App. E.2); this is evidence for reusable spatial structure, not letter recognition. All comparisons here use one training seed per architecture.

Overall, trained only to recall what it has seen, RWKV-7 learns a summary of the digit that completes what it has not seen, while a Transformer that can simply copy learns nothing beyond the copy.

4.3 Inference-time pressure also yields features and concepts

We now ask a deeper question: does inference-time pressure merely force a general smoothing of the data, or does it learn actual features and concepts?

To test this, we hide large parts of MNIST digits, building contexts unlike any seen during training (Fig. 6a). *Surprisingly, RWKV-7 completes digits far outside the given region!* It draws the whole lower loop of the 8, and completes the 3 whose top half is hidden as a 9, a reasonable guess from what is visible, while the Transformer again returns only the given pixels.

The learned state also makes classification more label-efficient. We train a new classification token and head on top of the recall-trained models (Fig. 6b): with 16 labels per class, RWKV-7 reaches 0.51 balanced accuracy against 0.17 for the Transformer (0.93 against 0.76 with all labels), while an MLP trained on the same shuffled records barely learns. This advantage is relative to models that read serialized records; a masked-recall autoencoder that sees the 2-D image does better (App. E.3).

Finally, we test on Hanzi, a script with far richer structure. Trained only to recall observed pixels of 930 characters, RWKV-7 reconstructs the stroke structure of held-out characters, while a Transformer with full memory does not, and one with a bounded memory (512 key–value tokens) does so only in some seeds (Fig. 6c; App. E.4). A linear probe on the RWKV-7 end-of-context state also reads out the characters' radicals with macro AUC 0.91 (mean of three seeds), against 0.79 for the bounded Transformer and 0.54 for full attention (Fig. 6d; App. E.4). The raw visible pixels give 0.95, so the state keeps the radical information in the context rather than adding to it; for full attention, the end-of-context state is also not where its memory lives.

Overall, the state learned under inference-time pressure is more than a smoothing: it completes digits from their shape, makes classification label-efficient, and keeps concept-level information such as Hanzi radicals.

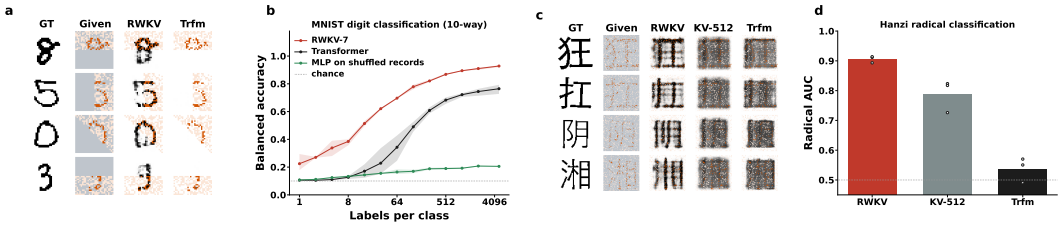


Figure 6: Features learned under inference-time pressure support completion, classification and concepts. **a.** With large regions hidden, RWKV-7 draws plausible missing strokes, while the Transformer returns only the given pixels. **b.** A classification token trained on top of the recall-trained RWKV-7 needs far fewer labels to classify digits than one on the Transformer, or an MLP given the same data format. **c.** On held-out Hanzi characters, RWKV-7 reconstructs stroke structure, while Transformers with bounded (KV-512) or full attention do not in these examples (seed 0); on hidden-pixel accuracy the bounded Transformer matches RWKV-7 in two of three seeds (App. E.4). **d.** A Hanzi character's radical is linearly readable from the RWKV-7 state, less so for KV-512 and barely for full attention (dots: three seeds; chance 0.5).

4.4 A PHASE DIAGRAM OF INFERENCE-TIME PRESSURE

Finally, we combine the two threads of this paper, informative cues about the future and inference-time pressure, and show that how much pressure a model feels depends strongly on both the architecture and the task.

To this end, we design a task similar to the MNIST one (Fig. 7a). A letter-shaped heat map on a 24×24 grid is observed at 192 random pixels, given as a list of (x, y, value) readings, and a cue before the list points to the pixel that will be queried, with a spread t. As in Sec. 4.2, during training we only ask the model to regurgitate observed pixels, and at test time we query pixels not given in the context. The amount of memory necessary for the task is now controlled by two variables: how much the model knows about the future query, and how much memory capacity it has. We therefore expect more pressure as t gets larger and as the state gets smaller.

Qualitatively, as expected, clean letters form when the cue is uninformative and the state is small (Fig. 7b). The phase diagram quantifies this (Fig. 7c, d): RWKV-7 trained with an uninformative cue reconstructs unseen pixels best with the smallest states (RMSE 0.59 at 256 floats, 0.63 at 512), the only settings that beat interpolating the four nearest observed readings (0.65), while the same small states trained with an informative cue ($t = 0.25$) fail (1.65). For completeness, we also measure how identifiable the letters are with a frozen letter classifier: it reads 97% of the 256-float, $t = 4$ reconstructions correctly, against 11% at $t = 0.25$ (Fig. 7e).

Interestingly, a Transformer given an artificial inference-time pressure, a bounded key–value budget, also learns better features: with 288 tokens it reconstructs the letters (RMSE 0.73, 85% legible), while full attention does not (1.63, 38%), though the cue matters much less for it (Fig. 7b; App. F.1).

Overall, inference-time compression pressure, whether from an uninformative cue or a small memory, can push a model toward features and concepts that generalize, when the data have structure to compress; ours is a controlled existence result, not a universal compression principle.

5 DISCUSSION

Why does language pretraining not teach cued saving? It is intriguing that a perfectly capable architecture (Sec. 3.4) does not learn cue-based memory saving from pretraining, a behavior any human easily demonstrates. We think that natural-language pretraining data rarely highlights such an ability: text seldom announces in advance what will be asked about later, and even more rarely rewards saving something because of such an announcement, let alone understanding a semantic cue or a negation. When text does refer back, it usually repeats the exact name or phrase, which is exactly the token-matching policy we observe. Perhaps more fundamentally, text mostly follows the arrow of time: next-token prediction rewards predicting what comes next from what came before, and a later question is rarely *caused* by an earlier instruction to remember. For cued saving, pretraining data is

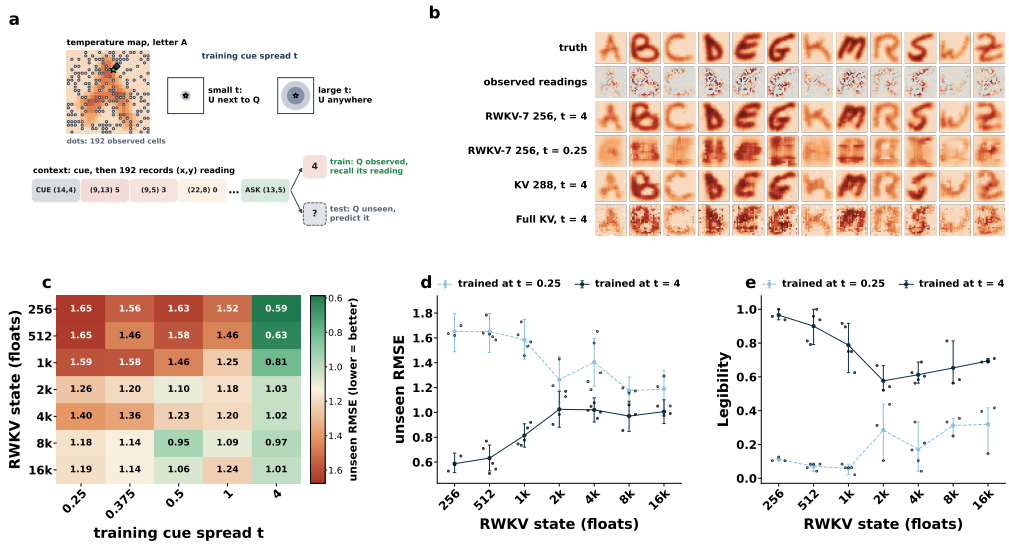


Figure 7: A phase diagram of inference-time pressure. a. A letter shaped heat map is observed at $\simeq 1/3$ pixels and a cue precedes the list of evidence given as $(x, y, value)$. The cue specifies the to-be-queried pixel with an error of t and sets how much it reveals about the query Q. Thus small t is very informative while big t isn't so much informative. **b.** A small-state RWKV-7 trained with an uninformative($t = 4$) cue and a Transformer with a 288-token KV-budget reconstruct the letters; while the same RWKV-7 trained with more informative cues($t = 0.25$) and a Transformer with full attention, do not do so as well. **c–e.** The phase diagram of inference-time pressure in action. For RWKV-7, a small state helps only when the training cue spoils little about the query. Same experiments for Transformers with a bounded key–value budget are in App. F.1. Dots: training seeds; error bars: 95% bootstrap intervals.

in this sense simply boring, and our fine-tuning results (Sec. 3) suggest that a modest amount of data that does reward it is enough to elicit it.

Why can inference-time memory pressure be an advantage? A limited memory, trivially, loses detail as more information arrives, which clearly seems like a limitation. However, we argue that limited memory is not always a disadvantage, especially when retrieval or storage tools are available (Malach et al., 2025). The most intuitive example is a scholar who has just read 100 papers. They probably cannot recall a single abstract perfectly, but they are most likely equipped with a rich, compressed model of the field. We think this is the right way to view inference-time pressure: as compressing a stream into a consistent world model (Schmidhuber, 2009).

Limitations. Our work is an analysis: we do not show any real-world performance gain, and the experiments of Sec. 4 are mostly synthetic proofs of concept. Many results rest on one training seed per model (all fine-tuned models, Figs. 4 and 5), and for the language models we measure recall only, not what they write to state. A fuller list is in App. H.

AI USE STATEMENT

We used generative AI tools for:

- **Code and analysis:** writing experimental, analysis, and plotting code. All code was checked by the authors and results were verified with control experiments; most were reproduced independently by multiple independent authors.
- **Experiments:** running variations around experiments specified by the authors. The hypotheses, experimental designs, and interpretations are the authors' own.
- **Writing:** polishing text, suggesting phrasings, and finding literature. The text, claims, and narrative are the authors' own.

Thus, we used AI for: *implement methods*, *create or edit software code*, *create or modify scientific figures or images*, *edit a research paper to improve readability*, and *identify relevant literature*. We did not use them to *propose or refine hypotheses* or *interpret results*. We have reviewed all AI-assisted work and take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none of which we feel must be specifically highlighted here.

REPRODUCIBILITY STATEMENT

The cued key–value recall task is described in Sec. 3.1, with the language-model evaluation in App. B.2 and the fine-tuning recipe in App. B.3. The synthetic task and models are in App. B.4, the pixel-record, latent-function and letter-heat tasks in App. B.5, B.6 and B.7, and the statistical procedures in App. G. All code will be released after the peer review process.

REFERENCES

Zeyuan Allen-Zhu. Physics of language models: Part 4.1, architecture design and the magic of canon layers, 2026. URL https://arxiv.org/abs/2512.17351.

Aryaman Arora, Neil Rathi, Nikil Roashan Selvam, Róbert Csordás, Dan Jurafsky, and Christopher Potts. Mechanistic evaluation of transformers and state space models, 2026. URL https://arxiv.org/abs/2505.15105.

Simran Arora, Sabri Eyuboglu, Aman Timalsina, Isys Johnson, Michael Poli, James Zou, Atri Rudra, and Christopher Ré. Zoology: Measuring and improving recall in efficient language models, 2023. URL https://arxiv.org/abs/2312.04927.

Simran Arora, Aman Timalsina, Aaryan Singhal, Benjamin Spector, Sabri Eyuboglu, Xinyi Zhao, Ashish Rao, Atri Rudra, and Christopher Ré. Just read twice: closing the recall gap for recurrent language models, 2024. URL https://arxiv.org/abs/2407.05483.

Jimmy Ba, Geoffrey Hinton, Volodymyr Mnih, Joel Z. Leibo, and Catalin Ionescu. Using fast weights to attend to the recent past, 2016. URL https://arxiv.org/abs/1610.06258.

Ali Behrouz, Peilin Zhong, and Vahab Mirrokni. Titans: Learning to memorize at test time, 2024. URL https://arxiv.org/abs/2501.00663.

Ali Behrouz, Meisam Razaviyayn, Peilin Zhong, and Vahab Mirrokni. Nested learning: The illusion of deep learning architectures, 2025. URL https://arxiv.org/abs/2512.24695.

Stella Biderman, Hailey Schoelkopf, Quentin Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, Aviya Skowron, Lintang Sutawika, and Oskar van der Wal. Pythia: A suite for analyzing large language models across training and scaling, 2023. URL https://arxiv.org/abs/2304.01373.

Loïc Cabannes, Maximilian Beck, Gergely Szilvasy, Matthijs Douze, Maria Lomeli, Jade Copet, Pierre-Emmanuel Mazaré, Gabriel Synnaeve, and Hervé Jégou. Short window attention enables long-term memorization, 2026. URL https://arxiv.org/abs/2509.24552.

Alan D. Castel, Aaron S. Benjamin, Fergus I. M. Craik, and Michael J. Watkins. The effects of aging on selectivity and control in short-term recall. *Memory & Cognition*, 30(7):1078–1085, October 2002. ISSN 1532-5946. doi: 10.3758/bf03194325. URL http://dx.doi.org/10.3758/BF03194325.

Barbara Pitts Cochran, Janet L McDonald, and Susan J Parault. Too smart for their own good: The disadvantage of a superior processing capacity for adult language learners. *Journal of Memory and Language*, 41(1):30–58, 1999.

Gregory Cohen, Saeed Afshar, Jonathan Tapson, and Andre van Schaik. Emnist: Extending mnist to handwritten letters. In *2017 International Joint Conference on Neural Networks (IJCNN)*, pp. 2921–2926. IEEE, 2017. doi: 10.1109/ijcnn.2017.7966217. URL http://dx.doi.org/10.1109/ijcnn.2017.7966217.

Tri Dao and Albert Gu. Transformers are ssms: Generalized models and efficient algorithms through structured state space duality, 2024. URL https://arxiv.org/abs/2405.21060.

Soham De, Samuel L. Smith, Anushan Fernando, Aleksandar Botev, George Cristian-Muraru, Albert Gu, Ruba Haroun, Leonard Berrada, Yutian Chen, Srivatsan Srinivasan, Guillaume Desjardins, Arnaud Doucet, David Budden, Yee Whye Teh, Razvan Pascanu, Nando De Freitas, and Caglar Gulcehre. Griffin: Mixing gated linear recurrences with local attention for efficient language models, 2024. URL https://arxiv.org/abs/2402.19427.

Jeffrey L. Elman. Learning and development in neural networks: the importance of starting small. *Cognition*, 48 (1):71–99, 1993. ISSN 0010-0277. doi: 10.1016/0010-0277(93)90058-4. URL http://dx.doi.org/10.1016/0010-0277(93)90058-4.

Alex Graves, Greg Wayne, and Ivo Danihelka. Neural turing machines, 2014. URL https://arxiv.org/abs/1410.5401.

Riccardo Grazzi, Julien Siems, Arber Zela, Jörg K. H. Franke, Frank Hutter, and Massimiliano Pontil. Unlocking state-tracking in linear rnns through negative eigenvalues, 2025. URL https://arxiv.org/abs/2411.12537.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces, 2024. URL https://arxiv.org/abs/2312.00752.

Kelsi J. Hall, Emily J. Fawcett, Kathleen L. Hourihan, and Jonathan M. Fawcett. Emotional memories are (usually) harder to forget: A meta-analysis of the item-method directed forgetting literature. *Psychonomic Bulletin & Review*, 28(4):1313–1326, April 2021. ISSN 1531-5320. doi: 10.3758/s13423-021-01914-z. URL http://dx.doi.org/10.3758/s13423-021-01914-z.

Ralph Hertwig and Peter M Todd. More is not always better: The benefits of cognitive limits. *Thinking: Psychological perspectives on reasoning, judgment and decision making*, pp. 213–231, 2003.

Joel Jang, Seonghyeon Ye, and Minjoon Seo. Can large language models truly understand prompts? a case study with negated prompts, 2022. URL https://arxiv.org/abs/2209.12711.

Samy Jelassi, David Brandfonbrener, Sham M. Kakade, and Eran Malach. Repeat after me: Transformers are better than state space models at copying, 2024. URL https://arxiv.org/abs/2402.01032.

Greg Kamradt. Needle in a haystack. https://github.com/gkamradt/needle-in-a-haystack, 2023. GitHub repository.

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are rnns: Fast autoregressive transformers with linear attention, 2020. URL https://arxiv.org/abs/2006.16236.

Barbara J. Knowlton and Alan D. Castel. Memory and reward-based learning: A value-directed remembering perspective. *Annual Review of Psychology*, 73(1):25–52, January 2022. ISSN 1545-2085. doi: 10.1146/annurev-psych-032921-050951. URL http://dx.doi.org/10.1146/annurev-psych-032921-050951.

Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. ISSN 0018-9219. doi: 10.1109/5.726791. URL http://dx.doi.org/10.1109/5.726791.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts, 2023. URL https://arxiv.org/abs/2307.03172.

Eran Malach, Omid Saremi, Sinead Williamson, Arwen Bradley, Aryo Lotfi, Emmanuel Abbe, Josh Susskind, and Etai Littwin. To infinity and beyond: Tool-use unlocks length generalization in state space models, 2025. URL https://arxiv.org/abs/2510.14826.

William Merrill, Jackson Petty, and Ashish Sabharwal. The illusion of state in state-space models, 2025. URL https://arxiv.org/abs/2404.08819.

Ali Modarressi, Hanieh Deilamsalehy, Franck Dernoncourt, Trung Bui, Ryan A. Rossi, Seunghyun Yoon, and Hinrich Schütze. Nolima: Long-context evaluation beyond literal matching, 2025. URL https://arxiv.org/abs/2502.05167.

Elissa L. Newport. Maturational constraints on language learning. *Cognitive Science*, 14(1):11–28, January 1990. ISSN 1551-6709. doi: 10.1207/s15516709cog1401_2. URL http://dx.doi.org/10.1207/s15516709cog1401_2.

Core Francisco Park, Ekdeep Singh Lubana, Itamar Pres, and Hidenori Tanaka. Competition dynamics shape algorithmic phases of in-context learning, 2025. URL https://arxiv.org/abs/2412.01003.

Bo Peng, Eric Alcaide, Quentin Anthony, Alon Albalak, Samuel Arcadinho, Stella Biderman, Huanqi Cao, Xin Cheng, Michael Chung, Matteo Grella, Kranthi Kiran GV, Xuzheng He, Haowen Hou, Jiaju Lin, Przemyslaw Kazienko, Jan Kocon, Jiaming Kong, Bartlomiej Koptyra, Hayden Lau, Krishna Sri Ipsit Mantri, Ferdinand Mom, Atsushi Saito, Guangyu Song, Xiangru Tang, Bolun Wang, Johan S. Wind, Stanislaw Wozniak, Ruichong Zhang, Zhenyuan Zhang, Qihang Zhao, Peng Zhou, Qinghua Zhou, Jian Zhu, and Rui-Jie Zhu. Rwkv: Reinventing rnns for the transformer era, 2023. URL https://arxiv.org/abs/2305.13048.

Bo Peng, Ruichong Zhang, Daniel Goldstein, Eric Alcaide, Xingjian Du, Haowen Hou, Jiaju Lin, Jiaxing Liu, Janna Lu, William Merrill, Guangyu Song, Kaifeng Tan, Saiteja Utpala, Nathan Wilce, Johan S. Wind, Tianyi Wu, Daniel Wuttke, and Christian Zhou-Zheng. Rwkv-7 "goose" with expressive dynamic state evolution, 2025. URL https://arxiv.org/abs/2503.14456.

Douglas L.T Rohde and David C Plaut. Language acquisition in the absence of explicit negative evidence: how important is starting small? *Cognition*, 72(1):67–109, August 1999. ISSN 0010-0277. doi: 10.1016/s0010-0277(99)00031-1. URL http://dx.doi.org/10.1016/S0010-0277(99)00031-1.

Juergen Schmidhuber. Driven by compression progress: A simple principle explains essential aspects of subjective beauty, novelty, surprise, interestingness, attention, curiosity, creativity, art, science, music, jokes, 2009. URL https://arxiv.org/abs/0812.4360.

Daniel J Simons and Christopher F Chabris. Gorillas in our midst: Sustained inattentional blindness for dynamic events. *Perception*, 28(9):1059–1074, 1999. ISSN 1468-4233. doi: 10.1068/p281059. URL http://dx.doi.org/10.1068/p281059.

Aaditya K. Singh, Stephanie C. Y. Chan, Ted Moskovitz, Erin Grant, Andrew M. Saxe, and Felix Hill. The transient nature of emergent in-context learning in transformers, 2023. URL https://arxiv.org/abs/2311.08360.

Yu Sun, Xinhao Li, Karan Dalal, Jiarui Xu, Arjun Vikram, Genghan Zhang, Yann Dubois, Xinlei Chen, Xiaolong Wang, Sanmi Koyejo, Tatsunori Hashimoto, and Carlos Guestrin. Learning to (learn at test time): Rnns with expressive hidden states, 2025. URL https://arxiv.org/abs/2407.04620.

Yutao Sun, Li Dong, Shaohan Huang, Shuming Ma, Yuqing Xia, Jilong Xue, Jianyong Wang, and Furu Wei. Retentive network: A successor to transformer for large language models, 2023. URL https://arxiv.org/abs/2307.08621.

Kimi Team, Yu Zhang, Zongyu Lin, Xingcheng Yao, Jiaxi Hu, Fanqing Meng, Chengyin Liu, Xin Men, Songlin Yang, Zhiyuan Li, Wentao Li, Enzhe Lu, Weizhou Liu, Yanru Chen, Weixin Xu, Longhui Yu, Yejie Wang, Yu Fan, Longguang Zhong, Enming Yuan, Dehao Zhang, Yizhi Zhang, T. Y. Liu, Haiming Wang, Shengjun Fang, Weiran He, Shaowei Liu, Yiwei Li, Jianlin Su, Jiezhong Qiu, Bo Pang, Junjie Yan, Zhejun Jiang, Weixiao Huang, Bohong Yin, Jiacheng You, Chu Wei, Zhengtao Wang, Chao Hong, Yutian Chen, Guanduo Chen, Yucheng Wang, Huabin Zheng, Feng Wang, Yibo Liu, Mengnan Dong, Zheng Zhang, Siyuan Pan, Wenhao Wu, Yuhao Wu, Longyu Guan, Jiawen Tao, Guohong Fu, Xinran Xu, Yuzhi Wang, Guokun Lai, Yuxin Wu, Xinyu Zhou, Zhilin Yang, and Yulun Du. Kimi linear: An expressive, efficient attention architecture, 2025. URL https://arxiv.org/abs/2510.26692.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2023. URL https://arxiv.org/abs/1706.03762.

Dustin Wang, Rui-Jie Zhu, Steven Abreu, Yong Shan, Taylor Kergan, Yuqi Pan, Yuhong Chou, Zheng Li, Jibin Wu, Ge Zhang, Wenhao Huang, and Jason Eshraghian. A systematic analysis of hybrid linear attention, 2026. URL https://arxiv.org/abs/2507.06457.

Daniel Wurgaft, Ekdeep Singh Lubana, Core Francisco Park, Hidenori Tanaka, Gautam Reddy, and Noah D. Goodman. In-context learning strategies emerge rationally, 2025. URL https://arxiv.org/abs/2506.17859.

Songlin Yang, Bailin Wang, Yikang Shen, Rameswar Panda, and Yoon Kim. Gated linear attention transformers with hardware-efficient training, 2024. URL https://arxiv.org/abs/2312.06635.

Songlin Yang, Jan Kautz, and Ali Hatamizadeh. Gated delta networks: Improving mamba2 with delta rule, 2025a. URL https://arxiv.org/abs/2412.06464.

Songlin Yang, Bailin Wang, Yu Zhang, Yikang Shen, and Yoon Kim. Parallelizing linear transformers with the delta rule over sequence length, 2025b. URL https://arxiv.org/abs/2406.06484.

Table 1 lists each appendix and the main-text material it supports.

Table 1: Appendix guide.

appendix	title	supports
A	Related work	all
Experimental setup		
B.1	Notation and metrics	Sec. 2
B.2	Language-model evaluation: models, prompts, and scoring	Sec. 3
B.3	Language-model fine-tuning	Sec. 3
B.4	Synthetic cued-recall task and models	Sec. 3.4
B.5	Pixel-record episodes and downstream protocol	Sec. 4
B.6	Structured latent-function task	Sec. 4
B.7	Letter-heat phase-diagram task	Sec. 4.4
Pretrained and fine-tuned language models (Sec. 3)		
C.1	Every pretrained model	Fig. 2
C.2	Scoring and measurement controls	Fig. 2
C.3	Fine-tuned models	Fig. 3
C.4	Multi-record and single-needle results	Sec. 3
Synthetic cued recall (Sec. 3.4)		
D.1	Architectures and training length	Fig. 4c
D.2	Acquisition rate and training budget	Sec. 3.4
D.3	Width tradeoff	Fig. 4d
D.4	Where the cue is held	Fig. 4e
Inference-time pressure and features (Sec. 4)		
E.1	Pixel completion gallery	Fig. 5c
E.2	OOD transfer to EMNIST letters	Fig. 5b
E.3	Input-matched classification baselines	Fig. 6b
E.4	Hanzi readout and completion	Fig. 6c, d
E.5	Attention-window pressure and post-hoc deletion	Sec. 4
Phase diagram (Sec. 4.4)		
F.1	Transformers with a bounded key–value budget	Fig. 7
F.2	Why the two memory types respond differently	Fig. 7
G	Statistical notes	Sec. 3–4.4
H	Limitations	all

A Related work

Efficient recurrent sequence models. RWKV, Mamba, Mamba-2 and Gated DeltaNet keep a fixed-size state instead of a KV cache that grows with the context (Peng et al., 2025; Gu & Dao, 2024; Dao & Gu, 2024; Yang et al., 2025a). What such a model can compute depends on its update rule and on the eigenvalues of its transitions (Grazzi et al., 2025). Our question is a different one: do the memory behaviors these designs were built for actually show up after training, and how does that depend on the data?

Linear attention, state tracking, copying and tools. Linear attention dropped the softmax, allowing parallel training with constant-cost inference (Katharopoulos et al., 2020). Mamba made the state update depend on the input (Gu & Dao, 2024), and the delta rule let the state overwrite stale associations rather than only accumulate them (Yang et al., 2025b). These models still could not track state (Merrill et al., 2025) until their transitions were allowed negative eigenvalues (Grazzi et al., 2025). How much the finite state matters has also been debated. RMs are worse than Transformers at copying long inputs (Jelassi et al., 2024), yet the gap largely closes once they can use external tools (Malach et al., 2025), much as people rely on notes.

Associative recall and long context. Associative-recall benchmarks expose retrieval gaps in efficient sequence models (Arora et al., 2023), and architectures with similar recall accuracy can implement it with different mechanisms (Arora et al., 2026). The work closest to ours is Arora et al. (2024). For a recurrent LM, the order in which information arrives changes how hard it is to pick what to store, and repeating the context so that the model sees every order improves recall. They did not vary what the question says. In Transformers, how well a model uses a long context depends on where the information sits (Liu et al., 2023), and it drops sharply when the question shares no words with the relevant passage (Modarressi et al., 2025). Larger LMs are also worse at following negated instructions (Jang et al., 2022). We ask the same about storage in RMs: when a cue comes before the content, does the model use what the cue means or just its exact tokens?

In-context algorithms and test-time learning. Whether an in-context learner retrieves or infers shifts with task diversity, context and training time (Park et al., 2025; Wurgaft et al., 2025). We look at a different knob, the finite inference state, and ask whether it pushes the model to keep the context literally, to store only what a cue asks for, or to build a structured representation of the episode. Training can also switch between a strategy that reads the context and one that generalizes. In Transformers, in-context learning can emerge and then vanish while the training loss keeps falling (Singh et al., 2023), and in hybrids, a short attention window during training forces the recurrent layers to learn long-term memory (Cabannes et al., 2026). Test-time learning trains the state as a memory while reading (Sun et al., 2025; Behrouz et al., 2024), but standard memory evaluations such as needle in a haystack (Kamradt, 2023) ask for the needle only after the content. They therefore cannot tell whether a memory writes what an advance cue asks for.

Bottlenecks and predictive representations. Information bottlenecks, predictive-state approaches and compact world models all suggest that a limited representation can keep the predictive structure that inputs share. But most of this work is about training objectives, latent codes or model size, not memory at inference time. Evidence for the popular "less is more" claims is also mixed. We take it only as motivation; our test is narrower, namely whether the pressure while an episode is processed causally changes what is learned.

Memory constraints and cognition. The cognitive analogy in this paper is functional, not mechanistic. Bounded memory may alter which information is encoded and which strategies are useful, but current RMs are not biological models and cognitive limits do not by themselves predict a benefit in neural networks. The idea that limits can help has a long history: Newport (1990) proposed that children learn language better partly because their cognitive abilities are limited, Cochran et al. (1999) argued that a superior processing capacity can be a disadvantage for adult language learners, and Hertwig & Todd (2003) argued more broadly for the benefits of cognitive limits. In neural networks, Elman (1993) found that a simple recurrent network learned a complex grammar only when its working memory started limited, but Rohde & Plaut (1999) found that starting small was unnecessary

and even hindered learning once the language was made more English-like. We therefore do not assume that limits help; we test when a bounded inference state does.

B EXPERIMENTAL SETUP

B.1 NOTATION AND METRICS

Cue effect. For a model, a list length N and a cue condition, the cue effect is the recall with the cue minus the recall without it, in percentage points. It is paired: every cued question is matched to the uncued question with the same document, record layout, target, question, answer scaffold and 16 candidates, and the effect is the mean of the per-question differences. In the synthetic task the cued and uncued queries fork from the same post-document state, so the cue effect there is also paired within a document.

Cue cells. A is the person the question will be about and B another person in the list. "Remember A" and "ignore B" are benign (a rational memory policy should gain from them); "remember B" and "ignore A" are adversarial (it should lose).

Recall scoring (language models). Greedy exact match: the model completes a cloze answer scaffold greedily, and the answer counts as correct if, after Unicode NFKC normalization, case folding, whitespace collapsing and removal of a trailing period, it equals the gold answer or one of its aliases. 16-way scoring: the probability the model assigns to the gold answer among 16 candidate answers (chance 6.25%). Recall is averaged over the six queried attributes (date of birth, nationality, place of birth, current city, phone number, email).

Balanced accuracy (pixels). The mean of the accuracy on ink pixels and on background pixels, so that predicting background everywhere scores 0.5.

Unseen RMSE and legibility (phase diagram). Unseen RMSE is the root mean squared error of the model's expected value (the mean of its predicted distribution over the 16 value tokens) against the clean, rounded field on the 384 unobserved cells, pooled over episodes and seeds. Legibility is the top-1 accuracy of a frozen letter classifier applied to the model's reconstructed map (App. B.7).

B.2 LANGUAGE-MODEL EVALUATION: MODELS, PROMPTS, AND SCORING

Models. We evaluate 15 pretrained runs:

- RWKV-7 g1 0.4B, 1.5B and 2.9B;
- RWKV-7 G1i 2.9B, 7.2B and 13.3B;
- RWKV-7 World 2.9B, evaluated twice as an independent replicate;
- DeltaNet, Gated DeltaNet, GLA and RetNet at 1.3B, trained on 100B tokens;
- two Gated DeltaNet–attention hybrids (12:1 and 3:1 linear-to-attention layers) from the same suite;
- Pythia 410M and 1.4B.

RWKV-7 and the 1.3B models run in bfloat16. Pythia runs in float32, because chunked and monolithic bfloat16 forwards differed by up to 2.6 logits. RWKV-7 checkpoints receive their official chat template: World uses `User: ...\n\nAssistant:`, and g1/G1i use the same template with an empty think block. Models without a chat template receive the same visible `User:/Assistant:` surface. Pythia and the two hybrids declare a 2,048-token context. Prompts exceed it from 32 biographies on (about 2,500 tokens), so their results beyond 16 biographies are extrapolation. No prompt is truncated.

Biographies. Profiles come from a pool of 1,024 fictional people generated with Gemini 3.1 Flash-Lite. Every candidate profile passes deterministic schema and uniqueness checks and a separate LLM judge, and nationality, sex and age are balanced. Phone numbers use the unassigned `+999` prefix and emails the reserved `.test` domain. A biography lists eight labeled fields (Fig. 8a). We query six of them: date of birth, nationality, place of birth, current city, phone number and email. Sex is shown but never queried. Documents contain $N \in \{1, 2, 4, \ldots, 256, 512\}$ biographies and are drawn from eight independent content banks. Each bank appears in four layouts: two record orders crossed with two field orders. Each document has up to four target people, and each target is asked about all six attributes. This gives 352 documents and 7,488 questions per cue condition.

Cues. A single cue sentence is placed before the records (Fig. 8b). Each cue condition reuses exactly the same documents, targets, questions and answers as the uncued baseline. Cue families:

- the full name;
- partial keys: first initial, last initial or both;
- an exact value from one of the six attributes, never the queried one;
- a coarse value: birth decade, birth continent, phone prefix or email format;
- the queried field.

Each family appears with positive ("I will ask about") and negative ("I will not ask about") wording, and each wording is either truthful or false about the queried person. In total the program has 61 cue conditions.

Scoring. The assistant turn ends with the cloze scaffold "⟨name⟩'s ⟨attribute⟩ is". The document is encoded once, and each question is answered from a separate copy of the state or cache. Greedy decoding stops at end-of-text, at a line break or at a role marker. It is capped at four tokens beyond the longest gold answer under each tokenizer. An answer counts as correct if it exactly matches the gold value after Unicode (NFKC) normalization, case folding, whitespace collapsing and removal of a trailing period. For 16-way scoring, the candidates are the gold value and the 15 other pool values of the same attribute with the closest token length under the RWKV World tokenizer. Recall is the softmax probability of the gold value over the 16 summed log-likelihoods. Free-answer recall (Fig. 10a) instead decodes after a bare `Assistant:`. Cue effects are paired within each question: recall with the cue minus recall on the same question without it. Intervals are 95% bootstrap intervals (10,000 resamples) over the eight content banks. Layouts and questions within a bank are not treated as independent. Each pretrained model is evaluated once.

a

```
<|rwkv_tokenizer_end_of_text|>User: Before
reading the records, please pay special
attention to and remember the biography of
Milica Jovanovic-Stankovic, because I will
ask you about this person afterward.

Read the fictional biography records below.
Then answer the question using only the
records. Return only the requested value,
with no explanation.

BIOGRAPHY RECORDS

Biography 1
Name: Milica Jovanovic-Stankovic
Phone number: +999-338-7742-0019
Place of birth: Birgunj, Nepal
Nationality: Serbian
Date of birth: 15 August 1939
Current city: Hobart, Australia
Email: mjstankovic@example.test
Sex: female

QUESTION
What is Milica Jovanovic-Stankovic's date of
birth?

Assistant: <think>
</think> Milica Jovanovic-Stankovic's date of
birth is
```

b

remember A (exact name)
```
Before reading the records, please pay
special attention to and remember the
biography of Milica Jovanovic-Stankovic,
because I will ask you about this person
afterward.
```

remember B (wrong person)
```
Before reading the records, please pay
special attention to and remember the
biography of Hamza Badawi, because I will
ask you about this person afterward.
```

only remember
```
Before reading the records, you only need
to pay attention to and remember the
biography of Milica Jovanovic-Stankovic.
You can ignore the other biographies,
because I will only ask you about Milica
Jovanovic-Stankovic afterward.
```

ignore B (foil)
```
Before reading the records, remember that
I will not ask you about Hamza Badawi
afterward, so you do not need to remember
this person's biography.
```

initials
```
Before reading the records, remember that
I will ask you about a person whose first
name begins with the letter M.
```

coarse feature
```
Before reading the records, remember that
I will ask you about a person who was born
in the 1930s.
```

field
```
Before reading the records, remember that
I will ask only for a person's date of
birth. You may ignore all other
biographical fields.
```

Figure 8: The biography prompt. a. A complete prompt with an exact-name cue and one biography; the cue sentence is highlighted. **b.** The cue sentence of each condition. Everything else in the prompt is identical across conditions.

B.3 LANGUAGE-MODEL FINE-TUNING

Corpus. Training profiles come from a second pool of 4,096 people, built with the same generator, judge and balance schedule. The pool shares no name, email or phone number with the evaluation pool, and 256 of its profiles are held out for validation loss. Each model is trained on 12,000 episodes. An episode has $N \in \{1, 2, 4, \ldots, 256\}$ biographies (512 is used only for evaluation), one cue and one question about one of the six attributes. 80% of questions use the cloze scaffold and 20% ask for a direct answer. Every cue is truthful and every target is the gold value: no adversarial or false cue and no uncued episode appears in training. Cue families are mixed as follows:

- exact name (remember A), 20%;
- ignore a foil (ignore B), 14%;
- initials, 14%;
- exact values of other fields, 18%;
- the queried field, 17%;
- coarse values, 17%.

All families except the exact name and ignore-B appear with both positive and negative wording. The cue wording is identical to the evaluation suites. Loss is computed only on the answer tokens and end-of-text. Tokenizer-specific corpora cap episodes at 22,528 tokens for RWKV-7 and 28,672 for the 1.3B models and Pythia. World 2.9B was also trained on a corpus that includes 30% uncued episodes. It reaches nearly the same uncued and cued recall and is not shown.

Optimization. All models are fully fine-tuned for one epoch: 187 optimizer steps of 64 episodes each (8 GPUs, gradient accumulation 8). We use AdamW ($\beta = (0.9, 0.95)$, $\epsilon = 10^{-8}$, weight decay 0.01) with learning rate 10^{-5}, 3% linear warm-up and cosine decay to zero, and gradient clipping at 1.0. Updates are applied to fp32 master weights. At this learning rate most bf16 updates round to zero; for example, only 3.6% of the 1.3B Gated DeltaNet's parameters could move at all. G1i 13.3B is trained with FSDP, with the sharded fp32 parameters serving as master weights. One training seed (0) per model; G1i 7.2B is not fine-tuned. Fine-tuned checkpoints are evaluated exactly as the pretrained models (App. B.2).

Multi-record fine-tune (Table 2). The earlier multi-record task fine-tunes RWKV-7 World 2.9B on 12,000 banks of $N \in \{16, 32, 64, 128, 256\}$ records, weighted 1:2:3:4:5. Records are literal project/access-code records, abstract `KEY | VALUE` records or semantic entity records queried by paraphrase, and four of each bank's records are targets. Four advance questions come before the bank. In the primed arm they are the real questions; in the control arm they are decoys absent from the bank, with the same token length. The same four real questions follow the bank in both arms. The balanced mixture trains on both arms in equal proportion. The optimizer and schedule match the biography fine-tune, but without fp32 master weights and with a maximum length of 18,432 tokens.

B.4 SYNTHETIC CUED-RECALL TASK AND MODELS

Episodes. The vocabulary has 4,096 key tokens, 256 value tokens and a few control tokens, all disjoint single tokens. An episode is `BOS`, then ten cue slots (`PRIME` followed by a key, or by `NULL`), then `DOC` and N records (`MARK` or `NOMARK`, key, value), then ten questions (`ASK` key `ANS` value). Keys are drawn without replacement and values uniformly at random. M of the N records are marked, the ten questions are drawn from the marked records, and K of the ten question keys are written into the cue slots in random order; the remaining slots hold `NULL`, so the sequence length does not depend on K. A *cued* question is one whose key was in a cue slot; an *uncued* question is one whose key was not.

Training. Each batch draws N uniformly from $\{32, 64, \ldots, N_{\max}\}$, and each row draws $M \sim \mathcal{U}\{10, \ldots, N\}$ and $K \sim \mathcal{U}\{0, \ldots, 10\}$. The loss is cross-entropy on the ten answer tokens only, with the questions teacher-forced in sequence. We train for 100k steps with batch size 32, AdamW ($\beta = (0.9, 0.95)$, weight decay 0.01), peak learning rate 3×10^{-4} with 2.5% linear warm-up and linear decay to zero, gradient clipping at 1.0, and seed 0 unless stated otherwise. "Trained with $N \leq N_{\max}$" means $N_{\max} \in \{128, 256, 512, 1024\}$; there is no length curriculum except for the two-stage Gated DeltaNet runs below.

Evaluation. The document is read once, and every question is answered greedily from its own copy of the same post-document state, so cued and uncued questions share one state. We evaluate $N \in \{32, 48, \ldots, 4096\}$ (15 lengths), $M \in \{10, 10 + \lfloor (N-10)/2 \rfloor, N\}$ and $K \in \{0, 2, 5, 8, 10\}$, with 16 documents per cell. Unless stated otherwise, figures use $M = N$, where the marks carry no information, and pool over K. Accuracy is exact match of the value token (chance $1/256$).

Models. All models have four layers, hidden size $d = 256$ and an MLP ratio of 4 unless stated otherwise. RWKV-7 uses head dimension 16, which gives 16,384 recurrent-matrix floats (plus 2,048 token-shift floats) and 5.95M parameters. Gated DeltaNet in Fig. 4 and Fig. 15 uses 12 heads of dimension 16 with value expansion 2 and a width-4 short convolution: 24,576 matrix floats and 6.20M parameters, so it is *not* state-matched. The Transformers have 4 heads, rotary (RoPE, $\theta = 10^4$) or no positional encoding (NoPE), and Canon layers (Allen-Zhu, 2026) (set ACD, kernel 4): 2,048 key–value floats per token and 6.47M parameters. The Transformer in Fig. 4 is the RoPE variant. GRU ($d = 352$) and simple RNN ($d = 480$) baselines, with Canon layer A, are parameter-matched (6.06M and 6.04M). The width sweep uses RWKV-7 with $d = 128$ (8 heads; 8,192 matrix floats; 2.19M parameters) and $d = 512$ (16 heads; 65,536 floats; 18.19M parameters). The hybrids of Fig. 18c put an RWKV-7 (R) or Gated DeltaNet (G) mixer in each layer of a common Gated DeltaNet shell (RMSNorm, residual, SwiGLU).

State-matched multi-seed sweep (Fig. 16). Here every recurrent model has exactly 16,384 matrix floats. Gated DeltaNet uses 16 heads of dimension 16, value expansion 1 and a width-2 convolution; DeltaNet uses 16 heads and a width-2 convolution. "Warm" Gated DeltaNet is trained for 100k steps with $N \leq 256$ at learning rate 10^{-4}, then continued from that checkpoint (same seed) for 100k steps with $N \leq 1024$ at 10^{-4}. The other variants change one training setting: `long300k` trains for 300k steps; `lr1e4`/`lr2e4` change the learning rate; `boost` oversamples $N \in \{32, 64\}$; `kall` reveals every cued key; `dtinit` re-initialises the decay-gate range; the partial warm starts (`warmbeta`, `warmgate`, `warmgates`) carry over only some parameter groups. Only `noconv` (no short convolution) and `rwshape` (RWKV-style decay layout) change the architecture. The key–value-budget Transformer (4 heads, Canon B, kernel 2) learns a keep-or-evict score for each document token and attends only to the top $B \in \{192, 384\}$ document tokens, plus the begin, cue and question tokens, which are always visible. It is trained with $N \leq 256$. These runs are evaluated on 3,600 documents (36,000 questions) for $N_{\max} = 1024$ and 3,120 documents for $N_{\max} = 256$. *Acquisition criterion* (Fig. 16c): during training, a small grid (2 documents per length) is evaluated every 2,500 steps. A run acquires the policy at the first step that starts three consecutive evaluations with cued minus uncued accuracy ≥ 0.1, about six times the grid's resolution of about $1/60$. Warm runs are excluded because they start from a checkpoint that already has the policy.

B.5 Pixel-record episodes and downstream protocol

MNIST episodes. We binarize MNIST (Lecun et al., 1998) with $\mathbb{1}[\text{pixel} \geq 128]$ and keep the 28×28 grid; the 60,000 official training images are split by hash into 55,000 training and 5,000 validation images. Each context reveals a uniformly random subset of $k \in \{196, 392, 588, 784\}$ pixels (25–100%), sampled without looking at pixel values, in a fresh random order. Every pixel is a four-token record `OBS ROW_`r `COL_`c `PIXEL_`v from a 64-token vocabulary, and an episode is `BOS` [records] `END_CONTEXT` followed by eight queries `QUERY ROW_`r `COL_`c `ANSWER`; only the binary answer at each `ANSWER` position receives loss. During training all eight queries ask for observed pixels (four ink, four background), so nothing ever supervises a missing pixel. Observed and missing queries use the same tokens.

Models and training. RWKV-7 (Peng et al., 2025): 4 layers, hidden size 128, head size 16 (1.08M parameters; 9,216 state floats). The Transformer has 4 layers, hidden size 128, 2 heads, RoPE and Canon layers (Allen-Zhu, 2026) (1.08M parameters; its KV cache grows to 3.2M floats for a full image). Both train for 15,000 steps with batch 64, AdamW, learning rate $3 \cdot 10^{-4}$ with 2.5% warmup, weight decay 0.01, gradient clipping 1.0 and bf16, one seed each.

Completion evaluation. Each context is encoded once, and its final state is queried at every missing pixel. The headline numbers use 16 fixed validation images at 196, 392 and 588 observed pixels (18,816 missing-pixel queries). We report balanced accuracy, the mean of ink and background

recall (chance 0.5). In the ink-matched control, the context is replaced by a different-digit image whose total ink is closest to the target's. The mask and order are the same, and the target's missing pixels are scored. For EMNIST (Cohen et al., 2017) we take ten ByClass test images for each of the 52 letter classes (transposed to reading orientation), each with 196 observed pixels. The pixel-frequency baseline thresholds the per-coordinate ink frequency of MNIST training images at 0.5. The ink/background split (Fig. 20c) and the galleries use a smaller panel of five images for each of eight letters (A–D, a–d).

Classification. We append a new `CLASSIFY` token after `END_CONTEXT` and add a new ten-way head. Both start random, and the whole recall-trained model is fine-tuned. Budgets are $\{1, 2, 4, \ldots, 2048, 4950\}$ labeled images per class, with three nested subset/initialization seeds and contexts at 196/392/588 observed pixels. Each fit uses a fixed learning rate of 10^{-4} with batch 50, validates every 250 steps, and runs for 2,000–10,000 steps. It stops after eight validations without improvement and keeps the checkpoint with the best validation balanced accuracy. The official test split is scored once. EMNIST classification uses the eight letters A–D, a–d (chance 0.125), with up to 2,726 labels per class. Its optimizers are not matched: RWKV-7 decays its learning rate from $3 \cdot 10^{-4}$, while the Transformer uses a fixed 10^{-4}.

Hanzi. Characters are rendered from one simplified-Chinese font (Heiti SC Light) on a 64×64 canvas, binarized at 128 and augmented with small affine and stroke-weight changes. The split is by character: 930 training characters with 512 instances each and 163 held-out characters with 256 instances each. Every component of a held-out character appears in at least three training characters. Records drop the `OBS`/`QUERY` markers (`ROW COL PIXEL`, 134 tokens), and 15–75% of the pixels are observed. RWKV-7, a full-attention Transformer+Canon and the same Transformer with a 512-token key–value budget use the MNIST architectures. Each trains for 100,000 steps (batch 16, learning rate $3 \cdot 10^{-4}$, fp32 master weights), with three seeds per model. The budgeted Transformer ranks context tokens with a learned per-token score. Each query attends only to the 512 highest-ranked observation tokens it has seen, and an evicted token never returns. The radical readout fits a logistic ridge probe on the normalized 128-d `END_CONTEXT` state at 614 revealed pixels (15%). It predicts eight radicals (four rarer radicals are dropped because some split has fewer than five characters with or without them), uses character-disjoint splits, and reports macro AUC over the 163 held-out characters. The completion ladder scores hidden-pixel balanced accuracy at 3, 5, 8 and 15% revealed pixels. It is compared against a Gaussian blur of the revealed ink, with σ fitted per exposure and thresholded at the true ink budget.

B.6 Structured latent-function task

One fixed basis gives each of $Q = 128$ keys a unit vector $a_q \in \mathbb{R}^3$. Every episode draws $z \sim \mathcal{N}(0, I_3)$, and key q takes the value $\mathrm{QuantileBin}(a_q^\top z)$ among $B = 4$ equiprobable bins. With probability p the value is instead replaced by a uniform draw (an "innovation"). A context shows $M = 32$ random keys as `BOS (KEY VALUE)`^{32} `END_CONTEXT` (66 tokens), followed by a single `QUERY KEY`. Training asks only for observed keys, and completion is scored on the 96 missing keys. Given z, the Bayes ceiling is $1 - p + p/B$. At $p = 0$ three numbers summarize the episode; at $p = 1$ every value is independent and completion is at chance (0.25). The model is the Transformer+Canon of App. B.5 (4 layers, width 128, 2 heads). It trains for 5,000 steps with batch 512, learning rate $3 \cdot 10^{-4}$ and bf16. The pressured arms train with a strict sliding attention window of 60, 52, 44 or 36 tokens. At the query this leaves 28, 24, 20 or 16 of the 32 records fully visible; the full-cache arm sees all 32. Two inference-only controls act on the full-cache weights. *Bounded-window recomputation* re-encodes the prefix under the smaller window. *Post-hoc deletion* encodes with full attention and then truncates each layer's keys and values to the budget. All budgets within a training seed share the initialization and the batch stream (generator seed = training seed + 17). Seeds 1–4 were fixed in advance as replications of the seed-0 result, and all seeds share one set of evaluation episodes. The independent-innovation null ($p = 1$) and the hybrid $p = 0.1$ endpoint use seed 0 only.

B.7 Letter-heat phase-diagram task

Field. Each scene is a 24×24 grid (576 cells) holding one uppercase EMNIST letter (Cohen et al., 2017). The letter class is drawn uniformly from the 26 classes, and an instance of it is drawn from a

bank of every official uppercase EMNIST-byclass image, resized to 22×22 pixels; training scenes use the training split and evaluation scenes the test split, with no overlap. The glyph is centred at $(12, 12)$, blurred with a Gaussian of width $\sigma = 1.25$, normalized to peak 1 and scaled by a strength drawn from $\{4, 6, 8\}$; the field is this map plus a background of 1, so it ranges from 1 to at most 9. A reading is the field plus Gaussian noise (s.d. 0.25), rounded and clipped to 16 integer values.

Observed cells and cue. The same 192 cells are observed in every scene (drawn once with a fixed seed); the other 384 are unseen. Each episode draws the query Q uniformly among the observed cells and the cue U from row Q of the heat kernel $\exp(-tL)$, where L is the Laplacian of the symmetrized 4-nearest-neighbour graph on the observed cells. Small t puts U at or next to Q; large t spreads it over the grid. Models are trained at $t \in \{0.25, 0.375, 0.5, 1, 4\}$.

Sequence and loss. An episode is $\texttt{BOS}\ \ \texttt{CUE}\ x_U\ y_U\ \texttt{DOC}\ (x\ y\ v) \times 192\ \texttt{END_DOC}\ \ \texttt{ASK}\ x_Q\ y_Q\ \texttt{ANS}\ v\ \texttt{EOS}$ (588 tokens), with the 192 readings in random order. The loss is applied only to the answer token, whose target is the reading at Q already present in the context, so training can be solved by retrieval alone.

Models. All models have 4 layers and width 256. The Transformer (4 heads, Canon layers) has 4.29M parameters and full attention over its context. The key–value-budget Transformer adds a learned retention score to every document token, computed from the token embedding and a cross-attention to the cue; at each position attention is restricted to the B highest-scoring document tokens seen so far, while the start token, the cue and the query stay visible. Scores are fixed per token, so the retained set only shrinks and the training mask equals streaming eviction; Gumbel noise ($\tau = 0.5$) on the scores explores the ranking during training. B is counted in evidence tokens (576 in total) and ranges from 24 to 575 (4.36M parameters). RWKV-7 has 16 heads of size 16, a recurrent state of 16,384 floats and 3.75M parameters. Smaller states are made on the same backbone by disabling heads and, below one head, key dimensions: $(16 - h) \times k \times 16 \times 4$ floats for h disabled heads and k active key dimensions, giving 256 ($h = 15, k = 4$), 512 ($h = 15, k = 8$), 1k ($h = 15$), 2k ($h = 14$), 4k ($h = 12$), 8k ($h = 8$) and 16k floats.

Training. AdamW ($\beta = (0.9, 0.95)$, weight decay 0.01), peak learning rate 3×10^{-4} with linear warmup over the first 2.5% of steps and linear decay to zero, batch size 128, gradient clipping 1.0, bfloat16, 100,000 steps. Seeds: five (full attention, budgets 24 and 192, RWKV-7 512, 1k and 4k floats) or three (all other settings) per training t; 300 runs in total.

Evaluation. Each run is evaluated on the same 48 test scenes with a common cue drawn at $t = 4$, whatever t it was trained at. Every one of the 576 cells is queried separately from the same prefix, and unseen RMSE is computed on the 384 unseen cells against the clean rounded field. Accuracy on observed readings (App. F.2) instead queries one observed cell per scene under the run's own training cue and scores an exact match to the noisy reading. As a reference, interpolating the four nearest observed readings (inverse-distance weighting) gives an unseen RMSE of 0.65.

Legibility. A small CNN letter classifier (142,842 parameters) is trained only on clean renders from the training split (background 1 plus a blurred glyph at strength 4, 6 or 8), never on model outputs, and is frozen. It classifies 98.5% of clean test-split renders correctly and 93.8% of the interpolation maps. Legibility is its top-1 accuracy on a model's reconstructed map, averaged over the 48 scenes.

C Pretrained and fine-tuned language models

C.1 Every pretrained model

Fig. 2 shows four RWKV-7 checkpoints and the architectures at one length. Fig. 9 shows every pretrained model over all lengths. Within RWKV-7, the cue effect grows with scale only in the G1i line. The g1 and World checkpoints peak at 11–24 points between 4 and 16 biographies. The other architectures have effects of either sign, up to about 15 points, only at short lengths, where their uncued recall is still well above its floor. None has an effect beyond 64 biographies.

C.2 Scoring and measurement controls

The main text scores recall as a greedy exact match of the cloze answer. Fig. 10 checks this choice on RWKV-7 G1i 13.3B, and Fig. 11 repeats Fig. 2b–d and f with 16-way scoring. The scale trend and

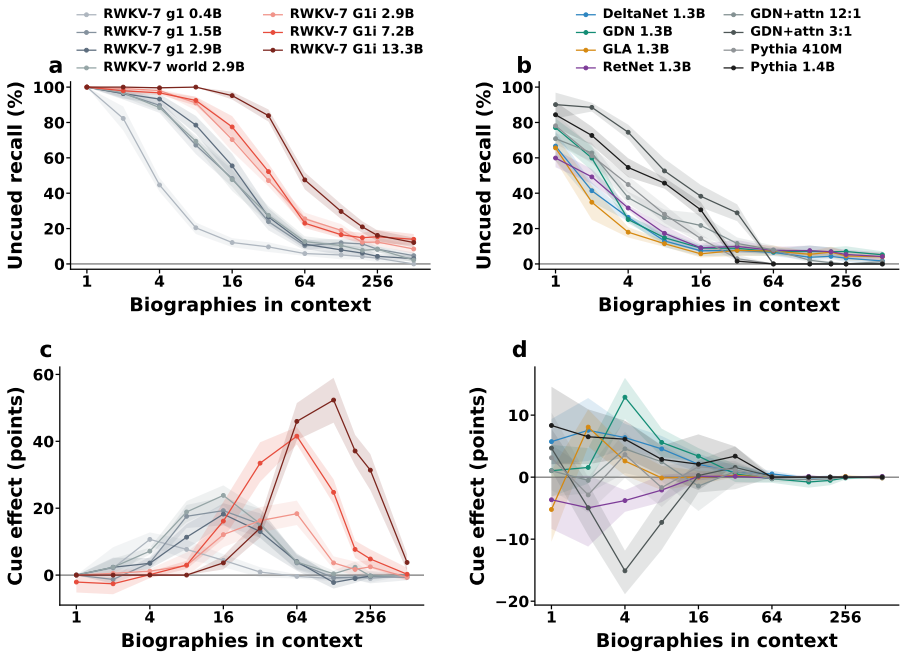


Figure 9: Every pretrained model. a, b. Uncued recall for the RWKV-7 checkpoints (a) and the other architectures (b). **c, d.** Exact-name cue effect (remember A minus no cue) for the same models. Shading: 95% bootstrap intervals over eight content banks.

the cue categories do not change: the exact-name cue effect still grows with scale, to 48 points for G1i 13.3B, and only exact text helps. The architecture comparison at 16 biographies is weaker under this scoring. Every model's effect is below 10 points, and DeltaNet (6 points) and Gated DeltaNet (8) exceed RWKV-7 g1 1.5B (5) (Fig. 11d).

C.3 Fine-tuned models

Fig. 3 shows World 2.9B, Gated DeltaNet and DeltaNet. Fig. 12 adds representative models of the other outcomes, and Fig. 13 repeats the cue-category comparison at three lengths. Fig. 14 shows the architectures that use the cue only over a short range, and the largest fine-tuned model, G1i 13.3B, over every length.

C.4 Multi-record and single-needle results

Model	Decoy questions	Real questions	Paired effect
Pretrained RWKV-7 World 2.9B	0.1319	0.1736	+0.0417
Balanced-mixture fine-tune	0.3941	0.9019	+0.5078

Table 2: Multi-record selective recall (RWKV-7 World 2.9B). Recall of four target records when the four advance questions before the bank are the real questions or decoys absent from the bank. The evaluation has 288 paired banks and 2,304 answers per model, covering literal, abstract and semantic records at 16–384 records (384 held out from training). The fine-tune's paired effect has a 95% bootstrap interval of [0.478, 0.537], and its exact McNemar p is 10^{-172}; the pretrained effect is concentrated in literal records.

Single-needle evaluation and fine-tune. An earlier single-needle test hides one fact in WikiText-2 filler and asks for it after the document, with or without the same question also stated before it. The grid covers 3 task types, contexts of 1K–32K tokens, 5 depths and 8 seeds: 1,440 items. For pretrained RWKV-7 World 2.9B, stating the question in advance does not help. Recall is 0.506 with it and 0.554 without it (Fisher $p = 0.073$), and at long contexts only the needle at the end of the document survives. Fine-tuning on 12,000 episodes, half with the question in advance and trained

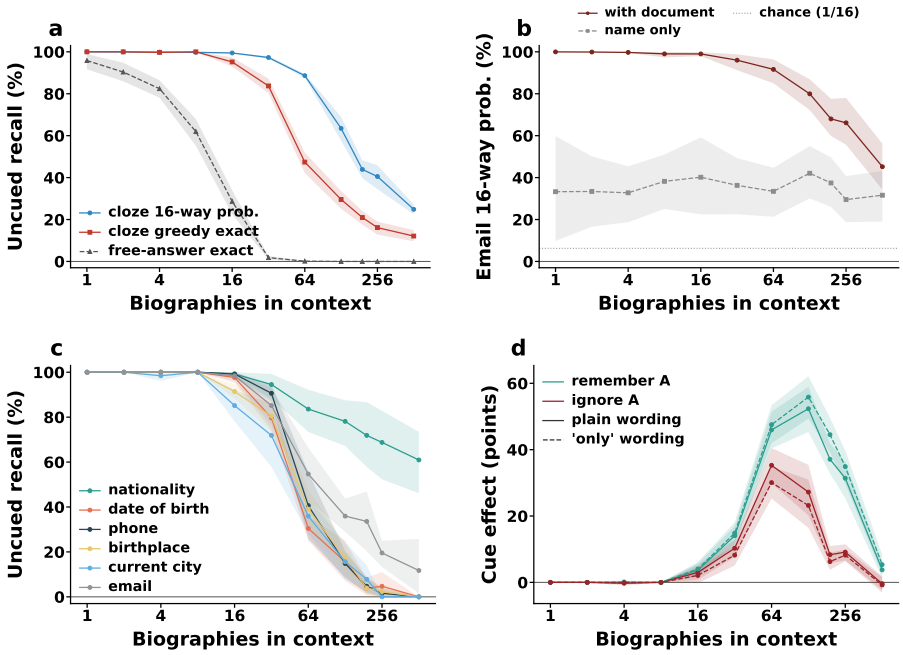


Figure 10: Scoring and measurement controls (RWKV-7 G1i 13.3B, uncued unless noted). a. Answer format: probability of the gold answer among 16 candidates, greedy exact match of the cloze answer (used in the main text), and exact match of a free answer. Free answers fail first, and the main-text cloze scoring lies between the two. **b.** Hard email control: 16-way probability of the gold email when all 15 alternatives are addresses built from the same person's name, with the biographies (solid) or with the name alone (dashed). Without the document the model still reaches about 35%, far above chance (dotted, 1/16), so part of email recall is guessing from the name. **c.** Recall by queried attribute. Nationality stays easiest at long contexts. **d.** Cue wording: "only remember A" (dashed) behaves like the plain "remember A" wording (solid), for both the remember and the ignore cue. Shading: 95% bootstrap intervals over eight content banks.

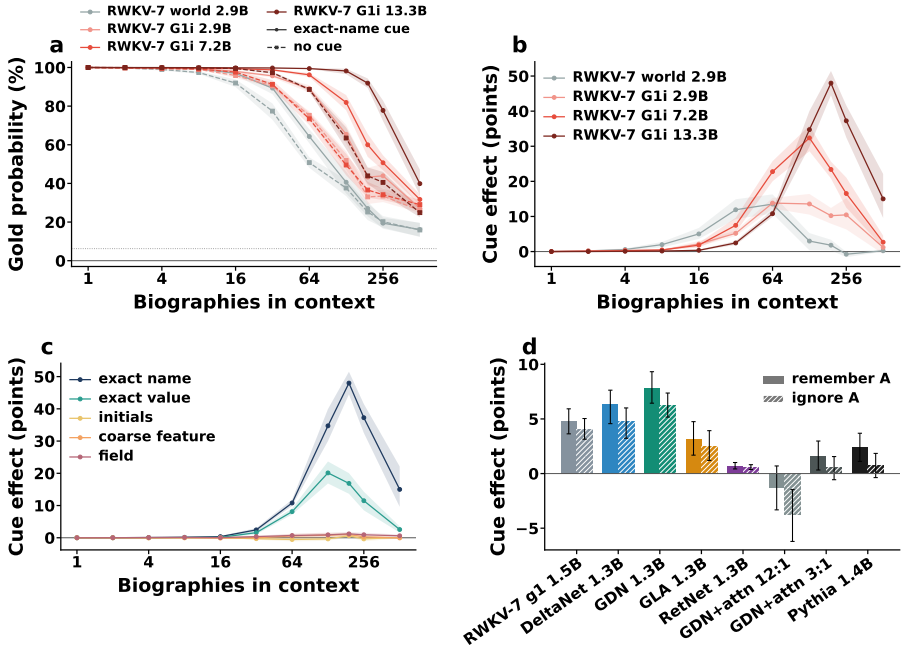


Figure 11: Fig. 2 with 16-way scoring. Recall is the probability of the gold answer among 16 candidates (chance: 6.25%, dotted in a). **a.** Recall with (solid) and without (dashed) an exact-name cue for four RWKV-7 checkpoints (Fig. 2b). **b.** Exact-name cue effect across scale (Fig. 2c). **c.** Cue categories for G1i 13.3B (Fig. 2d). **d.** Remember-A (solid) and ignore-A (hatched) cue effects across architectures at 16 biographies (Fig. 2f). Shading and error bars: 95% bootstrap intervals over eight content banks.

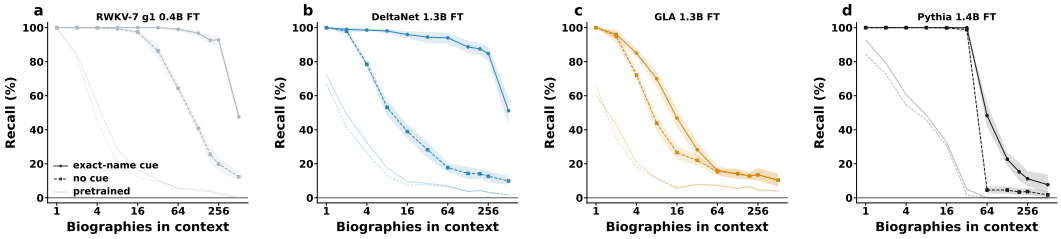


Figure 12: Fine-tuned models, one per outcome. Recall with (solid) and without (dashed) an exact-name cue after fine-tuning; faint lines: the same model before fine-tuning. **a.** The smallest RWKV-7 (g1 0.4B) learns strong uncued and cued recall and keeps cued recall above 90% up to 256 biographies. **b.** DeltaNet recalls well only what it was cued to keep. **c.** GLA gains little from the cue. **d.** The Transformer Pythia 1.4B recalls nearly everything up to 32 biographies (about 2,500 tokens) and fails from 64 on, although it was fine-tuned on up to 256 biographies; its pretraining context is 2,048 tokens. One training seed per model; shading: 95% bootstrap intervals over eight content banks.

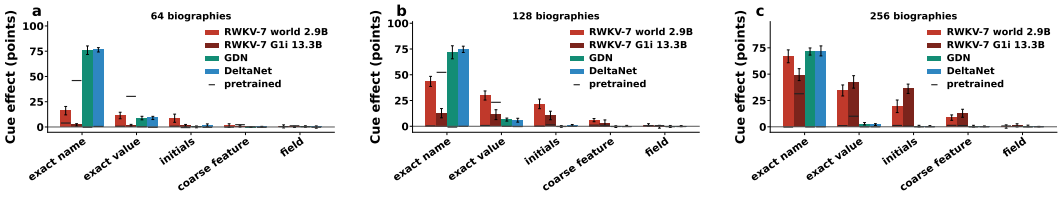


Figure 13: Cue categories after fine-tuning at 64 (a), 128 (b) and 256 (c) biographies (Fig. 3b is 256). Black ticks: the same model before fine-tuning. At shorter lengths the fine-tuned RWKV-7 models still recall most records without a cue, which caps their cue effects, while Gated DeltaNet and DeltaNet already gain 70–75 points from the exact name at 64. The ordering of the categories is the same at every length. Error bars: 95% bootstrap intervals over eight content banks.

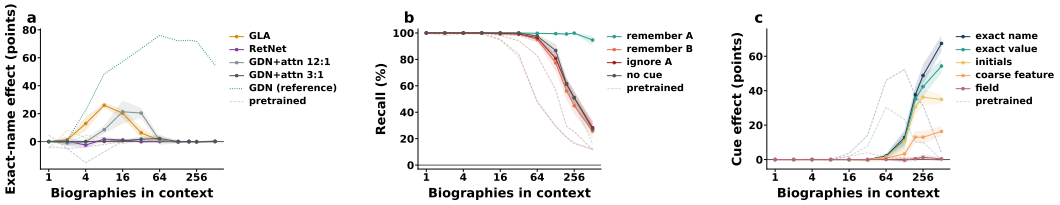


Figure 14: Short-range cue use, and fine-tuning at 13.3B. a. After fine-tuning, GLA and the 12:1 Gated DeltaNet–attention hybrid gain from the exact-name cue only at short lengths (at most 26 and 21 points, between 8 and 32 biographies), and RetNet and the 3:1 hybrid hardly at all. Dotted: fine-tuned Gated DeltaNet for reference; faint dashed lines: before fine-tuning. **b.** RWKV-7 G1i 13.3B after fine-tuning: recall under remember A, remember B, ignore A and no cue; faint dashed lines: before fine-tuning. "Ignore A" no longer helps. **c.** Cue categories for the same model over every length. One training seed; shading: 95% bootstrap intervals over eight content banks.

up to 8,192 tokens, raises recall to 0.844 in both conditions, a cue effect of exactly zero. The model learns to keep more of every needle, not to keep the one it was told about. A fine-tune without any advance questions gains almost as much (0.818). This fine-tune, like the multi-record one above, updated bf16 weights directly; at learning rate 10^{-5} most such updates round to zero (App. B.3), so its null cue effect is weak evidence.

D Synthetic cued recall

D.1 Architectures and training length

Fig. 15 shows cued and uncued accuracy for every architecture trained through four maximum lengths (one seed each). The full-attention Transformer without positional encoding solves both conditions at every length and so has no use for the cue; the rotary-position Transformer fails beyond its training length whatever the cue; GRU and simple RNN baselines (with Canon layers) do not learn the base task.

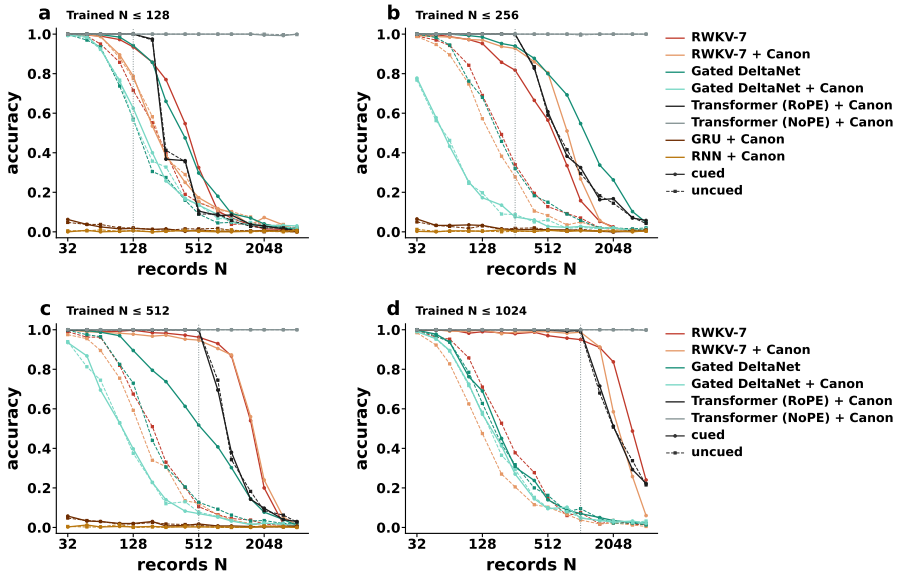


Figure 15: Every architecture, by training length. Cued (solid) and uncued (dashed) accuracy against the number of records N ($M = N$, pooled over K), for models trained with $N \leq 128$ (a), 256 (b), 512 (c) and 1024 (d); dotted vertical line: the training limit. "+ Canon": with Canon layers. GRU and RNN have no $N \leq 1024$ run. One training seed per model.

D.2 Acquisition rate and training budget

Fig. 16 holds the recurrent state fixed at 16,384 floats and repeats each architecture with three seeds. RWKV-7 acquires the cue policy in every seed. Gated DeltaNet acquires it in every seed when warm-started with full carry-over, in some seeds with longer training or other training changes, and not with partial carry-over or architectural changes. The step at which a run acquires the policy spans more than an order of magnitude across architectures. Transformers with a 192- or 384-token key–value budget (trained with $N \leq 256$) reach 70–75% recall but show no cue effect.

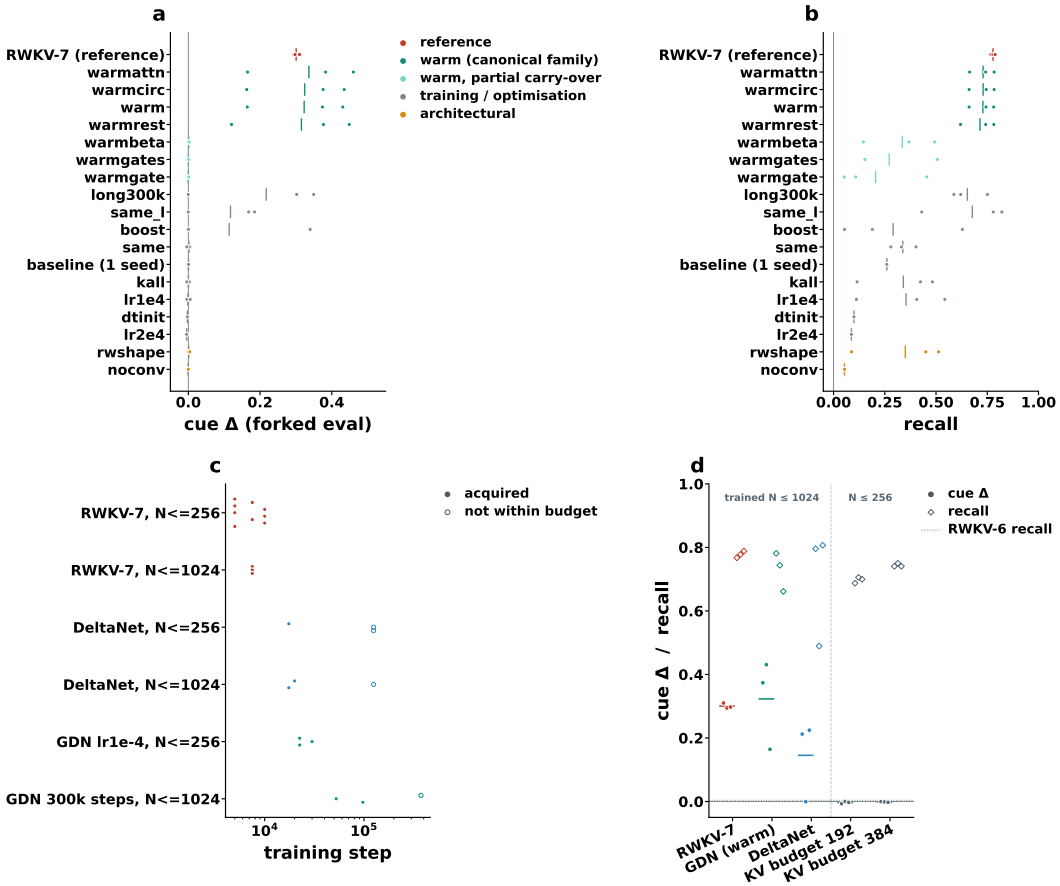


Figure 16: Acquisition rate and training budget (state-matched: 16,384 recurrent-matrix floats). a, b. Gated DeltaNet variants trained through $N = 1024$ (App. B.4): cue advantage (a) and recall (b) on the 36k-question evaluation; dots: seeds (three, except one for the baseline, `dtinit` and `lr2e4`); bars: means; RWKV-7 as reference; colours group the variants. **c.** Training step at which each run first meets the acquisition criterion (App. B.4); open circles: not met within the budget. RWKV-7 with $N \leq 256$ has nine seeds, every other arm three. **d.** Cue advantage (dots, bar: mean) and recall (diamonds) by architecture; KV-budget Transformers are trained with $N \leq 256$, the others with $N \leq 1024$; dotted: RWKV-6 recall, which never learns the base task.

D.3 Width tradeoff

Fig. 17 shows the width sweep behind Fig. 4d for every training length. A wider state moves the uncued frontier to longer documents. The heatmap shows the effect of training range: with $d = 512$ trained up to $N = 512$, the peak cue effect is only 0.04, whereas the same width trained up to $N = 1024$ reaches 0.79.

D.4 Where the cue is held

Fig. 4e transplants states before the document. Fig. 18 adds the transplant after the document, a probe for which keys were primed, hybrid models that mix RWKV-7 and Gated DeltaNet layers, and an

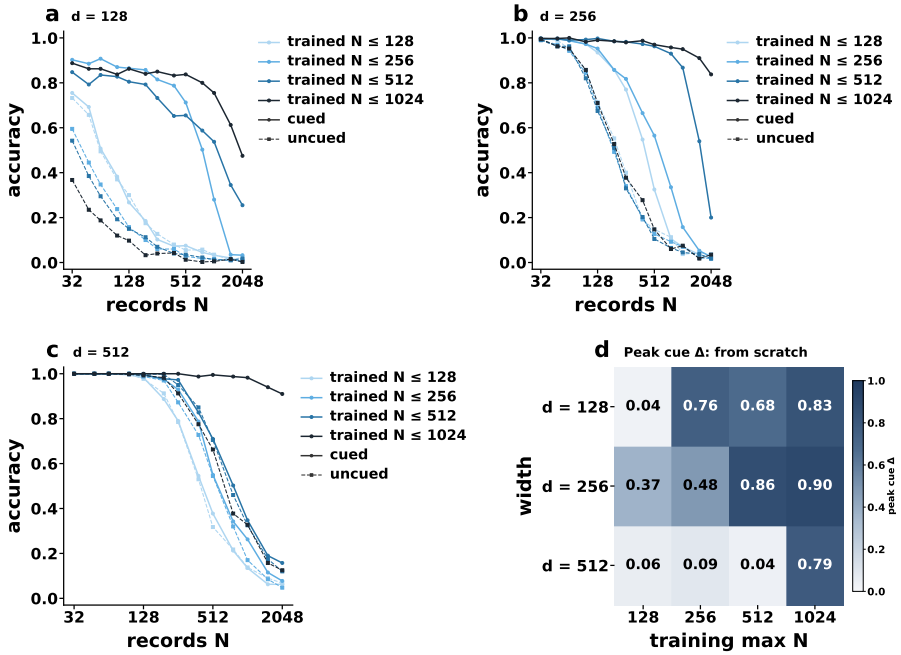


Figure 17: Width tradeoff. a–c. Cued (solid) and uncued (dashed) accuracy for RWKV-7 with hidden size $d = 128$, 256 and 512 (8,192, 16,384 and 65,536 matrix floats), one line per training length. **d.** Peak cue effect, the maximum over evaluated N of cued minus uncued accuracy, by width and training length (trained from scratch). Width also changes the parameter count (2.2M, 5.9M, 18.2M). One training seed per cell.

attempt to write the cue without a donor. After the document no single layer carries cued recall: layer 1 is necessary but not sufficient, and layers 2–4 each contribute. A counterfactual edit of a record's value is attributed almost entirely to layers 2–4 (Shapley attribution of the answer logit: 0.0, 9.2, 4.9 and 4.7 for layers 1–4). Every layer's state encodes which item was primed, but only hybrids whose two lowest layers are RWKV-7 learn the policy; swapping the local mixing between the architectures does not change this. Adding the probe direction for the primed key at the layer-1 output does not reproduce cued recall at any scale, and neither does adding a donor state there, so a donor-free write intervention remains open.

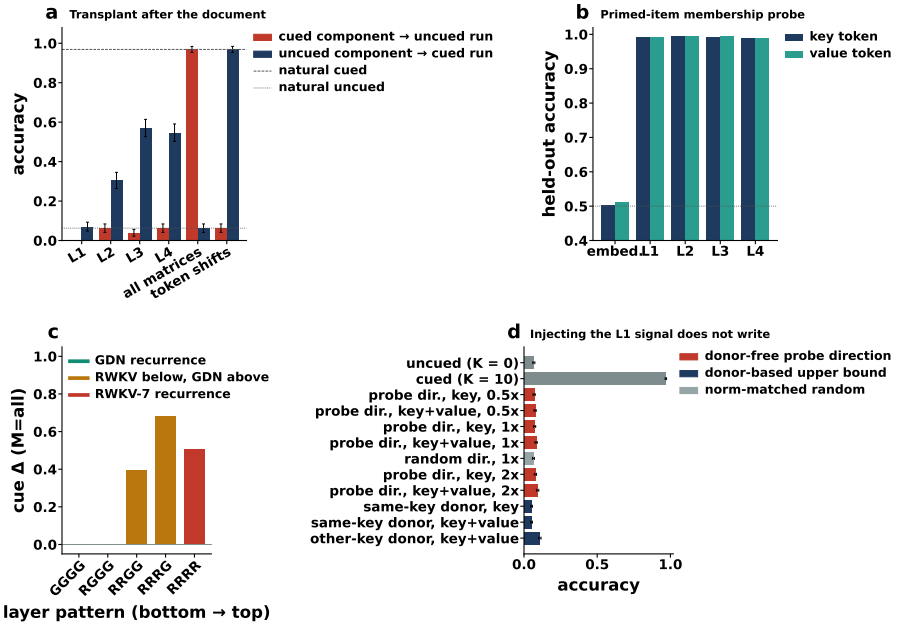


Figure 18: Where the cue is held (RWKV-7 trained with $N \leq 1024$, evaluated at $N = M = 1024$). **a.** Transplants after the document (128 documents, one cued question each): copying a component of the cued state into the uncued run (red), or of the uncued state into the cued run (navy). Moving all matrix states transfers cued recall and no single layer does (the cued layer 1 alone gives 0 of 128); replacing layer 1 with its uncued state removes cued recall (0.07), and replacing any of layers 2–4 lowers it (0.30–0.57). **b.** A linear probe reads from each layer's state whether a key or value was primed ($K = 10$); embeddings are at chance. **c.** Cue effect for four-layer hybrids, listed bottom to top (R: RWKV-7, G: Gated DeltaNet). The policy appears once the two lowest layers are RWKV-7. **d.** With no cue, adding the probe direction for each queried key (and value) to the layer-1 output at three scales leaves accuracy at the uncued level. Adding donor states at the same site, and a norm-matched random direction, does no better. One training seed; error bars: standard error over documents (a) or questions (d, 1,280).

E INFERENCE-TIME PRESSURE AND LEARNED FEATURES

E.1 PIXEL COMPLETION GALLERY

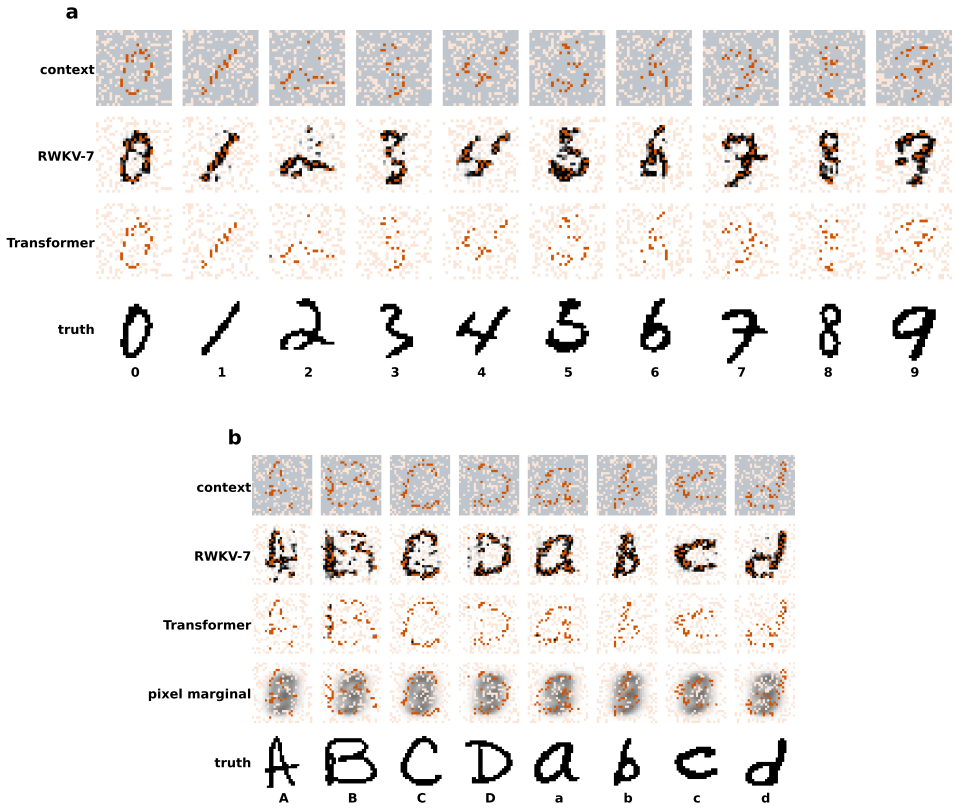


Figure 19: Pixel completion gallery (196 observed pixels, MNIST-trained models). a. One MNIST validation context per digit: the context, RWKV-7 and Transformer completions, and the true digit. **b.** The first test context of each of the eight EMNIST letters A–D, a–d, never seen in training, with the MNIST pixel-frequency map for comparison. Context row: observed ink (orange), observed background (pale), hidden (grey). Other rows: predicted ink probability at hidden pixels (black = 1), with the observed pixels overlaid.

E.2 OOD TRANSFER TO EMNIST LETTERS

Fig. 20 evaluates the MNIST-trained models on 52 EMNIST letter classes with 10 contexts each. Most missing pixels are background, so raw accuracy mostly measures how often a model predicts background. The Transformer's 0.833 equals the always-background rate (0.832). Balanced accuracy separates the models (0.801 against 0.502), and every letter class for RWKV-7 (0.75–0.86) lies above the pixel-frequency baseline's overall 0.74. On the eight-letter panel, the baseline finds more hidden ink than RWKV-7 (0.76 against 0.66), but at the cost of the background (0.75 against 0.95).

E.3 INPUT-MATCHED CLASSIFICATION BASELINES

Fig. 21 compares the classification heads of Fig. 6b with learners given the same pixels in other forms. Learners that read the serialized records stay far below RWKV-7: token autoencoders (0.19–0.29 at 16 MNIST labels per class) and from-scratch coordinate models (0.14–0.20). Learners that see the pixels laid out as a 2-D image do better. A from-scratch MLP on the partial image matches RWKV-7 at 8–16 labels per class (0.49 against 0.51) but falls behind with more labels (0.75 against 0.87 at 512). A masked-recall image autoencoder beats RWKV-7 at every MNIST budget (0.79 against 0.51 at 16), while on EMNIST letters RWKV-7 overtakes it from 1,024 labels per class. A full-image autoencoder, which sees every pixel, bounds what the input allows.

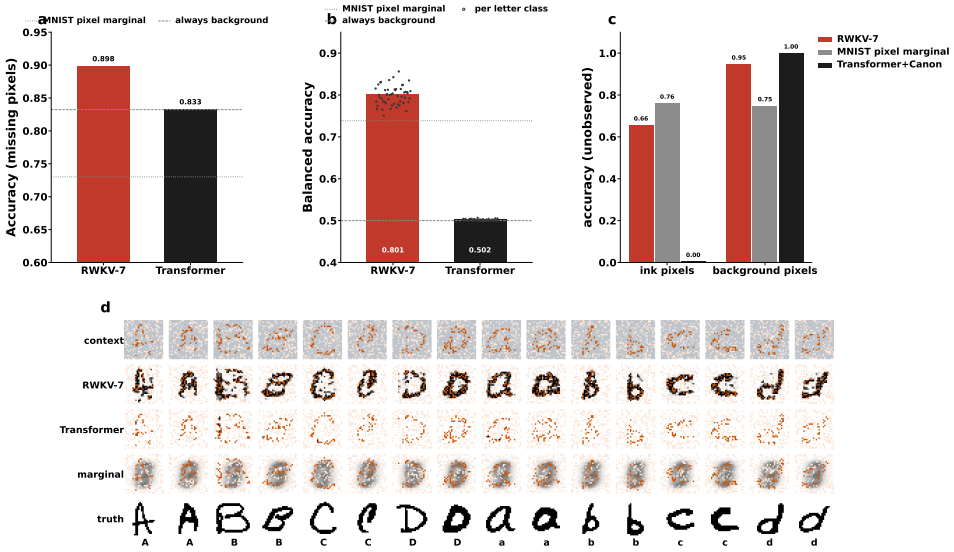


Figure 20: OOD transfer to EMNIST letters (MNIST-trained, 196 observed pixels). a. Raw accuracy on missing pixels over 520 contexts (52 classes $\times$ 10); dotted: MNIST pixel-frequency baseline; dashed: always predicting background. **b.** Balanced accuracy on the same contexts; dots: letter classes. **c.** Accuracy on hidden ink and hidden background pixels (eight letters A–D, a–d, five contexts each). The Transformer almost never predicts ink; RWKV-7 recovers two thirds of it while keeping the background clean. **d.** Two contexts per letter: context, RWKV-7, Transformer, pixel-frequency baseline and truth.

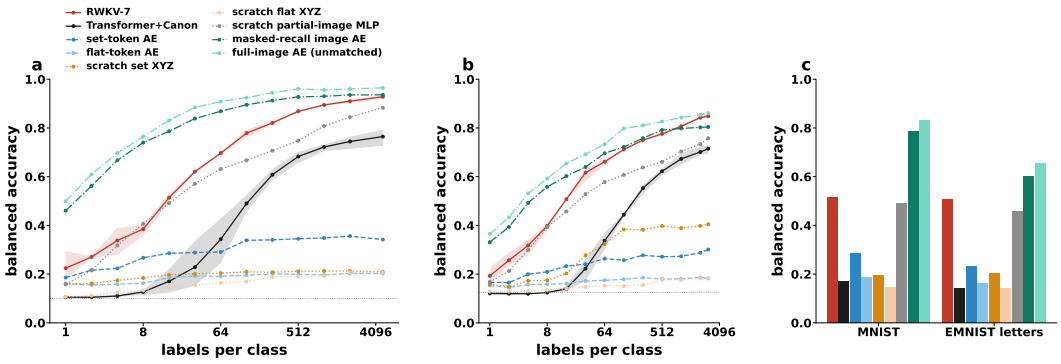


Figure 21: Input-matched classification baselines. a, b. Balanced accuracy against labels per class on MNIST (a) and the eight EMNIST letters A–D, a–d (b) for the recall-trained RWKV-7 and Transformer, autoencoders on the same records (set- and flat-token), from-scratch models on the records (set and flat (x, y, z)), a from-scratch MLP on the partial image, and image autoencoders (masked-recall and, as an unmatched upper control, full-image). Dotted: chance. The EMNIST sequence-model optimizers are not matched (App. B.5). **c.** All arms at 16 labels per class, in the same order and colours. One pretraining seed per arm; shading: range over three label-subset seeds.

E.4 HANZI READOUT AND COMPLETION

Fig. 22 shows how the radical readout of Fig. 6d develops over training. The RWKV-7 readout reaches its level within 10k steps and stays there (0.89–0.91 across seeds at 100k). The budgeted Transformer (0.73–0.82) is at the level of simple layout, stroke and ink-box statistics of the visible pixels (0.80). Full attention (0.49–0.57) is close to chance. On hidden-pixel completion, RWKV-7 beats the fitted blur at every exposure in all three seeds, but only by 0.02–0.03. The budgeted Transformer matches RWKV-7 in two seeds (0.89 and 0.87 at 15%) and fails in the third (0.66), which dominates its mean and its wide band.

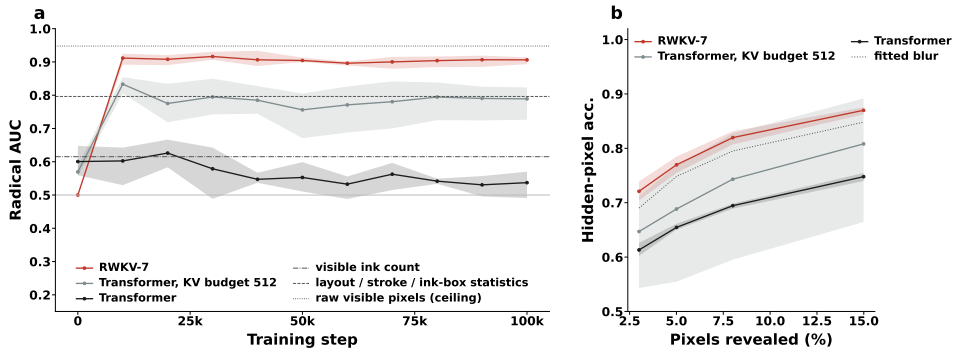


Figure 22: Hanzi readout and completion (three seeds). a. Radical macro AUC of a linear probe on the end-of-context state over training, for RWKV-7, a Transformer with a 512-token key–value budget and a full-attention Transformer. Horizontal lines (seed 0, no model): the same probe on image statistics of the visible pixels (ink count; layout, stroke and ink-box statistics) and on the raw visible pixels (ceiling). **b.** Balanced accuracy on hidden pixels of the 163 held-out characters against the fraction of pixels revealed; dotted: a Gaussian blur of the revealed ink, fitted per exposure. Lines: mean over seeds; shading: range over seeds.

E.5 ATTENTION-WINDOW PRESSURE AND POST-HOC DELETION

This is a third, smaller test of the claim in Sec. 4, on the latent-function task of App. B.6: a Transformer trained with a short sliding attention window is compared with the same-seed Transformer trained with its full cache. With windows of 60, 52, 44 and 36 tokens, completion exceeds the full-cache model in 5/5 seeds at every budget (median gains $+0.166$, $+0.240$, $+0.284$, $+0.292$). Every pressure-trained model keeps observed recall above 0.95; at 44 tokens, completion is 0.873 (seed SD 0.005). Bounded-window recomputation and post-hoc deletion of the full-cache weights lower completion in 5/5 seeds at every budget. Their recall also drops sharply at the exact visibility boundary, while the pressure-trained recall stays smooth across it. With only five pairs, the smallest possible two-sided sign-test p is 0.0625, so this supports the direction and a plateau from 44 tokens down, not a precise effect size. When every value is an independent innovation ($p = 1$, seed 0), all completion intervals include chance and recall of the bounded arms falls: pressure cannot create predictability.

Attention-pressure pilots on MNIST. Fig. 23 adds two softer pressures to the MNIST Transformer (one seed each, 512-token half-life). A normalized attention-logit bias raises missing-pixel completion from 0.50 to 0.83 (0.76 on EMNIST letters), but observed recall drops to 0.83 and the completions resemble the pixel-frequency map rather than the digit. This is therefore not a comparison at matched retrieval. An unnormalized post-softmax decay keeps recall (1.00) and does not recover completion (0.52).

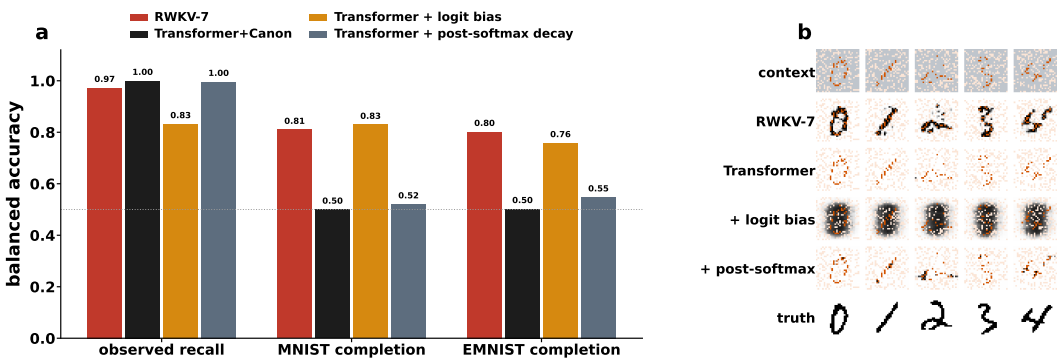


Figure 23: Attention-pressure pilots in the MNIST Transformer. a. Observed recall and missing-pixel completion (MNIST and EMNIST letters) for RWKV-7, the Transformer, and the Transformer with a logit bias or a post-softmax decay (512-token half-life); dotted: chance. Completion here uses a ten-context panel (RWKV-7 0.81), not the 16-image suite of Fig. 5. **b.** Five contexts: completions of each model and the true digit. One training seed per model.

F PHASE DIAGRAM

F.1 TRANSFORMERS WITH A BOUNDED KEY–VALUE BUDGET

Fig. 24 is the Transformer counterpart of Fig. 7c–e. Error is U-shaped in the budget: it is lowest at 288–384 tokens (0.73 at $t = 4$) and rises towards small budgets, which keep too few readings, and towards large ones. At 575 tokens, where the eviction module is still present but hardly binds, error jumps back to 1.41, close to full attention (1.63), so the gain comes from the budget rather than from the module. Legibility mirrors this, peaking at 288–480 tokens. The training cue spread changes the error by at most about 0.2 at any budget (largest spread 0.21, at 96 tokens).

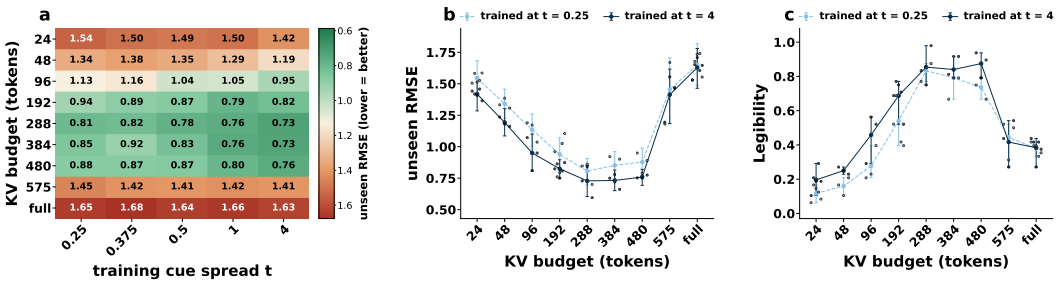


Figure 24: Transformers with a bounded key–value budget, as Fig. 7c–e. **a.** RMSE on unseen cells by budget and training cue spread t (full: unbounded attention); same colour scale as Fig. 7c. **b.** RMSE on unseen cells for models trained at $t = 0.25$ and $t = 4$. **c.** Legibility, the accuracy of a frozen letter classifier on the reconstructed map. Dots: training seeds (3–5 per setting); error bars: 95% bootstrap intervals over seeds and episodes (b) or range over seeds (c).

F.2 WHY THE TWO MEMORY TYPES RESPOND DIFFERENTLY

Training only asks about observed cells, so a model can solve training without learning the letter. Fig. 25 shows how each memory type does so. A small RWKV-7 state trained with an informative cue recalls almost every observed reading (0.97 at 256 floats): it can keep only the readings near the cue, because the cue sits next to the query. With an uninformative cue it cannot, its recall drops (0.83), and it learns the letter instead. The Transformer's recall of observed readings depends on the budget but hardly on the cue, so its pressure comes from how much evidence fits, not from what the cue reveals. The pressure effect, the error at $t = 0.25$ minus the error at $t = 4$, is about 1 for the smallest RWKV-7 states and below 0.2 for every Transformer budget when pooled over seeds (single seeds vary by up to about ± 0.2 around these values).

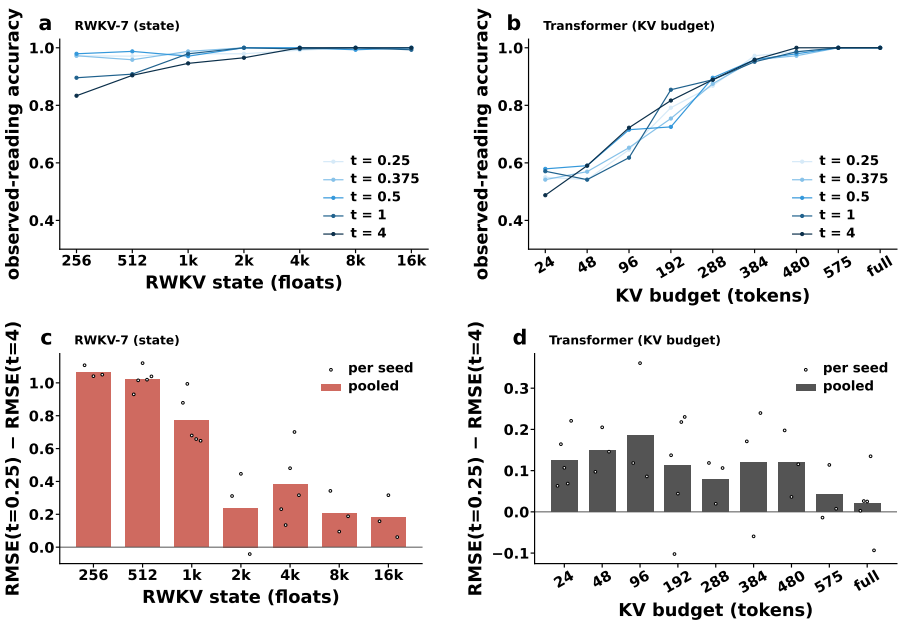


Figure 25: Why RWKV-7 and the budgeted Transformer respond differently to the cue. a, b. Accuracy on observed readings (the training task, queried under each model's own training cue) by state size (a) or budget (b), one line per training cue spread t. **c, d.** Pressure effect, RMSE on unseen cells at $t = 0.25$ minus at $t = 4$; bars: pooled over seeds; dots: individual seeds.

G STATISTICAL NOTES

Language models. Each biography condition is evaluated on eight independently generated content banks (sets of fictional people), each with several list layouts and questions. Layouts and questions within a bank are not treated as independent. Every reported value is the plain mean over questions; its 95% interval is a percentile bootstrap over the eight banks (10,000 resamples of per-bank means). Cue effects are computed on question-level paired differences before this aggregation. Pretrained models are evaluated once with greedy decoding; fine-tuned models have one training seed each.

Synthetic task. Error bars are standard errors over 128 evaluation documents within one trained model (Fig. 4); the three-seed sweep (Fig. 16) shows seeds individually.

Phase diagram. Each cell pools every (seed, scene) pair: RMSE is the square root of the mean per-scene squared error. Intervals are 95% percentile bootstraps that resample training seeds and scenes independently (2,000 resamples). Legibility bars show the range over seeds.

Seeds per claim.

Result	Independent runs
Pretrained language models (Fig. 2)	1 evaluation pass, 8 content banks
Fine-tuned language models (Fig. 3)	1 training seed per model
Synthetic cued recall, main (Fig. 4)	1 training seed per model
Synthetic, state-matched sweep (Fig. 16)	3 seeds per architecture
MNIST / EMNIST pixel completion and classification	1 seed (seed 0)
Hanzi readout (Fig. 6d)	3 seeds
Latent-function window pressure (App. E.5)	5 paired seeds
Phase diagram (Fig. 7)	3–5 seeds per setting, 48 test scenes

H LIMITATIONS

- Language-model results rest on one evaluation pass per pretrained model and one training seed per fine-tuned model, and are behavioral only: no language model's state was measured, and the post-bank cue control was not run on the fine-tuned checkpoints.
- Conditional answer behavior can arise from selective storage, selective access, or output gating. State/write interventions are required before using storage language for each practical model.
- Synthetic tasks isolate conditional memory but use random symbolic bindings and may not predict natural-language acquisition difficulty.
- Several architecture comparisons are sensitive to optimizer and training budget. A fixed-step null is an acquisition-rate result, not an impossibility theorem.
- The MNIST and EMNIST architecture comparisons use one training seed per architecture. The ink-matched context control strengthens the within-run result but does not replace seed replication. The classification advantage is relative to learners of serialized records: an image autoencoder with a masked-recall objective, which sees the 2-D layout, is more label-efficient on MNIST at every budget.
- The phase diagram uses one task, evaluates every model with the same uninformative cue, and has no control trained without a cue or with a late cue; only two RWKV-7 settings beat simple interpolation.
- Operational state size and attention-window budgets do not measure information in bits. Information-theoretic compression requires an explicit digital bottleneck and rate–distortion analysis.
- The discussion of human cognition is a functional research vision. No experiment establishes biological realism or a shared mechanism with human memory.

Table 3 summarizes where each claim currently stands.

Claim	Current boundary
Conditional memory in practical RMs	Biography evaluation complete for all pretrained and fine-tuned models, including negated, wrong-person and other cue types; behavior only, one fine-tuning seed per model.
Synthetic prospective selection	Main-figure runs are single-seed. A three-seed, state-matched sweep replicates RWKV-7 (cue effect 0.29–0.31); Gated DeltaNet acquires with a warm start (three of three seeds), 300k steps (two of three) or oversampled short documents (one of three), not at the default recipe; DeltaNet in two of three seeds.
Causal selective state	In the main synthetic RWKV-7 model, the cued layer-1 state before the document is necessary and sufficient for cued recall and a wrong-key state is not; no donor-free intervention reproduces it.
Pressure-induced features	MNIST/EMNIST comparisons use one training seed per architecture; the Hanzi readout and completion have three seeds (one budgeted-Transformer seed fails at low exposure); the latent-function window result (App. E.5) has five paired seeds but is a small task reported only in text.
Pressure phase diagram	Three to five seeds per setting on one task; only the two smallest RWKV-7 states beat nearest-neighbour interpolation; no control trained without a cue or with a late cue.
Information-theoretic compression	Not claimed: the current state budgets are operational, not explicit bit budgets or measured rate–distortion curves.

Table 3: Where each claim stands: the evidence behind it and what it does not yet show.