PAW-MEM: PREDICT, ADAPT, AND WRITE FOR EFFICIENT ONLINE ASSOCIATIVE MEMORY

Anonymous authors
Paper under double-blind review

ABSTRACT

Large language models increasingly need to retain and reuse information across long interaction histories, yet repeatedly carrying the entire history in the context window incurs growing memory and computation costs. Compact online associative memory offers an appealing alternative by compressing the past into a fixed-size recurrent state. Under aggressive compression, however, each persistent update must preserve useful history while incorporating an entire new segment, and intermediate memory states receive only indirect supervision from the final task objective. We introduce PAW-Mem (Predict, Adapt, and Write), an online associative-memory framework that addresses both issues. Predict uses the pre-write memory state to autoregressively predict the current segment, providing Forward Dynamics supervision along the memory trajectory. Adapt maintains a diagonal online scaling state that adjusts the write geometry across memory dimensions. Write then compresses token-level representations and performs one persistent associative update per segment. With a frozen Qwen3-4B backbone, PAW-Mem improves LoCoMo from 46.73 for the original δ-Mem to 50.02 while preserving competitive performance on HotpotQA and IFEval. Controlled ablations show complementary gains from Online Scaling and Forward Dynamics, and experiments with Qwen3-8B indicate that the design transfers across backbone scales.

Keywords: online associative memory, long-context language models, recurrent memory, delta-rule learning, test-time adaptation

1 INTRODUCTION

Large language models (LLMs) increasingly operate in settings where relevant information extends far beyond a single prompt, including long-running conversations, document streams, personalized assistants, and interactive agents. In such settings, a model must do more than process the current input: it must preserve information whose value may become apparent only much later. Although modern neural language models provide increasingly strong sequence representations (LeCun et al., 2015), retaining the complete interaction history in the context window remains costly, and access to a longer context does not by itself ensure reliable use of distant evidence. These limitations have motivated explicit mechanisms for retrieval, recurrence, compression, and persistent memory.

Existing approaches address this problem from complementary directions. Retrieval-augmented systems store past information explicitly and retrieve a sparse subset at query time (Lewis et al., 2020; Zhong et al., 2023; Modarressi et al., 2023; Packer et al., 2023). Recurrent and latent-memory models instead propagate or compress historical context into a bounded representation (Dai et al., 2019; Rae et al., 2020; Bulatov et al., 2022; Chevalier et al., 2023; Munkhdalai et al., 2024; Wang et al., 2024). A third line interprets the recurrent state as an online learner whose parameters are updated as new observations arrive (Schlag et al., 2021; Sun et al., 2024; Liu et al., 2025; Behrouz et al., 2025). These families trade off storage growth, retrieval latency, differentiability, and the fidelity with which the distant past can be reconstructed.

Among these approaches, compact online associative memory is attractive because its persistent state does not grow with the interaction history. δ-Mem, for example, augments a frozen full-attention language model with a small matrix state $S_t \in \mathbb{R}^{r\times r}$ that is updated online and read

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.

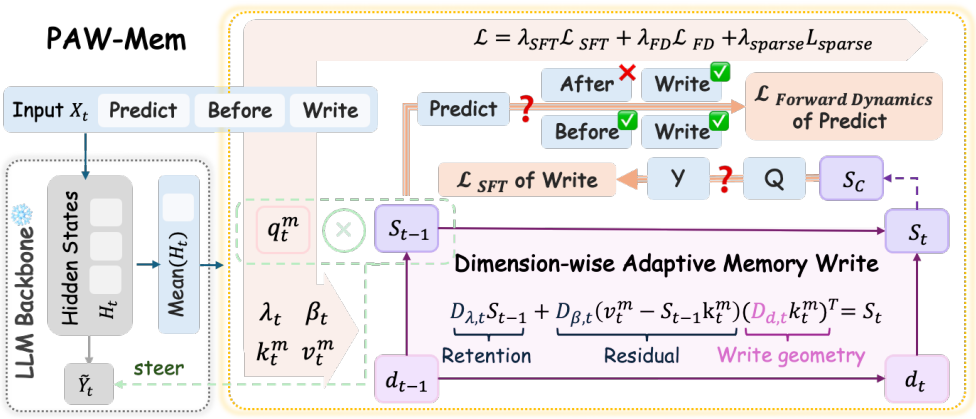


Figure 1: Overview of PAW-Mem. The incoming segment is processed by the frozen backbone using the pre-write state S_{t-1}. Predict produces token-level Forward Dynamics supervision before the segment is written; Adapt updates the diagonal scaling state d_t; and Write commits the segment-level associative update to obtain S_t. Terminal SFT is applied to the final state S_C.

through a lightweight interface (Lei et al., 2026). This design decouples persistent-state size from context length and avoids repeatedly modifying the pretrained backbone. Its update is also closely related to delta-rule fast-weight programming, which casts a recurrent matrix as an editable key–value memory (Schlag et al., 2021).

The cost of maintaining this state can be reduced further by lowering the frequency of persistent writes. Suppose a context contains N tokens grouped into T segments, where $T \ll N$. Rather than updating recurrent memory after every token, the model can aggregate the hidden representations within a segment and write once. The backbone still performs token-level computation, but persistent memory evolves at segment frequency, reducing the recurrent update cost from approximately $O(Nr^2)$ to $O(Tr^2)$.

Lower write frequency, however, changes the memory problem rather than eliminating it. When fewer updates must absorb the same history into a fixed-size state, each update carries more information and creates greater interference risk. The original δ-Mem uses dimension-wise retention and residual gates, but these controls operate on the state/value side of the update. They do not independently adapt the coordinate geometry of the write key to the observed online trajectory.

Efficient writing alone also does not ensure that intermediate states remain useful. With terminal supervised fine-tuning, the final state S_T is optimized directly for downstream answering, whereas $S_1, \ldots, S_{T-1}$ are supervised only through their eventual effect on the final prediction. This credit-assignment path becomes especially indirect when many tokens are compressed between persistent updates. Predictive objectives in context-compression models suggest a complementary principle: an intermediate memory should be judged by whether it supports prediction of information that has not yet been written (Chevalier et al., 2023).

We address these issues jointly with PAW-Mem: Predict, Adapt, and Write. PAW-Mem organizes every segment-level transition into three operations. *Predict* uses the pre-write state S_{t-1} to autoregressively predict the current segment, yielding a Forward Dynamics objective that supervises the memory trajectory without first exposing memory to the target segment. *Adapt* updates a compact diagonal scaling state d_t that changes the coordinate geometry of the upcoming write. *Write* then compresses the segment representation and commits a single associative update to persistent memory.

Figure 1 summarizes the architecture. The frozen backbone continues to operate at token resolution, whereas persistent memory advances once per segment. A single pre-write pass produces both the token predictions used by Forward Dynamics and the hidden representations used to construct the subsequent write. Online Scaling then adapts the write geometry before the transition to S_t.

The resulting temporal transition is

$$(S_{t-1}, d_{t-1}) \to \mathrm{Predict}(X_t) \to H_t \to \bar{h}_t \to d_t \to S_t.$$

PAW-Mem therefore separates persistent content, represented by S_t, from online write adaptation, represented by d_t, and trains the resulting trajectory with both intermediate predictive supervision and a terminal task objective.

Our contributions are fourfold:

1. We formulate a Predict–Adapt–Write cycle that combines token-level backbone computation with segment-level persistent memory updates.
2. We introduce dimension-wise Online Scaling, adding adaptive right-side write geometry to the retention and residual controls of δ-Mem.
3. We introduce Forward Dynamics supervision, which evaluates the current segment under the pre-write state and supplies a direct learning signal to intermediate memory states.
4. We demonstrate gains on long-horizon memory while maintaining competitive question-answering and instruction-following performance, with consistent behavior across Qwen3-4B and Qwen3-8B backbones.

2 Preliminaries

2.1 Compact Online Associative Memory

We build on δ-Mem (Lei et al., 2026), which augments a frozen full-attention Transformer with a compact online associative state

$$S_t \in \mathbb{R}^{r\times r}, \qquad r \ll d,$$

where d denotes the hidden dimension of the backbone. The backbone parameters remain frozen, while a lightweight trainable interface maps its hidden representations into a lower-dimensional memory space.

Given memory key k_t^m, value v_t^m, and vector-valued gates $\boldsymbol{\beta}_t, \boldsymbol{\lambda}_t \in \mathbb{R}^r$, the baseline associative update is

$$S_t = D_{\lambda,t} S_{t-1} + D_{\beta,t}\,(v_t^m - S_{t-1}k_t^m)\,(k_t^m)^\top,$$

where

$$D_{\lambda,t} = \mathrm{Diag}(\boldsymbol{\lambda}_t), \qquad D_{\beta,t} = \mathrm{Diag}(\boldsymbol{\beta}_t).$$

This residual has the standard delta-rule interpretation used in fast-weight models (Schlag et al., 2021). For

$$\ell_t(S) = \frac{1}{2}\,\|Sk_t^m - v_t^m\|_2^2\,,$$

the gradient is

$$\nabla_S \ell_t(S) = (Sk_t^m - v_t^m)(k_t^m)^\top.$$

At token position i, memory is read using a projected query,

$$r_{i,t} = S_{t-1} q_{i,t}^m,$$

and mapped into low-rank corrections of the frozen backbone, for example,

$$\Delta q_{i,t} = W_q^{\Delta} r_{i,t}, \qquad \Delta o_{i,t} = W_o^{\Delta} r_{i,t}.$$

The pretrained weights therefore remain fixed, while the sequence-specific online state steers the backbone through low-rank corrections.

2.2 SEGMENT-LEVEL PERSISTENT WRITING

Let segment t contain

$$X_t = (x_{1,t}, \ldots, x_{n_t,t}).$$

Processing the segment produces token-level hidden states

$$H_t = \begin{bmatrix} h_{1,t}^\top \\ \vdots \\ h_{n_t,t}^\top \end{bmatrix} \in \mathbb{R}^{n_t \times d}.$$

To construct one persistent write for the segment, we aggregate these states as

$$\bar{h}_t = \frac{1}{n_t} \sum_{i=1}^{n_t} h_{i,t}.$$

The key, value, and gate variables for the memory update are subsequently derived from $\bar{h}_t$.

For a context with

$$N = \sum_{t=1}^{T} n_t$$

tokens, token-wise persistent memory requires N recurrent state updates, whereas segment-level writing requires only T. The recurrent memory-update complexity therefore changes from

$$O(Nr^2)$$

to

$$O(Tr^2).$$

This reduction applies only to persistent memory maintenance: every input token is still processed by the frozen Transformer. We therefore describe the architecture as token-level computation with segment-level persistence.

2.3 TERMINAL SUPERVISION

After processing T segments, the final context memory is

$$S_C = S_T.$$

Given query Q and target response $Y = (y_1, \ldots, y_{|Y|})$, terminal supervised fine-tuning minimizes

$$\mathcal{L}_{\mathrm{SFT}} = -\frac{1}{|Y|} \sum_{j=1}^{|Y|} \log p_{\phi,\theta}\left(y_j \mid Q, y_{<j}, S_C\right),$$

where ϕ denotes the frozen backbone parameters and θ the trainable memory/interface parameters. This objective directly evaluates the terminal state S_C, but constrains earlier states only through their downstream influence. PAW-Mem augments it with a predictive objective defined along the memory trajectory.

3 PAW-MEM: PREDICT, ADAPT, AND WRITE

As illustrated in Fig. 1, PAW-Mem treats each segment as a single online transition composed of three operations:

$$\text{Predict} \rightarrow \text{Adapt} \rightarrow \text{Write}.$$

Predict evaluates the existing memory before the current segment is stored. Adapt changes the geometry of the upcoming compressed write. Write then commits that segment to the persistent associative state. This ordering is essential: prediction must precede writing if the predictive loss is to measure information already retained in memory rather than information copied from the current segment.

3.1 PREDICT: AUTOREGRESSIVE FORWARD DYNAMICS

Before X_t is persistently written, PAW-Mem processes it using the pre-write state S_{t-1}. For token $x_{i,t}$, the autoregressive Forward Dynamics loss is

$$\ell_{\mathrm{FD}}^{(t,i)} = -\log p_{\phi,\theta}\left(x_{i,t} \mid x_{<i,t}, S_{t-1}\right).$$

Across the complete context,

$$\mathcal{L}_{\mathrm{FD}} = -\frac{1}{N}\sum_{t=1}^{T}\sum_{i=1}^{n_t}\log p_{\phi,\theta}\left(x_{i,t} \mid x_{<i,t}, S_{t-1}\right).$$

The objective remains autoregressive: when predicting $x_{i,t}$, the model observes only the causal within-segment prefix $x_{<i,t}$. The critical distinction is that persistent memory remains fixed at S_{t-1}. Replacing it with S_t would leak information because S_t is constructed from the complete current segment. PAW-Mem therefore enforces the predict-before-write dependency

$$S_{t-1}, x_{<i,t} \to p(x_{i,t}).$$

The same pre-write forward pass also produces token-level hidden states H_t. Prediction and representation construction therefore share the same token computation:

$$(X_t, S_{t-1}) \to \begin{cases} \text{token logits} \to \mathcal{L}_{\mathrm{FD}}^{(t)}, \\ H_t \to \bar{h}_t. \end{cases}$$

Forward Dynamics thus supplies a learning signal to intermediate memory states without introducing another memory representation or a second persistent state. It asks whether the history summarized by S_{t-1} remains useful for modeling what comes next, while causal token context accounts for dependencies internal to the current segment.

3.2 ADAPT: DIMENSION-WISE ONLINE SCALING

Because one segment-level update must integrate multiple tokens into a compact state, PAW-Mem maintains an additional adaptation state

$$d_t \in \mathbb{R}^r.$$

Unlike S_t, which represents content, d_t controls the coordinate geometry of online writing. Let

$$s_t = (k_t^m)^{\odot 2}, \qquad z_t = \max\left(\|k_t^m\|_2^2, \epsilon\right).$$

Using the write statistics observed at step t, Online Scaling updates

$$\bar{d}_t = d_{t-1} + \eta\bar{\beta}_t \frac{1 - \bar{\beta}_t\langle d_{t-1}, s_t\rangle}{z_t} s_t,$$

followed by projection

$$d_t = \Pi_{\mathcal{D}}(\bar{d}_t).$$

The updated d_t is immediately applied to the current write. Defining

$$D_{d,t} = \mathrm{Diag}(d_t),$$

we obtain

$$\widetilde{k}_t^m = D_{d,t}k_t^m = d_t \odot k_t^m.$$

Crucially, the memory residual is evaluated with the unscaled key,

$$u_t = v_t^m - S_{t-1}k_t^m,$$

whereas the scaled key determines how that residual is committed to memory. The construction separates estimation of the current associative error from coordinate-wise control of the update direction. The scaling rule follows the diagonal right-preconditioning interpretation developed for DeltaNet by OSDN (Zhou et al., 2026).

3.3 WRITE: DIMENSION-WISE ASSOCIATIVE UPDATE

PAW-Mem combines the original dimension-wise retention and residual gates with the Online Scaling state:

$$S_t = D_{\lambda,t} S_{t-1} + D_{\beta,t} \left(v_t^m - S_{t-1} k_t^m\right) \left(D_{d,t} k_t^m\right)^\top .$$

The three diagonal operators have distinct roles:

$$D_{\lambda,t} \quad \text{controls retention,}$$

$$D_{\beta,t} \quad \text{controls residual write magnitude,}$$

and

$$D_{d,t} \quad \text{controls write geometry.}$$

The scaling state therefore complements rather than replaces the original δ-Mem gates: retention and residual magnitude act on the state/value side, whereas $D_{d,t}$ acts on the right-side write geometry.

The complete PAW transition is

$$(S_{t-1}, d_{t-1}) \xrightarrow{\text{FD}} H_t \xrightarrow{\text{Mean}} \bar{h}_t \xrightarrow{\text{Adapt}} d_t \xrightarrow{\text{Write}} S_t.$$

3.4 TRAINING OBJECTIVE

PAW-Mem jointly trains intermediate predictive utility and terminal task utility:

$$\mathcal{L} = \lambda_{\text{SFT}} \mathcal{L}_{\text{SFT}} + \lambda_{\text{FD}} \mathcal{L}_{\text{FD}} + \lambda_{\text{sparse}} \mathcal{L}_{\text{sparse}}.$$

We use

$$\lambda_{\text{SFT}} = 1, \qquad \lambda_{\text{FD}} = 0.1, \qquad \lambda_{\text{sparse}} = 0.01.$$

Here, $\mathcal{L}_{\text{FD}}$ trains the evolving memory trajectory, whereas $\mathcal{L}_{\text{SFT}}$ trains the terminal state for the downstream task. During offline training, the language-model backbone is frozen and only the memory/interface parameters are optimized. At inference time, all learned parameters remain fixed; only the sequence-specific states S_t and d_t evolve online. Thus, "online" refers to state adaptation rather than gradient-based parameter updates at deployment.

4 EXPERIMENTS

4.1 EXPERIMENTAL SETUP

We evaluate PAW-Mem on four complementary benchmarks. LoCoMo measures long-term conversational memory across multi-hop, temporal, open-domain, and single-hop questions (Maharana et al., 2024); MemoryAgentBench (MAB) evaluates retrieval, test-time learning, long-range understanding, and selective forgetting under incremental interactions (Hu et al., 2025); HotpotQA tests explainable multi-document reasoning (Yang et al., 2018); and IFEval measures compliance with verifiable natural-language instructions (Zhou et al., 2023). All scores are reported as percentages, with both exact match (EM) and F1 shown for HotpotQA.

Our primary experiments use Qwen3-4B-Instruct as the frozen backbone, and we additionally evaluate Qwen3-8B to test whether the memory mechanism transfers across model scale (Yang et al., 2025). The QASPER LoRA baseline is trained on the QASPER scientific question-answering corpus (Dasigi et al., 2021). To preserve direct comparability, we report a method in a benchmark block only when its results are available for all metrics shown in that block.

The primary PAW-Mem configuration uses

$$d_{\max} = 2, \qquad \alpha = 16, \qquad \texttt{online_gain} = 0.05.$$

All hyperparameters are selected on development data and are fixed before evaluation on the test sets.

Table 1: Main benchmark results and LoCoMo category-wise scores. Each method is reported in a single row, with grouped benchmark and category-wise headers.

		Overall benchmark results				LoCoMo category-wise results			
Backbone	Method	IFEval	HotpotQA EM / F1	MAB	LoCoMo Avg.	Multi-hop	Temporal	Open-domain	Single-hop
Qwen3-4B	Qwen3-Instruct	81.15	43.21 / 56.82	40.00	41.12	39.78	30.64	12.25	48.85
	+ QASPER LoRA	81.70	50.24 / 63.65	39.46	43.65	35.02	33.46	11.55	54.10
	+ MLP Memory	41.04	4.50 / 16.15	40.18	18.30	28.38	11.15	5.81	19.07
	Original δ-Mem	**82.44**	52.63 / 67.05	44.53	46.73	37.41	37.45	**18.32**	56.65
	AutoCompressor + δ-Mem	80.78	**53.19 / 67.12**	43.89	46.90	38.56	37.57	15.08	56.88
	PAW-Mem	82.07	52.36 / 66.36	**44.96**	**50.02**	**40.25**	**39.69**	17.08	**61.00**
Qwen3-8B	Qwen3-Instruct	79.67	32.96 / 41.79	45.66	48.16	44.22	31.81	13.80	59.64
	AutoCompressor + δ-Mem	**80.78**	53.63 / 67.28	**45.69**	**51.79**	**48.22**	35.21	17.05	**63.28**
	PAW-Mem	80.59	**54.09 / 67.96**	45.56	51.76	47.01	**37.48**	**17.51**	62.71

Table 2: LoCoMo ablation results.

Method	Overall	Multi-hop	Temporal	Open-domain	Single-hop
Original δ-Mem	45.96	37.29	37.03	14.84	55.83
+ Online Scaling	48.65	39.51	38.81	14.76	59.34
+ Autoregressive Forward Dynamics	47.43	37.62	38.25	18.22	57.55
+ FD + Online Scaling (Token-256)	49.11	40.82	39.06	14.57	59.68
PAW-Mem (Session/history-turn)	**50.02**	**40.25**	**39.69**	**17.08**	**61.00**

4.2 Main Results and Cross-Backbone Generalization

On Qwen3-4B, PAW-Mem raises LoCoMo from 46.73 for the original δ-Mem to 50.02, a gain of 3.29 points, and achieves the highest MAB score among the methods with complete results. Its HotpotQA and IFEval scores remain close to those of the original δ-Mem. The gains are therefore concentrated on the benchmarks most directly tied to persistent memory, rather than reflecting a uniform shift across all evaluated capabilities.

On Qwen3-8B, PAW-Mem obtains a LoCoMo score of 51.76, essentially matching the 51.79 of AutoCompressor+δ-Mem, while achieving the strongest HotpotQA result in the 8B block at $54.09/67.96$. These results support compatibility with a larger backbone, but do not by themselves establish monotonic gains from model scaling.

4.3 Ablation Study

We use the controlled LoCoMo ablation suite to isolate the effects of Autoregressive Forward Dynamics, Online Scaling, and segment structure.

Adding Online Scaling raises overall LoCoMo from 45.96 to 48.65, a gain of 2.69 points. Improvements on Multi-hop, Temporal, and Single-hop questions show that adapting dimension-wise write geometry benefits the associative-memory baseline even without the additional predictive objective.

Forward Dynamics alone increases the overall score from 45.96 to 47.43. Its largest category-level gain occurs on Open-domain questions, where performance rises from 14.84 to 18.22. This result is consistent with the intended role of Forward Dynamics: intermediate predictive supervision contributes a useful signal beyond terminal SFT.

Combining Forward Dynamics and Online Scaling under Token-256 segmentation reaches 49.11, outperforming either component in isolation. This complementarity matches their design: Online Scaling changes how compressed information is written, whereas Forward Dynamics constrains what the evolving state should remain useful for predicting.

Finally, replacing fixed Token-256 boundaries with Session/history-turn segmentation raises the overall score to 50.02. The resulting PAW-Mem model attains 39.69 on Temporal and 61.00 on Single-hop questions, the highest values in the ablation. Since each segment triggers exactly one persistent write, the result indicates that the semantic unit of compression is itself an important modeling choice.

4.4 Discussion

The experiments reveal a consistent pattern: PAW-Mem improves the tasks most directly associated with long-horizon memory while maintaining competitive question-answering and instruction-following performance. The ablations further distinguish the two mechanisms. Online Scaling improves the update rule, Forward Dynamics improves trajectory-level supervision, and their combination performs better than either component alone.

The effect of Session/history-turn segmentation further suggests that write frequency and write semantics should not be treated independently. Once multiple tokens are compressed into one transition, the segment boundary determines which information must be integrated jointly by a single update.

Overall, the evidence supports PAW-Mem as a targeted improvement to compact online memory dynamics rather than a generic modification of the pretrained language model.

5 Conclusion

We introduced PAW-Mem, a compact online associative-memory framework for frozen full-attention language models. Reducing the frequency of persistent writes lowers recurrent update cost, but also requires each remaining update to absorb more information without erasing useful history. PAW-Mem addresses this tension through a three-step cycle: Predict evaluates the current segment under the pre-write state, Adapt updates a dimension-wise scaling state for the write geometry, and Write commits one compressed segment-level associative update.

PAW-Mem improves LoCoMo over the original δ-Mem while maintaining competitive HotpotQA and IFEval performance, and its two principal components provide complementary gains in controlled ablations. Results with Qwen3-8B further indicate that the design is not specific to a single backbone size. More broadly, the findings suggest that effective long-term memory depends not only on storage capacity, but also on when information is written, how each write is geometrically conditioned, and whether intermediate states receive a direct learning signal.

References

Ali Behrouz, Peilin Zhong, and Vahab Mirrokni. Titans: Learning to memorize at test time, 2025. URL https://arxiv.org/abs/2501.00663.

Aydar Bulatov, Yuri Kuratov, and Mikhail S. Burtsev. Recurrent memory transformer, 2022. URL https://arxiv.org/abs/2207.06881.

Alexis Chevalier, Alexander Wettig, Anirudh Ajith, and Danqi Chen. Adapting language models to compress contexts. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, 2023. URL https://aclanthology.org/2023.emnlp-main.232/.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V. Le, and Ruslan Salakhutdinov. Transformer-XL: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 2978–2988, 2019. doi: 10.18653/v1/P19-1285. URL https://aclanthology.org/P19-1285/.

Pradeep Dasigi, Kyle Lo, Iz Beltagy, Arman Cohan, Noah A. Smith, and Matt Gardner. A dataset of information-seeking questions and answers anchored in research papers. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 4599–4610, 2021. doi: 10.18653/v1/2021.naacl-main.365. URL https://aclanthology.org/2021.naacl-main.365/.

Yuanzhe Hu, Yu Wang, and Julian McAuley. Evaluating memory in LLM agents via incremental multi-turn interactions, 2025. URL https://arxiv.org/abs/2507.05257.

Yann LeCun, Yoshua Bengio, and Geoffrey E. Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015. doi: 10.1038/nature14539.

Jingdi Lei, Di Zhang, Junxian Li, Weida Wang, Kaixuan Fan, Xiang Liu, Qihan Liu, Xiaoteng Ma, Baian Chen, and Soujanya Poria. δ-mem: Efficient online memory for large language models, 2026. URL https://arxiv.org/abs/2605.12357.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich K"uttler, Mike Lewis, Wen-tau Yih, Tim Rockt"aschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems*, volume 33, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/6b493230205f780e1bc26945df7481e5-Abstract.html.

Bo Liu, Rui Wang, Lemeng Wu, Yihao Feng, Peter Stone, and Qiang Liu. Longhorn: State space models are amortized online learners. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=8jOqCcLzeO.

Adyasha Maharana, Dong-Ho Lee, Sergey Tulyakov, Mohit Bansal, Francesco Barbieri, and Yuwei Fang. Evaluating very long-term conversational memory of LLM agents. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 13851–13870, 2024. doi: 10.18653/v1/2024.acl-long.747. URL https://aclanthology.org/2024.acl-long.747/.

Ali Modarressi, Ayyoob Imani, Mohsen Fayyaz, and Hinrich Schuetze. RET-LLM: Towards a general read-write memory for large language models, 2023. URL https://arxiv.org/abs/2305.14322.

Tsendsuren Munkhdalai, Manaal Faruqui, and Siddharth Gopal. Leave no context behind: Efficient infinite context transformers with infini-attention, 2024. URL https://arxiv.org/abs/2404.07143.

Charles Packer, Sarah Wooders, Kevin Lin, Vivian Fang, Shishir G. Patil, Ion Stoica, and Joseph E. Gonzalez. MemGPT: Towards llms as operating systems, 2023. URL https://arxiv.org/abs/2310.08560.

Jack W. Rae, Anna Potapenko, Siddhant M. Jayakumar, Chloe Hillier, and Timothy P. Lillicrap. Compressive transformers for long-range sequence modelling. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=SylKikSYDH.

Imanol Schlag, Kazuki Irie, and Juergen Schmidhuber. Linear transformers are secretly fast weight programmers. In *Proceedings of the 38th International Conference on Machine Learning*, 2021. URL https://proceedings.mlr.press/v139/schlag21a.html.

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, 2024. URL https://arxiv.org/abs/2407.04620.

Yu Wang, Yifan Gao, Xiusi Chen, Haoming Jiang, Shiyang Li, Jingfeng Yang, Qingyu Yin, Zheng Li, Xian Li, Bing Yin, Jingbo Shang, and Julian McAuley. MEMORYLLM: Towards self-updatable large language models, 2024. URL https://arxiv.org/abs/2402.04624.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2369–2380, 2018. doi: 10.18653/v1/D18-1259. URL https://aclanthology.org/D18-1259/.

Wanshu Zhong, Liang Guo, Qiqi Gao, and Yanlin Wang. Memorybank: Enhancing large language models with long-term memory, 2023. URL https://arxiv.org/abs/2305.10250.

Chenyu Zhou, Hongpei Li, Yuerou Liu, Jianghao Lin, Dongdong Ge, and Yinyu Ye. OSDN: Improving delta rule with provable online preconditioning in linear attention, 2026. URL https://arxiv.org/abs/2605.13473.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. Instruction-following evaluation for large language models, 2023. URL https://arxiv.org/abs/2311.07911.