# BALANCING MEMORY PATHWAYS: ANALYZING AND IMPROVING MEMORY UTILIZATION IN LLMS

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Recurrent–attention hybrid language models (LMs), which interleave attention and recurrent layers, are increasingly used to combine the efficiency of the recurrent layers with the strong performance of attention layers. Prior work suggests that attention and recurrent layers offer complementary pathways to use past information: attention supports precise memory recall from earlier tokens, while recurrent layers support consolidation of disparate information over long spans. However, we observe that simply having access to both pathways does not mean that hybrid LMs are effectively using them. When we analyze how much the models rely on each pathway, we find a substantially greater reliance on attention than on the recurrent state. Standard supervised fine-tuning improves overall performance but does not improve how the two memory pathways are coordinated: the model becomes more reliant on information propagated by attention layers, while its use of information propagated by recurrent layers remains limited. To encourage better coordination between the two memory pathways, we add an *auxiliary loss* that limits attention's access to earlier context while the recurrent state propagates through the full sequence. By backpropagating the auxiliary loss through the recurrent layers, this objective encourages the model to retain information from earlier context in recurrent state and use it alongside information accessed through attention. We observe increased use of the recurrent memory pathway and improved overall performance. Crucially, this imbalance and the benefit of our auxiliary loss generalize: they apply to multiple recurrent-attention LMs in question-answering and agentic tasks, as well as to attention-based LMs that combine different forms of memory. Together, our findings show that simply providing multiple memory pathways does not ensure their effective use, and that targeted supervision is needed to coordinate them.

## 1 INTRODUCTION

Recurrent–attention hybrid language models (Qwen Team, 2026; Ren et al., 2025; Dong et al., 2024; NVIDIA et al., 2025), which consist of recurrent and attention layers, are widely used to leverage efficient recurrent computation and expressive attention memory that allows flexible access to prior context. These architectures contain two memory pathways for retaining and using past information: attention and recurrent layers. Prior work (Lee et al., 2025; Michalak & Abreu, 2025; Bick et al., 2026) has analyzed distinct characteristics and strengths of these components: *attention layers* support precise memory recall by directly accessing earlier information, while *recurrent layers* synthesize information distributed across long contexts. Extending these findings, we observe that attention plays a larger role when the answer appears directly in the context or can be found from a single piece of evidence, whereas the recurrent state plays a larger role when the answer must be inferred rather than copied or requires combining multiple pieces of evidence across the context. These complementary strengths motivate an important question: how well do hybrid LMs use both memory pathways, and how can their coordination be improved?

We first examine how existing hybrid LMs use their attention and recurrent memory pathways. We find that having both pathways does *not* mean that models use both effectively; models retain most of their accuracy even when the recurrent pathway's access to past information is limited (30.2% $\rightarrow$ 20.2%), whereas accuracy drops substantially when the same restriction is applied to attention layers (30.2% $\rightarrow$ 5.1%). This imbalance in memory utilization persists even after standard supervised fine-



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.

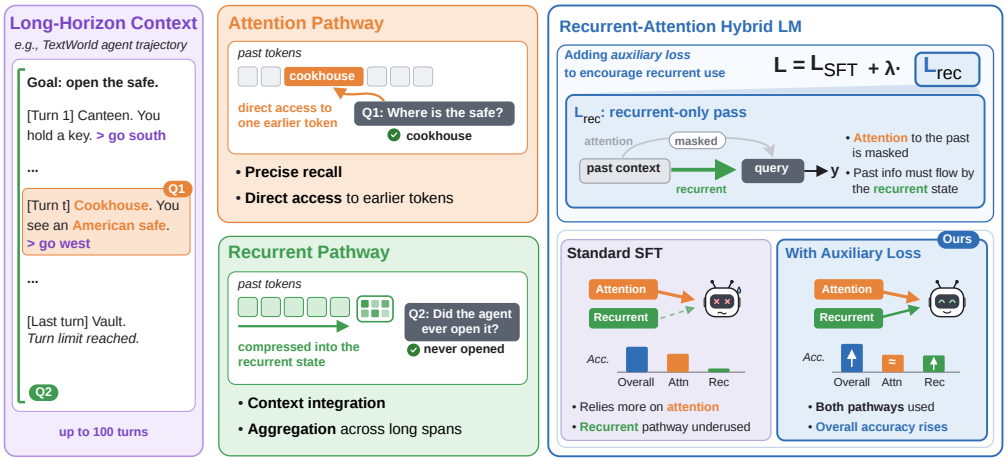


Figure 1: Overview of memory pathways in recurrent-attention hybrid LMs and our findings. (**Middle**) Two memory pathways in recurrent-attention hybrid LMs play different roles: *attention* supports precise memory recall of earlier tokens, while the *recurrent* state supports aggregating information across long spans. (**Right**) With Standard SFT ($L_{SFT}$), the model relies mostly on attention and underuses the recurrent pathways. We add an auxiliary loss ($L_{rec}$) that masks attention to the past context, so that past information can reach the current input only through the recurrent layers. We observe that adding auxiliary loss increases use of the recurrent pathway while preserving use of attention, and thus improves overall performance.

tuning (SFT) on downstream tasks such as question answering. Although the overall performance improves, the gains are mostly from better utilization of attention layers whereas their use of the recurrent pathway remains limited (Figure 1 (*Standard SFT* box)).

To address this imbalance in memory utilization, as shown in Figure 1 (*Adding auxiliary loss* box), we add a recurrent auxiliary loss that complements standard SFT by encouraging the model to use information retained by recurrent layers. In an auxiliary forward pass, we limit the attention layers' access to past information, encouraging the model to retain and use the information in the recurrent state. This simple intervention improves question-answering performance, with average absolute accuracy gains of 3.9% on Qwen3.5-4B (Qwen Team, 2026) and 5.2% on Nemotron-H-4B (NVIDIA et al., 2025) across six datasets. The gains are particularly large on longer-context datasets (8.0%) compared to shorter context (0.2%), suggesting that the auxiliary loss helps the model better utilize recurrent layers' previously identified strength in processing long contexts. Reversing the restriction to encourage attention leads to smaller gains, suggesting that the improvement from the auxiliary loss depends on which pathway is encouraged, with greater improvement from strengthening the underutilized recurrent pathway.

The benefits of auxiliary supervision also extend to agentic tasks, where agents must explore, accumulate observations and use them to choose subsequent actions. When averaged across both models, adding recurrent auxiliary supervision improves the success rate by 3.5% and 9.3% on Quest and Treasure tasks in TextWorld (Côté et al., 2019), and by 22.0% and 13.5% on the PutNextLocal and GoToObjMaze tasks in BabyAI (Chevalier-Boisvert et al., 2019), respectively. We observe that the larger gains on BabyAI mirror our question answering results, where the auxiliary loss tends to show stronger improvement on longer contexts; BabyAI trajectories (avg. 134.0 steps) are substantially longer compared to TextWorld (avg. 42.2 steps).

We further examine how the auxiliary loss affects the use of past information. When models are allowed to retry after failures, with their previous attempts stored in memory so that they can revise their answers or actions (Shinn et al., 2023; Zhao et al., 2024), models trained with the auxiliary loss recover more successfully, improving the cumulative recovery rate over standard SFT by 16.6% on question answering and 19.2% on agentic tasks. In question answering, the gain is especially large on multi-evidence questions compared to single-evidence questions (30.3% vs. 2.8%). We attribute this to the auxiliary loss encouraging greater use of recurrent layers, which we observe to play a larger role in multi-evidence questions. In the agentic setup, when given multiple suboptimal suc-

cessful demonstrations, we observe that agents trained with the auxiliary loss more often generate trajectories shorter than any provided demonstrations (6.0% on average), with larger gains when the demonstrations are longer. Finally, the same imbalance and the strength of the auxiliary loss generalize to attention-based LMs with multiple memories: standard SFT favors one memory pathway over another, and adding auxiliary supervision that encourages the use of the underutilized pathway increases usage of the pathway and improves overall performance in question-answering. Across architectures and tasks, our findings highlight that **simply providing multiple memory pathways does *not* lead to balanced, complementary use; models must instead be trained to coordinate these memory pathways.**

## 2 Background & Related Work

In this section, we provide an overview of recurrent-attention hybrid LMs in Section 2.1, then review and extend prior findings on the roles of the recurrent and attention memory pathways in Section 2.2.

### 2.1 Recurrent-Attention Hybrid LMs

Recurrent-attention hybrid LMs (Ren et al., 2025; Dong et al., 2024; NVIDIA et al., 2025; Zhang et al., 2026) integrate attention and recurrent layers to combine strong language modeling performance with computational efficiency. An *attention layer* computes each token's representation by attending to the current and preceding tokens, providing direct access to past information at the cost of quadratic computation in sequence length and KV cache that grows linearly. A *linear recurrent layer*, such as Mamba2 (Dao & Gu, 2024) or Gated DeltaNet (Yang et al., 2025), instead updates a fixed-size state as new tokens are processed, enabling computation that scales linearly with sequence length and constant memory size. Throughout the paper, we refer to such linear recurrent layers simply as *recurrent layers*. Many hybrid LMs stack these layers sequentially, combining attention's direct access to earlier tokens with recurrent layers' compact integration of past information. As a result, hybrid LMs can match or outperform full attention-based models with only a small portion of attention layers, while gaining efficiency from the larger portion of recurrent layers (Kimi Team et al., 2025; NVIDIA et al., 2025).

### 2.2 Roles and Strengths of Attention and Recurrent Memory Pathways

**Findings from Prior Work.** Prior work has identified distinct roles for attention and recurrent components in hybrid LMs. Attention is particularly important for precise retrieval, where removing attention causes severe retrieval failure in the evaluated hybrids, while retaining a small subset of attention heads preserves strong retrieval performance (Michalak & Abreu, 2025; Bick et al., 2026). Recurrent components contribute substantially to language modeling, where removing the recurrent pathway causes larger degradations in likelihood-based evaluation than removing attention (Lee et al., 2025; Borobia et al., 2026a). Fine-tuning can also affect the two components differently, where Borobia et al. (2026b) find that training only the attention pathway consistently outperforms full-model adaptation. *While these studies characterize the distinct strengths of the two memory pathways, they do not analyze whether hybrid LMs make effective use of both pathways' strengths or how to improve their use.*

**Revisiting and Extending Prior Findings.** We first revisit the distinct roles of attention and recurrent layers identified in the studies reviewed above. Following the setup from Lee et al. (2025), we use Based NQ (Arora et al., 2024) to evaluate memory recall and PG19 (Rae et al., 2019) to evaluate language modeling, and measure the increase in negative log-likelihood (NLL) when either attention or recurrent layers are ablated relative to the unablated model. Figure 2 shows a pattern consistent with prior findings: on retrieval QA, ablating attention increases NLL more than ablating recurrent layers (5.3 vs. 1.5). The pattern reverses for language modeling, where ablating recurrent layers has a larger effect (1.3 vs. 3.7). This contrast suggests that attention contributes more to precise recall, while recurrent layers contribute more to integrating information across the context.

We then extend the analysis to different question types to test whether the two memory pathways have distinct roles or strengths for different information needs. We consider two distinctions. First, we classify a question as *verbatim* if its normalized gold answer appears in the source context,

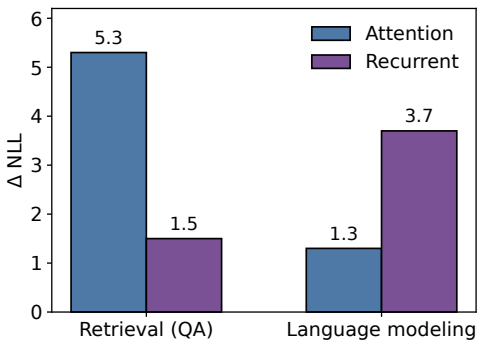


Figure 2: Increase in negative log-likelihood (y-axis) for Qwen3.5 when ablating attention or recurrent layers compared to unablated model, where *high* value indicates a *greater contribution of the ablated component*. Attention contributes more to retrieval QA, while recurrent layers contribute more to language modeling.

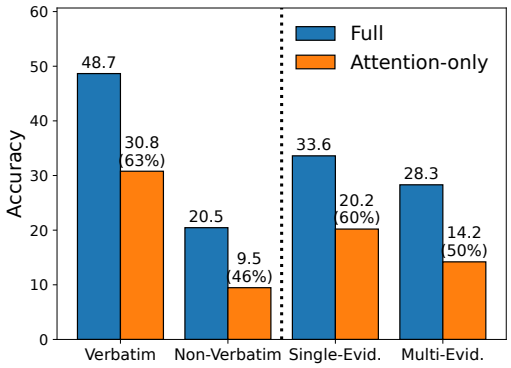


Figure 3: Accuracy (y-axis) of Qwen3.5 under *Full* (unrestricted) and *Attention-only* (past context accessible only through attention) across question types. Percentages show the fraction of Full accuracy retained under Attention-only: high and low retention indicates greater reliance on attention and recurrent state, respectively.

and as *non-verbatim* otherwise. Second, we classify questions as *single-evidence* or *multi-evidence* based on the number of evidence required to answer them, where evidence annotations are available. We match the number of examples across the two categories in each distinction. Figure 3 compares pathway use across question types. We measure *attention-only retention* as the fraction of full-model accuracy preserved when cross-context information is accessible only through attention layers; higher retention indicates greater reliance on attention and less on recurrent state. We observe that retention is higher for verbatim than non-verbatim questions (63% vs. 46%) and for single-evidence than multi-evidence questions (60% vs. 50%). This aligns with the findings above: attention layers play a larger role in recovering precise details, but contribute less when information must be abstracted or integrated across the context. Further details are in Appendix A.

## 3 Experimental Setup

**Models.** We use two recurrent-attention hybrid LMs: `Qwen3.5-4B` (Qwen Team, 2026) and `Nemotron-H-4B-Instruct` (NVIDIA et al., 2025). `Qwen3.5-4B` contains 24 *Gated DeltaNet* (Yang et al., 2025) and 8 full-attention layers. `Nemotron-H-4B-Instruct` consists of 24 `Mamba2` (Dao & Gu, 2024) and 4 full-attention layers.

**Training.** We train all models using LoRA (Hu et al., 2021) on all layers with rank 16, scaling factor 32, and dropout 0.05.[1] We use AdamW with a learning rate of $1e^{-4}$, a cosine learning rate schedule, a warmup period covering 3% of training steps, and an effective batch size of 16. Since objectives with an auxiliary loss require additional computation, we halve the number of optimization steps to match the total training compute with standard SFT. Unless otherwise specified, we use the same optimization setting across all compared variants.

**Task & Evaluations.** We experiment on long-context question answering and long-horizon agentic tasks, which are two common settings for studying memory in which the model needs to retain and use information from earlier contexts or interactions for strong performance (Packer et al., 2024; Zhao et al., 2024; Shinn et al., 2023). For **Question Answering**, we pool training examples from four datasets so that the model is trained over diverse context lengths and reasoning types: DROP (Dua et al., 2019), HotpotQA (Yang et al., 2018), Qasper (Dasigi et al., 2021), NarrativeQA (Kočiský et al., 2017). This yields 12k training and 2k validation instances. We evaluate each

[1]We use LoRA for parameter-efficient fine-tuning, as prior works (Yoshimura et al., 2025; Xin et al., 2024) have shown its effectiveness for both attention-based and recurrent-based models.

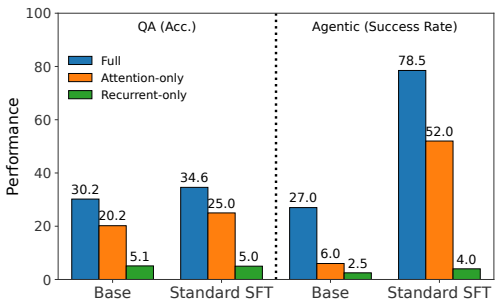


Figure 4: Performance of Qwen3.5-4B before (*Base*) and after standard SFT (*Standard SFT*) on question answering and agentic tasks. *Standard SFT* substantially improves *Full* and *Attention-only* performance but little improvement in *Recurrent-only* performance.

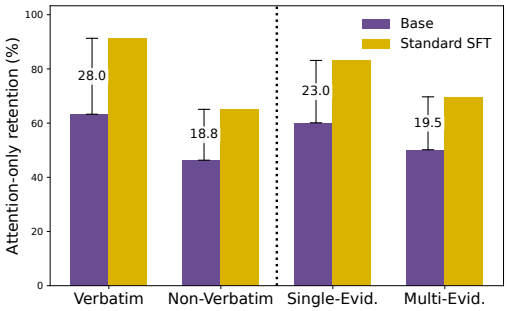


Figure 5: Attention-only retention (the fraction of *Full* accuracy retained when past context is accessible only through attention) before (*Base*) and after SFT (*Standard SFT*). SFT increases attention-only retention across all question types, especially verbatim and single-evidence.

dataset separately, and additionally evaluate on MINTEval (Lee et al., 2026) and MuSR (Sprague et al., 2024), leading to a total of 8.2k test examples with an average context length of 84k tokens. For **Agentic Tasks**, we experiment over TextWorld (Côté et al., 2019) and BabyAI (Chevalier-Boisvert et al., 2019), where agents must explore the environment, accumulate information through sequences of observations and actions, and use it to make subsequent decisions. The environments differ in their observation spaces, action spaces, and in the structure of their trajectories. In TextWorld, we train on Quest and evaluate on both Quest and Treasure. In BabyAI, we train on MiniBossLevel and evaluate on PutNextLocal and GoToObjMaze. Further details are in Appendix B.

## 4 Standard SFT Favors One Memory Pathway

**Key Observation:** Applying standard SFT to recurrent-attention hybrid LMs increases their use of attention layers.

In Section 2.2, we observed complementary strengths and roles of the two pathways: attention supports precise recall, while recurrent layers integrate information across the context. Thus, in this section, we examine whether standard supervised fine-tuning (SFT) leads the model to use both pathways in ways that leverage these complementary strengths. We first define *attention-only* and *recurrent-only* conditions to analyze and assess the model's use of each memory pathway (Section 4.1). We then evaluate how overall performance and pathway utilization change after SFT on question answering and agentic tasks (Section 4.2).

### 4.1 Disentangling Two Memory Pathways

To analyze how past information is carried through attention and recurrence, we divide each input into a past portion and a current portion, and selectively restrict which memory mechanism can transfer information from the past to the current input. For example, in question answering, the past portion corresponds to the context and the current portion to the question. For agentic tasks, they correspond to the previous history (e.g., trajectory, reflection), and the current observation and decision step, respectively. We construct two complementary interventions, **recurrent-only** and **attention-only**; these indicate **which mechanism can carry information from the past to the current portion**. In the recurrent-only setting, attention cannot access the past portion, so past information reaches the current portion only through the recurrent state; the attention-only setting does the opposite. In both, all layers remain active within each portion.

Specifically, in the *recurrent-only* condition, we block attention across segment boundaries while allowing the recurrent state to propagate normally:

$$M_{t,s} = \begin{cases} 0, & s \le t \text{ and } \sigma(s) = \sigma(t), \\ -\infty, & \text{otherwise.} \end{cases}$$

$M_{t,s}$ is added to the attention score between query position $t$ and key position $s$ before normalization, and $\sigma(t)$ denotes the segment containing token $t$. The mask therefore retains standard causal attention within each segment, while blocking attention to all tokens in preceding segments.

Conversely, in the *attention-only* condition, we retain the standard causal attention over the preceding context, while restricting the recurrent state from propagating across segment boundaries ($\sigma(\cdot)$). For $t > 1$, we modify the recurrent update as:

$$S_t = \mathbb{1}[\sigma(t) = \sigma(t-1)]\, \mathcal{A}_t(S_{t-1}) + w_t, \qquad S_0 = 0,$$

where $S_t$ denotes the recurrent state after processing token $t$, $\mathcal{A}_t$ denotes the recurrent transition applied to the previous state, $w_t$ denotes the contribution of the current token, and $\mathbb{1}[\cdot]$ denotes an indicator function. At a segment boundary ($\sigma(t) \neq \sigma(t-1)$), the indicator is zero, so the previous recurrent state does not contribute to the new segment and within a segment, the recurrent update is unchanged. Thus, information from preceding segments remains accessible only through attention.

### 4.2 Results

We evaluate both the original (*Base*) and SFT-trained (*Standard SFT*) checkpoints under the full, attention-only, and recurrent-only conditions defined in Section 4.1. As shown in Figure 4, standard SFT improves the QA accuracy of Qwen3.5 from 30.2% to 34.6%. Attention-only accuracy also increases from 20.2% to 25.0%, whereas recurrent-only accuracy remains nearly unchanged (5.1% to 5.0%). We observe a similar pattern on the agentic Quest task: the success rate rises from 27.0% to 78.5%, attention-only also rises from 6.0% to 52.0%, while recurrent-only increases only from 2.5% to 4.0%. These results indicate that the model retains a large portion of its full performance when recurrent layers are restricted to local information (attention-only), but drops sharply when attention is restricted (recurrent-only), and this gap widens after SFT.

This shift toward the attention pathway also appears across the question types discussed in Section 2.2. We observe that attention-only retention, the fraction of full-model accuracy preserved under the attention-only condition, increases for all question types (Figure 5). The gain is larger for question types that already relied more on attention: verbatim and single-evidence questions show an average increase of 25.5% compared to 19.2% for non-verbatim and multi-evidence questions.

Together, these results show that standard SFT improves overall performance. However, examining how the model uses its two memory pathways shows that simply providing both pathways and applying SFT does not lead to balanced utilization: the model shifts further toward using attention pathway, while its use of the recurrent pathway changes little. Given the distinct strengths of the two pathways observed in Section 2.2, we next investigate whether encouraging more effective use of both pathways can improve memory coordination and further improve overall performance.

## 5 Encouraging the Underutilized Pathway Improves Performance

**Key Observation:** Adding an auxiliary loss that encourages use of the recurrent layers increases their utilization and improves overall performance.

In Section 2.2, we observe that attention and recurrent layers play complementary roles, each carrying important information for different types of questions. However, Section 4.2 shows that standard SFT does not effectively utilize both pathways, instead increasing the model's reliance on attention while leaving the recurrent pathway underutilized. We thus examine whether encouraging the underutilized recurrent pathway leads to better memory coordination and stronger overall performance.

### 5.1 Recurrent Auxiliary Supervision

We augment standard SFT with an auxiliary loss ($\mathcal{L}_{\text{rec}}(\theta)$) that uses the same supervised next-token prediction loss as standard SFT ($\mathcal{L}_{\text{SFT}}$), but masks attention to the preceding context, a mechanism commonly used to control information flow in LMs (Mu et al., 2024; Dong et al., 2019). Unlike attention, recurrent layers remain unrestricted: they process the entire sequence and propagate their state from the preceding context. Thus, during the auxiliary forward pass, information from the

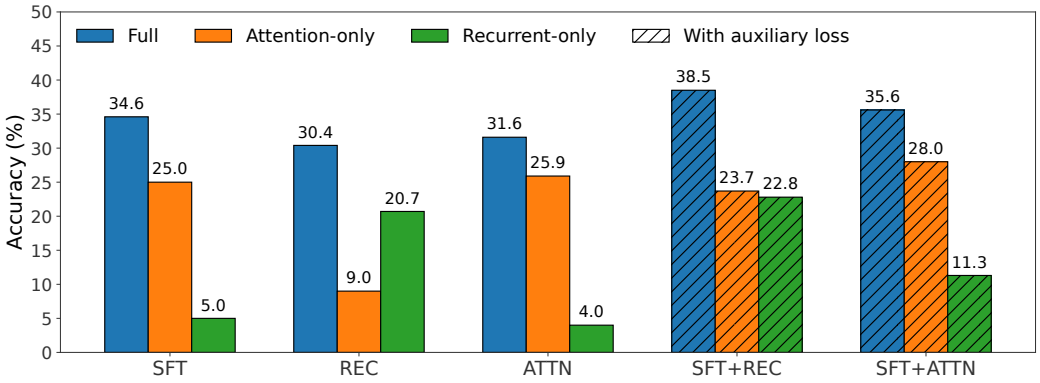


Figure 6: QA accuracy of Qwen3.5-4B across various training objectives (details of objectives in Section 5.2). Hatched bars indicate that it adds an auxiliary loss. *SFT+REC* increases Recurrent-only accuracy while largely preserving Attention-only accuracy and best Full accuracy.

Table 1: QA accuracy across six datasets. First two rows compare standard SFT with our objective; the remaining rows ablate the training objective. Avg. is micro-average over all test examples.

| Model | Training Obj. | DROP | MuSR | HQA | Qasper | NQA | MINT | Avg. |
|---|---|---|---|---|---|---|---|---|
| Qwen3.5 | $\mathcal{L}_{\text{SFT}}$ | 48.6 | 45.6 | 43.1 | 30.1 | 24.4 | 21.1 | 34.6 |
| | $\mathcal{L}_{\text{SFT}} + \mathcal{L}_{\text{rec}}$ (Ours) | 46.9 | 44.8 | 44.0 | **35.9** | **33.7** | **28.9** | **38.5** |
| | $\mathcal{L}_{\text{rec}}$ | 35.0 | 36.5 | 38.0 | 32.1 | 20.1 | 23.6 | 30.4 |
| | $\mathcal{L}_{\text{attn}}$ | **56.9** | 48.1 | 35.7 | 31.3 | 15.6 | 10.3 | 31.6 |
| | $\mathcal{L}_{\text{SFT}} + \mathcal{L}_{\text{attn}}$ | 54.0 | **49.1** | **45.7** | 30.3 | 23.7 | 17.1 | 35.6 |
| Nemotron-H | $\mathcal{L}_{\text{SFT}}$ | 42.4 | 40.9 | 38.7 | 28.1 | 21.9 | 18.8 | 31.0 |
| | $\mathcal{L}_{\text{SFT}} + \mathcal{L}_{\text{rec}}$ (Ours) | 44.5 | 38.1 | 42.1 | **34.1** | **31.7** | **27.8** | **36.2** |
| | $\mathcal{L}_{\text{rec}}$ | 31.6 | 37.6 | **42.8** | 28.0 | 24.1 | 18.4 | 29.8 |
| | $\mathcal{L}_{\text{attn}}$ | 44.7 | **47.6** | 35.8 | 29.4 | 24.0 | 14.7 | 31.4 |
| | $\mathcal{L}_{\text{SFT}} + \mathcal{L}_{\text{attn}}$ | **50.7** | 47.1 | 40.1 | 29.0 | 26.1 | 18.0 | 34.1 |

preceding context can influence the answer only through recurrent memory, directly encouraging greater use of the recurrent pathway. We jointly optimize the unrestricted SFT objective ($\mathcal{L}_{\text{SFT}}(\theta)$) and the recurrent-constrained auxiliary objective ($\mathcal{L}_{\text{rec}}(\theta)$):

$$\mathcal{L}(\theta) = \mathcal{L}_{\text{SFT}}(\theta) + \lambda\, \mathcal{L}_{\text{rec}}(\theta),$$

where $\lambda$ controls the strength of the auxiliary supervision.[2]

### 5.2 Results

**Question Answering.** Table 1 shows accuracy across six question-answering datasets. When comparing standard SFT ($L_{SFT}$) and when adding auxiliary recurrent supervision ($L_{SFT}$+$L_{rec}$), we observe that adding recurrent auxiliary loss improves average accuracy from 34.6% to 38.5% on Qwen3.5 and from 31.0% to 36.2% on Nemotron-H. The gains are particularly large on Qasper, NarrativeQA, and MINTEval, which suggests that the auxiliary loss is especially helpful when answering questions from longer contexts. These larger gains on long-context datasets suggest that the auxiliary loss encourages the model to make better use of recurrent state, consistent with its strength in aggregating information across long contexts observed earlier in Section 2.2.

To understand which part of the objective drives these gains, we conduct analysis with several training variants. For these ablations, we also consider $L_{attn}$, which is the counterpart of $L_{rec}$: it blocks recurrent-state propagation from the preceding context while leaving attention unrestricted. We compare training with auxiliary losses ($L_{rec}$ or $L_{attn}$) alone, and adding $L_{attn}$ to standard

[2] Ablation of different $\lambda$ in Appendix C.1

Table 2: Performance on TextWorld and BabyAI under different training objectives. For each model, the first two rows compare standard SFT with our objective; the remaining rows ablate the training objective. Best in **bold**.

| Model | Training Obj. | TextWorld | | BabyAI | |
|---|---|---|---|---|---|
| | | Quest | Treasure | PutNextLocal | GoToObjMaze |
| Qwen3.5 | $\mathcal{L}_{\text{SFT}}$ | 78.5 | 44.0 | 46.0 | 63.0 |
| | $\mathcal{L}_{\text{SFT}} + \mathcal{L}_{\text{rec}}$ (Ours) | **80.5** | **56.0** | **73.0** | **83.0** |
| | $\mathcal{L}_{\text{rec}}$ | 75.5 | 45.0 | 13.0 | 37.0 |
| | $\mathcal{L}_{\text{attn}}$ | 76.5 | 44.0 | 31.0 | 50.0 |
| | $\mathcal{L}_{\text{SFT}} + \mathcal{L}_{\text{attn}}$ | 77.0 | 49.0 | 47.0 | 70.0 |
| Nemotron-H | $\mathcal{L}_{\text{SFT}}$ | 70.5 | 40.5 | 30.0 | 77.0 |
| | $\mathcal{L}_{\text{SFT}} + \mathcal{L}_{\text{rec}}$ (Ours) | **75.5** | **47.0** | **47.0** | **84.0** |
| | $\mathcal{L}_{\text{rec}}$ | 68.5 | 40.0 | 17.0 | 20.0 |
| | $\mathcal{L}_{\text{attn}}$ | 71.5 | 42.0 | 37.0 | 54.0 |
| | $\mathcal{L}_{\text{SFT}} + \mathcal{L}_{\text{attn}}$ | 72.0 | 44.5 | 43.0 | 80.0 |

SFT ($L_{SFT} + L_{attn}$). We observe that $L_{SFT} + L_{attn}$ yields a smaller gain than $L_{SFT} + L_{rec}$ for both models, with average absolute improvement of 2.1% and 4.6%, respectively. This suggests that encouraging the underutilized recurrent pathway is more effective than further encouraging the use of attention. Finally, training with only the auxiliary loss ($L_{attn}$ or $L_{rec}$) yields lower average accuracy than standard SFT ($L_{SFT}$) when averaged over both models, by 1.3% and 2.7%, respectively, suggesting that restricted losses are most effective as auxiliary objectives alongside unrestricted SFT.

**Agentic Tasks.** Table 2 shows performance on two agentic tasks: TextWorld and BabyAI. We observe similar trend as we observed from question answering task: when adding recurrent auxiliary supervision to standard SFT ($\mathcal{L}_{\text{SFT}} + \mathcal{L}_{\text{rec}}$), it improves success rate over standard SFT ($\mathcal{L}_{\text{SFT}}$) by a large gain: when averaged across Qwen3.5 and Nemotron-H, the absolute gains are 3.5% and 9.3% in TextWorld Quest and Treasure, respectively and 22.0% and 13.5% in BabyAI PutNextLocal and GoToObjMaze, respectively. Similarly, $\mathcal{L}_{\text{SFT}} + \mathcal{L}_{\text{attn}}$ yields a smaller gain than $\mathcal{L}_{\text{SFT}} + \mathcal{L}_{\text{rec}}$ across all four games, with average absolute improvements of 4.1% and 12.1%, respectively. Furthermore, the larger gains on BabyAI mirror the question-answering results, where we observe that adding recurrent auxiliary supervision yields larger improvement especially with longer context datasets. We observe that BabyAI trajectories are generally longer than those in TextWorld: successful trajectories average 40.2 and 13.8 steps, and all trajectories average 134.0 and 42.2 steps, respectively.

## 6 Auxiliary Supervision Generalizes

In this section, we further analyze whether encouraging the use of the recurrent pathway also improves performance in common use cases of memory, where models store and leverage past trajectories or experience in memory (Zhao et al., 2024; Shinn et al., 2023). We then examine whether our observations generalize to attention-based LMs with multiple memories.

**Learning from Failure Trajectories.** We further analyze whether models can recover from their own errors when given additional attempts (Shinn et al., 2023; Gou et al., 2024). We evaluate on MINTEval as it contains long context with multiple question types. Figure 7 shows the cumulative recovery rate (y-axis) for different maximum numbers of attempts (x-axis). When averaged over all questions (solid lines), both models recover steadily as the number of attempts increases. Adding the auxiliary loss yields consistently higher recovery at every budget, with an absolute improvement of 16.6% on average across budgets. When looking into two splits of the dataset, single-evidence and multi-evidence questions, we observe that retrying is in general more effective on multi-evidence questions for both models: multi- vs. single-evidence show 27.3% vs. 14.8% for standard SFT and 57.6% vs. 17.6% with the auxiliary loss. The improvement from the auxiliary loss is largest on multi-evidence questions, with an absolute improvement of 30.3% compared to 2.8% on single evidence questions. We apply the same retry setup to agentic tasks, providing trajectories in context, and observe the same trend of auxiliary loss improving the cumulative recovery rate by 19.2% on

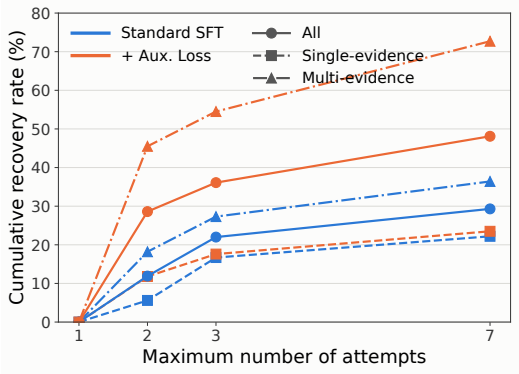


Figure 7: Cumulative recovery rate (y-axis) in QA as the maximum number of attempts increases (x-axis), where previous failed attempts are provided in context. The auxiliary loss improves recovery over standard SFT, especially on multi-evidence questions.

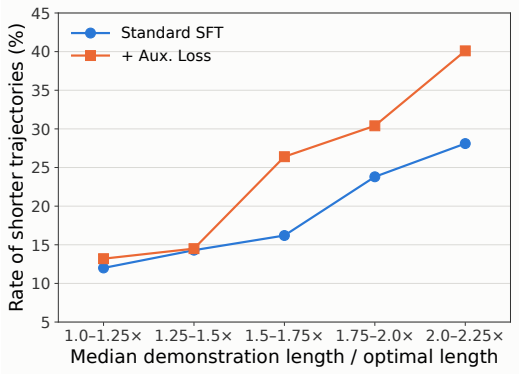


Figure 8: Rate of generating trajectories shorter than all provided successful demonstrations (y-axis), as average demonstration length becomes longer relative to the optimal path (x-axis). The auxiliary loss yields shorter trajectories more often, especially with longer demonstrations.

average (Figure 11 in Appendix). The model trained with the auxiliary loss continues to improve as the number of attempts increases, whereas standard SFT plateaus.

**Learning from Successful Trajectories.** We examine whether the agents can learn from successful trajectories. Since they are often suboptimal, containing unnecessary steps, we provide multiple successful trajectories of varying lengths and measure how often the agent generates a trajectory shorter than the shortest demonstration. As shown in Figure 8, the auxiliary loss improves this rate by 6.0% on average, with larger gains as the demonstrations become longer, suggesting that auxiliary loss helps the model synthesize a better strategy from suboptimal demonstrations.

**Generalizes to Attention-Based LMs.** We find that both the imbalance in memory use and the benefits of auxiliary supervision extend to attention-based LMs with multiple memory types (textual and latent memory). As shown in Figure 9, standard SFT (*SFT*) with both memories achieves 24.1% accuracy, far below the *Oracle* that selects the better of the two single-memory predictions for each example (32.0%) and even below the text-only model (25.9%). This suggests that standard SFT fails to exploit both memory pathways. Adding an auxiliary loss that encourages the use of both memory pathways improves accuracy to 28.6%. Further analysis shows that standard SFT relies heavily on textual memory, whereas the auxiliary loss increases the use of latent memory while preserving the use of textual memory, leading to stronger overall accuracy (Appendix C.3).

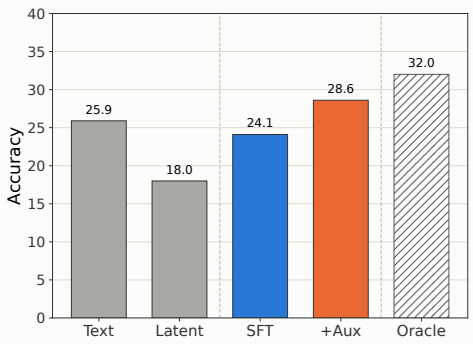


Figure 9: QA accuracy of attention-based models given a single memory type (*Text*, *Latent*) or both (*SFT*, *+Aux*), with a per-example *Oracle* upper bound.

## 7 Conclusion

In this paper, we show that recurrent-attention hybrid LMs tend to rely predominantly on the attention pathway despite the complementary strengths of their recurrent layers, and that standard SFT tends to further reinforce this imbalance. We show that adding a simple auxiliary loss that encourages the model to use the underutilized pathway increases its use and improves overall performance. These findings generalize across multiple recurrent-attention hybrid LMs, across question answering and agentic tasks, and to attention-based LMs that combine different forms of memory.

## 8 AI USE STATEMENT

In this work, we have not used generative AI tools for tasks in the required disclosure. We used generative AI tools for create or edit software code and edit a research paper to improve readability. We have reviewed all AI-assisted work.

## REFERENCES

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.

Aviv Bick, Eric P. Xing, and Albert Gu. Retrieval-aware distillation for transformer-ssm hybrids, 2026. URL https://arxiv.org/abs/2602.11374.

Hector Borobia, Elies Seguí-Mas, and Guillermina Tormo-Carbó. Component ablation for efficient hybrid language model architectures: Performance, resilience, and compression implications, 2026a. URL https://arxiv.org/abs/2603.22473.

Hector Borobia, Elies Seguí-Mas, and Guillermina Tormo-Carbó. Where should lora go? component-type placement in hybrid language models, 2026b. URL https://arxiv.org/abs/2604.22127.

Maxime Chevalier-Boisvert, Dzmitry Bahdanau, Salem Lahlou, Lucas Willems, Chitwan Saharia, Thien Huu Nguyen, and Yoshua Bengio. Babyai: A platform to study the sample efficiency of grounded language learning, 2019. URL https://arxiv.org/abs/1810.08272.

Marc-Alexandre Côté, Ákos Kádár, Xingdi Yuan, Ben Kybartas, Tavian Barnes, Emery Fine, James Moore, Ruo Yu Tao, Matthew Hausknecht, Layla El Asri, Mahmoud Adada, Wendy Tay, and Adam Trischler. Textworld: A learning environment for text-based games, 2019. URL https://arxiv.org/abs/1806.11532.

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.

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, 2021. URL https://arxiv.org/abs/2105.03011.

Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. Unified language model pre-training for natural language understanding and generation, 2019. URL https://arxiv.org/abs/1905.03197.

Xin Dong, Yonggan Fu, Shizhe Diao, Wonmin Byeon, Zijia Chen, Ameya Sunil Mahabaleshwarkar, Shih-Yang Liu, Matthijs Van Keirsbilck, Min-Hung Chen, Yoshi Suhara, Yingyan Lin, Jan Kautz, and Pavlo Molchanov. Hymba: A hybrid-head architecture for small language models, 2024. URL https://arxiv.org/abs/2411.13676.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. Drop: A reading comprehension benchmark requiring discrete reasoning over paragraphs, 2019. URL https://arxiv.org/abs/1903.00161.

Tao Ge, Jing Hu, Lei Wang, Xun Wang, Si-Qing Chen, and Furu Wei. In-context autoencoder for context compression in a large language model, 2024. URL https://arxiv.org/abs/2307.06945.

Zhibin Gou, Zhihong Shao, Yeyun Gong, Yelong Shen, Yujiu Yang, Nan Duan, and Weizhu Chen. Critic: Large language models can self-correct with tool-interactive critiquing, 2024. URL https://arxiv.org/abs/2305.11738.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models, 2021. URL https://arxiv.org/abs/2106.09685.

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.

Tomáš Kočiský, Jonathan Schwarz, Phil Blunsom, Chris Dyer, Karl Moritz Hermann, Gábor Melis, and Edward Grefenstette. The narrativeqa reading comprehension challenge, 2017. URL https://arxiv.org/abs/1712.07040.

Hyunji Lee, Wenhao Yu, Hongming Zhang, Kaixin Ma, Jiyeon Kim, Dong Yu, and Minjoon Seo. Understanding and enhancing mamba-transformer hybrids for memory recall and language modeling, 2025. URL https://arxiv.org/abs/2510.26912.

Hyunji Lee, Justin Chih-Yao Chen, Joykirat Singh, Zaid Khan, Elias Stengel-Eskin, and Mohit Bansal. Minteval: Evaluating memory under multi-target interference in long-horizon agent systems, 2026. URL https://arxiv.org/abs/2605.18565.

Zongqian Li, Yixuan Su, and Nigel Collier. 500xcompressor: Generalized prompt compression for large language models, 2024. URL https://arxiv.org/abs/2408.03094.

Felix Michalak and Steven Abreu. Some attention is all you need for retrieval, 2025. URL https://arxiv.org/abs/2510.19861.

Jesse Mu, Xiang Lisa Li, and Noah Goodman. Learning to compress prompts with gist tokens, 2024. URL https://arxiv.org/abs/2304.08467.

NVIDIA, :, Aaron Blakeman, Aarti Basant, Abhinav Khattar, Adithya Renduchintala, Akhiad Bercovich, Aleksander Ficek, Alexis Bjorlin, Ali Taghibakhshi, Amala Sanjay Deshmukh, Ameya Sunil Mahabaleshwarkar, Andrew Tao, Anna Shors, Ashwath Aithal, Ashwin Poojary, Ayush Dattagupta, Balaram Buddharaju, Bobby Chen, Boris Ginsburg, Boxin Wang, Brandon Norick, Brian Butterfield, Bryan Catanzaro, Carlo del Mundo, Chengyu Dong, Christine Harvey, Christopher Parisien, Dan Su, Daniel Korzekwa, Danny Yin, Daria Gitman, David Mosallanezhad, Deepak Narayanan, Denys Fridman, Dima Rekesh, Ding Ma, Dmytro Pykhtar, Dong Ahn, Duncan Riach, Dusan Stosic, Eileen Long, Elad Segal, Ellie Evans, Eric Chung, Erick Galinkin, Evelina Bakhturina, Ewa Dobrowolska, Fei Jia, Fuxiao Liu, Gargi Prasad, Gerald Shen, Guilin Liu, Guo Chen, Haifeng Qian, Helen Ngo, Hongbin Liu, Hui Li, Igor Gitman, Ilia Karmanov, Ivan Moshkov, Izik Golan, Jan Kautz, Jane Polak Scowcroft, Jared Casper, Jarno Seppanen, Jason Lu, Jason Sewall, Jiaqi Zeng, Jiaxuan You, Jimmy Zhang, Jing Zhang, Jining Huang, Jinze Xue, Jocelyn Huang, Joey Conway, John Kamalu, Jon Barker, Jonathan Cohen, Joseph Jennings, Jupinder Parmar, Karan Sapra, Kari Briski, Kateryna Chumachenko, Katherine Luna, Keshav Santhanam, Kezhi Kong, Kirthi Sivamani, Krzysztof Pawelec, Kumar Anik, Kunlun Li, Lawrence McAfee, Leon Derczynski, Lindsey Pavao, Luis Vega, Lukas Voegtle, Maciej Bala, Maer Rodrigues de Melo, Makesh Narsimhan Sreedhar, Marcin Chochowski, Markus Kliegl, Marta Stepniewska-Dziubinska, Matthieu Le, Matvei Novikov, Mehrzad Samadi, Michael Andersch, Michael Evans, Miguel Martinez, Mike Chrzanowski, Mike Ranzinger, Mikolaj Blaz, Misha Smelyanskiy, Mohamed Fawzy, Mohammad Shoeybi, Mostofa Patwary, Nayeon Lee, Nima Tajbakhsh, Ning Xu, Oleg Rybakov, Oleksii Kuchaiev, Olivier Delalleau, Osvald Nitski, Parth Chadha, Pasha Shamis, Paulius Micikevicius, Pavlo Molchanov, Peter Dykas, Philipp Fischer, Pierre-Yves Aquilanti, Piotr Bialecki, Prasoon Varshney, Pritam Gundecha, Przemek Tredak, Rabeeh Karimi, Rahul Kandu, Ran El-Yaniv, Raviraj Joshi, Roger Waleffe, Ruoxi Zhang, Sabrina Kavanaugh, Sahil Jain, Samuel Kriman, Sangkug Lym, Sanjeev Satheesh, Saurav Muralidharan, Sean Narenthiran, Selvaraj Anandaraj, Seonmyeong Bak, Sergey Kashirsky, Seungju Han, Shantanu Acharya, Shaona Ghosh, Sharath Turuvekere Sreenivas, Sharon Clay, Shelby Thomas, Shrimai Prabhumoye, Shubham Pachori, Shubham Toshniwal, Shyamala Prayaga, Siddhartha Jain, Sirshak Das, Slawek Kierat, Somshubra Majumdar, Song Han, Soumye Singhal, Sriharsha

Niverty, Stefania Alborghetti, Suseella Panguluri, Swetha Bhendigeri, Syeda Nahida Akter, Szymon Migacz, Tal Shiri, Terry Kong, Timo Roman, Tomer Ronen, Trisha Saar, Tugrul Konuk, Tuomas Rintamaki, Tyler Poon, Ushnish De, Vahid Noroozi, Varun Singh, Vijay Korthikanti, Vitaly Kurin, Wasi Uddin Ahmad, Wei Du, Wei Ping, Wenliang Dai, Wonmin Byeon, Xiaowei Ren, Yao Xu, Yejin Choi, Yian Zhang, Ying Lin, Yoshi Suhara, Zhiding Yu, Zhiqi Li, Zhiyu Li, Zhongbo Zhu, Zhuolin Yang, and Zijia Chen. Nemotron-h: A family of accurate and efficient hybrid mamba-transformer models, 2025. URL https://arxiv.org/abs/2504.03624.

Charles Packer, Sarah Wooders, Kevin Lin, Vivian Fang, Shishir G. Patil, Ion Stoica, and Joseph E. Gonzalez. Memgpt: Towards llms as operating systems, 2024. URL https://arxiv.org/abs/2310.08560.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026. URL https://qwen.ai/blog?id=qwen3.5.

Jack W. Rae, Anna Potapenko, Siddhant M. Jayakumar, and Timothy P. Lillicrap. Compressive transformers for long-range sequence modelling, 2019. URL https://arxiv.org/abs/1911.05507.

Liliang Ren, Yang Liu, Yadong Lu, Yelong Shen, Chen Liang, and Weizhu Chen. Samba: Simple hybrid state space models for efficient unlimited context language modeling, 2025. URL https://arxiv.org/abs/2406.07522.

Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning, 2023. URL https://arxiv.org/abs/2303.11366.

Zayne Sprague, Xi Ye, Kaj Bostrom, Swarat Chaudhuri, and Greg Durrett. Musr: Testing the limits of chain-of-thought with multistep soft reasoning, 2024. URL https://arxiv.org/abs/2310.16049.

Yu Wang, Ryuichi Takanobu, Zhiqi Liang, Yuzhen Mao, Yuanzhe Hu, Julian McAuley, and Xiaojian Wu. Mem-$\alpha$: Learning memory construction via reinforcement learning, 2025. URL https://arxiv.org/abs/2509.25911.

Chunlei Xin, Yaojie Lu, Hongyu Lin, Shuheng Zhou, Huijia Zhu, Weiqiang Wang, Zhongyi Liu, Xianpei Han, and Le Sun. Beyond full fine-tuning: Harnessing the power of LoRA for multi-task instruction tuning. In Nicoletta Calzolari, Min-Yen Kan, Veronique Hoste, Alessandro Lenci, Sakriani Sakti, and Nianwen Xue (eds.), *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, May 2024. URL https://aclanthology.org/2024.lrec-main.206/.

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

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, 2018. URL https://arxiv.org/abs/1809.09600.

Masakazu Yoshimura, Teruaki Hayashi, and Yota Maeda. Mambapeft: Exploring parameter-efficient fine-tuning for mamba, 2025. URL https://arxiv.org/abs/2411.03855.

Yi Yu, Liuyi Yao, Yuexiang Xie, Qingquan Tan, Jiaqi Feng, Yaliang Li, and Libing Wu. Agentic memory: Learning unified long-term and short-term memory management for large language model agents, 2026. URL https://arxiv.org/abs/2601.01885.

Zeyu Zhang, Ziliang Guo, Yihang Sun, Xichong Zhang, Xixuan Hao, Zehao Lin, Yang Zhang, Xiaoyan Zhao, Tong Shen, Bo Tang, Zhi-Qin John Xu, Junchi Yan, Haofen Wang, Xu Chen, Feiyu Xiong, Zhiyu Li, and Tat-Seng Chua. Metis: Memory foundation model, 2026. URL https://arxiv.org/abs/2607.26760.

Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. Expel: Llm agents are experiential learners, 2024. URL https://arxiv.org/abs/2308.10144.

# A Details on Role analysis of hybrid LMs

## A.1 Verbatim vs. Non-Verbatim Questions

We first distinguish questions based on whether their answers can be directly recovered from the source context. We define a question as *verbatim* when its normalized gold answer appears in the source document and *non-verbatim* otherwise. Approximately 34.6% of the test examples (about 2.8k examples) are verbatim and to match the number of instances for the two categories, we randomly sample the same number of examples from the remaining non-verbatim questions.

## A.2 Single- vs. Multi-Evidence Questions

We next examine whether the relative strengths of the two memory pathways change with the amount of evidence required to answer a question. Using the evidence annotations provided by the datasets, we divide questions into those that rely on a single piece of evidence and those that require multiple pieces of evidence. Among the test examples, approximately 30.2% (about 2.5k) are single-evidence and 39.7% (about 3.3k) are multi-evidence, and the rest does not provide explicit evidence annotations. To balance the two categories, we randomly sample the same number of multi-evidence questions as single-evidence ones, resulting in around 2.5k examples each.

# B Details on Experimental Setup

## B.1 Question Answering

We evaluate each dataset with its standard metric after text normalization: token-level F1 for NarrativeQA and Qasper, and exact match (EM) for DROP, HotpotQA, and MINTEval. For MuSR, we provide all answer options in the input and ask the model to generate one of them, following the same generation format as the other datasets, and compute EM. For MINTEval, we evaluate on the wiki-revision and git-commit subsets, excluding the state-tracking subset, which is synthetic, and the dialogue subset, whose contexts are too long for our setting. For brevity, we refer to these metrics collectively as accuracy in the paper. We train models for 5 epochs with early stopping. For evaluation, we subsample large test sets (DROP, NarrativeQA, HotpotQA, and MINTEval) to 1.5k due to compute cost.

## B.2 Agentic Setup

We report success rate, the fraction of games in which the agent completes the goal within the step limit. For TextWorld, we set the maximum number of steps to 100 and use 1,000 training, 150 validation and 200 test games, generated with non-overlapping random seeds. For BabyAI, we use the official maximum number of steps for each level with 100 instances for each training level (MiniBossLevel) and test levels (PutNextLocal, GoToObjMaze). In the retry experiments (Section 6), the agent is allowed up to 7 attempts per game, with its previous trajectories provided in context. In the demonstration experiments, we run the agent multiple times on each game to build a pool of successful trajectories with varying lengths, and sample five trajectories from this pool as demonstrations. We train the agents using the gold trajectories for 5 epochs with early stopping.

# C Additional Analysis

## C.1 Effect of the auxiliary loss weight $\lambda$

Figure 10 shows the success rate of Qwen3.5 on TextWorld Quest as we vary the weight of the auxiliary loss $\lambda$. Adding the auxiliary loss consistently improves over standard SFT ($\lambda = 0$, 78.5%) for all values of $\lambda$. We use $\lambda = 0.25$, which achieves the best performance, in all experiments unless otherwise specified.

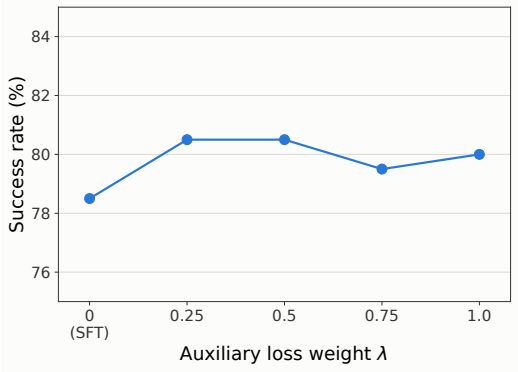


Figure 10: Success rate (y-axis) of Qwen3.5 on TextWorld Quest across different auxiliary loss weights $\lambda$ (x-axis), where $\lambda = 0$ denotes standard SFT. The auxiliary loss improves over standard SFT for all $\lambda$.

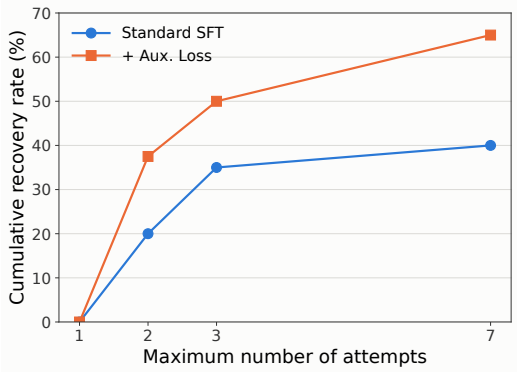


Figure 11: Cumulative recovery rate (y-axis) on agentic tasks as the maximum number of attempts increases (x-axis), where previous failed trajectories are provided in context. The auxiliary loss continues to improve as attempts accumulate, whereas standard SFT plateaus.

### C.2 Agentic Tasks

We examine whether agents can learn from their own failures: when the agent fails to complete a game, it retries the same game up to a maximum number of attempts, with its previous trajectories provided in context. As shown in Figure 11, the auxiliary loss improves the cumulative recovery rate by 19.2% on average. Notably, the standard SFT model tends to converge as the number of attempts increases, whereas the model with the auxiliary loss continues to improve, suggesting that it learns better from past failure trajectories.

### C.3 Generalizes to Attention-Based Models

To examine whether our findings generalize to attention-based models, which do not have multiple built-in memory pathways, we provide them with two commonly used forms of memory: textual (Packer et al., 2024; Yu et al., 2026; Wang et al., 2025) and latent (Mu et al., 2024; Li et al., 2024) memory. We conduct the experiment with `Qwen3-4B-Instruct` model.

#### C.3.1 Textual & Latent Memory.

For **textual memory**, we summarize a given document and train the model to generate the answer conditioned on the resulting summary and the question. For **latent memory**, following prior work (Mu et al., 2024; Ge et al., 2024), we represent each document as a layer-wise KV prefix with $K$ memory slots. Specifically, we append $K$ learnable memory tokens to the end of the document and run the resulting sequence through the model. Because these memory tokens follow the document, they can attend to and aggregate information from the preceding tokens. We retain the key-value representations at these $K$ memory tokens positions from every transformer layer; together, these representations form the latent memory. Training proceeds in two stages: in Stage 1, we train the model to encode document information into these latent memory slots using reasoning supervision. In Stage 2, we provide the latent memory as a prefix cache to the model, which is trained to generate the answer conditioned on the latent memory and the question. The answer-generation loss is backpropagated through the entire computation, including the forward pass used to construct the latent memory.

**Textual and Latent Memory Play Complementary Roles.** We compare textual and latent memory under a range of matched memory budgets along two axes: whether the answer appears word-for-word in the context (verbatim vs. non-verbatim) and whether answering requires one or multiple pieces of evidence (single- vs. multi-evidence). See Appendix A for details. As shown in Figure 12, textual memory shows a larger advantage when the required information is directly recoverable (ver-

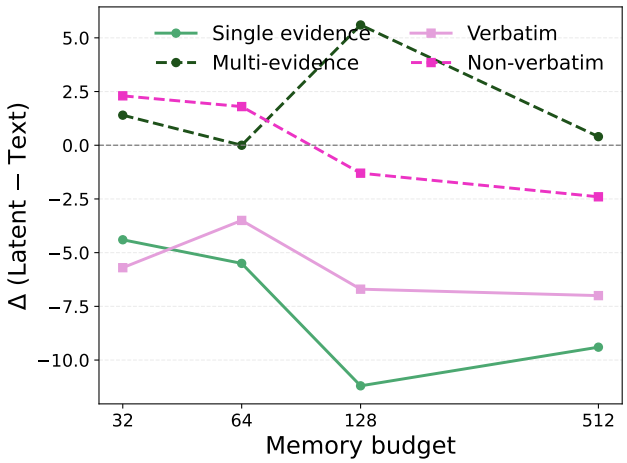


Figure 12: Relative performance of latent versus textual memory (y-axis) under matched memory budgets (x-axis). Textual memory is stronger (negative $\Delta$) on verbatim and single-evidence questions, while latent memory is relatively more competitive (less negative, or even positive $\Delta$) on non-verbatim and multi-evidence questions.

batim) or localized in a single piece of evidence (single-evidence), whereas latent memory becomes more competitive when the answer must be inferred (non-verbatim) or requires combining multiple pieces of evidence (multi-evidence). This suggests that textual memory is especially effective when prediction relies on precise information explicitly retained in memory, whereas latent memory is relatively more effective when information must be transformed or aggregated.

### C.3.2 Complementary Information Does Not Guarantee Effective Coordination

**Key Observation:** Naively training hybrid memory with standard SFT fails to coordinate its components effectively, instead favoring textual memory.

**Naive Hybrid Memory.** For *naive hybrid memory*, we follow the same Stage 1 as for latent memory, training the model with reasoning supervision to encode document information into the $K$ latent memory slots. In Stage 2, we provide both the latent memory and textual memory to the model and train it with the same question-answering objective. Thus, both memory representations are *jointly* available during training, without any additional supervision that explicitly encourages coordination between them.

**Analysis.** Figure 9 shows question answering performance with different types of memory, using a textual memory budget of 8k tokens (Wang et al., 2025) and a latent memory budget of 128 slots (Ge et al., 2024). The *Oracle*, which selects the correct prediction whenever either single-memory model succeeds, measures the complementary information available across the two memories. It outperforms the single-memory models (*Text* and *Latent*) by 10.1% on average, indicating that the two memories capture complementary information. However, the model trained with both memories under standard SFT (*SFT*) remains 7.9% below the *Oracle* and even falls below the *Text* (24.1% vs. 25.9%), suggesting that naively combining the two memories does not ensure better coordination or improved performance, and can instead lead them to interfere with each other. These results show that **the presence of complementary information across memories does not by itself guarantee effective coordination under joint training**.

### C.3.3 Encouraging Memory Coordination with Auxiliary Supervision

**Key Observation:** Adding an auxiliary loss that encourages the model to use a designated memory pathway improves memory coordination and overall performance. The gains are largest when pathway selection follows the verbatim criterion identified in Appendix C.3.1.

The previous section shows that naive joint training leads to imbalanced use of the two memory pathways, with the model relying predominantly on textual memory and underutilizing information captured by latent memory. We therefore examine whether an auxiliary objective that explicitly encourages the model to make use of the underutilized latent memory improves memory coordination and overall performance.

**Auxiliary Loss for Memory Coordination.** To examine whether encouraging the use of the underutilized latent memory improves memory coordination and, in turn, improves overall performance, we add an auxiliary loss that assigns each memory an explicit role based on its strengths (Appendix C.3.1).

The auxiliary loss encourages the model to use a designated memory pathway by making the other pathway uninformative for the target. For each example $i$, the standard hybrid-memory objective is:

$$\ell(T_i, L_i) = -\log p_\theta(y_i \mid q_i, T_i, L_i),$$

where $q_i$ and $y_i$ denote the query and target, and $T_i$ and $L_i$ denote the textual and latent memories constructed from the source context of example $i$.

We augment the objective with an auxiliary term ($\mathcal{L}_i^{\text{aux}}$):

$$\mathcal{L}_i = \ell(T_i, L_i) + \lambda \mathcal{L}_i^{\text{aux}}, \qquad \mathcal{L}_i^{\text{aux}} = \begin{cases} \ell(T_i, \text{sg}[L_j]), & \text{role}(i) = \text{text}, \\ \ell(T_j, L_i), & \text{role}(i) = \text{latent}. \end{cases} \tag{1}$$

Here, $\lambda$ controls the weight of the auxiliary loss, and $j$ denotes a sampled irrelevant example[3]. For a *text-role* example ($\text{role}(i) = \text{text}$), we keep the correct textual memory $T_i$ and replace the latent memory with $L_j$, so that the model must rely on $T_i$ to predict $y_i$; we stop gradients ($\text{sg}[\cdot]$) through $L_j$ so that this auxiliary term does not update the latent-memory writer. For a *latent-role* example ($\text{role}(i) = \text{latent}$), we instead keep the correct latent memory $L_i$ and replace the textual memory with $T_j$, encouraging the model to preserve and use the target-relevant information through the latent pathway. We use the same $\lambda$ to 0.25 for all experiments.

Since the optimal role for each example is not known a priori, we experiment over four criteria for assigning $\text{role}(i)$. Outcome assigns roles based on which single-memory model answers the example correctly: $\text{role}(i) = \text{latent}$ if the latent-memory model answers example $i$ correctly, and $\text{role}(i) = \text{text}$ otherwise. Random assigns each example to the text or latent role uniformly at random. Verbatim follows the distinction identified in Appendix C.3.1, assigning verbatim examples to the text role and non-verbatim examples to the latent role: $\text{role}(i) = \text{text}$ if $i$ = verbatim else $\text{role}(i) = \text{latent}$. Reverse uses the opposite assignment, assigning verbatim examples to the latent role and non-verbatim examples to the text role: $\text{role}(i) = \text{latent}$ if $i$ = verbatim else $\text{role}(i) = \text{text}$.

**Results.** We examine whether auxiliary supervision improves coordination between textual and latent memory, and how the criterion used to assign $\text{role}(i)$ affects this coordination. Table 3 reports accuracy together with the use of each memory ($\Delta$Latent and $\Delta$Text). Standard SFT without auxiliary supervision (Naive) largely ignores latent memory: replacing it with an unrelated one even slightly improves accuracy ($\Delta\text{Latent} = -2.4$).

*Role-conditioned supervision.* Among the role assignment criteria, Verbatim achieves the best accuracy, improving it from 24.1% to 28.6%, and the largest increase in latent-memory use ($-2.4 \rightarrow +4.7$) while largely preserving textual-memory use ($+11.4 \rightarrow +10.4$). Outcome also improves accuracy (26.1%) but yields a smaller increase in latent-memory use; we hypothesize that

[3]To reduce the unrelated factor impact, we sample $j$ from the same source dataset with a similar document length.

Table 3: Accuracy and memory use across auxiliary-training variants. ΔLatent and ΔText denote the accuracy drop when the latent or textual memory, respectively, is replaced with one constructed from an unrelated context; higher values indicate greater use of that memory. *Role-conditioned* indicates whether auxiliary supervision uses per-example role assignment.

| Role-conditioned | Variant | Acc. | ΔLatent | ΔText |
|---|---|---|---|---|
| - | NAIVE | 24.1 | -2.4 | +11.4 |
| Yes | OUTCOME | 26.1 | +0.8 | +11.5 |
| | VERBATIM | **28.6** | **+4.7** | +10.4 |
| | REVERSE | 24.5 | +1.1 | +7.2 |
| | RANDOM | 23.9 | -1.0 | +10.5 |
| No | LATENT-AUX | 26.1 | +3.4 | +8.5 |
| | TEXT-AUX | 24.9 | +0.0 | **+14.5** |
| | UNROUTED | 24.4 | -1.2 | +9.1 |
| - | ORACLE | 32.0 | | |

per-example correctness is influenced by factors beyond the consistent strengths of each memory, making it a noisier signal for role assignment. REVERSE increases latent-memory use only modestly while reducing textual-memory use, and RANDOM does not improve accuracy, showing that the assignment criterion is critical for coordinating the two memories.

*Supervision without role assignment.* We also apply auxiliary supervision without per-example role assignment: LATENT-AUX and TEXT-AUX assign all examples to the latent or text role, respectively, and UNROUTED applies both auxiliary terms to every example. LATENT-AUX increases latent-memory use and improves accuracy to 26.1%, but at the cost of textual-memory use, whereas TEXT-AUX mainly increases textual-memory use with little gain in accuracy, and UNROUTED improves neither memory use nor accuracy. Together, these results suggest that increasing the use of a single memory can help, but the largest gains come from assigning each example to the memory suited to it, according to the complementary strengths of the two memories.