# LM2: LARGE MEMORY MODELS FOR LONG CONTEXT REASONING

**Jikun Kang, Wenqi Wu, Filippos Christianos, Alex J. Chan**
Convergence AI
London, UK
{jikun, wenqi, filippos, alex}@convergence.ai

**Fraser Greenlee, George Thomas, Marvin Purtorab, Andy Toulis**
Convergence AI
London, UK
{fraser, george, marvin, andy}@convergence.ai

## ABSTRACT

This paper introduces the *Large Memory Model* (LM2), a decoder-only Transformer architecture enhanced with an auxiliary memory module that aims to address the limitations of standard Transformers in multi-step reasoning, relational argumentation, and synthesizing information distributed over long contexts. The proposed LM2 incorporates a memory module that acts as a contextual representation repository, interacting with input tokens via cross attention and updating through gating mechanisms. To preserve the Transformer's general-purpose capabilities, LM2 maintains the original information flow while integrating a complementary memory pathway. Experimental results on the BABILong benchmark demonstrate that the LM2model outperforms both the memory-augmented RMT model by 37.1% and the baseline Llama-3.2 model by 86.3% on average across tasks. LM2 exhibits exceptional capabilities in multi-hop inference, numerical reasoning, and large-context question-answering. On the MMLU dataset, it achieves a 5.0% improvement over a pre-trained vanilla model, demonstrating that its memory module does not degrade performance on general tasks. Further, in our analysis, we explore the memory interpretability, effectiveness of memory modules, and test-time behavior. Our findings emphasize the importance of explicit memory in enhancing Transformer architectures.

## 1 INTRODUCTION

Transformer-based models have achieved remarkable success. Landmark architectures such as GPT-3 Brown et al. (2020), BERT Kenton & Toutanova (2019), and Vision Transformers Dosovitskiy (2020) have established state-of-the-art performance across a wide array of applications, including machine translation Zhu et al. (2020), text summarization Liu & Lapata (2019), question-answering Li et al. (2023), and image recognition Dosovitskiy (2020). As demonstrated by studies on large-scale models, their generalization capabilities improve significantly with increased data and model size, leading to emergent behaviors that extend beyond their original training objectives Kaplan et al. (2020); Kang et al. (2024). Despite their significant contributions, current Transformer models encounter critical limitations when applied to long context reasoning tasks Kuratov et al. (2024). For instance, in the *needle-in-a-haystack* problem, models must answer questions that require reasoning across facts scattered throughout exceedingly long documents. Effectively addressing tasks with extensive context demands the model's ability to discern essential information from vast amounts of irrelevant data.

Recent memory-augmented architectures (e.g., Bulatov et al., 2022; Ko et al., 2024) attempt to tackle these challenges by using recurrent prompts to track long context information. However, these architectures primarily summarize previous answers into prompts without fully integrating long-term information, leading to performance degradation over long contexts. For example, on Task

2 (see appendix A), MemReasoner Ko et al. (2024) achieves a performance score of 60.6 for context lengths under 8K, but drops significantly to 18.5 when the context length exceeds 16K. Additionally, these models are specifically tailored for memory-based tasks, thereby sacrificing the generalization capabilities inherent to large language models (LLMs).

To address these limitations, we propose the Large Memory Model (LM2), a novel architecture that enhances the Transformer framework with a dedicated memory module. This module functions as an auxiliary storage and retrieval mechanism, dynamically interacting with input embeddings to improve performance. The memory module follows a structured process: initializing with a memory bank, leveraging cross attention for efficient interaction with sequence embeddings, and using gating mechanisms, such as forget and input gates, to selectively update stored information. By decoupling memory storage and retrieval from immediate processing, LM2 provides a robust solution for modeling long-term dependencies, overcoming the shortcomings of existing methods while maintaining computational efficiency. This architecture is particularly well-suited for tasks requiring long context and complex reasoning, offering a practical and scalable alternative to current approaches.

Moreover, as illustrated in Figure 1, we maintain the original information flow—-namely, the output embeddings passed from one block to the next—-while introducing an additional, complementary memory information flow represented by the memory embeddings. The memory information flow is controlled by a learnable output gate, which uses cross attention to dynamically regulate the amount of memory information passed to subsequent layers. This design ensures that the original attention information flow remains intact while dynamically incorporating relevant memory information as needed.

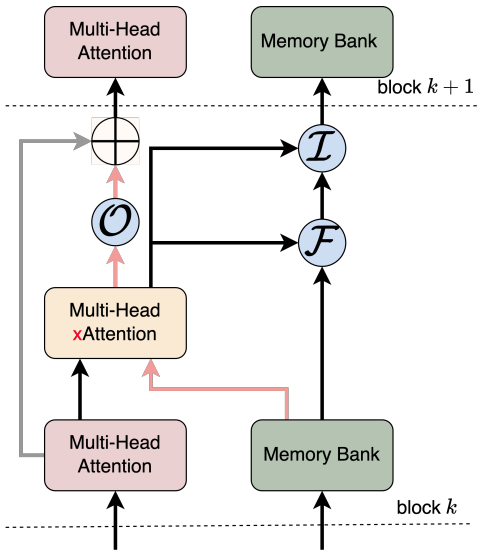



Figure 1: Illustration of LM2 overall architecture. It consists of a separate memory bank, which updates the main information flow through cross attention, and is updated using the input ($\mathcal{I}$), output ($\mathcal{O}$), and forget ($\mathcal{F}$) gates. For the information flow from one block to another, the gray curve shows the normal attention flow and the pink curve shows the extra memory flow.

We first evaluate the effectiveness of LM2 on the *BABILong* dataset Kuratov et al. (2024), a challenging benchmark specifically designed to test memory-intensive reasoning capabilities. To verify that our memory-based approach does not undermine general performance, we also assess LM2 on the MMLU benchmark Hendrycks et al. (2021), which spans a broad array of academic subjects and difficulty levels. Across both evaluations, LM2 outperforms state-of-the-art (SOTA) memory model Recurrent Memory Transformer (RMT) Bulatov et al. (2022) by up to 80.4%, illustrating enhanced proficiency in multi-hop inference, numerical reasoning, and relational argumentation. These improvements underscore the value of incorporating our explicit memory mechanisms within Transformer architectures, enabling more robust handling of extended contexts.

The contributions of this work are summarized as follows: (1) We propose a novel memory-augmented Transformer architecture that incorporates a dynamic memory module capable of capturing and leveraging long-term dependencies in sequential data. (2) We introduce an additional memory information flow within the decoder block that complements the existing attention mechanism, enabling the integration of enriched memory information while preserving the original attention information. (3) Through extensive experiments on long context reasoning tasks (up to a context of 128K tokens), LM2 outperforms SOTA memory-augmented model RMT and non-memory baseline Llama-3.2 on average 37.1% and 86.3%, respectively, demonstrating the practical benefits of our approach.

## 2 LARGE MEMORY MODEL (LM2)

We present Large Memory Model (LM2), a memory-augmented Transformer model designed to enhance its long-term memory capabilities. LM2 consists of multiple Transformer decoder blocks, augmented with a memory module that dynamically stores and updates intermediate sequences of representations. The decoder block processes input sequences using positional embeddings, while the memory module interacts with these embeddings via cross attention mechanisms. We use a skip connection between the multi-head attention and the memory modules to facilitate learning and maintain the original intermediate embeddings of the Transformer. The memory updates are controlled by learnable control gates, denoted as $\mathcal{F}$, $\mathcal{I}$, and $\mathcal{O}$, which correspond to the *forget*, *input*, and *output* gates, respectively. The memory module operates through two primary stages: memory information flow, and memory updates. Each of these stages is elaborated on in the following sections.

### 2.1 MEMORY INFORMATION FLOW

As depicted in Figure 1, we introduce an explicit memory module, named the memory bank $\mathbf{M} \in \mathbb{R}^{N \times d \times d}$, designed to store long-term memory. Here, $N$ denotes the number of memory slots, while $d$ represents the hidden dimension of each slot. For simplicity, each memory slot is initialized as an identity matrix: $\mathbf{M}_r = \mathbf{I}_{d \times d}$, where $r \in \{1, \ldots, N\}$ and $\mathbf{I}_{d \times d}$ is the identity matrix.

We use a cross attention-based mechanism between the memory bank and input embeddings to locate memory slots that contain relevant information. This approach is based on the idea that humans tend to store and group related information together (e.g., in Documentation Science and Archival Science (Dooley, 2007)). Note that the input embeddings $\mathbf{E}$ are encoded by the positional encoder, which embeds the input tokens and persists the temporal correlations between states and actions. Concretely, each input embedding $\mathbf{E}$ acts as the *query*, while the memory bank $\mathbf{M}$ serves as both the *key* and the *value* store. Intuitively, this means we look up "where" (via the key) in $\mathbf{M}$ to find relevant information and then retrieve it (via the value). To enable cross attention, the input embeddings $\mathbf{E} \in \mathbb{R}^{T \times d}$ (where $T$ is the sequence length) and memory bank $\mathbf{M}$ are projected into query ($\mathbf{Q}$), key ($\mathbf{K}$), and value ($\mathbf{V}$) spaces:

$$\mathbf{Q} = \mathbf{E}_t \mathbf{W}^Q, \quad \mathbf{K} = \mathbf{M}_t \mathbf{W}^K, \quad \mathbf{V} = \mathbf{M}_t \mathbf{W}^V, \tag{1}$$

where $\mathbf{W}^Q, \mathbf{W}^K, \mathbf{W}^V \in \mathbb{R}^{d \times d}$ are learnable projection matrices, and $t$ stands for decoder block $t$.

The attention scores are computed as the scaled dot product of the query and key matrices: $\mathbf{A} = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d}}\right)$, where $\mathbf{A} \in \mathbb{R}^{T \times N}$ represents the alignment between the input sequence and memory slots. The resultant attention output is $\mathbf{E}_{\text{mem}} = \mathbf{A}\mathbf{V}$, where $\mathbf{E}_{\text{mem}} \in \mathbb{R}^{T \times d}$ integrates information from the input and memory. To ensure temporal consistency, causal masking is applied, and optionally, top-$k$ attention is used to retain only the most relevant memory interactions.

To regulate the influence of the memory information (gray path in Figure 1) on the existing attention information flow (pink path in Figure 1), an output gate is introduced. The output gate dynamically controls the contribution of the memory retrieval based on the cross attention output $\mathbf{E}_{\text{mem}}$:

$$g_{\text{out}} = \sigma\left(\mathbf{E}_{\text{mem}} \mathbf{W}_{\text{out}}\right), \tag{2}$$

where $\mathbf{W}_{\text{out}} \in \mathbb{R}^{d \times d}$ is a learnable parameter matrix, and $\sigma$ is the sigmoid activation function. The gated memory output is then computed as:

$$\mathbf{E}_{\text{gated}} = g_{\text{out}} \cdot \mathbf{M}_t. \tag{3}$$

The gated memory output is integrated into the standard attention flow of the Transformer decoder through a skip connection. Specifically, the output of the self-attention mechanism, $\mathbf{E}_{\text{attn}}$, is combined with the gated memory output as $\mathbf{E}_{\text{next}} = \mathbf{E}_{\text{attn}} + \mathbf{E}_{\text{gated}}$. This skip connection ensures that the standard attention output and the memory-augmented features jointly contribute to the next decoder layer. By dynamically gating the memory retrieval and integrating it with the attention flow, LM2 effectively balances the use of memory and contextual information, enhancing its ability to model long-term dependencies while preserving the core Transformer operations.

### 2.2 MEMORY UPDATES

As illustrated in Figure 2, the update process is divided into three distinct phases: the *input*, *forget*, and *output* (previously described). By gating how much new information is introduced and how much old information is discarded, the memory module avoids overwriting crucial long-term facts while also eliminating irrelevant or outdated content when processing long context sequences.

**Input Phase** During the input phase, the model decides how much of the newly computed embeddings ($\mathbf{E}_{\text{mem}}$) to incorporate into the memory. To achieve this, first an *input gate* is computed:

$$g_{\text{in}} = \sigma\big(\mathbf{E}_t \mathbf{W}_{\text{in}}\big), \tag{4}$$

where $\mathbf{W}_{\text{in}} \in \mathbb{R}^{d \times d}$ is a learnable parameter matrix, $\mathbf{E}_t$ is the current input representation, and $\sigma$ is the sigmoid activation function. This gating mechanism serves as a filter, deciding which relevant information should be "written" into memory, while also preventing the influx of noise or redundant details.

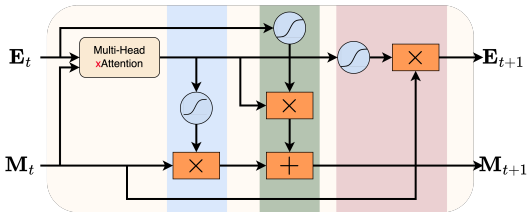



Figure 2: Illustration of how memory module works inside of each decoding block, where blue, green, and red box corresponds to forget, input, and output phase.

**Forgetting Phase** Once new information is made available during the input phase, the memory must also decide which parts of its existing content to discard. This is governed by the *forget gate*:

$$g_{\text{forget}} = \sigma\big(\mathbf{E}_{\text{mem}} \mathbf{W}_{\text{forget}}\big), \tag{5}$$

where $\mathbf{W}_{\text{forget}} \in \mathbb{R}^{d \times d}$. By outputting values less than one, the forget gate selectively "erases" memory slots that are no longer relevant, allowing the model to focus on more recent or salient information.

**Memory Update** Combining these two gating mechanisms leads to the updated memory state:

$$\mathbf{M}_{t+1} = g_{\text{in}} \cdot \tanh(\mathbf{E}_{\text{mem}}) \; + \; g_{\text{forget}} \cdot \mathbf{M}_t, \tag{6}$$

where a $\tanh$ function is applied to keep the new memory content bounded. Through these regulated phases, the memory module memorizes the most relevant information and removes outdated details, ensuring that it remains both concise and informative over time.

## 3 PRE-TRAINING LM2

We base our work on the Llama-3 model framework Dubey et al. (2024), employing it as the foundation for our Transformer architecture. Its architecture comprises 16 decoder blocks, each with a model dimension of 2,048. The feed-forward networks within these blocks have an inner dimension of 8,192. The model utilizes 32 attention heads, with 8 dedicated key/value heads.

Our memory module extends this architecture, consisting of 2,048 memory slots, each with a dimension of 2,048. Memory modules are integrated into all 16 decoder blocks, as this configuration empirically achieves the best performance (see Section 4.3 for detailed results). The Llama-3 framework comprises approximately 1.2 billion parameters, with an additional 0.5 billion parameters introduced by the memory module, resulting in a total of 1.7 billion parameters for the LM2 model.

For pre-training, we leverage a high-quality dataset sourced from the SmolLM-Corpus Loubna et al. (2023). The dataset is structured into three distinct sections: synthetic test-books and stories, educational web content, and python codes. To ensure a focused evaluation on language tasks, we exclude Python sample training data from this process. The specific details of the training dataset are outlined as follows: **Synthetic Textbooks and Stories**: Generated using advanced language models to cover a wide range of topics, providing 28 billion tokens of diverse educational content. **Educational Web Content**: Filtered and deduplicated web pages from FineWeb-Edu Penedo et al. (2024), contributing 220 billion tokens of high-quality educational material.

Table 1: Performance on the BABILong dataset: All models are evaluated on various context lengths ranging from 0K, 1K, 2K, and 4K to an aggregated average length of $\geq 8K$. Qa stands for various subsets. Due to page limits, we aggregate the results for 8K, 16K, 32K, 64K, and 128K into a single metric, with detailed results provided in Appendix B.

| model | qa1 | qa2 | qa3 | qa4 | qa5 | qa6 | qa7 | qa8 | qa9 | qa10 | Avg. |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0K | | | | | | | | | | | |
| Llama-3.2-1.2B | 54.0 | 25.0 | 29.0 | 62.0 | 59.0 | 49.0 | 14.0 | 52.0 | 41.0 | 22.0 | 40.7 |
| vanilla-Llama-1.7B | 86.0 | 57.0 | 46.0 | 59.0 | 85.0 | 83.0 | 95.0 | 79.0 | 83.0 | 77.0 | 75.0 |
| RMT-1.7B | 85.0 | 49.0 | 49.0 | 81.0 | 95.0 | 84.0 | 82.0 | 78.0 | 85.0 | 76.0 | 76.4 |
| LM2-1.7B | **99.0** | **89.0** | **70.0** | **88.0** | **98.0** | **95.0** | **96.0** | **97.0** | **99.0** | **94.0** | **92.5** |
| 1K | | | | | | | | | | | |
| Llama-3.2-1.2B | 48.0 | 22.0 | 24.0 | 55.0 | 69.0 | 49.0 | 9.0 | 31.0 | 55.0 | 33.0 | 39.5 |
| Llama-3.2-1.2B-RAG | 51.0 | 14.0 | 19.0 | 59.0 | 80.0 | 49.0 | 10.0 | 38.0 | 40.0 | 46.0 | 40.6 |
| vanilla-Llama-1.7B | 31.0 | 21.0 | 44.0 | 43.0 | 71.0 | 60.0 | 71.0 | 40.0 | 67.0 | 58.0 | 50.6 |
| RMT-1.7B | 35.0 | 26.0 | 29.0 | 33.0 | 61.0 | 50.0 | 83.0 | 41.0 | 68.0 | 53.0 | 47.9 |
| LM2-1.7B | **85.0** | **59.0** | **72.0** | **68.0** | **91.0** | **84.0** | **96.0** | **69.0** | **82.0** | **77.0** | **78.3** |
| 2K | | | | | | | | | | | |
| Llama-3.2-1.2B | 44.0 | 18.0 | 19.0 | **50.0** | 64.0 | 52.0 | 18.0 | 24.0 | 55.0 | 42.0 | 38.6 |
| Llama-3.2-1.2B-RAG | 52.0 | 11.0 | 12.0 | 49.0 | 75.0 | 48.0 | 5.0 | 33.0 | 50.0 | 43.0 | 37.8 |
| vanilla-Llama-1.7B | 25.0 | 22.0 | 37.0 | 34.0 | 58.0 | 60.0 | 65.0 | 38.0 | 66.0 | 58.0 | 46.3 |
| RMT-1.7B | 44.0 | 21.0 | 43.0 | 41.0 | 79.0 | 47.0 | 78.0 | 41.0 | 69.0 | 51.0 | 51.4 |
| LM2-1.7B | **58.0** | **43.0** | **64.0** | 43.0 | **87.0** | **73.0** | **93.0** | **53.0** | **75.0** | **69.0** | **65.8** |
| 4K | | | | | | | | | | | |
| Llama-3.2-1.2B | 37.0 | 16.0 | 25.0 | 56.0 | 56.0 | 50.0 | 14.0 | 27.0 | 55.0 | 32.0 | 36.8 |
| Llama-3.2-1.2B-RAG | 47.0 | 3.0 | 16.0 | **58.0** | 68.0 | 58.0 | 3.0 | 36.0 | 45.0 | 39.0 | 37.3 |
| vanilla-Llama-1.7B | 21.0 | 18.0 | 38.0 | 28.0 | 55.0 | 61.0 | 64.0 | 35.0 | 49.0 | 53.0 | 42.2 |
| RMT-1.7B | 24.0 | 20.0 | 22.0 | 24.0 | 28.0 | 46.0 | 75.0 | 35.0 | **65.0** | 45.0 | 38.4 |
| LM2-1.7B | **46.0** | **37.0** | **48.0** | 34.0 | **78.0** | **66.0** | **93.0** | **45.0** | 62.0 | **50.0** | **55.9** |
| AVG. Length $\geq$8K | | | | | | | | | | | |
| Llama-3.2-1.2B | 19.0 | 8.0 | 17.8 | 27.3 | 36.5 | 49 | 21.3 | 12.8 | 48.0 | 41.8 | 28.2 |
| Llama-3.2-1.2B-RAG | 29.3 | 1.0 | 5.0 | **55.8** | **72.0** | **49.8** | 4.8 | 22.8 | 46.3 | 36.8 | 32.3 |
| vanilla-Llama-1.7B | 11.3 | 15.0 | 21.3 | 14.5 | 31.0 | 44.0 | 63.0 | 33.5 | 42.0 | 36.3 | 31.2 |
| RMT-1.7B | 17.5 | 14.5 | 20.5 | 22.5 | 20.3 | 47.0 | 73.3 | 34.5 | 62.5 | **43.0** | 35.5 |
| LM2-1.7B | **23.8** | **15.0** | **24.5** | 24.0 | 38.8 | 47.3 | **92.8** | **37.0** | **53.8** | 42.0 | **39.9** |

## 4 EXPERIMENTS

We design our experiments to answer the following questions: **Q1:** How does LM2 perform in memory tasks? **Q2:** Does LM2 harm the performance in general tasks? **Q3:** Do we need to include the memory module in all decoder blocks? **Q4:** What is stored in the memory bank? **Q5:** How is the memory module updated at test-time?

To evaluate LM2, we compare its performance against the following baselines: **vanilla-Llama-1.7B:** The Llama 3.2 architecture, pre-trained from scratch on the same datasets as LM2. We scale this model to 1.7 billion parameters for a fair comparison. **RMT-1.7B:** Recurrent Memory Transformer (RMT) Bulatov et al. (2022) is a memory-augmented framework that generates memory tokens, serving as an additional module built on top of existing LLMs. We use the LLaMA-1.7B model as the backbone and fine-tune it on the bAbI training dataset Weston et al. (2016), following the methodology outlined in Kuratov et al. (2024) and Ko et al. (2024). **Llama-3.2-1.2B:** To show case the effectiveness of LM2 we also compared the model against the original model trained by Meta, with the same total number of pure Transformer parameters (1.2B), but trained on far more high-quality tokens. **Llama-3.2-1.2B-RAG:** Lastly, we compare with a version of Llama with retrieval-augmented generation (RAG) to better handle long context problems.

### 4.1 PERFORMANCE ON MEMORY TASKS

**BABILong** The BABILong dataset Kuratov et al. (2024) extends bAbI benchmark Weston et al. (2016) by incorporating significantly longer contexts and more intricate queries, thus demanding advanced memory capabilities and multi-step reasoning. By increasing both contextual and computa-

tional challenges, BABILong offers a rigorous evaluation benchmark for testing memory-augmented models.

Table 1 presents a comparison of our model against the baselines on the BABILong dataset. We report results across multiple context lengths, from 0K context-length, which is identical to bAbI dataset, to the maximum context length of 128K, which is the target context-length of the backbone Llama-3.2 model. From this table, we observe several key findings as follows:

**Performance at bAbI benchmark (0K).** Without additional context, LM2-1.7B achieves the highest average accuracy of 92.5%, surpassing Llama-3.2-1.2B, vanilla-LLama-1.7B, and RMT-1.7B, which average results are 40.7%, 75.0% and 76.4%, respectively. Because Llama-3.2-1.2B-RAG is designed for retrieval-augmented generation and evaluated only at longer contexts, it is not included in the 0K setting. This suggests that LM2's underlying modeling improvements enhance its core reasoning ability.

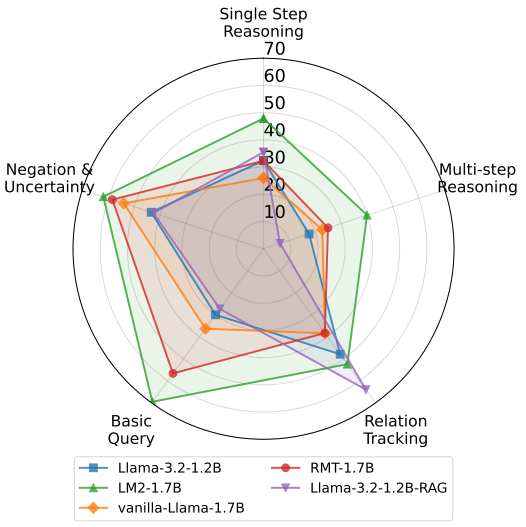



Figure 3: Performance on BABILong benchmark with respect to different reasoning abilities.

**Performance at Long Context Lengths (1K–4K).** As context length increases, performance generally degrades for all models, but LM2-1.7B maintains a noticeable improvement over both standard and retrieval-augmented Llama variants and RMT. For instance, at 4K, LM2-1.7B's average accuracy (55.9%) is higher than Llama-3.2-1.2B, vanilla-LLama-1.7B, and RMT-1.7B, which average results are 36.8%, 42.2% and 48.4%, respectively. This gap underscores LM2's effectiveness for long-term memory ranging from 1K to 4K.

**Performance at Long Contexts (8K–128K).** Although all models exhibit some accuracy decline at these extreme long context lengths, LM2-1.7B remains robust. RMT-1.7B shows reasonable robustness, yet still falls short of LM2-1.7B on most tasks. RAG methods demonstrate some improvements over the baseline Llama, but still falls behind memory-based methods. These results highlight LM2's ability to handle long context problems where Transformer-based models struggle.

**Performance at Different Reasoning Types** To further understand how LM2 performs in different reasoning scenarios, we group the BABILong dataset into five categories: (1) Single-step Reasoning (qa1), (2) Multi-step Reasoning (qa2–3), (3) Relation Tracking (qa4–5), (iv) Basic Queries (qa6–8), and (v) Negation & Uncertainty (qa9–10). Figure 3 depicts the results in a radar chart, where higher values indicate better performance. Across nearly all task categories except for *Relation Tracking*, LM2-1.7B demonstrates the best performance. Notably, LM2 outperforms the other methods on both single and multi-step reasoning, indicating that it can handle more complex, multi-hop inferences and direct fact retrieval with fewer errors. The improvement margin is larger for Basic Queries, Single-Step Reasoning, and Multi-step Reasoning, suggesting that LM2 has strong abilities to retrieve long-term facts and apply them in complex reasoning tasks. The marginally lower performance on Relation Tracking can be attributed to RAG's approach of chunking the context into smaller, more focused "documents" and retrieving only the

Table 2: Performance on MMLU dataset. For better visualization, the dataset is categorized on two criteria - subject and difficulty.

| | | vanilla Llama | RMT | LM2 |
|---|---|---|---|---|
| Subject Category | STEM | 27.2 | 25.7 | **28.1** |
| | Humanities | 28.7 | 26.7 | **32.2** |
| | Social Sciences | 29.2 | 27.0 | **31.6** |
| | Others | 27.7 | 27.1 | **28.0** |
| Difficulty Level | High School | 28.8 | 26.5 | **30.4** |
| | College | 27.7 | 27.1 | **29.0** |
| | Professional | 27.5 | 26.6 | **27.6** |
| | General Knowledge | 27.2 | 25.6 | **28.5** |
| Average | | 28.0 | 26.5 | **29.4** |

most relevant pieces at inference time. RAG makes it much easier to precisely identify which facts are associated with the queried relationship, thus serving as an extremely strong baseline for this task category.

### 4.2 Performance on General Benchmarks

To further evaluate if introducing an extra memory module affects LLMs' general performance, we evaluate the proposed memory-based model, LM2, on the MMLU benchmark Hendrycks et al. (2021), which tests a broad spectrum of subject areas—STEM, Humanities, Social Sciences, and Others—as well as varied difficulty levels—High School, College, Professional, and General Knowledge. Table 2 presents the results of LM2 in comparison to vanilla-Llama and RMT.

Overall, LM2 demonstrates a clear performance gain, improving the average accuracy of vanilla-Llama from 28.0% to 29.4%. On the contrary, despite sharing the same pre-trained model, RMT degrades the performance of vanilla-Llama to 26.5%. Notably, LM2 achieves substantial gains in Humanities and Social Sciences, where LM2 surpasses vanilla-Llama by 3.5% and 2.4%, respectively. These categories often involve context-rich questions, suggesting that LM2's memory-based approach is advantageous for retaining and leveraging more nuanced and interconnected information. Meanwhile, LM2 also sustains competitive performance in STEM and Others, indicating its robustness beyond highly specialized domains.

These results illustrate that LM2 overcomes the drawback associated with memory-augmented models: performance degradation on more general tasks. Current memory-based architectures are carefully designed for memory tasks, weakening their ability to general LLM tasks. However, LM2's performance on all categories of MMLU dataset indicates that the proposed memory mechanism does not impede its general applicability.

### 4.3 Impact of memory modules

We evaluate the effectiveness of proposed memory modules using perplexity as the primary metric across varying numbers of training tokens (measured in billions). Figure 4 illustrates the perplexity trends for the baseline vanilla-Llama and LM2 with varying degrees of memory integration (i.e., 1, 6, 12, and 16 blocks), where 16 is the maximum number of blocks used in Llama-3.2-1B.

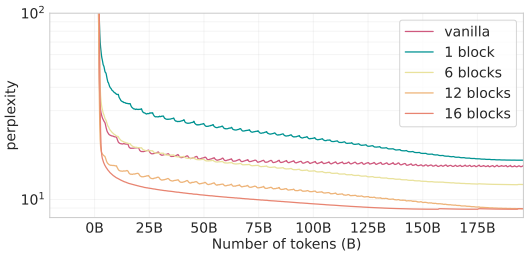



Figure 4: We evaluate variations of integrating memory within the decoder blocks. The number indicates how many of the initial decoder blocks include the memory module, as we found that the order of implementing memory modules does not affect performance.

The results demonstrate that integrating memory information more extensively throughout the decoder leads to improved model performance. Specifically, implementing the memory module in only the first block achieves similar results to the vanilla Llama, but with slower convergence. This suggests that introducing a single memory flow does not degrade overall performance but may slow down training because of extra memory optimization. In contrast, incorporating more memory flows, such as in the 6-block configuration, leads to lower perplexity, highlighting the effectiveness of the proposed memory flow design. The 16-block configuration significantly outperforms the limited 1-block integration, validating that the proposed memory flow is highly advantageous for reducing perplexity and enhancing the overall capabilities of the model.

### 4.4 Analysis of Memory Representations

To gain deeper insights into the information encoded within the memory module, we utilize the Neuron Explainer method Bills et al. (2023). It generates natural language explanations of neuron behavior, simulates activations using these descriptions, and evaluates their accuracy through predictive scoring. We utilize this approach to explain the latent representations of specific memory slots, which helps understand how these slots process and retain task-relevant information. By analyzing activations

within the memory module, the Neuron Explainer identifies patterns in latent representations of each memory slot, mapping them to specific elements of the input text.

We evaluate LM2 using the input text illustrated in Figure 5. Subsequently, we identify and rank the most relevant memory slots, selecting two for sampling (slots 1679 and 1684) along with one of the least relevant memory slot (slot 1). Utilizing the neuron explainer, we investigate the relevance rationales.

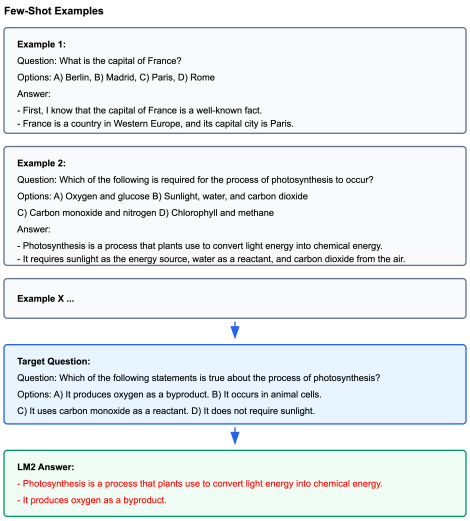



Figure 5: We sample a question from MMLU to test the LM2 in a few-shot fashion. To study how the memory module focuses on relevant information, we place useful information inside one of the few-shot examples.

**Explanation 4.1: Memory Slot 1679**

This memory slot's representations for this specific input text suggest that the memory module's focus is likely on detecting factual information, question and answer structures.

These observations suggest that **Memory Slot 1679** specializes in retrieving and synthesizing factual information for the target question, functioning as a repository for domain-specific knowledge and structured reasoning.

**Explanation 4.2: Memory Slot 1684**

This representations in this memory slot is designed to focus on specific elements within the input text, as evidenced by the pattern in the memory bank.

**Memory Slot 1684**, in contrast, demonstrated a focus on structural elements within the input text. Its activations aligned closely with linguistic markers and contextual cues, such as "Options:" or "Answer:". This behavior implies that Memory Slot 1684 facilitates the model's comprehension of input organization, enabling effective parsing of complex instruction formats and multi-part structures.

**Explanation 4.3: Memory Slot 1**

The representations in this memory slot for the provided input text are primarily negative. This suggests that the module is not detecting the specific aspects it was designed to recognize in the input text.

**Memory Slot 1**, on the other hand, showed predominantly negative activations across the input text, indicating minimal engagement with the task-specific content.

These findings underline the importance of memory modules in gathering information for the generation tasks.

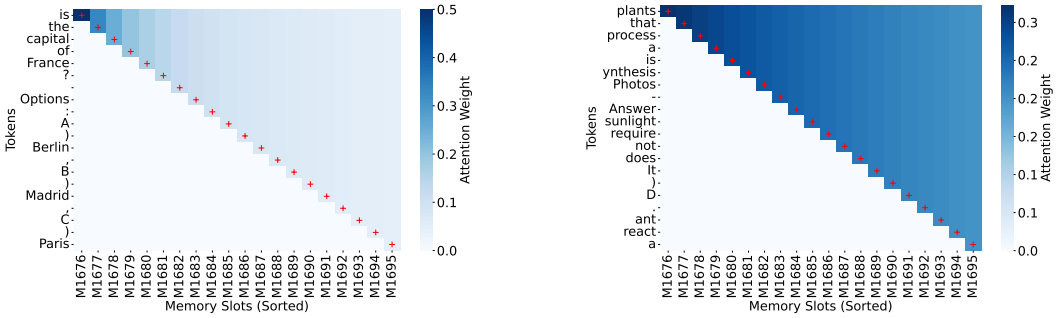
### 4.5 Test-time memory adaptations

We further investigate how memory updates influence model generation during test time. To explore this, we analyze the example illustrated in Figure 5. Cross attention heatmaps, presented in Figure 6, provide key insights into these memory updates.

Figure 6a shows the cross attention heatmap prior to memory updates. In this figure, tokens such as "France" and "Paris" strongly engage with the memory. These tokens do not pertain specifically to the target question about photosynthesis. Instead, on the first pass, memory initially focuses on the structure of question as well as identifying factual information.

Next, we examine the memory heatmap after various inference update steps (one inference step corresponds to a single forward pass for one token). As depicted in Figure 6b, the tokens attended to by the memory slots shift toward those relevant to the target question. Since cross attention

exclusively computes the relationships between input tokens and memory, this shift reflects the influence of test-time memory updates. These changes highlight the adaptive nature of memory during inference.




(a) Heatmaps before memory update.

(b) Heatmaps after memory update.

Figure 6: Cross-attention heatmaps between input tokens and memory. The x-axis shows the memory slots sorted by slot number. The y-axis shows the most attended tokens. Diagonal attentions are marked with "+".

## 5 Related Work

Various methods have been proposed to augment Transformers with memory. One direction is to optimize the attention mechanisms and use some global representations acting as memory points to ensure input coverage. Models like Longformer Beltagy et al. (2020), Big Bird Zaheer et al. (2020), GMAT Gupta & Berant (2020) and Extended Transformer Construction Ainslie et al. (2020) all proposed some sparse attention mechanisms to reduce the quadratic dependency of self-attention to linear and introduced global tokens to encode the information from the entire sequence. Another line of work introduces memorization capabilities to Transformers through recurrence. Transformer-XL Dai et al. (2019) addresses the limitation of fixed-length context by introducing segment-level recurrence and relative position encodings. However, during training, gradients are restricted to individual segments, limiting the model's ability to capture long-term temporal dependencies. Recurrent Memory Transformer (RMT) Bulatov et al. (2022) mitigates these limitations by introducing a more efficient memory mechanism. It adds recurrence to Transformers via a small number of special overlapping memory tokens between segments of long sequences, enabling gradients to propagate across them while significantly reducing memory usage. RMT outperforms Transformer-XL for sequence processing tasks and is on par with Transformer-XL on language modeling, but requires less memory. Associative RMT (ARMT) Rodkin et al. (2024) is a follow-up to RMT that addresses its time complexity issues. Similarly, MemReasoner Ko et al. (2024) introduces a memory-augmented LLM architecture designed for temporal reasoning. However, as demonstrated by Kuratov et al. (2024) and Ko et al. (2024), RMT continues to outperform these subsequent models, maintaining its status as the state-of-the-art (SOTA) method. Therefore, we primarily consider RMT as the SOTA memory-based model and compare LM2 against it.

## 6 Conclusion

In this paper, we introduced Large Memory Model (LM2), a memory-augmented Transformer architecture designed to address long context reasoning challenges. The key innovation is the memory module, integrated inside the decoder blocks, which augments the model with additional memory information while also updating itself. Empirical results on the BABILong benchmark highlights LM2's advantages on various long context tasks. On average across tasks, LM2 outperforms the SOTA memory-augmented RMT model by 37.1%, and a non-memory baseline Llama-3.2 model by 86.3%. Furthermore, LM2 achieves improvement over baselines on the MMLU benchmark, evidencing that its memory module does not degrade performance on general tasks. Overall, these findings underscore the importance of explicit memory mechanisms, and lay a foundation for further research on integrating long-term memory into large language models.

## REFERENCES

Joshua Ainslie, Santiago Ontañón, Chris Alberti, Philip Pham, Anirudh Ravula, and Sumit Sanghai. ETC: encoding long and structured data in transformers. *CoRR*, abs/2004.08483, 2020. URL https://arxiv.org/abs/2004.08483.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. Longformer: The long-document transformer. *CoRR*, abs/2004.05150, 2020. URL https://arxiv.org/abs/2004.05150.

Steven Bills, Nick Cammarata, Dan Mossing, Henk Tillman, Leo Gao, Gabriel Goh, Ilya Sutskever, Jan Leike, Jeff Wu, and William Saunders. Language models can explain neurons in language models. https://openaipublic.blob.core.windows.net/neuron-explainer/paper/index.html, 2023.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

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

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime G. Carbonell, Quoc V. Le, and Ruslan Salakhutdinov. Transformer-xl: Attentive language models beyond a fixed-length context. *CoRR*, abs/1901.02860, 2019. URL http://arxiv.org/abs/1901.02860.

Jackie Dooley. The archival advantage: Integrating archival expertise into management of born-digital library materials. *Archival Science Special Issue on Archiving Research Data*, 7(1), March 2007.

Alexey Dosovitskiy. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Ankit Gupta and Jonathan Berant. GMAT: global memory augmentation for transformers. *CoRR*, abs/2006.03274, 2020. URL https://arxiv.org/abs/2006.03274.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021.

Jikun Kang, Romain Laroche, Xingdi Yuan, Adam Trischler, Xue Liu, and Jie Fu. Think before you act: Decision transformers with working memory. In *ICML*. OpenReview.net, 2024.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of naacL-HLT*, volume 1, pp. 2. Minneapolis, Minnesota, 2019.

Ching-Yun Ko, Sihui Dai, Payel Das, Georgios Kollias, Subhajit Chaudhury, and Aurelie Lozano. Memreasoner: A memory-augmented llm architecture for multi-hop reasoning. In *The First Workshop on System-2 Reasoning at Scale, NeurIPS'24*, 2024.

Yuri Kuratov, Aydar Bulatov, Petr Anokhin, Ivan Rodkin, Dmitry Sorokin, Artyom Sorokin, and Mikhail Burtsev. Babilong: Testing the limits of llms with long context reasoning-in-a-haystack, 2024.

Qianxi Li, Yingyue Cao, Jikun Kang, Tianpei Yang, Xi Chen, Jun Jin, and Matthew E Taylor. Laffi: Leveraging hybrid natural language feedback for fine-tuning language models. *arXiv preprint arXiv:2401.00907*, 2023.

Yang Liu and Mirella Lapata. Hierarchical transformers for multi-document summarization. In Anna Korhonen, David Traum, and Lluís Màrquez (eds.), *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 5070–5081, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1500. URL https://aclanthology.org/P19-1500/.

Allal Loubna, Ben, Lozhkov Anton, and Bakouch Elie. Small language models: Efficient, accessible, and effective. https://huggingface.co/blog/smollm, 2023. Accessed: 2025-01-16.

Guilherme Penedo, Hynek Kydlíček, Anton Lozhkov, Margaret Mitchell, Colin Raffel, Leandro Von Werra, Thomas Wolf, et al. The fineweb datasets: Decanting the web for the finest text data at scale. *arXiv preprint arXiv:2406.17557*, 2024.

Ivan Rodkin, Yuri Kuratov, Aydar Bulatov, and Mikhail Burtsev. Associative recurrent memory transformer, 2024. URL https://arxiv.org/abs/2407.04841.

Jason Weston, Antoine Bordes, Sumit Chopra, and Tomás Mikolov. Towards ai-complete question answering: A set of prerequisite toy tasks. In *ICLR (Poster)*, 2016.

Manzil Zaheer, Guru Guruganesh, Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontañón, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Amr Ahmed. Big bird: Transformers for longer sequences. *CoRR*, abs/2007.14062, 2020. URL https://arxiv.org/abs/2007.14062.

Jinhua Zhu, Yingce Xia, Lijun Wu, Di He, Tao Qin, Wengang Zhou, Houqiang Li, and Tie-Yan Liu. Incorporating bert into neural machine translation. *arXiv preprint arXiv:2002.06823*, 2020.

## A BABILONG DATASET

This section provides an overview of the tasks in BABILong. Each task targets a specific aspect of language understanding and reasoning, forming a core benchmark for assessing model performance on retrieve factors from long context.

- **Task 1: Single Supporting Fact**
  *Goal*: Identify and use exactly one piece of relevant information from the text to answer a question.
  *Key Challenge*: Pinpointing the specific sentence or fact that directly yields the correct answer.
- **Task 2: Two Supporting Facts**
  *Goal*: Answer questions using two pieces of interconnected information.
  *Key Challenge*: Linking separate facts and understanding how they combine to produce the correct answer.
- **Task 3: Three Supporting Facts**
  *Goal*: Extend the reasoning chain to three distinct pieces of information.
  *Key Challenge*: Maintaining accuracy over longer inference chains and managing multiple pieces of related text.
- **Task 4: Two Argument Relations**
  *Goal*: Understand relationships involving two entities (arguments) to answer questions.
  *Key Challenge*: Correctly interpreting and manipulating relational information (e.g., who gave what to whom) with two entities.
- **Task 5: Three Argument Relations**
  *Goal*: Similar to Task 4 but introduces a third entity in the relationship.
  *Key Challenge*: Tracking more complex interactions among three entities while maintaining clarity and correctness.
- **Task 6: Yes/No Questions**
  *Goal*: Provide binary (yes/no) answers based on the facts.
  *Key Challenge*: Determining whether sufficient evidence exists in the text to affirm or deny the query.
- **Task 7: Counting**
  *Goal*: Count the number of times or entities that meet certain conditions.
  *Key Challenge*: Performing numerical reasoning and accurately tracking quantities within the text.
- **Task 8: Lists/Sets**
  *Goal*: Gather all items satisfying specific criteria into a list or set.
  *Key Challenge*: Aggregating multiple elements from different parts of the text into a cohesive list/set.
- **Task 9: Simple Negation**
  *Goal*: Handle statements containing negation.
  *Key Challenge*: Understanding how negative statements (e.g., "John did not pick up the apple") alter the truth value and impact the answer.
- **Task 10: Indefinite Knowledge**
  *Goal*: Work with statements that contain incomplete or uncertain information.
  *Key Challenge*: Managing and expressing knowledge not explicitly stated (e.g., "Someone picked up the apple, but we don't know who").

## B BABILONG BENCHMARK RESULTS

In Table 3, we present the whole expeirments of compared models on BABILong benchmark.

Table 3: Detailed performance of BABILong benchmark

| model | qa1 | qa2 | qa3 | qa4 | qa5 | qa6 | qa7 | qa8 | qa9 | qa10 |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | 0K | | | | | | |
| Llama-3.2-1.2B | 54.0 | 25.0 | 29.0 | 62.0 | 59.0 | 49.0 | 14.0 | 52.0 | 41.0 | 22.0 |
| Llama-3.2-3.2B | 62.0 | 37.0 | 29.0 | 64.0 | 82.0 | 53.0 | 25.0 | 53.0 | 65.0 | 56.0 |
| vanilla-Llama-1.7B | 86.0 | 57.0 | 46.0 | 59.0 | 85.0 | 83.0 | 95.0 | 79.0 | 83.0 | 77.0 |
| RMT-1.7B | 85.0 | 49.0 | 49.0 | 81.0 | 95.0 | 84.0 | 82.0 | 78.0 | 85.0 | 76.0 |
| LM2-1.7B | 99.0 | 89.0 | 70.0 | 88.0 | 98.0 | 95.0 | 96.0 | 97.0 | 99.0 | 94.0 |
| | | | | 1K | | | | | | |
| Llama-3.2-1.2B | 48.0 | 22.0 | 24.0 | 55.0 | 69.0 | 49.0 | 9.0 | 31.0 | 55.0 | 33.0 |
| Llama-3.2-1.2B-RAG | 51.0 | 14.0 | 19.0 | 59.0 | 80.0 | 49.0 | 10.0 | 38.0 | 40.0 | 46.0 |
| vanilla-Llama-1.7B | 31.0 | 21.0 | 44.0 | 43.0 | 71.0 | 60.0 | 71.0 | 40.0 | 67.0 | 58.0 |
| RMT-1.7B | 35.0 | 26.0 | 29.0 | 33.0 | 61.0 | 50.0 | 83.0 | 41.0 | 68.0 | 53.0 |
| LM2-1.7B | 85.0 | 59.0 | 72.0 | 68.0 | 91.0 | 84.0 | 96.0 | 69.0 | 82.0 | 77.0 |
| | | | | 2K | | | | | | |
| Llama-3.2-1.2B | 44.0 | 18.0 | 19.0 | 50.0 | 64.0 | 52.0 | 18.0 | 24.0 | 55.0 | 42.0 |
| Llama-3.2-1.2B-RAG | 52.0 | 11.0 | 12.0 | 49.0 | 75.0 | 48.0 | 5.0 | 33.0 | 50.0 | 43.0 |
| LM2-1.7B | 58.0 | 43.0 | 64.0 | 43.0 | 87.0 | 73.0 | 93.0 | 53.0 | 75.0 | 69.0 |
| RMT-1.7B | 44.0 | 21.0 | 43.0 | 41.0 | 79.0 | 47.0 | 78.0 | 41.0 | 69.0 | 51.0 |
| vanilla-Llama-1.7B | 25.0 | 22.0 | 37.0 | 34.0 | 58.0 | 60.0 | 65.0 | 38.0 | 66.0 | 58.0 |
| | | | | 4K | | | | | | |
| Llama-3.2-1.2B | 37.0 | 16.0 | 25.0 | 56.0 | 56.0 | 50.0 | 14.0 | 27.0 | 55.0 | 32.0 |
| Llama-3.2-1.2B-RAG | 47.0 | 3.0 | 16.0 | 58.0 | 68.0 | 58.0 | 3.0 | 36.0 | 45.0 | 39.0 |
| LM2-1.7B | 46.0 | 37.0 | 48.0 | 34.0 | 78.0 | 66.0 | 93.0 | 45.0 | 62.0 | 50.0 |
| RMT-1.7B | 24.0 | 20.0 | 22.0 | 24.0 | 28.0 | 46.0 | 75.0 | 35.0 | 65.0 | 45.0 |
| vanilla-Llama-1.7B | 21.0 | 18.0 | 38.0 | 28.0 | 55.0 | 61.0 | 64.0 | 35.0 | 49.0 | 53.0 |
| | | | | 8K | | | | | | |
| Llama-3.2-1.2B | 26.0 | 11.0 | 24.0 | 40.0 | 52.0 | 44.0 | 25.0 | 19.0 | 44.0 | 40.0 |
| Llama-3.2-1.2B-RAG | 36.0 | 1.0 | 5.0 | 57.0 | 72.0 | 49.0 | 8.0 | 28.0 | 44.0 | 35.0 |
| LM2-1.7B | 34.0 | 12.0 | 31.0 | 26.0 | 63.0 | 53.0 | 95.0 | 40.0 | 57.0 | 49.0 |
| RMT-1.7B | 14.0 | 15.0 | 25.0 | 28.0 | 25.0 | 47.0 | 74.0 | 38.0 | 65.0 | 46.0 |
| vanilla-Llama-1.7B | 17.0 | 19.0 | 26.0 | 20.0 | 41.0 | 51.0 | 60.0 | 37.0 | 42.0 | 45.0 |
| | | | | 16K | | | | | | |
| Llama-3.2-1.2B | 24.0 | 6.0 | 19.0 | 33.0 | 46.0 | 55.0 | 20.0 | 13.0 | 47.0 | 48.0 |
| Llama-3.2-1.2B-RAG | 26.0 | 2.0 | 9.0 | 59.0 | 76.0 | 45.0 | 5.0 | 29.0 | 52.0 | 36.0 |
| LM2-1.7B | 23.0 | 17.0 | 28.0 | 28.0 | 39.0 | 44.0 | 93.0 | 38.0 | 48.0 | 42.0 |
| RMT-1.7B | 23.0 | 9.0 | 18.0 | 23.0 | 19.0 | 47.0 | 75.0 | 33.0 | 62.0 | 42.0 |
| vanilla-Llama-1.7B | 10.0 | 11.0 | 21.0 | 11.0 | 37.0 | 59.0 | 61.0 | 34.0 | 46.0 | 46.0 |
| | | | | 32K | | | | | | |
| Llama-3.2-1.2B | 15.0 | 7.0 | 15.0 | 24.0 | 46.0 | 54.0 | 23.0 | 13.0 | 53.0 | 46.0 |
| Llama-3.2-1.2B-RAG | 28.0 | 1.0 | 2.0 | 51.0 | 74.0 | 51.0 | 2.0 | 19.0 | 41.0 | 32.0 |
| LM2-1.7B | 19.0 | 13.0 | 20.0 | 23.0 | 31.0 | 50.0 | 92.0 | 35.0 | 59.0 | 39.0 |
| RMT-1.7B | 12.0 | 16.0 | 20.0 | 18.0 | 22.0 | 46.0 | 74.0 | 34.0 | 62.0 | 43.0 |
| vanilla-Llama-1.7B | 10.0 | 17.0 | 24.0 | 13.0 | 30.0 | 54.0 | 71.0 | 33.0 | 39.0 | 53.0 |
| | | | | 64K | | | | | | |
| Llama-3.2-1.2B | 11.0 | 8.0 | 13.0 | 12.0 | 42.0 | 43.0 | 17.0 | 6.0 | 48.0 | 33.0 |
| Llama-3.2-1.2B-RAG | 27.0 | 0.0 | 4.0 | 56.0 | 66.0 | 54.0 | 4.0 | 15.0 | 48.0 | 44.0 |
| LM2-1.7B | 19.0 | 18.0 | 19.0 | 19.0 | 22.0 | 42.0 | 91.0 | 35.0 | 51.0 | 38.0 |
| RMT-1.7B | 21.0 | 18.0 | 19.0 | 21.0 | 15.0 | 48.0 | 70.0 | 33.0 | 61.0 | 41.0 |
| vanilla-Llama-1.7B | 8.0 | 13.0 | 14.0 | 14.0 | 16.0 | 52.0 | 60.0 | 30.0 | 41.0 | 41.0 |
| | | | | 128K | | | | | | |
| Llama-3.2-1.2B-RAG | 17.0 | 0.0 | 3.0 | 51.0 | 73.0 | 49.0 | 5.0 | 11.0 | 46.0 | 39.0 |
| LM2-1.7B | 15.0 | 16.0 | 12.0 | 19.0 | 23.0 | 48.0 | 91.0 | 34.0 | 54.0 | 38.0 |
| RMT-1.7B | 17.0 | 13.0 | 20.0 | 21.0 | 18.0 | 47.0 | 72.0 | 35.0 | 64.0 | 42.0 |
| vanilla-Llama-1.7B | 7.0 | 14.0 | 19.0 | 12.0 | 13.0 | 52.0 | 63.0 | 28.0 | 46.0 | 42.0 |