Say as It Is: Verbatim Fidelity Evaluation of Long-Context Language Model

Kyu Won Kim[*1] **Suhwan Choi**[*2 3] **Myeongho Jeon**[4]

Abstract

Accurately processing long texts and generating precise responses remains a significant challenge for large language models (LLMs). While existing benchmarks evaluate long-text comprehension, they often overlook the models' ability to faithfully preserve the exact wording, formatting, and sequence of prompts in their responses. To address this gap, we propose a novel evaluation framework with two key advantages: (i) adaptability across diverse domains and data sources, and (ii) tunable difficulty through dynamic variation of text length. Across three tasks—mathematical, contextual, and semantic reasoning—we find that even state-of-the-art long-context LLMs exhibit notable difficulty in maintaining verbatim fidelity during long-text generation.

1. Introduction

Recent advancements in large language models (LLMs) have dramatically expanded context windows, i.e., long-context language models (LCLMs). For instance, OpenAI's o1 (Jaech et al., 2024) and o3-mini (OpenAI, 2025) handle up to 200K input tokens. This expanded capacity enables these models to process vast amounts of raw data across domains. However, even with explicit instructions to preserve text verbatim, LLMs often exhibit omissions, oversimplifications, or hallucinations when managing large volumes of information (Liu et al., 2023; Huang et al., 2023). This is nontrivial, as even minor omissions or distortions in certain domains such as law, medicine, and regulatory compliance can have serious consequences.

Prior studies have examined the capabilities of LCLMs in leveraging extended input contexts across various tasks. These[1] include maintaining performance in summarization and multi-hop reasoning (Kuratov et al., 2024; Zhang et al., 2024b; Bai et al., 2023), retrieving target information amid irrelevant contexts via Needle-in-a-Haystack tasks (Agrawal et al., 2024; Li et al., 2024), and measuring token-level copy accuracy of long input texts (Liu et al., 2024c). However, to our knowledge, no prior work focused on LCLMs' ability to retain the exact wording, formatting, and word order of long-context inputs while reasoning—a capability we term verbatim fidelity, which remains largely unexplored.

To further investigate the verbatim fidelity of recent LCLMs, we present VERBATIMEVAL, a proxy evaluation framework for assessing the long-text generation capabilities of LLMs. VERBATIMEVAL encompasses mathematical, contextual, and semantic reasoning tasks, implemented through numeric sorting, arranging shuffled sentences, and entity grouping. Additionally, we introduce tailored evaluation metrics for each task to assess both overall task performance and the precise reproduction of details. Figure 1 illustrates how VERBATIMEVAL evaluates a model's ability to memorize the context and specific words from the prompt and leverage them when generating responses. A key advantage of our proposed evaluation framework is its flexibility across *domains* and *difficulty levels*. For the shuffled sentence arrangement task, no annotations are required, making it easily applicable to any domain. Moreover, the difficulty can be adjusted by varying the retention demands.

Finding. We evaluate state-of-the-art LCLMs, including o1, o3-mini, gpt-4o (Hurst et al., 2024), gpt-4o-mini (OpenAI, 2024), Gemini 2.0 Flash (Google DeepMind, 2025), and Gemini 1.5 Pro (Team et al., 2024a), using VERBATIMEVAL. Our results show that while these models demonstrate strong memorization capabilities and effectively recall lengthy texts, they still exhibit omissions and hallucinations in tasks requiring both comprehension and precise retention. Specifically, compared to short-text inputs, performance declines by 40% in numeric sorting, 48% in arranging shuffled sentences, and 37% in entity grouping when handling longer inputs. This gap underscores that despite recent increases in context-window sizes, maintaining verbatim retention in long-text generation remains an open challenge.

[*]Equal contribution [1]Yonsei University, Seoul, South Korea [2]Seoul National University, Seoul, South Korea [3]CRABs, Seoul, South Korea [4]École Polytechnique Fédérale de Lausanne, Lausanne, Switzerland. Correspondence to: Myeongho Jeon <myeongho.jeon@epfl.ch>.

Proceedings of the 2nd Workshop on Long-Context Foundation Models, Vancouver, Canada. 2025. *Copyright 2025 by the author(s).*

[1]For a more detailed review of related work, see Appendix B.

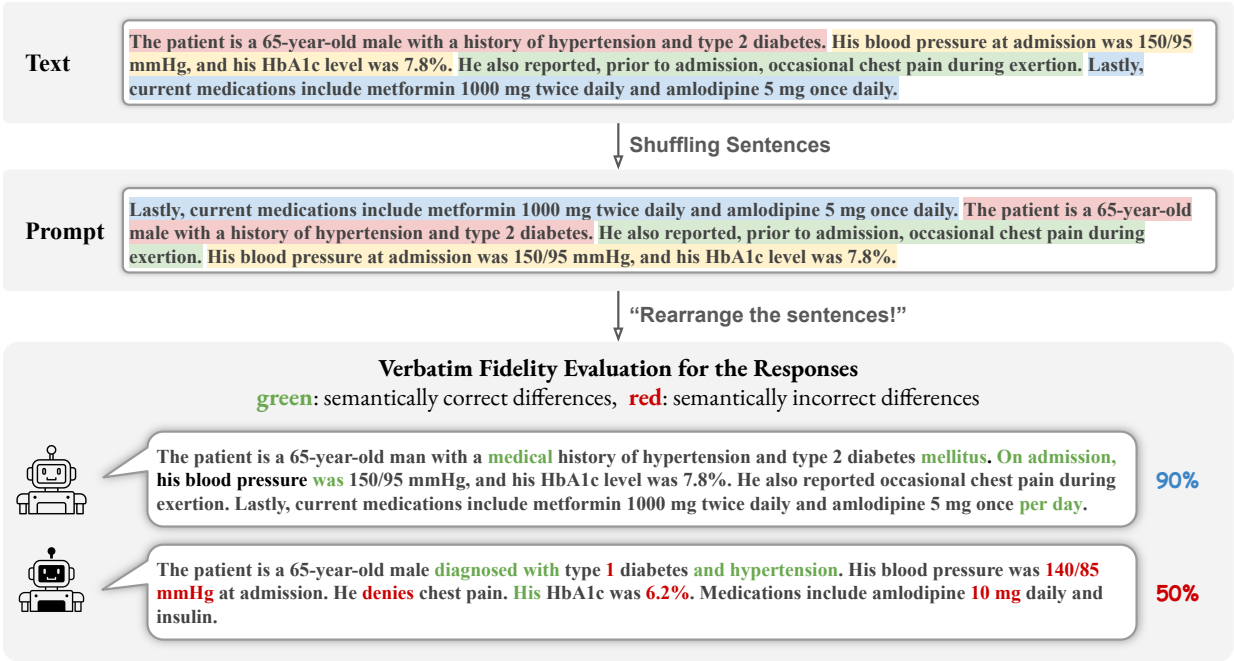

Figure 1. **An example of VerbatimEval.** While rearranging the sentences, the second response alters key medical values, reflecting poor retention of the prompt information. We impose a penalty on tokens that differ from the original ones, and an additional penalty is applied if the generated sentences are also semantically different from the original.

2. VERBATIMEVAL

The core idea behind VERBATIMEVAL is that as LLMs continue to evolve, it becomes increasingly important for them not only to handle extended contexts but also to accurately retain prompt details and reflect them in their responses. For this, we first define *verbatim fidelity*.

Definition 2.1. Verbatim fidelity measures how accurately a model, even while reasoning, preserves the exact wording, formatting, and order of the original text at the word level, ensuring precise reproduction of token sequences, numerical values, and named entities.

With this definition, to systematically assess this critical yet underexplored capability, VERBATIMEVAL introduces three tasks that collectively measure key skills essential for robust long-context generation: *Numeric Sorting*, *Sentence Arrangement*, and *Entity Grouping*. We specifically design the evaluation metrics for each task to prioritize both task performance and the accurate reproduction of details, providing a comprehensive assessment of each model's ability to maintain verbatim fidelity under challenging conditions. Diverse datasets can be utilized for sentence arrangement, covering a broad range of domains. Each task includes adjustable difficulty levels, e.g., increasing the number of items to recall, ensuring the benchmark remains relevant as LCLMs advance.

2.1. Numeric sorting

Models are given a list of large numbers and tasked with rearranging them in ascending or descending order. This task evaluates both the arithmetic comparison of numeric values and the model's ability to retain verbatim numeric information as the number of items and the magnitude of the values increase.

Method. We sample N integers, $x_1, x_2, \ldots, x_N$, from a uniform range between A and B where $A < B$. In this work, we set $A = 10^8$ and $B = 10^9$. Then, the model outputs the numbers in either ascending or descending sequence. Task difficulty can be scaled by increasing N and by using larger numbers. The more digit the number has, the more tokens it contains, which increases the in-context memory load.

Metric. We measure *verbatim fidelity* by computing Levenshtein similarity between output texts and ground-truth texts[2]:

$$\text{lev}_{sim}(a, b) := \frac{(|a| + |b|) - \text{lev}(a, b)}{|a| + |b|}, \quad (1)$$

The Levenshtein distance (Levenshtein, 1965) $\text{lev}(a, b)$ signifies the minimum number of edits (insertions, deletions, substitutions) required to transform a into b. The detailed formula is provided in Appendix A.

[2]For implementation, we represent both the output list of numbers and the ground-truth list as strings in comparison.

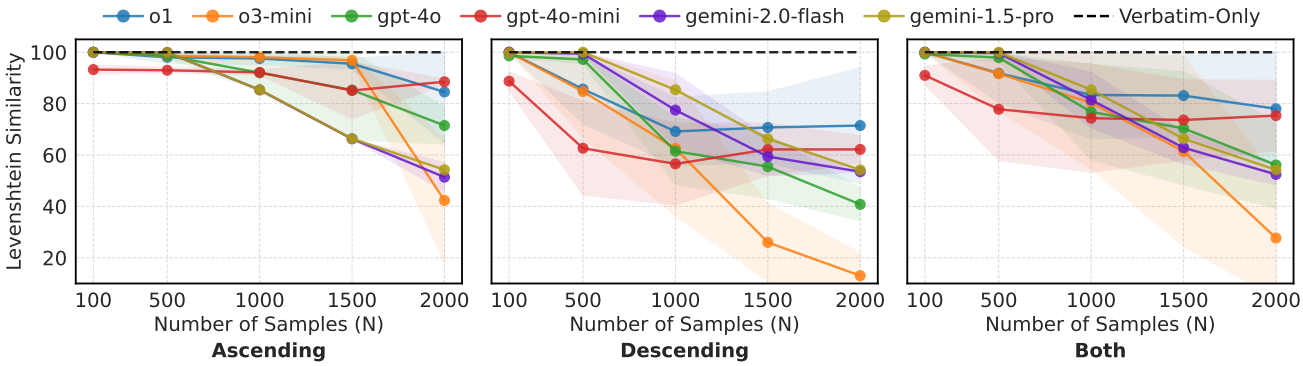

Figure 2. **Numeric sorting results.** The "Verbatim-Only" baseline represents the performance when LLMs are instructed to simply repeat the input without sorting, providing an upper bound on their verbatim retention of information. We compute the standard deviation over five runs for ascending and descending orders with varied sampling seeds. The detailed results are provided in Section G.

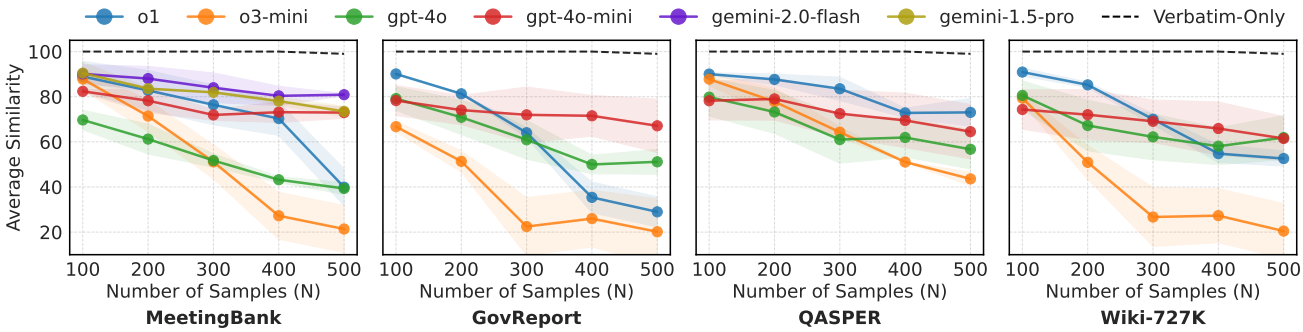

Figure 3. **Arranging shuffled sentences results.** Average similarity represents the mean of semantic similarity and Levenshtein similarity. The shaded area spans the minimum and maximum of the two similarity measures. The Gemini models generated responses only for the MeetingBank dataset due to Gemini's RECITATION flag. The results for each similarity measure are presented in Section G.

2.2. Arranging Shuffled Sentences

In this task, the model receives a paragraph with shuffled sentences and aims to restore the original order while preserving both semantic coherence and exact wording. Each passage is a self-contained text, allowing reconstruction without external context. By evaluating reconstruction across diverse domains, we assess how well models maintain sentence integrity and semantic flow across varying vocabulary, content, and discourse styles.

Method. We randomly select passages from the dataset from each domain, ensuring their length falls between N and $N + 100$ sentences. Each selected passage is then shuffled at the sentence level, and the model is tasked with reconstructing the original order. The difficulty increases as N grows, making the passages longer and more challenging to restore.

Metric. We evaluate the output on two fronts: *verbatim fidelity* and *semantic fidelity*. For *verbatim fidelity*, we first compute the Levenshtein similarity (Eq. 1) between each ground-truth sentence and every generated sentence. Specifically, let $T = \{t_n\}_{n=1}^{N}$ be the set of ground-truth sentences, and $Q = \{q_m\}_{m=1}^{M}$ be the set of generated sentences. For each $t_n \in T$, we calculate the Levenshtein similarity lev(t_n,q_m) for all $q_m \in Q$ and take the maximum value as the matching score for t_n. The overall verbatim fidelity, $V(T, Q)$, is then defined as the average of these maximum scores across all ground-truth sentences:

$$V(T, Q) := \frac{1}{N} \sum_{n=1}^{N} \max_{1 \leq m \leq M} [\text{lev}(t_n, q_m)]. \quad (2)$$

By comparing one ground-truth sentence against all generated sentences and taking the highest Levenshtein similarity, it accommodates scenarios where the generated text may be out of order or duplicated. Averaging these best-match scores over all ground-truth sentences then gives an overall measure of how faithfully the model reproduced the original text, sentence by sentence. For *semantic fidelity*, we compute cosine similarity between the generated passage and the original passage. This captures whether the model has reconstructed a passage that remains semantically equivalent to the original.

Remark 2.2. Numeric Sorting and Sentence Arrangement require no annotations. Sentence Arrangement is especially useful, as it can be applied across domains by simply shuffling sentences within a large corpus.

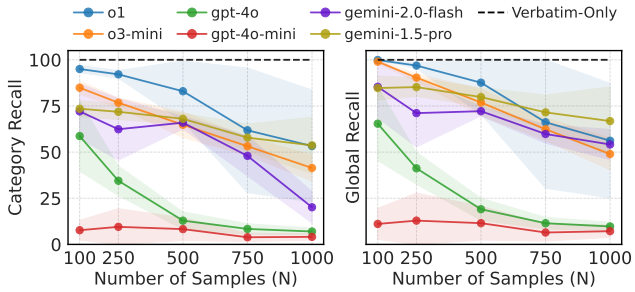

Figure 4. **Entity grouping results.** The standard deviation is computed over 10 runs with different sampling seeds. The detailed results are provided in Appendix G.

2.3. Entity Grouping

The *entity grouping* task challenges the model to classify a set of entities into predefined categories without altering or omitting the entity names. This requires both precise verbatim retention of each entity and an understanding of its semantic category.

Method. We randomly sample N entities without replacement from a set of X entities with $N < X$, ensuring that at least one entity per category is selected. The model is tasked with grouping these entities into K categories without altering their original text forms. The difficulty can be increased by scaling N to include more entities.

Metric. For *task performance* evaluation, we calculate recall for each category and average these scores across all K categories:

$$\textbf{Recall}_{\text{category}}(M, G) := \sum_{k=1}^{K} \frac{|M_k \cap G_k|}{|G_k|}, \tag{3}$$

where G_k represent the set of ground-truth entities in a category k, and M_k denote the set of entities the model assigns to the category k. This approach offers a comprehensive assessment of the model's performance, accounting for the variations in grouping accuracy specific to each category. For *verbatim fidelity* evaluation, we measure recall at a global level, comparing the set of all predicted entities ($M^K = \bigcup_{k=1}^{K} M_k$) and the set of all ground-truth entities ($G^K = \bigcup_{k=1}^{K} G_k$). This metric quantifies the model's ability to retain the full set of entities from the original input, reflecting the extent of both omissions and incorrect inclusions.

3. Experiments

3.1. Experimental Setup

Baseline. We evaluate state-of-the-art LLMs, gpt-4o (Hurst et al., 2024), o1 (Jaech et al., 2024), o3-mini (OpenAI, 2025), gpt-4o-mini (OpenAI, 2024), Gemini 2.0 Flash (Google DeepMind, 2025), and Gemini 1.5 Pro (Team et al., 2024a) using VERBATIMEVAL. For each task, We also include a *Verbatim-Only Baseline*, where the model is instructed to replicate the input text without performing any concurrent task. First, we identify the lowest-performing model at the largest tested N, and then track its verbatim retention score across various N. This baseline helps isolate the model's capacity for faithful reproduction of input text from any additional reasoning requirements.

Dataset. *Entity grouping*: DBpedia Ontology (Zhang et al., 2015). *Arranging shuffled sentences*: MeetingBank(Hu et al., 2023), GovReport(Cao & Wang, 2022), QASPER(Dasigi et al., 2021), and Wiki-727K. (Koshorek et al., 2018). Please refer to Appendix C.1 for more details.

Implementation detail. The additional implementation details and prompts employed in these tasks are outlined in Appendix C.2 and E, respectively.

3.2. Results and Discussion

Remark 3.1. LCLMs can reliably memorize and reproduce long inputs (`Verbatim-Only` in Figure 2, 3, and 4), but falter on tasks that require reasoning or comprehension — i.e., the typical way LCLMs are used.

Numeric sorting. Models maintain high fidelity to the sorted list for small N, but Levenshtein similarity drops by 40% as N increases (Figure 2). Notably, descending order proves more difficult than ascending, and among small models, o3-mini experiences greater performance degradation than gpt-4o-mini. Performance across different upper bounds of sample numbers, B, is illustrated in Figure 5.

Arranging shuffled sentences. As N increases, all models show declines in both semantic and Levenshtein similarity, averaging a 48% reduction (Figure 3). gpt-4o-mini performs unexpectedly well, while o3-mini struggles with semantic similarity despite comparable Levenshtein scores, indicating its performance degradation under higher retention demands.

Entity grouping. Increasing N raises task difficulty, reducing average recall by 37% (Figure 4). Unlike previous tasks, GPT-4o models underperform significantly in this task compared to OpenAI's other models and Gemini models.

4. Concluding Remark

We introduce VERBATIMEVAL, an evaluation framework for measuring the verbatim fidelity of LCLMs. Applying it across diverse domains and difficulty levels reveals that even state-of-the-art LCLMs struggle to maintain verbatim accuracy, a limitation that may pose challenges in domains requiring precise reproduction of input. Future work could introduce a metric that better captures numeric or textual item order and refine the evaluation of semantic accuracy in generated texts for the sentence-arrangement task.

Code Availability

The code for the experiments is available at: https://github.com/schoi828/VerbatimEval.

Impact Statement

This paper introduces a benchmark for evaluating verbatim fidelity in long-text generation. We believe it serves as a practical and versatile proxy applicable across various domains. While our work may have societal implications, we do not identify any that require specific emphasis at this time.

Acknowledgment

This work was supported by 1) the Starting growth Technological R&D Program (RS-2024-00506994) funded by the Ministry of SMEs and Startups(MSS, Korea) and 2) Culture, Sports and Tourism R&D Program (RS-2024-00399433) through the Korea Creative Content Agency grant funded by the Ministry of Culture, Sports and Tourism in 2024.

References

Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F. L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Agrawal, A., Dang, A., Nezhad, S. B., Pokharel, R., and Scheinberg, R. Evaluating multilingual long-context models for retrieval and reasoning. *arXiv preprint arXiv:2409.18006*, 2024.

An, C., Gong, S., Zhong, M., Li, M., Zhang, J., Kong, L., and Qiu, X. L-eval: Instituting standardized evaluation for long context language models. *ArXiv*, abs/2307.11088, 2023.

Anthropic. The claude 3 model family: Opus, sonnet, haiku, March 2024. URL https://docs.anthropic.com/en/resources/claude-3-model-card.

Bai, Y., Lv, X., Zhang, J., Lyu, H., Tang, J., Huang, Z., Du, Z., Liu, X., Zeng, A., Hou, L., et al. Longbench: A bilingual, multitask benchmark for long context understanding. *arXiv preprint arXiv:2308.14508*, 2023.

Cao, S. and Wang, L. Hibrids: Attention with hierarchical biases for structure-aware long document summarization, 2022.

Cohere. Command r and command r+ model card, October 2024. URL https://docs.cohere.com/docs/responsible-use.

Dasigi, P., Lo, K., Beltagy, I., Cohan, A., Smith, N. A., and Gardner, M. A dataset of information-seeking questions and answers anchored in research papers. *arXiv preprint arXiv:2105.03011*, 2021.

Google DeepMind. Gemini flash, 2025. URL https://ai.google.dev/gemini-api/docs/models?hl=ko#gemini-2.0-flash.

Hsieh, C.-P., Sun, S., Kriman, S., Acharya, S., Rekesh, D., Jia, F., and Ginsburg, B. Ruler: What's the real context size of your long-context language models? *ArXiv*, abs/2404.06654, 2024.

Hu, Y., Ganter, T., Deilamsalehy, H., Dernoncourt, F., Foroosh, H., and Liu, F. Meetingbank: A benchmark dataset for meeting summarization. *arXiv preprint arXiv:2305.17529*, 2023.

Huang, L., Yu, W., Ma, W., Zhong, W., Feng, Z., Wang, H., Chen, Q., Peng, W., Feng, X., Qin, B., and Liu, T. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *ArXiv*, abs/2311.05232, 2023.

Hurst, A., Lerer, A., Goucher, A. P., Perelman, A., Ramesh, A., Clark, A., Ostrow, A., Welihinda, A., Hayes, A., Radford, A., et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.

Jaech, A., Kalai, A., Lerer, A., Richardson, A., El-Kishky, A., Low, A., Helyar, A., Madry, A., Beutel, A., Carney, A., et al. Openai o1 system card. *arXiv preprint arXiv:2412.16720*, 2024.

Kamradt, G. Llmtest_needleinahaystack: A simple "needle in a haystack" analysis to test in-context retrieval ability of long context llms. https://github.com/gkamradt/LLMTest_NeedleInAHaystack, 2024.

Karpinska, M., Thai, K., Lo, K., Goyal, T., and Iyyer, M. One thousand and one pairs: A "novel" challenge for long-context language models. *ArXiv*, abs/2406.16264, 2024.

Koshorek, O., Cohen, A., Mor, N., Rotman, M., and Berant, J. Text segmentation as a supervised learning task. *ArXiv*, abs/1803.09337, 2018.

Kuratov, Y., Bulatov, A., Anokhin, P., Rodkin, I., Sorokin, D., Sorokin, A., and Burtsev, M. Babilong: Testing the limits of llms with long context reasoning-in-a-haystack. *ArXiv*, abs/2406.10149, 2024.

Levenshtein, V. I. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet physics. Doklady*, 10:707–710, 1965.

Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Kuttler, H., Lewis, M., tau Yih, W., Rocktäschel, T., Riedel, S., and Kiela, D. Retrieval-augmented generation for knowledge-intensive nlp tasks. *ArXiv*, abs/2005.11401, 2020.

Li, J., Wang, M., Zheng, Z., and Zhang, M. Loogle: Can long-context language models understand long contexts? *arXiv preprint arXiv:2311.04939*, 2023.

Li, M., Zhang, S., Liu, Y., and Chen, K. Needlebench: Can llms do retrieval and reasoning in 1 million context window? *URL https://arxiv. org/abs/2407*, 11963, 2024.

Liu, J., Tian, J. L., Daita, V., Wei, Y., Ding, Y., Wang, Y. K., Yang, J., and Zhang, L. Repoqa: Evaluating long context code understanding. *ArXiv*, abs/2406.06025, 2024a.

Liu, N. F., Lin, K., Hewitt, J., Paranjape, A., Bevilacqua, M., Petroni, F., and Liang, P. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173, 2023.

Liu, X., Dong, P., Hu, X., and Chu, X. Longgenbench: Long-context generation benchmark. *ArXiv*, abs/2410.04199, 2024b.

Liu, X., Zhao, R., Huang, P., Xiao, C., Li, B., Wang, J., Xiao, T., and Zhu, J. Forgetting curve: A reliable method for evaluating memorization capability for long-context models. *arXiv preprint arXiv:2410.04727*, 2024c.

OpenAI. Gpt-4o mini: Advancing cost-efficient intelligence. https://url.kr/y85fps, May 2024.

OpenAI. New embedding models and api updates, January 2024. URL https://openai.com/index/new-embedding-models-and-api-updates/.

OpenAI. tiktoken: A fast bpe tokenizer for use with openai's models, 2024. URL https://github.com/openai/tiktoken.

OpenAI. Openai o3-mini system card, January 2025. URL https://url.kr/nvcqnj.

Su, J., Lu, Y., Pan, S., Wen, B., and Liu, Y. Roformer: Enhanced transformer with rotary position embedding. *ArXiv*, abs/2104.09864, 2021.

Team, G., Georgiev, P., Lei, V. I., Burnell, R., Bai, L., Gulati, A., Tanzer, G., Vincent, D., Pan, Z., Wang, S., et al. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*, 2024a.

Team, J., Lenz, B., Arazi, A., Bergman, A., Manevich, A., Peleg, B., Aviram, B., Almagor, C., Fridman, C., Padnos, D., et al. Jamba-1.5: Hybrid transformer-mamba models at scale. *arXiv preprint arXiv:2408.12570*, 2024b.

Tworkowski, S., Staniszewski, K., Pacek, M., Wu, Y., Michalewski, H., and Milo's, P. Focused transformer: Contrastive training for context scaling. *ArXiv*, abs/2307.03170, 2023.

Vodrahalli, K., Ontanon, S., Tripuraneni, N., Xu, K., Jain, S., Shivanna, R., Hui, J., Dikkala, N., Kazemi, M., Fatemi, B., et al. Michelangelo: Long context evaluations beyond haystacks via latent structure queries. *arXiv preprint arXiv:2409.12640*, 2024.

Wang, M., Chen, L., Fu, C., Liao, S., Zhang, X., Wu, B., Yu, H., Xu, N., Zhang, L., Luo, R., et al. Leave no document behind: Benchmarking long-context llms with extended multi-doc qa. *arXiv preprint arXiv:2406.17419*, 2024.

Xiong, W., Liu, J., Molybog, I., Zhang, H., Bhargava, P., Hou, R., Martin, L., Rungta, R., Sankararaman, K. A., Oğuz, B., Khabsa, M., Fang, H., Mehdad, Y., Narang, S., Malik, K., Fan, A., Bhosale, S., Edunov, S., Lewis, M., Wang, S., and Ma, H. Effective long-context scaling of foundation models. In *North American Chapter of the Association for Computational Linguistics*, 2023.

Yen, H., Gao, T., Hou, M., Ding, K., Fleischer, D., Izsak, P., Wasserblat, M., and Chen, D. Helmet: How to evaluate long-context language models effectively and thoroughly. *ArXiv*, abs/2410.02694, 2024.

Zhang, J., Bai, Y., Lv, X., Gu, W., Liu, D., Zou, M., Cao, S., Hou, L., Dong, Y., Feng, L., and Li, J. Longcite: Enabling llms to generate fine-grained citations in long-context qa. *ArXiv*, abs/2409.02897, 2024a.

Zhang, X., Zhao, J., and LeCun, Y. Character-level convolutional networks for text classification. In Cortes, C., Lawrence, N., Lee, D., Sugiyama, M., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015.

Zhang, X., Chen, Y., Hu, S., Xu, Z., Chen, J., Hao, M. K., Han, X., Thai, Z. L., Wang, S., Liu, Z., et al. "∞Bench: Extending long context evaluation beyond 100K tokens". *arXiv preprint arXiv:2402.13718*, 2024b.

A. Levenshtein distance

- If $|b| = 0$, then $\text{lev}(a, b) = |a|$.
- If $|a| = 0$, then $\text{lev}(a, b) = |b|$.
- If $\text{head}(a) = \text{head}(b)$, then:

$$\text{lev}(a, b) = \text{lev}(\text{tail}(a), \text{tail}(b))$$

- Otherwise:

$$\text{lev}(a, b) = 1 + \min \begin{array}{c} \text{lev}(\text{tail}(a), b), \\ \text{lev}(a, \text{tail}(b)), \\ \text{lev}(\text{tail}(a), \text{tail}(b)), \end{array}$$

where *head*(x) refers to the first string of x, and *tail*(x) refers to the substring consisting of all characters except the first.

B. Related Work

Benchmarks for long-context language model. A wide range of benchmarks have evaluated LLMs in long-context scenarios, mainly focusing on semantic correctness in comprehension, retention, and retrieval. These include ensuring valid QAanswers (An et al., 2023; Wang et al., 2024; Hsieh et al., 2024), high-quality summaries (Yen et al., 2024; Zhang et al., 2024b), and retrieval accuracy in Needle-in-a-Haystack-like settings (Li et al., 2023; 2024; Kamradt, 2024). Other benchmarks target more specific tasks, such as long-form generation (Liu et al., 2024b), long-context token-level copy memory (Liu et al., 2024c), mathematics-focused reasoning (Wang et al., 2024), latent structure queries (Vodrahalli et al., 2024), massive multi-hop QA (Kuratov et al., 2024), multilingual retrieval and reasoning (Agrawal et al., 2024), function search in code repositories (Liu et al., 2024a), veracity checks (Karpinska et al., 2024), and QA with fine-grained citations (Zhang et al., 2024a) explore distinct challenges. While prior benchmarks assess long-context comprehension in terms of semantic correctness and retention, they do not explicitly evaluate verbatim memorization and reproduction, leaving a critical gap in assessing exact text fidelity.

Long-context language model. Recent advancements in LLMs have significantly extended the window size for text processing, addressing limitations in handling lengthy documents. GPT-4 Turbo (Achiam et al., 2023) supports up to 128K tokens, while Claude 3 (Anthropic, 2024) extends this to 200K tokens, both optimizing retrieval and contextual coherence. Gemini 2.0 Flash (Google DeepMind, 2025) sets a new benchmark with a 1M-token context window, enabling ultra-long document comprehension. Efforts to expand the context length of open-source models continue. LLaMA 2-Long (Xiong et al., 2023) applies RoPE (Su et al., 2021) to reach 32K tokens, and LongLLaMA (Tworkowski et al., 2023)—with its Focused Transformer—scales up to 256K tokens. Additionally, Command R+ (Cohere, 2024) (128K tokens) and Jamba 1.5 (Team et al., 2024b) (256K tokens) integrate retrieval-augmented generation (RAG) (Lewis et al., 2020) for knowledge-intensive tasks.

C. Experimental Setup

C.1. Dataset

MeetingBank contains 6,892 city council English transcripts with dialogic, unstructured text. GovReport includes 19,463 government reports with dense, technical language in English. QASPER features 1,585 NLP research papers requiring specialized content handling in English. Wiki-727K comprises 582K Wikipedia passages in English, testing structured information flow. DBpedia Ontology contains 70,000 named entities across 14 categories, including Company, Artist, Athlete, Building, Natural Place, Animal, Film, and Written Work, with 5,000 entities per category. For each dataset, we merge the training, test, and validation sets, and then sample instances from this combined dataset. Table 1 reports the mean input-token count for every dataset–sample-size configuration.

C.2. Implementation Details

We use the openai-python and google-generativeai libraries to access the APIs for our chosen models, employing their default generation configurations (e.g., temperature). For the sentence-shuffling task, we employ text-embedding-3-large (OpenAI, 2024) to calculate the semantic similarity between ground-truth passages and the corresponding generated outputs. In the entity-grouping task, we initially require the models to produce a structured output consisting of a list of categories, each containing a category name and its associated entities. The exact formatting code for this structure is detailed in Appendix F. As N grows, however, some models fail to adhere to this format. In those cases, we parse their free-text responses instead. Despite explicit formatting instructions, larger inputs often lead models to disregard the requested structure. We therefore tailor our parsing strategy for each experimental instance. Notably, the o1 model fails to generate structured outputs at $N = 250, 500, 750, 1000$, while gemini-2.0-flash fails at $N = 500, 750, 1000$.

C.3. Model API Detail

Table 2 displays the specific model versions used throughout the experiments.

Table 3 illustrates the tokens used in Figure 2, Figure 3, Figure 4, and Figure 5.

Numeric Sorting					
Number of Samples	**100**	**500**	**1000**	**1500**	**2000**
-	498 ± 0	2498 ± 0	4998 ± 0	7498 ± 0	9998 ± 0
Arranging Shuffled Sentences					
Number of Samples	**100**	**200**	**300**	**400**	**500**
MeetingBank	2129.8 ± 421.3	3386.8 ± 606.0	5547.8 ± 637.8	7155.5 ± 1061.3	7976.4 ± 1369.5
GovReport	5105.7 ± 1041.0	7367.0 ± 963.7	9451.8 ± 1489.2	12521.9 ± 1688.1	16704.8 ± 1804.2
QASPER	4333.9 ± 1077.8	5968.2 ± 1065.9	9548.0 ± 1521.7	10437.3 ± 3873.5	13944.1 ± 1721.8
Wiki-727K	3472.6 ± 1081.1	7724.0 ± 1136.7	10713.1 ± 1592.9	13356.3 ± 1601.6	16523.4 ± 3014.7
Entity Grouping					
Number of Samples	**100**	**250**	**500**	**750**	**1000**
DBpedia Ontology	594.0 ± 17.4	1487.4 ± 38.3	2981.8 ± 39.3	4463.3 ± 63.4	5900.0 ± 50.1

Table 1. **Token statistics for all tasks.** Each cell signifies the mean token count (± standard deviation) computed over 10 independent subsampling runs for the corresponding "number-of-samples" setting, which varies across tasks. Token counts naturally depend on the tokenizer used; the number of tokens here were obtained with the `tiktoken` tokenizer (OpenAI, 2024) employed by the OpenAI model family.

Model	**Version / Release**
o1	o1-2024-12-17
o3-mini	o3-mini-2025-01-31
gpt-4o	gpt-4o-2024-08-06
gpt-4o-mini	gpt-4o-mini-2024-07-18
gemini-2.0-flash	gemini-2.0-flash-001
gemini-1.5-pro	gemini-1.5-pro-002

Table 2. **Model versions used in the experiments.**

	Input	**Output**	**Reasoning**
o1	3.10M	6.05M	3.64M
o3-mini	3.05M	4.38M	3.31M
gpt-4o	3.01M	1.28M	–
gpt-4o-mini	3.03M	2.05M	–
gemini-2.0-flash	4.67M	1.82M	–
gemini-1.5-pro	4.24M	1.51M	–
Average	3.52M	2.85M	3.47M
Total	21.1M	17.1M	6.95M

Table 3. **Comparison of total input, output, and reasoning tokens used by various models in the experiments.**

D. Licensing and Terms of Use

We rely on several publicly available datasets in this work and strictly comply with their respective licenses. MeetingBank is distributed under the Creative Commons Attribution Non Commercial Share Alike 4.0 license ("huuuyeah/meetingbank"). Wiki-727K, obtained from "TankNee/wiki-727k," does not explicitly specify a license; however, since the passages are derived from Wikipedia, they are subject to the Creative Commons Attribution-ShareAlike 4.0 International License (CC-BY-SA 4.0), which we assume applies. Additionally, GovReport ("ccdv/govreport-summarization") and QASPER ("allenai/qasper") are both released under the Creative Commons Attribution 4.0 license. We adhere to the terms of these licenses in our use and distribution of these datasets.

We carefully reviewed all data sources for personally identifying or offensive content. The MeetingBank dataset was compiled from publicly accessible city council meetings and evaluated—through consultation with legal experts—to ensure that it does not contain confidential or uniquely identifying information. Similarly, while Wikipedia includes detailed profiles of public figures, its content is subject to strict editorial standards and is publicly available under the CC-BY-SA 4.0 license. We rely on these established protocols to maintain ethical standards in our work.

E. Prompts

This section presents prompts used in the experiments in Section 3.2.

Verbatim-Only

Repeat the same text as provided in the original text.
Original Text: {context}

In the *Verbatim-Only* prompt, *context* represents the original text that the model must reproduce verbatim.

Numeric Sorting

Rearrange the numbers so that they are in {mode} order: {context}
Only provide the answer.

For the *Numeric Soring* prompt, *context* represents the list of numbers to be sorted, and *mode* indicates whether they should be arranged in ascending or descending order.

Arranging Shuffled Sentences

You are given sentences from a Wikipedia article. These sentences have been randomly shuffled. Your task is to restore them to their original order to form a coherent text. Follow these rules:

1: Do not omit or modify any words, punctuation, or other details from the sentences.

2: Do not add any new content or commentary.

3: Only output the reordered text, with each sentence in its correct position to recreate the original passage.

4: Do your best to restore the original order of the provided sentences.

Shuffled Sentences: {context}

In the *Shuffled Sentences* prompt, *context* is the randomly shuffled sentences that the model must reorder into their original sequence.

Entity Grouping

Below is a list of entity names. Your task is to assign each entity to exactly one of the following 14 categories:
- Company
- Educational Institution
- Artist
- Athlete
- Office Holder
- Mean of Transportation
- Building
- Natural Place
- Village
- Animal
- Plant
- Album
- Film
- Written Work

`Requirement:`

1: Each entity must be assigned to only one category.

2: No entity should be left unclassified.

3: Use the exact entity names as they appear in the input; do not modify them.

The list of entities: {context}

In the *Entity Grouping* prompt, context is the set of entities, each assigned to a single category without modifying the original text.

F. Structure Output Format

Here, we provide the structured output code for the entity grouping task in Figure 4.

```
from pydantic import BaseModel
class Category(BaseModel):
    entities:list[str]
    category˙name: str
class Grouping(BaseModel):
    categories: list[Category]
```

G. Results

Table 4, Table 5, Table 6, and Table 7 provide detailed results for Figure 2, Figure 3 (Tables 5, 6), and Figure 4, respectively. We also report semantic and Levenshtein similarity for the shuffled sentences task (Figure 6), recall for each category for the entity grouping task (Figure 7), and Levenshtein similarity across the sample upper bounds for the numeric sorting task (Figure 5).

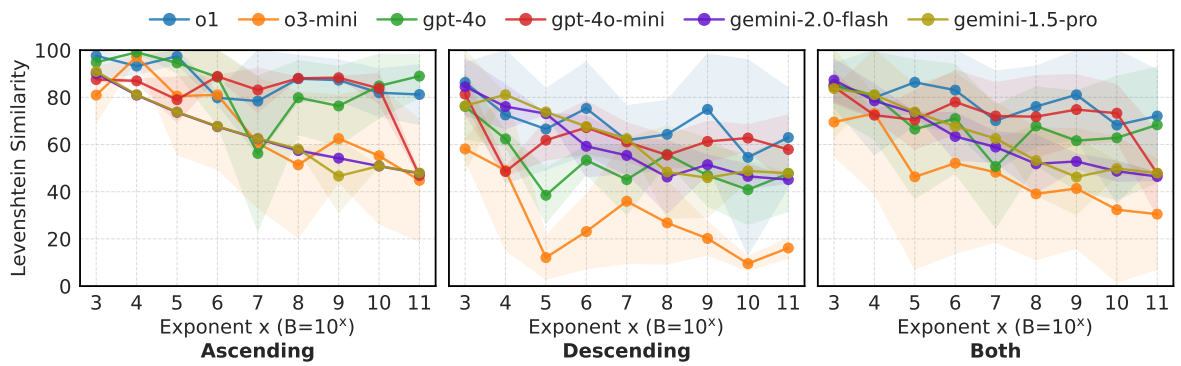

Figure 5. **Numeric sorting performance across different sample upper bounds.** The models are evaluated on $N = 2000$ integers sampled from $[10^{x-1}, 10^x]$. The x-axis represents the exponent x, with $B = 10^x$ as the sample upper bound. For all models, overall performance decreases at higher exponents.

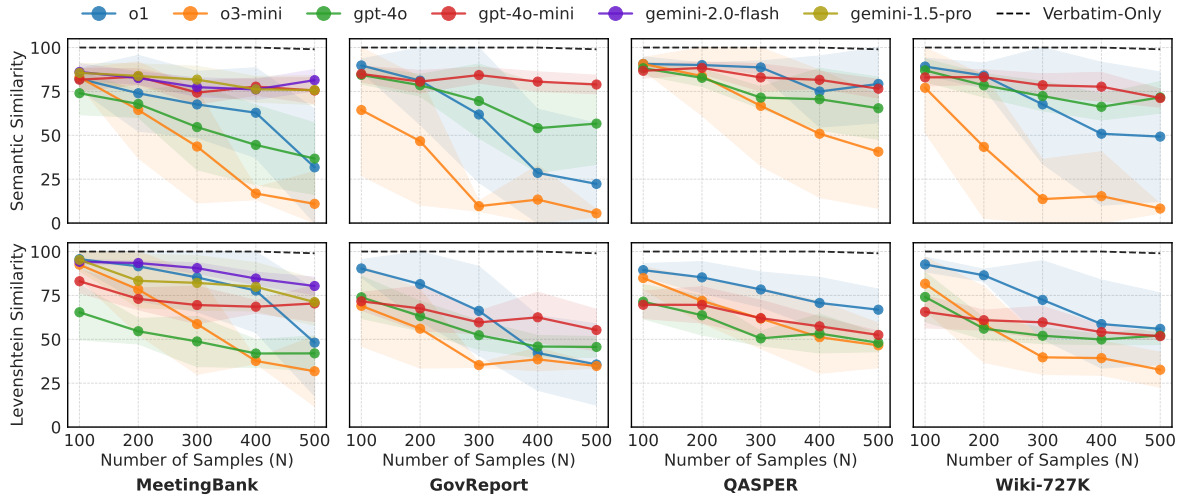

Figure 6. **Arranging shuffled sentence results.** Each column corresponds to one dataset (MeetingBank, GovReport, QASPER, Wiki-727K). The top row displays semantic similarity, indicating the semantic coherence between the ground-truth and generated text. The bottom row shows Levenshtein similarity, reflecting the verbatim fidelity of each model's output.

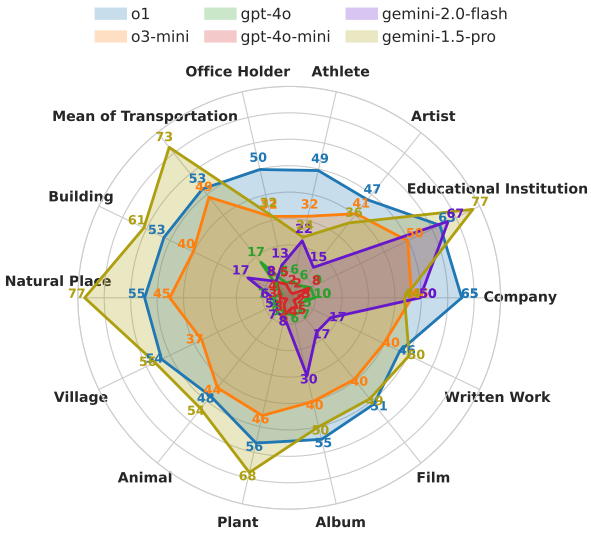

Figure 7. **Category-wise recall at N=1000**. Each axis represents one of 14 entity categories in the entity grouping task, with higher values indicating better recall for that category.

Model	100	500	1000	1500	2000
Ascending					
o1	100.00 ± 0.00	98.04 ± 2.36	97.56 ± 2.36	95.52 ± 2.05	84.54 ± 19.73
o3-mini	100.00 ± 0.00	98.59 ± 0.69	98.07 ± 0.32	96.84 ± 1.42	42.33 ± 24.05
gpt-4o	100.00 ± 0.00	98.65 ± 0.99	92.01 ± 7.19	85.30 ± 19.42	71.48 ± 7.17
gpt-4o-mini	93.25 ± 1.68	92.98 ± 1.16	92.15 ± 0.97	85.09 ± 10.99	88.52 ± 1.03
gemini-2.0-flash	99.95 ± 0.00	99.91 ± 0.10	85.34 ± 0.05	66.29 ± 0.07	51.39 ± 5.57
gemini-1.5-pro	99.95 ± 0.00	99.95 ± 0.05	85.38 ± 0.00	66.36 ± 0.00	54.26 ± 0.03
Descending					
o1	100.00 ± 0.00	85.65 ± 13.53	69.17 ± 13.17	70.73 ± 13.87	71.46 ± 22.47
o3-mini	100.00 ± 0.00	84.72 ± 17.99	62.55 ± 26.27	26 ± 15.08	13.04 ± 8.88
gpt-4o	98.57 ± 0.67	97.18 ± 1.61	61.50 ± 13.03	55.47 ± 12.32	40.79 ± 6.37
gpt-4o-mini	88.75 ± 3.59	62.63 ± 18.24	56.54 ± 15.87	62.17 ± 10.17	62.18 ± 5.31
gemini-2.0-flash	99.95 ± 0.00	99.45 ± 0.33	77.48 ± 14.21	59.40 ± 7.46	53.50 ± 1.82
gemini-1.5-pro	99.75 ± 0.40	99.99 ± 0.01	85.38 ± 0.01	66.34 ± 0.04	54.23 ± 0.03

Table 4. **Numeric sorting results.** The experimental setup follows the one depicted in Figure 2.

Model	100	200	300	400	500
MeetingBank					
o1	95.66 ± 4.80	91.60 ± 19.73	85.26 ± 19.57	77.78 ± 23.92	48.09 ± 30.30
o3-mini	92.52 ± 8.18	78.41 ± 25.75	58.63 ± 28.74	37.67 ± 19.73	31.80 ± 1.56
gpt-4o	65.41 ± 15.71	54.60 ± 7.27	48.77 ± 14.35	41.92 ± 7.98	41.98 ± 10.86
gpt-4o-mini	83.08 ± 6.67	73.03 ± 6.21	69.53 ± 10.74	68.56 ± 3.65	70.44 ± 10.41
gemini-2.0-flash	94.16 ± 1.38	93.45 ± 1.15	90.56 ± 3.09	84.63 ± 3.67	80.37 ± 5.04
gemini-1.5-pro	95.29 ± 6.39	83.33 ± 12.27	82.15 ± 15.15	79.92 ± 13.79	71.21 ± 13.32
GovReport					
o1	90.36 ± 5.08	81.45 ± 21.04	66.21 ± 25.36	42.20 ± 21.26	35.65 ± 23.19
o3-mini	69.07 ± 22.98	56.05 ± 22.45	35.29 ± 1.29	38.58 ± 6.56	34.78 ± 1.31
gpt-4o	73.93 ± 12.08	63.25 ± 8.49	52.32 ± 8.28	45.87 ± 6.37	45.65 ± 5.88
gpt-4o-mini	71.64 ± 5.12	67.58 ± 12.78	59.64 ± 6.51	62.51 ± 14.37	55.28 ± 11.79
QASPER					
o1	89.41 ± 3.77	85.29 ± 9.00	78.38 ± 10.04	70.71 ± 14.62	66.85 ± 11.8
o3-mini	84.92 ± 6.54	71.91 ± 12.68	61.80 ± 17.18	51.19 ± 20.62	46.56 ± 12.74
gpt-4o	71.43 ± 9.81	63.71 ± 11.34	50.53 ± 4.22	53.27 ± 11.07	48.02 ± 5.09
gpt-4o-mini	69.68 ± 7.83	69.61 ± 10.37	62.05 ± 10.29	57.40 ± 5.51	52.48 ± 2.18
Wiki-727K					
o1	92.67 ± 4.20	86.46 ± 3.52	72.44 ± 22.38	58.70 ± 25.11	55.95 ± 20.51
o3-mini	81.65 ± 15.26	58.45 ± 21.65	39.73 ± 9.76	39.29 ± 9.78	32.65 ± 10.04
gpt-4o	74.14 ± 14.12	56.05 ± 6.25	52.02 ± 4.53	49.89 ± 2.96	52.18 ± 5.22
gpt-4o-mini	65.66 ± 9.18	60.9 ± 5.72	59.68 ± 9.19	54.12 ± 3.92	51.78 ± 1.93

Table 5. **Levenshtein similarity results for arranging shuffled sentences task.** Each cell shows the average Levenshtein similarity (higher is better) between the model's re-ordered text and the ground-truth sentences. The experimental setup follows the one depicted in Figure 3.

Model	100	200	300	400	500
MeetingBank					
o1	82.20 ± 5.94	73.96 ± 21.6	67.59 ± 18.86	62.85 ± 25.66	31.73 ± 33.31
o3-mini	83.1 ± 4.73	64.43 ± 27.47	43.61 ± 32.19	16.83 ± 16.87	10.94 ± 3.58
gpt-4o	73.98 ± 12.26	67.86 ± 7.68	54.68 ± 24.28	44.51 ± 22.69	36.72 ± 24.85
gpt-4o-mini	81.67 ± 6.37	83.46 ± 4.54	74.31 ± 6.53	77.72 ± 4.94	75.5 ± 7.46
gemini-2.0-flash	86.08 ± 4.72	82.61 ± 3.6	77.41 ± 5.31	76.13 ± 5.75	81.41 ± 6.16
gemini-1.5-pro	85.59 ± 5.33	83.31 ± 6.93	83.73 ± 7.64	76.21 ± 7.36	75.77 ± 8.26
GovReport					
o1	89.81 ± 3.66	81.14 ± 25.02	61.85 ± 38.69	28.59 ± 36.32	22.32 ± 34.57
o3-mini	64.41 ± 37.43	46.66 ± 36.39	9.64 ± 2.97	13.37 ± 19.57	5.53 ± 2.23
gpt-4o	84.41 ± 5.24	78.52 ± 6.18	69.54 ± 21.15	54.08 ± 25.4	56.67 ± 23.23
gpt-4o-mini	84.95 ± 4.58	80.58 ± 6.38	84.30 ± 4.52	80.57 ± 5.62	78.94 ± 5.31
QASPER					
o1	90.64 ± 4.12	90.03 ± 1.25	88.67 ± 3.72	74.93 ± 20.27	79.26 ± 22.16
o3-mini	90.66 ± 3.7	83.73 ± 22.52	66.77 ± 34.19	50.9 ± 36.31	40.64 ± 32.27
gpt-4o	88.21 ± 3.8	82.75 ± 4.96	71.5 ± 6.2	70.58 ± 17.61	65.41 ± 17.76
gpt-4o-mini	86.75 ± 4.28	88.4 ± 3.44	82.96 ± 7.79	81.59 ± 4.93	76.54 ± 5.28
Wiki-727K					
o1	89.17 ± 4.8	84.04 ± 7.18	67.61 ± 33.88	50.89 ± 40.72	49.26 ± 36.89
o3-mini	77.06 ± 25.44	43.4 ± 40.75	13.66 ± 22.68	15.31 ± 25.38	8.28 ± 2.73
gpt-4o	87.06 ± 6.51	78.45 ± 6.66	72.36 ± 8.44	66.24 ± 7.54	71.6 ± 9.02
gpt-4o-mini	83.01 ± 4.83	83.17 ± 4.23	78.53 ± 6.50	77.69 ± 8.19	71.22 ± 5.22

Table 6. **Semantic similarity results for arranging shuffled sentences task.** Each cell shows the average semantic similarity (higher is better) between the model's re-ordered text and the ground-truth sentences. The experimental setup follows the one depicted in Figure 3.

Model	100	250	500	750	1000
Category Recall					
o1	95.02 ± 1.96	92.13 ± 2.07	82.98 ± 16.37	61.82 ± 33.75	53.27 ± 30.36
o3-mini	84.89 ± 3.54	76.78 ± 2.53	64.82 ± 6.90	53.18 ± 5.04	41.42 ± 7.03
gpt-4o	58.69 ± 18.99	34.48 ± 8.01	12.89 ± 4.73	8.39 ± 2.83	6.99 ± 1.82
gpt-4o-mini	7.69 ± 5.36	9.52 ± 9.82	8.28 ± 6.12	3.88 ± 2.29	4.09 ± 1.72
gemini-2.0-flash	72.07 ± 14.89	62.39 ± 16.72	65.95 ± 3.34	47.97 ± 10.83	20.13 ± 8.5
gemini-1.5-pro	73.5 ± 4.32	71.83 ± 4.24	68.05 ± 3.03	57.85 ± 7.38	53.67 ± 15.2
Global Recall					
o1	99.90 ± 0.30	96.89 ± 1.58	87.62 ± 17.03	66.18 ± 35.71	56.18 ± 31.06
o3-mini	99.00 ± 1.00	90.36 ± 2.04	76.88 ± 7.19	62.39 ± 6.16	48.97 ± 8.66
gpt-4o	65.40 ± 19.92	41.24 ± 8.64	19.05 ± 5.95	11.47 ± 3.15	9.73 ± 2.32
gpt-4o-mini	11.08 ± 8.38	12.89 ± 14.94	11.51 ± 9.16	6.45 ± 4.68	7.17 ± 3.31
gemini-2.0-flash	85.40 ± 16.82	71.12 ± 18.3	72.17 ± 2.56	59.81 ± 4.98	54.14 ± 8.17
gemini-1.5-pro	84.71 ± 6.54	85.22 ± 5.02	79.82 ± 5.09	71.53 ± 9.37	66.81 ± 18.54

Table 7. **Entity grouping results.** The experimental setup follows the one depicted in Figure 4.