

Characterizing the Compression-Benefit Threshold in Long-Context LLMs

Alex Knusel
knuse001@umn.edu

Nick Hinds
hinds084@umn.edu

Zach Kaupp
kaupp047@umn.edu

Team Small Language Model

Abstract

Long-context language models often lose task accuracy well before their advertised context windows are reached, and prompt compression has been proposed as a way to recover that loss. We define the compression-benefit threshold as the smallest context length at which a model’s task accuracy on a compressed prompt matches or exceeds its accuracy on the uncompressed baseline, and we estimate it for LLMLingua-2, LongLLMLingua, and SelectiveContext using Llama-3.2-1B-Instruct on RULER. We evaluate context lengths from 4,096 to 65,536 tokens and compression ratios from 0.1 to 0.9, with SelectiveContext included for its completed subset of ratios. The threshold is strongly compression-method-dependent. LongLLMLingua matches or exceeds the raw baseline at every length from 8,192 tokens upward and at every ratio, while LLMLingua-2 and SelectiveContext fail to match the raw baseline in any of our experiments. For models of this scale, the choice of compression method matters more than choosing when to compress.

1 Introduction

Large language models can now process far longer inputs than earlier systems, with several recent models advertising context windows extending into million-token ranges (Gemini Team et al., 2024). These larger windows allow users to pass full documents, long conversations, retrieved evidence, or code repositories directly into a model. However, a larger context window does not guarantee that the model can reliably use the entire input. Prior evaluations show that long-context models often lose accuracy as input length increases, even when the input remains within the model’s advertised limit (Hsieh et al., 2024; Paulsen, 2026). This gap between the maximum supported context length and the effective context length motivates the central question of our project.

One response to this problem is prompt compression. A compression method removes tokens that are less relevant to the task while preserving the information needed to answer correctly. In principle, compression can help in two ways. It can reduce the cost of inference by shortening the prompt, and it can make the relevant evidence easier for the model to identify. Existing methods such as LLMLingua-2, LongLLMLingua, and SelectiveContext follow this general strategy, although they differ in how they estimate token importance and reconstruct the final prompt (Pan et al., 2024; Jiang et al., 2024; Li et al., 2023).

The remaining question is when to apply compression. At short context lengths, the model may already have enough capacity to process the raw prompt, so compression can remove useful evidence without providing much benefit. At longer context lengths, the raw prompt may contain enough irrelevant information that a shorter compressed prompt becomes preferable. This creates the possibility of a transition point at which compression matches, or even outperforms, the raw context. We define this point as the compression-benefit threshold: the smallest context length at which a compressed prompt achieves accuracy at least as high as the uncompressed baseline.

This report estimates that threshold empirically. We evaluate three compression methods against a raw-context baseline on NVIDIA’s RULER benchmark, varying both context length and compression ratio. RULER is appropriate for this study because it spans retrieval, multi-hop tracing, aggregation, and question-answering tasks rather than relying only on needle-in-a-haystack retrieval (Hsieh et al., 2024). Our results show that the threshold is not a property of context length alone. Instead, it depends strongly on the compression method. LongLLMLingua achieves compressed accuracies near or above the raw baseline across much of our tested range, placing its threshold near the lower

end of the tested context lengths. LLMLingua-2 and SelectiveContext exhibit different behavior, suggesting that compression can be harmful when it removes information the downstream model needs.

We make three contributions. First, we define the compression-benefit threshold as an accuracy-based criterion for evaluating when prompt compression becomes useful in long-context inference. Second, we provide an evaluation pipeline for estimating this threshold across compression methods, context lengths, and compression ratios. Third, we show that the relationship between retained context and downstream accuracy is more method-dependent than a simple distractor-removal explanation would predict.

2 Previous work

Our work bridges two related but largely separate research areas: long-context evaluation and prompt or context compression. Long-context evaluation assesses how a model’s accuracy changes as input length increases. Context compression studies how selectively removing tokens affects accuracy and inference cost. Both areas examine the relationship between input content and model performance, but they have not been systematically studied together.

2.1 Long-context evaluation

A growing body of work shows that increasing a model’s maximum context window does not guarantee that it can effectively use the full input. RULER is a synthetic long-context benchmark designed to quantify this gap (Hsieh et al., 2024). Rather than relying on natural documents with uncontrolled structure, RULER places target information at controlled depths within long sequences, which allows accuracy to be measured cleanly as a function of context length. The benchmark spans retrieval, multi-hop tracing, aggregation, and question-answering tasks, which makes it broader than simple needle retrieval. Across these tasks, RULER reports a consistent decline in accuracy as input length grows, even when the input remains within the model’s advertised context window, and this pattern holds across multiple model families and sizes (Hsieh et al., 2024). Subsequent work reaches a similar conclusion using a different evaluation methodology, finding that effective context length is often a small fraction of the advertised maximum (Paulsen, 2026).

2.2 Prompt and context compression

A separate line of work attempts to mitigate long-context degradation without modifying the model. Prompt and context compression methods preprocess the input to retain semantically important tokens and discard the rest, producing a shorter prompt that the model can use more efficiently. We evaluate three methods that follow this strategy.

LLMLingua-2 frames compression as a token-classification problem (Pan et al., 2024). A small auxiliary model is trained on data distilled from a larger LLM to predict which tokens are most useful for downstream tasks. At inference time, the auxiliary model labels each prompt token, and the lowest-scoring tokens are removed until the target compression ratio is reached. The method is task-agnostic, meaning it does not need to see the question before deciding which tokens to keep.

LongLLMLingua is a question-aware member of the LLMLingua family designed for long-context settings (Jiang et al., 2024). It conditions its token-importance scores on the question, reorders context segments so that high-importance content appears in positions where the model attends more reliably, and adapts the compression rate across segments based on their estimated relevance. These changes target known long-context failure modes, including the lost-in-the-middle effect, where information placed near the middle of a long input is often harder for models to retrieve than information near the beginning or end (Liu et al., 2023).

SelectiveContext takes an information-theoretic approach (Li et al., 2023). It uses a small language model to compute the self-information of each lexical unit in the prompt, then prunes units with low self-information, assuming that low-information content contributes little to the model’s ability to answer correctly. The method is independent of the question and does not require a specialized auxiliary model trained for compression.

Empirical evaluations of each method report that compressed prompts can preserve and, in some settings, improve task accuracy compared to the raw prompt (Pan et al., 2024; Jiang et al., 2024; Li et al., 2023). However, these evaluations are typically run at one or two fixed context lengths and one or two fixed compression ratios, leaving the relationship between context length, compression ratio, and the benefit of compression uncharacterized. Our work fills this gap by comparing compressed and uncompressed prompts across six compression ratios and

five context lengths spanning 4,096 to 65,536 tokens, which lets us estimate a compression-benefit threshold separately for each method.

3 Methodology

We measure how task accuracy changes when prompts are compressed before being passed to a long-context model. Each compressed condition is compared with an uncompressed baseline of the same context length, allowing us to isolate the effect of compression from that of increasing input length. The full experiment is implemented in two notebooks: a prediction notebook that adapts the RULER generation and model-wrapper code to run in Google Colab and writes predictions incrementally to disk, and an evaluation notebook that reads the saved prediction files and compressed prompts to compute accuracy, retention, and figures. This separation between prediction and evaluation makes the results easier to reproduce, since the evaluation can be rerun from saved artifacts without regenerating model outputs. Our experimental settings are summarized in Table 1.

3.1 Model and inference settings

All experiments use meta-llama/llama-3.2-1B-Instruct, a 1.2B-parameter instruction-tuned model with an advertised context window of 128k tokens (Meta AI, 2024). We originally planned to evaluate a larger Llama model, but running the full grid with the 3B model took significantly more time than our available compute allowed, so we switched to the 1B model, which fit within our Google Colab High-RAM L4 GPU setup at 65,536-token contexts while still exhibiting long-context degradation.

The model is loaded in bfloat16 with FlashAttention2 (Dao et al., 2022). Inference uses batch size 1 and deterministic decoding with temperature 0, top- k 32, and top- p 1.0. These settings remove generation-time randomness from the comparison, so differences between conditions reflect changes in context length and compression rather than sampling variation.

3.2 Benchmark

We evaluate on the synthetic portion of RULER, a benchmark that generates prompts at target context lengths with known reference answers (Hsieh et al., 2024). This allows direct comparison of raw and compressed versions of the same underlying example. RULER consists of 13 tasks, including

eight needle-in-a-haystack variants, variable tracking, common-word extraction, frequent-word extraction, and two question-answering tasks based on SQuAD and HotpotQA (Rajpurkar et al., 2016; Yang et al., 2018). The full task inventory is summarized in Table 2. We generate 30 validation samples per task for each context length and compression ratio. The tested context lengths are 4,096, 8,192, 16,384, 32,768, and 65,536 tokens.

For each context length, the pipeline first generates raw RULER prompts, then applies each compression method as a post-processing step to those prompts rather than during task generation. Every compression method therefore operates on the same underlying haystack as the raw baseline, so differences in performance can be attributed to the compression method rather than to different randomly generated examples.

3.3 Compression configurations

We compare an uncompressed raw baseline against three compression methods: LLMLingua-2, LongLLMLingua, and SelectiveContext (Pan et al., 2024; Jiang et al., 2024; Li et al., 2023). For LLMLingua-2 and LongLLMLingua, we evaluate compression ratios of 0.1, 0.3, 0.5, 0.7, 0.8, and 0.9. The ratio denotes the target fraction of the original context retained after compression, so lower ratios correspond to more aggressive compression. SelectiveContext is reported at 0.1, 0.5, and 0.7 because it was substantially slower than the other methods and consumed a large share of our Colab compute budget. The full set of completed runs is summarized in Table 3.

Compression is applied only to the haystack portion of each prompt. The pipeline preserves the task instruction and final question, compresses the haystack between them, and reconstructs the full prompt. This prevents the compressor from deleting the task’s instructions, which would create failures unrelated to long-context evidence selection.

LLMLingua-2 is implemented through the Microsoft LLMLingua package using the microsoft/llmlingua-2-xlm-roberta-large-meetingbank checkpoint (Pan et al., 2024). Long RULER contexts can exceed the auxiliary model’s practical input length, so the pipeline splits the haystack into chunks of at most 400 compressor tokens before compression, then concatenates the compressed chunks. We protect newlines, question marks, and colons as structural tokens to prevent the compression from breaking the prompt format.

LongLLMLingua is also implemented through the LLMLingua package, but uses the long-context compression mode (Jiang et al., 2024). Unlike LLMLingua-2, it conditions compression on the task question; we parse the question text from the prompt when available, and otherwise use the generic placeholder “Find the information needed to answer the task.” The method estimates a target token budget from the requested ratio, ranks context chunks by relevance, reorders high-importance content, and applies context-level filtering. We disable token-level filtering for these runs.

SelectiveContext uses a GPT-2-based self-information model and prunes at the phrase level (Li et al., 2023). Its `reduce_ratio` parameter specifies the fraction of content to remove, whereas the LLMLingua family’s `rate` parameter specifies the fraction to retain. We invert the parameter at the call site so that all three methods are configured by the same retained-fraction convention used throughout this paper.

3.4 Accuracy metric

We report task accuracy using the RULER evaluation metrics. Needle-in-a-haystack, variable-tracking, and extraction tasks are scored with a case-insensitive string-match-all metric, which gives partial credit according to the fraction of required reference strings that appear in the model output. The question-answering tasks are scored with a case-insensitive string-match-part metric, which gives credit when any acceptable reference answer appears in the output. We compute a per-task score for each context length and compression variant, then average across the 13 tasks to obtain an overall RULER score for that setting.

3.5 Context retention metric

Accuracy alone does not reveal whether compression fails because the answer was removed or because the model failed to use information that remained in the prompt. To separate these cases, we also compute a context-retention score on the compressed prompt before model inference. This score measures how much task-relevant information survives compression.

For needle-in-a-haystack and question-answering tasks, retention is measured by checking whether the reference answer strings appear in the compressed prompt. For common-word and frequent-word extraction, the evaluation pipeline recomputes word frequencies in the compressed

text and checks whether the target words remain among the most frequent words. For variable tracking, the pipeline parses surviving assignments with a regular expression and traces whether each target variable still resolves to the expected value.

Retention is diagnostic rather than the main success metric. A compressed prompt with low retention likely lost necessary evidence; one with high retention but low downstream accuracy suggests the information remained present but the model did not use it correctly. The compression papers we evaluate do not report retention separately from accuracy, so this diagnostic is part of our methodological contribution.

4 Results

We organize the results around three observations from the experimental grid. First, the compression-benefit threshold we set out to estimate lies outside the tested range for two of the three methods. Second, LongLLMLingua effectively lacks an observable threshold within our grid, as it matches or exceeds the raw baseline at almost every ratio and context-length cell. Third, retention and accuracy are decoupled in a way that a simple distractor-removal account of compression does not predict. Figure 1 shows average RULER scores by compression ratio across methods, aggregated over context lengths. Figure 2 summarizes the best-scoring method per (compression ratio, context length) cell. Figure 3 pairs accuracy and retention at the cell level, and Figure 4 shows performance over context length at a fixed compression ratio of 0.7. The full numerical grid is provided in Table 7.

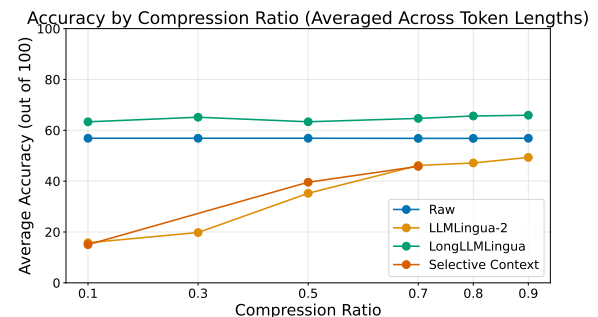


Figure 1: Average RULER score by compression ratio, averaged across all context lengths. LongLLMLingua exceeds the raw baseline at every ratio when averaged across context lengths, while SelectiveContext and LLMLingua-2 remain below the baseline.

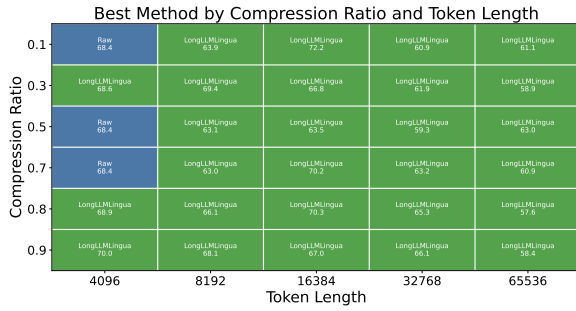


Figure 2: Best-scoring method at each (compression ratio, context length) cell. LongLLMLingua outperforms all other methods except at 4,096 tokens with ratios 0.1, 0.5, and 0.7, where raw is highest.

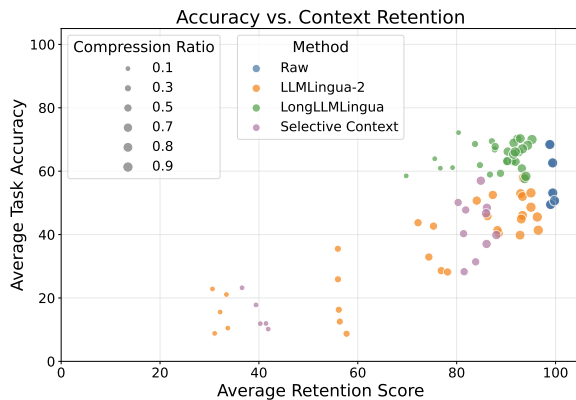


Figure 3: Accuracy vs. context retention for every compression method, compression ratio, and context length combination. Retention alone does not determine accuracy: LongLLMLingua at low retention matches raw, while LLMLingua-2 at high retention can fall below.

4.1 Threshold behavior across methods

The original hypothesis of the project assumed that each compression method would reveal a compression-benefit threshold. At short context lengths, compression would likely be harmful because the context length is not long enough to degrade the model’s performance, and the compression risks removing necessary data. At longer context lengths, compression would become helpful because distractor accumulation would make the raw prompt more difficult to use. The reasons we find no threshold differ by compression method.

Figure 2 and Table 6 show that LongLLMLingua matches or exceeds the raw baseline at context lengths of 8,192 tokens and longer, at every compression ratio we tested. At 4,096 tokens it is close to the raw baseline at most ratios, ranging from -9.9 to $+1.6$ percentage points across ratios according to Table 7; the largest gap occurs at the most aggressive ratio of 0.1. Its threshold is

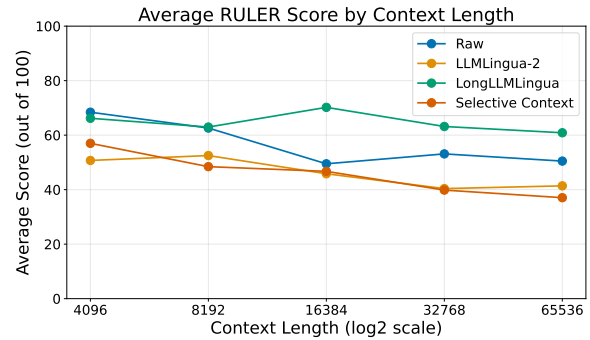


Figure 4: Average RULER score by context length at a compression ratio of 0.7. Raw and LongLLMLingua are roughly tied at 4,096 tokens, but LongLLMLingua dominates at longer lengths, while LLMLingua-2 and SelectiveContext stay below raw across the full range.

therefore at or near the shortest length we tested, which means we cannot observe the predicted short-context penalty within our grid. The largest gains occur at 16,384 tokens, where LongLLMLingua exceeds the raw baseline by 14–23 percentage points across ratios.

LLMLingua-2 and SelectiveContext fail to match the raw baseline at any setting we tested, as shown in Table 7. Across the 30 ratio-length cells for LLMLingua-2 and the 15 cells for SelectiveContext, compressed accuracy is lower than raw accuracy in every cell. The closest LLMLingua-2 comes is at 16,384 tokens with a ratio of 0.9, where it trails the raw context by 0.8 percentage points; the closest SelectiveContext comes is at 16,384 tokens with a ratio of 0.7, trailing raw by 2.8 percentage points. For these two methods, any threshold lies beyond the longest context we tested. Figure 2 shows that LongLLMLingua is the best-scoring method at every cell except 4,096 tokens at ratios 0.1, 0.5, and 0.7, where raw is the highest.

4.2 Method dominance over context length

The clearest pattern in Figure 1 and Table 4 is that the compression method matters more than the context length or the ratio. LongLLMLingua’s worst configuration (ratio 0.1 at 4,096 tokens) scores 58.5, higher than LLMLingua-2’s best at any context length (ratio 0.9 at 4,096 tokens, 57.9) and matching or beating SelectiveContext’s best completed configuration (ratio 0.7 at 4,096 tokens, 57.0). The same pattern holds at the longest context. At 65,536 tokens, LongLLMLingua scores 58.4–63.0 across all six ratios, while LLMLingua-2 reaches 8.7–44.9 and SelectiveContext 10.2–37.1.

Figure 4 illustrates this divergence at a fixed com-

pression ratio of 0.7. Raw and LongLLMLingua sit within about 2 percentage points of each other at 4,096 tokens, but LongLLMLingua opens a 10–21 percentage point gap over raw from 16,384 tokens onward, while LLMLingua-2 and SelectiveContext stay below raw across the full range. The figure also illustrates that the raw baseline is itself non-monotonic in context length. It drops sharply at 16,384 tokens before partially recovering, a pattern that reflects the long-context degradation that motivated this study in the first place.

This demonstrates that knowing the context length and target compression ratio is not enough to predict whether compression will help because the compression method has the greatest impact. The same context length and ratio can produce a 23 percentage point gain over the raw baseline, as LongLLMLingua does at 16,384 tokens with a ratio of 0.1, or a 38 percentage point loss, as SelectiveContext does at the same context length and ratio. Method choice carries most of the variation.

4.3 Decoupling of retention and accuracy

We expected accuracy to match retention. A compressed prompt that retains most task-relevant content should, in principle, allow the model to answer correctly, while a prompt that drops the relevant content should not. LLMLingua-2 breaks the first half of that expectation.

Figure 3 plots accuracy against retention for every method, ratio, and context-length cell, with average retention values aggregated by ratio reported in Table 5. LLMLingua-2 at ratio 0.9 retains roughly 93%–97% of task-relevant content across context lengths, comparable to the raw baseline’s 99%. Despite this, LLMLingua-2’s accuracy at ratio 0.9 falls 10–27 percentage points below raw at every length, with the gap widening as context grows. The answer survives compression in roughly the same proportion as the raw baseline, but the model performs much worse when reading the compressed version.

LongLLMLingua exhibits the opposite pattern, breaking the second half of the expectation. At ratio 0.1 and 4,096 tokens, its retention is 69.8%, the lowest in the compressed grid, yet its accuracy is 58.5, within 10 percentage points of the raw baseline at the same length. LongLLMLingua appears to remove content the model would not have used while keeping content the model needs: what it discards matters more than how much it discards.

The simplest reading of these patterns is

that LongLLMLingua’s question-conditioning and context-reordering steps do work that LLMLingua-2’s task-agnostic classifier cannot, even when LLMLingua-2 retains a higher fraction of relevant content. SelectiveContext sits between the two on retention (85.9% at a ratio of 0.7 and 16,384 tokens, comparable to LLMLingua-2 at lower ratios), but its accuracy still tracks closer to LLMLingua-2 than to LongLLMLingua. Distractor removal alone does not explain what makes a compressed prompt useful; the way surviving content is presented to the model likely matters as well.

4.4 Compression algorithm error analysis

The primary shortcomings of the compression methods differ by task family. Single-evidence retrieval tasks are the least sensitive to compression when the method is question-aware. The model only needs one short span from the haystack, so LongLLMLingua can remove large amounts of irrelevant context while preserving the evidence that answers the final query, which is consistent with its largest gains at 16,384 tokens. Raw context, by contrast, sometimes fails because the answer is present but buried among many distractors.

Tasks that require combining or preserving several pieces of information are more fragile. Variable tracking is especially vulnerable because a correct answer depends on an ordered chain of assignments, not just the presence of the final value. If compression deletes an intermediate assignment, changes the order of assignments, or removes enough surrounding structure that the chain becomes ambiguous, the downstream model can no longer reconstruct the final state. Beyond the obvious case where heavy compression deletes assignments outright, surrounding structure also matters. For example, when running with our same experimental settings, in LLMLingua-2’s first compressed variable tracking prompt at a compression ratio of 0.5 and a context length of 4096, the compression preserves all the necessary variable assignments. However, it removes the newline characters that put the variable assignments on their own lines. For this prompt, the model only identified the first variable assignment, and performed worse than the raw baseline, even with a shorter prompt that contained all the necessary assignments.

Common-word and frequent-word extraction have a similar issue: the answer depends on aggregate statistics over the context, and phrase- or token-level pruning can change the frequency distri-

bution even when many individual relevant words remain. For example, in LLM-Lingua-2's second compressed frequent word extraction prompt at a compression ratio of 0.1 and context length of 4096, the compressed prompt's word-frequency distribution no longer matched the raw prompt's: the random-string token "emswml" appeared more often than the expected answer "akrkxo".

A different group of errors appears at higher compression ratios for LLM-Lingua-2: the relevant string survives compression, but the model still fails to use it. At ratio 0.9, LLM-Lingua-2 keeps most task-relevant content by our retention metric, yet accuracy stays well below the raw baseline. This suggests that some failures are caused by presentation rather than deletion. Chunkwise compression can remove formatting cues, local transitions, or nearby distractors in a way that leaves the answer text visible but harder for the model to identify as the answer. In multi-step tasks, it can also separate or reorder the necessary facts enough that the model does not combine them correctly.

SelectiveContext exhibits a more direct failure. Because it prunes by self-information rather than by the final question, it can remove tokens deemed to have low significance but that are semantically necessary for the task. This is particularly harmful for synthetic RULER prompts, where the crucial evidence may be short, formulaic, or repeated. SelectiveContext behaves like a lossy shortening method rather than a compressor, which matches its consistently lower accuracy across our tests.

Overall, these errors indicate that compression quality is not captured by retained length or by answer-string retention alone. Successful compression must preserve the evidence, preserve the structure that makes the evidence usable, and place that evidence where the model can use it. LongLLM-Lingua is the only method we tested that reliably satisfies those conditions.

5 Discussion

5.1 Replicability

The full pipeline is implemented in two notebooks that are released alongside this report. Inference is deterministic, with temperature 0, greedy decoding, and batch size 1, so rerunning the prediction notebook against the same model checkpoint, RULER seed, and compression library versions will produce the same outputs we report. The compression methods all use publicly available check-

points or models released by their original authors: LLM-Lingua-2 uses the microsoft/llmlingua-2-xlm-roberta-large-meetingbank checkpoint, LongLLM-Lingua uses the default LLM-Lingua compressor, and SelectiveContext uses a GPT-2 self-information model. The model under test, Llama-3.2-1B-Instruct, is also publicly available through Hugging Face. We pin the LongLLM-Lingua hyperparameters in the methodology section so that a reader can reproduce our configuration without referring to the notebook.

Two practical barriers to exact reproduction remain. RULER generates synthetic prompts at runtime using random seeds, so a rerun with a different random state will produce different underlying haystacks; we use a fixed seed of 42. SelectiveContext was also substantially slower at long contexts than the LLM-Lingua methods, so we did not complete a full grid sweep within our compute budget. A reader with more compute can complete the missing 0.3 and 0.9 cells without changes to our code.

5.2 Datasets

We do not introduce a new dataset. RULER generates the synthetic prompts we use (Hsieh et al., 2024), which draws on SQuAD (Rajpurkar et al., 2016) and HotpotQA (Yang et al., 2018) for the question-answering subtasks and on the Paul Graham essay corpus as the haystack source for several needle-in-a-haystack variants. RULER, SQuAD, and HotpotQA are publicly released for research use, and the RULER authors distribute the essay corpus as part of the benchmark setup.

Our work does not produce annotations or labeled data for other researchers to adopt. Its reusable artifact is a comparison grid of accuracy and retention scores across method, ratio, and context-length cells. Others can compare against this grid by extending RULER-based compression evaluation to other models or compression methods. The CSV exports from our evaluation pipeline are the natural artifact for that kind of reuse.

The main dataset limitation is that RULER is synthetic. That design is useful here because it gives controlled context lengths, known answers, and a repeatable task structure, which makes the threshold easier to estimate cleanly. However, synthetic prompts do not capture the ambiguity, redundancy, and diffuse evidence found in real long-document tasks, so results on code repositories, multi-document retrieval inputs, long conversation histories, legal filings, or medical records may dif-

fer from the patterns we observe here.

5.3 Ethics

Prompt compression is a low-risk technique because it operates on the input to a language model and does not change the model’s parameters or training data. Even so, three considerations are worth flagging.

The first concern is silent quality degradation. Our results suggest that LLMLingua-2 and SelectiveContext underperform raw context at every completed ratio in our grid for this model. Practitioners who deploy these methods solely based on fixed-length results from prior papers may apply them in settings where they degrade output quality. The mitigation is straightforward but important: compression should be evaluated on the deployment task before production use, especially when the output affects decisions beyond a benchmark.

The second concern is uneven failure across task types. Some RULER tasks lose much more accuracy under compression than others, which means a method that looks acceptable on average may be much worse on the subset of inputs a downstream user depends on. If compression removes or obscures relevant evidence in legal review, medical summarization, or compliance work, the model may produce an answer that appears grounded while omitting important information.

The third concern is transparency. End users of a compressed-context system may not realize their input is shortened before the model sees it. Systems that use compression should disclose that preprocessing step and preserve access to the source text. For sensitive applications, compressed inference should be paired with evidence tracing or retrieval checks to verify whether the model’s answer depends on information that survived compression.

5.4 Limitations and future work

The most consequential limitation is the model. Our results come from a single 1B-parameter model, and the strength of our headline finding depends on how that model attends to long inputs. Larger and more recent models may have better raw-context performance, which could reduce the gap between LongLLMLingua and the raw baseline, and may also be better at using compressed prompts, which could improve LLMLingua-2 and SelectiveContext. We cannot determine from this experiment whether the patterns we report generalize across model scales.

A second limitation is that our threshold is solely accuracy-based, so the report does not determine whether the tested methods are practical in real environments. Compression that improves accuracy is not necessarily worth using if it adds latency or memory overhead exceeding what compression saves. Conversely, compression that lowers accuracy may still be useful if it substantially reduces inference cost. We do not measure those trade-offs.

A third limitation concerns SelectiveContext: we completed only three of the six compression ratios, so our evaluation of this method is incomplete. SelectiveContext might cross the raw baseline at one of the ratios we did not run, or at a context length beyond 65,536 tokens.

Future work should extend this report in three directions. The first is model scale: rerunning our pipeline with Llama-3.2-3B and a larger model such as Llama-3.1-8B would show whether the compression benefit transfers, shrinks, or reverses with model size. The second is task realism: replacing or supplementing RULER with a naturally occurring long-context benchmark like LongBench would test whether question-aware compression dominates when the input is less synthetic. The third is the retention-accuracy gap. An ablation that varies position, format, and chunking artifacts independently would determine whether LongLLMLingua’s advantage comes from selecting better tokens, presenting them in a better order, or both.

6 Conclusion

We estimated the compression-benefit threshold for LLMLingua-2, LongLLMLingua, and SelectiveContext on RULER using Llama-3.2-1B-Instruct, and did not observe the predicted harmful-to-helpful transition for any of the three methods within our tested range. LongLLMLingua’s threshold sits at or near the bottom of the range, while LLMLingua-2 and SelectiveContext fail to match the raw baseline at any cell in the completed grid. The compression method, not the context length or compression ratio, explains the largest observed performance differences: for the model we tested, the question of which compressor to use, if any, is more important than the question of when to apply compression. Whether these patterns hold at larger model scales, on naturally occurring long-context inputs, or once latency and memory are considered remain open questions for future research.

References

- Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. 2022. [Flashattention: Fast and memory-efficient exact attention with io-awareness](#).
- Gemini Team, Petko Georgiev, Ving Ian Lei, Ryan Burrell, Libin Bai, et al. 2024. [Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context](#).
- Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekish, Fei Jia, Yang Zhang, and Boris Ginsburg. 2024. [Ruler: What’s the real context size of your long-context language models?](#)
- Huiqiang Jiang, Qianhui Wu, Xufang Luo, Dongsheng Li, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. 2024. [LongLLMLingua: Accelerating and enhancing LLMs in long context scenarios via prompt compression](#). In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1658–1677, Bangkok, Thailand. Association for Computational Linguistics.
- Yucheng Li, Bo Dong, Chenghua Lin, and Frank Guerin. 2023. [Compressing context to enhance inference efficiency of large language models](#).
- Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2023. [Lost in the middle: How language models use long contexts](#).
- Meta AI. 2024. [Llama 3.2: Revolutionizing edge AI and vision with open, customizable models](#).
- Zhuoshi Pan, Qianhui Wu, Huiqiang Jiang, Menglin Xia, Xufang Luo, Jue Zhang, Qingwei Lin, Victor Rühle, Yuqing Yang, Chin-Yew Lin, H. Vicky Zhao, Lili Qiu, and Dongmei Zhang. 2024. [LLMLingua-2: Data distillation for efficient and faithful task-agnostic prompt compression](#). In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 963–981, Bangkok, Thailand. Association for Computational Linguistics.
- Norman Paulsen. 2026. [Context is what you need: The maximum effective context window for real world limits of llms](#).
- Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. [Squad: 100,000+ questions for machine comprehension of text](#).
- Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. [Hotpotqa: A dataset for diverse, explainable multi-hop question answering](#).

A Appendix

This appendix summarizes the collected evaluation data used in our analysis. All reported scores are percentages. Accuracy is the average RULER task score, and retention is the average proportion of task-relevant context preserved after compression.

Field	Value
Benchmark	RULER synthetic benchmark
Model	meta-llama/Llama-3.2-1B-Instruct
Context lengths	4,096; 8,192; 16,384; 32,768; 65,536 tokens
Compression ratios	0.1, 0.3, 0.5, 0.7, 0.8, 0.9
Prompt variants	raw, LLMLingua-2, LongLLMLingua, Selective Context
Number of tasks	13 synthetic task types
Examples per task	30 validation examples
Primary metric	Mean task accuracy (%)
Auxiliary metric	Mean context retention (%)

Table 1: Evaluation setup for the collected data.

Task subset	Task IDs	Evaluation target
Needle-in-a-haystack		
Single key	niah_single_1	Single-key retrieval
	niah_single_2	
	niah_single_3	
Multi-key	niah_multikey_1	Multi-key retrieval
	niah_multikey_2	
	niah_multikey_3	
Multi-query/value	niah_multiquery	Multi-query and multi-value retrieval
	niah_multivalue	
Question answering		
Long-context QA	qa_1	Long-context QA
	qa_2	
Word extraction		
Common/frequent words	cwe	Common and frequent word extraction
	fwe	
Variable tracking		
State tracking	vt	State tracking over long context

Table 2: Synthetic RULER task groups included in the evaluation.

Compression ratio	Units	Variants included
0.1	20	raw, LLMingua-2, LongLLMingua, Selective Context
0.3	15	raw, LLMingua-2, LongLLMingua
0.5	20	raw, LLMingua-2, LongLLMingua, Selective Context
0.7	20	raw, LLMingua-2, LongLLMingua, Selective Context
0.8	15	raw, LLMingua-2, LongLLMingua
0.9	15	raw, LLMingua-2, LongLLMingua

Table 3: Evaluation run inventory. A unit is one context-length by prompt-variant condition evaluated over all 13 tasks.

Compression ratio	Raw	LLMingua-2	LongLLMingua	Selective Context
0.1	56.87	15.77	63.33	15.01
0.3	56.87	19.79	65.14	–
0.5	56.87	35.22	63.35	39.57
0.7	56.82	46.17	64.68	45.81
0.8	56.82	47.15	65.63	–
0.9	56.87	49.33	65.93	–

Table 4: Mean accuracy across context lengths for each compression ratio and prompt variant. Missing entries indicate that the variant was not collected for that ratio.

Compression ratio	Raw	LLMingua-2	LongLLMingua	Selective Context
0.1	99.30	32.19	76.32	39.92
0.3	99.30	56.44	85.99	–
0.5	99.30	75.38	89.86	81.77
0.7	99.30	86.87	91.57	86.20
0.8	99.30	93.09	92.44	–
0.9	99.30	95.27	93.74	–

Table 5: Mean context retention across context lengths for each compression ratio and prompt variant.

Compression ratio	Context length	Best prompt variant	Accuracy
0.1	4,096	raw	68.4
0.1	8,192	LongLLMLingua	63.9
0.1	16,384	LongLLMLingua	72.2
0.1	32,768	LongLLMLingua	60.9
0.1	65,536	LongLLMLingua	61.1
0.3	4,096	LongLLMLingua	68.6
0.3	8,192	LongLLMLingua	69.4
0.3	16,384	LongLLMLingua	66.8
0.3	32,768	LongLLMLingua	61.9
0.3	65,536	LongLLMLingua	58.9
0.5	4,096	raw	68.4
0.5	8,192	LongLLMLingua	63.1
0.5	16,384	LongLLMLingua	63.5
0.5	32,768	LongLLMLingua	59.3
0.5	65,536	LongLLMLingua	63.0
0.7	4,096	raw	68.4
0.7	8,192	LongLLMLingua	63.0
0.7	16,384	LongLLMLingua	70.2
0.7	32,768	LongLLMLingua	63.2
0.7	65,536	LongLLMLingua	60.9
0.8	4,096	LongLLMLingua	68.9
0.8	8,192	LongLLMLingua	66.1
0.8	16,384	LongLLMLingua	70.3
0.8	32,768	LongLLMLingua	65.3
0.8	65,536	LongLLMLingua	57.6
0.9	4,096	LongLLMLingua	70.0
0.9	8,192	LongLLMLingua	68.1
0.9	16,384	LongLLMLingua	67.0
0.9	32,768	LongLLMLingua	66.1
0.9	65,536	LongLLMLingua	58.4

Table 6: Best-performing prompt variant by compression ratio and context length. Accuracy is reported as a percentage.

Compression ratio	Context length	Raw	LLMLingua-2	Long LLMLingua	Selective Context
0.1	4,096	68.4	22.9	58.5	23.2
0.1	8,192	62.6	21.1	63.9	17.8
0.1	16,384	49.5	15.6	72.2	12.0
0.1	32,768	53.1	8.8	60.9	11.9
0.1	65,536	50.7	10.5	61.1	10.2
0.3	4,096	68.4	35.5	68.6	—
0.3	8,192	62.6	25.9	69.4	—
0.3	16,384	49.5	16.3	66.8	—
0.3	32,768	53.1	12.6	61.9	—
0.3	65,536	50.7	8.7	58.9	—
0.5	4,096	68.4	43.7	67.8	50.1
0.5	8,192	62.6	42.7	63.1	47.8
0.5	16,384	49.5	32.9	63.5	40.3
0.5	32,768	53.1	28.6	59.3	31.4
0.5	65,536	50.7	28.2	63.0	28.3
0.7	4,096	68.4	50.7	66.2	57.0
0.7	8,192	62.6	52.5	63.0	48.4
0.7	16,384	49.5	45.8	70.2	46.7
0.7	32,768	53.1	40.4	63.2	39.8
0.7	65,536	50.5	41.4	60.9	37.1
0.8	4,096	68.4	53.0	68.9	—
0.8	8,192	62.6	52.0	66.1	—
0.8	16,384	49.5	46.1	70.3	—
0.8	32,768	53.1	39.8	65.3	—
0.8	65,536	50.5	44.9	57.6	—
0.9	4,096	68.4	57.9	70.0	—
0.9	8,192	62.6	53.2	68.1	—
0.9	16,384	49.5	48.6	67.0	—
0.9	32,768	53.1	45.6	66.1	—
0.9	65,536	50.7	41.4	58.4	—

Table 7: Accuracy by compression ratio, context length, and prompt variant. Missing entries indicate that the variant was not collected for that ratio.