

Understanding and Improving Information Preservation in Prompt Compression for LLMs

Weronika Łajewska*

University of Stavanger, weronika.lajewska@uis.no

Momchil Hardalov Laura Aina Neha Anna John Hang Su
Amazon

{momchilh, eailaura, nehajohn, shawnsu}@amazon.com

Lluís Màrquez†

Technical University of Catalonia (UPC), lluis.marquez@upc.edu

Abstract

Recent advancements in large language models (LLMs) have enabled their successful application to a broad range of tasks. However, in information-intensive tasks, the prompt length can grow fast, leading to increased computational requirements, performance degradation, and induced biases from irrelevant or redundant information. Recently, various prompt compression techniques have been introduced to optimize the trade-off between reducing input length and retaining performance. We propose a holistic evaluation framework that allows for in-depth analysis of prompt compression methods. We focus on three key aspects, besides compression ratio: (i) downstream task performance, (ii) grounding in the input context, and (iii) information preservation. Using our framework, we analyze state-of-the-art soft and hard compression methods and show that some fail to preserve key details from the original prompt, limiting performance on complex tasks. By identifying these limitations, we are able to improve one soft prompting method by controlling compression granularity, achieving up to +23% in downstream performance, +8 BERTScore points in grounding, and 2.7× more entities preserved in compression. Ultimately, we find that the best effectiveness/compression rate trade-off is achieved with soft prompting combined with sequence-level training.

1 Introduction

Recent advancements in large language models have enabled their successful application to a broader range of tasks that require long-context input, detailed instructions, and in-context learning (ICL) demos (Brown et al., 2020; Wei et al., 2022; Touvron et al., 2023; Yao et al., 2024; Dubey et al., 2024; Abdin et al., 2024). The resulting prompts can become very long, especially for tasks

requiring extensive contextual information. Even though state-of-the-art LLMs are able to process hundreds of thousands of tokens (Anthropic., 2023; Munkhdalai et al., 2024; Gemini Team, 2024; Yang et al., 2024; Amazon AGI, 2024), providing them with long input context introduces computational inefficiencies (Shi et al., 2023; Liu et al., 2024).

Recent research has focused on developing methods for prompt compression to reduce the length of the LLMs inputs while preserving the information needed to successfully complete the task (Chang et al., 2024; Li et al., 2024a). The two main approaches to compression are: *soft-prompting*, which compresses the context into dense memory slots by learning continuous representations of the information in the latent space (Wingate et al., 2022; Mu et al., 2023; Cheng et al., 2024); and *hard-prompting*, which operates on the surface form and involves removing unnecessary or low-information content, e.g., by pruning parts of the original text, or summarizing it (Sennrich et al., 2016; Jiang et al., 2023b; Pan et al., 2024a). The resulting hard-compressed prompts still use natural language, but they may be less fluent. Soft compression methods allow for significantly higher compression rates (Li et al., 2024b; Cheng et al., 2024), as they operate in the embedding space. However, it is not easy to understand the information that is contained in soft prompts since, in contrast to hard prompting, these are not directly interpretable.

A limitation of the work done on prompt compression is that it focuses mostly on downstream performance on tasks like question answering (QA, Mu et al. (2023)) or reading comprehension (RC, Li et al. (2024b)). We argue that downstream performance itself is insufficient to evaluate the quality and limitations of a compression method, and to understand its ability to preserve information from the original context. To address this gap, we propose a holistic evaluation framework to analyze the performance of prompt compression ap-

*Work conducted during an internship at AWS AI Labs.

†Work done while at Amazon

proaches, which considers three key dimensions, beyond compression rate: (i) downstream task performance, (ii) grounding of the compressed context to the original text, and (iii) type and amount of information preserved during compression. The framework includes the evaluation on various text generation tasks with different complexity and context lengths (from 850 to 6.5K tokens), including multi-hop reasoning, conversation QA, long document summarization, and mathematical reasoning.

We use our newly proposed framework to analyze a representative hard-prompting method, LLMingua (Jiang et al., 2023b), and two soft-prompting approaches, xRAG (Cheng et al., 2024) and PISCO (Louis et al., 2025). All these methods are limited in handling long-context scenarios, exhibiting performance drops across various generation tasks and an increased number of ungrounded responses. We also demonstrate that the target LLM struggles to accurately reconstruct the original context from soft prompts, particularly for xRAG, often hallucinating information and failing to regenerate crucial details. Based on these observations, we explore modifications of xRAG, where we incorporate fine-grained data samples during pre-training and operate on more granular data at generation. Our results show substantial gains: 23% improvement on downstream tasks, up to 8 BERTScore F1 points in response grounding, and 2.7× more entities preserved after compression. Still, we see limitations in the ability to integrate information across separate units. Our results show that this gap can be filled by soft prompting methods such as PISCO, which achieve the best trade-off between compression rate and effectiveness.

In summary, this paper provides the following contributions: (i) it reveals major limitations of state-of-the-art soft and hard prompt compression methods; (ii) it advocates for a holistic framework to evaluate prompt compression methods beyond downstream task performance; (iii) it presents promising directions to address the limitations of soft prompting methods, showing that manipulating the granularity of the compressed content is key to improving performance.

2 Related Work

Prompt compression is a technique to improve the efficiency of LLMs by creating an approximate representation of a prompt with reduced size (Wan et al., 2024; Wingate et al., 2022). Approaches

can be categorized into two main groups: *hard* prompting and *soft* prompting (Li et al., 2024a).

Hard Prompting compression methods focus on producing a shorter text version of the input prompt, with minimal token usage to retain the key information. These methods operate at various granularity levels, from omitting low-information instruction content and ICL demonstrations to refining token selection (Sennrich et al., 2016; Choi et al., 2024). Approaches include optimizing task definitions (Yin et al., 2023), dynamic compression allocation (Jiang et al., 2023b), syntax-guided compression (Yin et al., 2023) and demonstration selection via diversity-based sampling or relevance scoring (Yao et al., 2024; Gupta et al., 2023). Demonstration ordering can mitigate position bias (Jiang et al., 2024), while token classification ensures faithfulness to the original text (Pan et al., 2024a; Jiang et al., 2023b). However, hard prompting may disrupt grammar, introduce unfamiliar input distributions, and require re-encoding, impacting efficiency. Additionally, extreme compression is challenging, and compressing long inputs can be computationally expensive (Li et al., 2024a).

Soft Prompting methods compress a given text into continuous representations (Lester et al., 2021). Soft prompts can achieve higher compression rates (i.e., fewer input tokens for the LLMs), but fall short in explainability—the content encoded in the soft prompt is not directly interpretable by humans. Approaches like AutoCompressors recursively generate summary vectors (Chevalier et al., 2023), while Gisting condenses prompts into transformer activations using virtual gist tokens (Mu et al., 2023). ICAE (Ge et al., 2024) and 500xCompressor (Li et al., 2024b) encode long contexts into compact memory slots: ICAE leverages a LoRA-adapted encoder and 500xCompressor uses key-value representations for richer information retention. xRAG (Cheng et al., 2024) projects dense retrieval embeddings into the LLM’s representation space via a trainable modality bridge. It employs a frozen embedding model as the encoder, with a lightweight adapter between the encoder and the decoder LLM as the only trainable component. Unlike other soft prompting methods, xRAG keeps both the encoder and decoder frozen, making it a modular and efficient solution for rapid development. PISCO (Louis et al., 2025) is another compressor-decoder model relying on knowledge distillation from document-based questions. While

#	Dataset	Task	Input type	Output type	# samples
1.	HotpotQA (Yang et al., 2018)	Multi-hop QA	Multiple documents	Short-form answer	7,405 QA pairs (dev)
2.	QuAC (Choi et al., 2018)	Conversational QA	Wikipedia section	Spans from the text	863 QA pairs (validation)
3.	TriviaQA (Joshi et al., 2017)	RC with reasoning	Evidence documents	Short-form answer	5,743 QA pairs (validation)
4.	arXiv-sum. (Cohan et al., 2018)	Long-doc. summary	Scientific paper	Summary	6,440 articles
5.	GSM8K (Cobbe et al., 2021)	Math reasoning	Math problem	Answer with explanation	1,000 questions (test)

Table 1: Overview of the datasets used for evaluating prompt compression methods.

xRAG minimizes the Kullback-Leibler divergence between the logits of the encoder and target LLM, PISCO is trained using sequence-level knowledge distillation from a teacher generator.

Most prompt compression methods are evaluated on downstream tasks, including classification (Yin et al., 2023; Chevalier et al., 2023), natural language inference (Yao et al., 2024), reasoning (Chen et al., 2023), QA (Jiang et al., 2024; Cheng et al., 2024; Xu et al., 2024), and summarization (Pan et al., 2024a; Fei et al., 2024). However, downstream performance alone does not reveal a method’s limitations or assess information loss, a key issue in prompt compression. Reconstructing the original text from the compressed prompt has been proposed as a way to measure information preservation (Li, 2023; Wingate et al., 2022; Wang et al., 2024c), but standardized metrics for this evaluation are still lacking. This work goes beyond downstream performance by evaluating: (i) the grounding of the generated responses in the original input, and (ii) LLM’s ability to reconstruct the input context from its compressed version. This approach enables us to study the quantity and nature of information preserved by compression methods.

3 Prompt Compression Evaluation Framework

3.1 Benchmarking Data Design

Currently, the main benchmarking of prompt compression methods (Jiang et al., 2024) focuses on a limited range of tasks such as text classification (e.g., MNLI (Yao et al., 2024); SuperGLUE (Lester et al., 2021; Chevalier et al., 2023)), or extractive QA/RC (e.g., DROP, Gupta et al. 2023; RACE, Li et al. 2024b). Solving these tasks is often possible by only capturing a high-level topic representation or the main entities. We argue that a more comprehensive evaluation is needed to truly cover the properties of the compression methods. Therefore, we adopt generation tasks containing much longer contexts, such as long-form QA based on multiple documents, long document summarization, and con-

versational search, with retrieval-augmented generation (RAG) based on multiple source documents. Table 1 summarizes the target tasks characteristics. The list is not exhaustive and should evolve with advances in compression methods and LLMs.¹

3.2 Compression Quality Evaluation

We outline the following key dimensions for measuring the quality of the compression methods: (i) downstream task performance, (ii) response grounding, and (iii) information preservation.

Downstream Task Performance Following previous work, we use BERTScore (F1, Zhang et al. (2020)) for summarization and long-form QA, and Exact Match (EM) for reasoning and short-form QA (more details can be found in Appendix B.1).

Grounding Grounding is another important dimension of compression quality, providing insights into a method’s ability to preserve key information from the context to be used in the generated responses (Kim et al., 2024). We explored different approaches to automatically evaluate grounding, to find a generalizable metric across different long-context tasks. We selected FABLES (Kim et al., 2024) as it produced results better aligned with human evaluation (see the discussion in Appendix B.2). FABLES first extracts a set of decontextualized claims from the generated response and then rates the faithfulness of each claim given the evidence. Both steps are performed by prompting an LLM (Claude 3 Haiku). For each claim, we derive a score by comparing each 10-sentence context chunk against the claim and then taking the maximum score across chunks. The final score is the average across all claims in the response.

Information Preservation A key factor indicating the success of compression is the amount of main factual claims preserved from the original context. To evaluate the capability of prompt compression methods to provide the target LLM with access to key information from the text, we look at

¹See Fig. 3 in Appendix C.2 for context length distribution.

Method	HotpotQA (EM)	HotpotQA* (EM)	arXiv-sum. (F1)	QuAC (F1)	TriviaQA (EM)	GSM8K (EM)
Mistral-7B	0.664	0.772	0.834	0.869	0.773	0.477
Mistral-7B (no cont.)	0.276 (-58%)	0.276 (-64%)	—	0.834 (-4%)	0.590 (-24%)	0.440 (-1%)
xRAG	0.297 (-55%)	0.374 (-52%)	0.803 (-4%)	0.838 (-4%)	0.691 (-11%)	0.336 (-30%)
PISCO	0.318 (-52%)	0.589 (-24%)	0.818 (-2%)	0.861 (-1%)	0.738 (-5%)	0.393 (-18%)
LLMLingua	0.306 (-54%)	0.696 (-10%)	0.805 (-4%)	0.846 (-3%)	0.727 (-6%)	0.305 (-36%)

Table 2: Model performance on long-context datasets. Percentages in brackets show the relative **drop** from the Mistral-7B baseline, calculated as $(score - mistral_score) / mistral_score$. “No context” (*no cont.*) for GSM8K means removing ICL demos, while for arXiv-sum, the context is the document itself, making this setup not applicable.

the content from the original text that is preserved after compression. This evaluation is particularly interesting for soft prompting methods since compressed tokens are not interpretable by design. To capture the information preserved from the original text, we prompt the target LLM to reconstruct the content encoded in the soft prompt tokens. Then, the reconstructed text is compared with the original one using similarity metrics like ROUGE (Lin, 2004) or BERTScore (Zhang et al., 2020).

Compression Rates The last dimension that our framework monitors is the *compression rate* – the ratio between the size of the compressed and the original prompt. The computational cost of achieving compression is another important factor to be considered when comparing different methods.

4 Prompt Compression Methods Analysis

For our experiments, we select two soft prompting and one hard prompting method, using the same target LLM – Mistral-7B Instruct v0.2 (Jiang et al., 2023a) – across all setups, with the no-compression case as an upper bound.

Soft Prompting We adopt xRAG (Cheng et al., 2024) and PISCO (Louis et al., 2025) as our soft prompting baselines. Both methods were shown to work in long context scenarios, making them the most promising starting point for our research. The input text in xRAG is encoded as a single token passed to the target LLM. This representation is obtained from an encoder model, followed by a modality bridge – the only trainable component – mapping the encoder’s representation to the target LLM’s embedding space. PISCO is a successor of xRAG containing two adapters around a backbone LLM: 1) an encoder adapter trained to compress input contexts into a set of embedding vectors, and 2) a decoder adapter providing an answer based on document embedding vectors and a query.

Hard Prompting We adopt LLMLingua (Jiang et al., 2023b) as our hard prompting baseline. It dynamically allocates compression ratios to different prompt components (using a budget controller) while preserving semantic integrity. It then applies a token-level iterative pruning algorithm for fine-grained compression that accounts for conditional dependencies.² Following the LLMLingua setup (Jiang et al., 2023b), the compressed prompt is provided as textual input to the target LLM.

4.1 Downstream Task Performance

We first evaluate the out-of-the-box compression methods on five diverse tasks (see Section 3.1).³ Table 2 compares the performance of Mistral-7B with/without input compression. We see that fully removing the context, such as background for QA or ICL examples for GSM8K, leads to large performance drops, confirming that the model’s parametric memory alone is insufficient for these tasks.

On this set of complex tasks, all compression methods result in sizeable performance losses (3% to 55% relative difference). The highest performance drop is observed on HotpotQA, as it requires multi-hop reasoning and information aggregating. This indicates that applying compression removes crucial pieces of information needed to derive the correct answer. If we simplify the task, by retaining only the relevant paragraphs instead of the whole context (*HotpotQA**), we see an improvement of 8 points for xRAG and 39 for LLMLingua, indicating their difficulty with handling long, noisy contexts.

On the long document summarization tasks (*arXiv-sum.*), differences between the base Mistral and the compression methods are smaller, with only a 4% relative difference in F1 BERTScore. We attribute this to the nature of the task, i.e., summarizing very long texts (~5800 words) into concise abstracts (~200 words) requires focusing on high-

²The compression budget of LLMLingua is 350 tokens.

³A reproducibility study confirmed that our results align with those reported in the original papers (see Appendix D).

Method	Encodings	HotpotQA	HotpotQA*	TriviaQA
xRAG	1 per context	0.297	0.374	0.691
	1 per paragraph	0.211 (-30%)	0.400 (+7%)	—
	1 per sentence	0.055 (-81%)	0.264 (-29%)	0.224 (-68%)
PISCO	8 per context	0.318	0.589	0.738
	8 per paragraph	0.512 (+61%)	0.625 (+6%)	—
	8 per sentence	0.399 (+25%)	0.559 (-5%)	0.719 (-3%)

Table 3: Exact match scores for xRAG and PISCO responses with varying compression granularity.

level information rather than details.

QuAC and *TriviaQA* tasks are less dependent on the details from the context. On these datasets, we see a smaller gap in performance between the compressed and uncompressed inputs (5 EM points for LLMingua and 8 points for xRAG). On the math reasoning task (*GSM8K*), we find that both prompt compression methods—in this case compressing ICL examples—yield substantially worse performance than providing no demonstrations at all. This means that compressed ICL demonstrations are not well interpreted by the model, which makes the response generation even more difficult.

To assess the impact of the compression input granularity, we encode the input on context-, paragraph-, and sentence-level using xRAG tokens. We also run similar experiments with PISCO using separate sets of 8 embedding vectors per unit. Results presented in Table 3 indicate that compressing context into smaller segments does not improve performance for xRAG, likely because the tokens capture high-level topics without key details. However, PISCO trained with sequence-level knowledge distillation objective shows significant improvements on HotpotQA that uses longer contexts compared to the remaining two datasets when operating on smaller granularity. This suggests that enhancing the model to handle finer details is a worthwhile direction to explore for soft compression methods.

4.2 Grounding

We evaluate how prompt compression affects the faithfulness of LLM-generated text. Depending on the task, we have different grounding texts: source documents for summarization, and background sections for QA and RC. We compare grounding scores for responses with and without compression to assess its impact. Results in Table 4 show that compression leads to a 30 points drop in groundedness score on HotpotQA, and around 50 points on QuAC and arXiv-summarization. This indicates that higher compression rates used in xRAG re-

Method	HotpotQA	HotpotQA*	arXiv-sum.	QuAC	TriviaQA	GSM8K
Mistral-7B	0.80	0.75	0.97	0.93	0.78	0.50
xRAG	0.52	0.57	0.39	0.45	0.73	0.42
PISCO	0.59	0.76	0.74	0.63	0.84	0.48
LLMLingua	0.45	0.75	0.62	0.49	0.72	0.44

Table 4: FABLES grounding scores for responses generated with different methods, averaged over 5 random sets of 100 samples (stdev (σ) < 0.04).

Method	Data	Encodings	Sample Lengths				Preserved Entities
			All	1-sent	5-sents	10-sents	
xRAG	Unseen	1 per sample	0.66	0.57	0.72	0.70	0.28
		1 per sent.	0.42	0.57	0.38	0.31	0.19
	Seen	1 per sample	0.65	0.50	0.73	0.73	0.25
		1 per sent.	0.35	0.50	0.33	0.21	0.12
PISCO	Unseen	8 per sample	0.89	0.91	0.89	0.87	0.49
		8 per sent.	0.90	0.90	0.90	0.89	0.59

Table 5: Information preservation results: BERTScore F1 between original and reconstructed context for different context lengths, and fraction of preserved entities.

sult in generating text that is less faithful to the context. The hallucinated content appears in the entire response/summary, including the first claim of the generated text, which intuitively contains the most important information (see Table 16 in Appendix E.2). As expected, responses generated with LLMingua are less prone to hallucinations as the compressed input retains direct information from the original context. PISCO produces the most faithful responses to the source despite high compression rates, likely due to the use of sequence-level knowledge distillation from a teacher model.

4.3 Information Preservation

We reconstruct the original text from the compressed xRAG representation by prompting the target LLM to recreate the information encoded in the tokens. Using a subset of xRAG pre-training prompts (Table 21 in Appendix F), we apply two scenarios: (i) encoding the entire context in one xRAG token, and (ii) encoding each context sentence into separate tokens. Similarly, for PISCO, we prompt the decoder to reconstruct the provided information, encoded on sample- or sentence-level, restoring as many details as possible. We hypothesize that compressing entire contexts into a single token risks significant information loss for longer, more complex inputs. Our evaluation of information preservation covers only soft prompting methods, not LLMingua, as hard-prompting methods do not add new content and are fully interpretable.

We evaluate xRAG on two datasets: “seen” data—random samples from xRAG’s pre-training set, and “unseen” data—samples from HotpotQA, QuAC, and

xRAG variant	HotpotQA (EM)		HotpotQA* (EM)		arXiv-sum. (F1)		TriviaQA (EM)		QuAC (F1)	
	Cont.	Sent.	Cont.	Sent.	Cont.	Sent.	Cont.	Sent.	Cont.	Sent.
xRAG (Reproduced)	0.286	0.014	0.375	0.174	0.775	0.760	0.696	0.115	0.829	0.843
w/ Sentence PT + FT	0.313 (9%)	0.066 (371%)	0.422 (13%)	0.423 (143%)	0.803 (3%)	0.739 (3%)	0.712 (2%)	0.361 (214%)	0.833 (0%)	0.855 (1%)
w/ Two-Step PT + FT	0.277 (-3%)	0.145 (936%)	0.406 (8%)	0.462 (166%)	0.785 (1%)	0.696 (8%)	0.685 (-2%)	0.521 (353%)	0.834 (1%)	0.823 (-2%)

Table 6: Performance of the xRAG variants at different compression granularities, calculated on a subset of 1,000 examples, uniformly sampled from each dataset. Relative **improvement/drop** is calculated wrt the reproduced xRAG.

Model	Data	Encodings	Avg.	PERSON	GPE	DATE	CARD.	ORG
xRAG	Unseen	1 per sample	0.28	0.31	0.39	0.22	0.26	0.32
		1 per sentence	0.19	0.21	0.25	0.14	0.13	0.24
	Seen	1 per sample	0.25	0.16	0.38	0.24	0.38	0.21
		1 per sentence	0.12	0.06	0.18	0.11	0.19	0.14
PISCO	Unseen	1 per sample	0.49	0.56	0.58	0.39	0.42	0.50
		1 per sent.	0.60	0.65	0.71	0.51	0.49	0.63

Table 7: Fraction of entities preserved in the reconstructed context using xRAG and PISCO models.

TriviaQA. Each dataset contains 450 examples, split into three 150-example sets with 1, 5, and 10 sentences per example, respectively. Our primary metric is BERTScore, which can identify meaning-preserving paraphrases as accurate reconstructions. Table 5 indicates that the xRAG method is far from being able to reconstruct the original context (average BERTScore F1 is 0.66). On average the performance on *seen* and *unseen* examples is similar, suggesting that xRAG is learning a general representation rather than memorizing its pre-training data. Additionally, the target LLM is not able to handle more than one xRAG token, and the reconstruction scores drop by 20-30 BERTScore F1 points when using one token per sentence.

Higher scores for examples containing multiple sentences likely come from the fact that xRAG pre-training data contains longer samples and the model is not able to preserve more granular pieces of information (see Figure 1). The qualitative analysis shows that xRAG tokens primarily capture the general topic of the compressed content (see Table 24 in Appendix E) but fail to retain key details. This pattern is further supported by our entity preservation experiments, where we measure the fraction of text entities from the input retained in the reconstructed version (last column in Table 5).

The fraction of entities’ types preserved in the reconstructed context per entity type is presented in Table 7. We observe particularly low scores for *dates* and *numerical* values. Additionally, entity preservation for *people* is notably poor on in-domain samples, likely due to noise or ambiguous references in the pre-training data (e.g., “*Thomas*

Method	MFLOPs
xRAG	3.7×10^7
PISCO	1.3×10^7
LLMLingua	6.6×10^6

Table 8: Computational cost of different prompt compression methods measured in floating-point operations (FLOPs). Values are reported in MFLOPs, representing the total number of FLOPs required for processing a single input sample.

Scott 1806–1816” annotated as *person*). In contrast, *geographical locations* are the most consistently preserved across all input types. Overall, our findings indicate that the model struggles with maintaining entities during prompt compression.

PISCO results are reported only for xRAG “*unseen*” data for fair comparison and show significantly higher information preservation—both in terms of semantic similarity between the reconstructed and original text, and the percentage of preserved entities. Notably, PISCO maintains a similar level of information retention across different encoding levels, likely due to its higher encoding capacity: while xRAG compresses information into a single token, PISCO uses 8 embeddings.

4.4 Compression Computational Efficiency

We evaluate the computational cost of various prompt compression methods by measuring the total number of floating-point operations (FLOPs) required to compress a standard input sample (context documents for a randomly selected HotpotQA dataset sample). The results, summarized in Table 8, provide a comparison of the efficiency of each compression method, highlighting trade-offs between compression strategy and computational demand. Dominating operation remains the number of matrix multiplication. The results show that all compression methods used in our experiments are comparable in terms of efficiency.

5 Improving xRAG

The results presented in Section 4 show a large disproportion between PISCO and the remaining

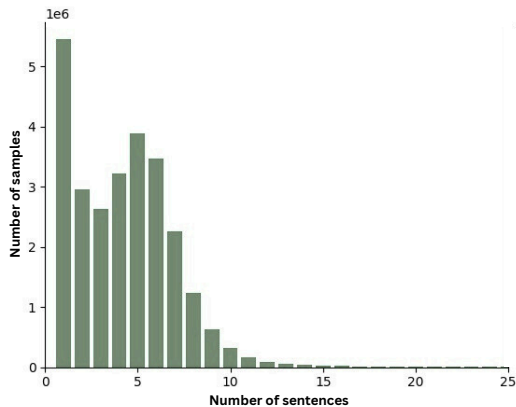


Figure 1: Distribution of the lengths of the examples measured in terms of number of sentences per training sample in pre-training data. The training partition contains 26,541,264 samples overall. The longest sample contains 137 sentences.

two prompt compression methods. In particular, we highlight xRAG’s inability to preserve detailed information from the original input in long-context scenarios. We here explore directions to mitigate these limitations of xRAG, while still retaining the advantage of high compression rates. We use our prompt compression evaluation framework to verify whether the modifications lead to improvements over the original xRAG method. We re-train an xRAG model using the data and code released with its paper to ensure that the models are fully comparable (details in Appendix D). We denote this reproduced version as *xRAG (Reproduced)*.

5.1 Sentence Pre-Training and Sentence-Level Fine-Tuning (*w/ Sentence PT + FT*)

The only trainable component of the xRAG is a bridge model, whose main role is to align two embedding spaces – the one from the encoder (dense retrieval) model and the one from the target LLM. This function is learned via unsupervised pre-training on a reconstruction task, and then further fine-tuned on a set of target downstream tasks.

In xRAG, the pre-training (PT) samples are encoded into single xRAG tokens, regardless of their length. Since the majority of the examples are longer than one sentence (see Figure 1), the bridge model fails to transform the representations of texts with smaller granularity (see Table 3) and thus to capture details beyond the topic of the samples. To address this limitation, we use single-sentence samples in pre-training to ensure that the model can effectively handle these basic cases. We use the

Method	xRAG	HotpotQA	HotpotQA*	arXiv-sum.	QuAC	TriviaQA
xRAG (Reproduced)		0.50	0.58	0.38	0.43	0.72
w/ Sentence PT + FT	1 per cont.	0.52	0.62	0.44	0.49	0.73
w/ Two-Step PT + FT		0.40	0.45	0.35	0.42	0.68
xRAG (Reproduced)		0.31	0.47	0.26	0.47	0.71
w/ Sentence PT + FT	1 per sent.	0.34	0.52	0.46	0.42	0.65
w/ Two-Step PT + FT		0.23	0.58	0.30	0.39	0.55

Table 9: Grounding scores w.r.t. the context, averaged across 5 random sets of 100 examples ($\sigma < 0.1$), calculated over the non-empty outputs (see Appendix E.2).

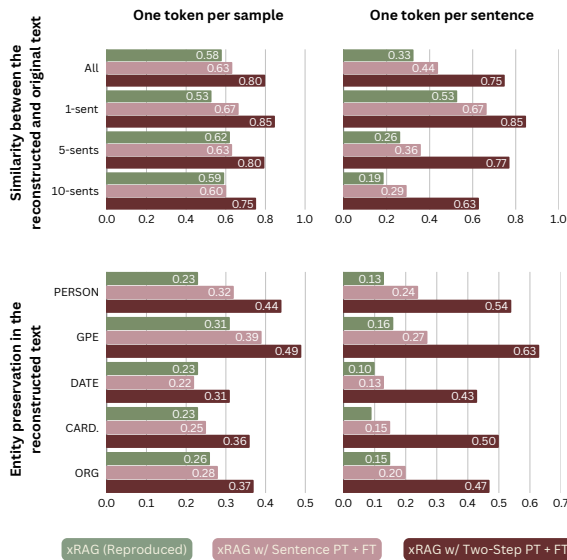


Figure 2: Information preservation results for xRAG variants. Similarity is measured with BERTScore between the original and reconstructed text. Entity preservation is based on EM of entities in the reconstruction.

same pre-training data, but consider only the first sentence from each sample. In the fine-tuning (FT) phase, original xRAG chunks each context into texts of 180 tokens with each chunk encoded into one xRAG token. Instead, we replace token-based chunking with sentence-based segmentation with each sentence encoded into a separate xRAG token to preserve information continuity. We denote this variant of xRAG as *xRAG w/ Sentence PT + FT*.

On the downstream evaluation, *xRAG w/ Sentence PT + FT* brings sizable improvements on all tasks (Table 6). With context-level (*Cont.*) compression we see 1-5 points absolute gains on all tasks (up to 13% relative), suggesting that this method allows to capture the information better. This is further supported by the increase in grounding of the generated text (Table 9). We also see that the model is able to handle sentence-level compression better (*Sent.*) – especially on HotpotQA* (downstream performance increases from 0.174 to 0.423) and TriviaQA (from 0.115 to 0.361). Interestingly, sentence-level representations are still not

performing as well as the context-level ones.

Finally, in the context reconstruction experiments, we see improved performance when encoding at sentence level.⁴ The *xRAG w/ Sentence PT + FT* model is able to preserve more entities on average—increase from 13% to 20% (see Figure 2). However, in absolute terms, the numbers still remain low (preserving between 13%-39% entities). In all three dimensions, the performance on sentence-level tokens is lower than when encoding the entire context at once, implying that the model is not able to handle multiple tokens at a time.

5.2 Two-Step Pre-Training and Sentence-Level Fine-Tuning (*w/ Two-Step PT + FT*)

Even though we see improvements in downstream performance and grounding with the *xRAG w/ Sentence PT + FT* model (§ 5.1), the sentence-level encoding is still falling behind. We hypothesize that the main reason is that the model is not able to handle information spread across multiple tokens.

To verify and refine this hypothesis, we introduce an additional pre-training step that focuses on helping the model reason about information from multiple xRAG tokens. We adopt a two-stage pre-training procedure with these steps: (i) encoding one sentence per pre-training sample; (ii) chunking samples into sentences with each sentence encoded separately. The fine-tuning stage is the same as in *xRAG w/ Sentence PT + FT* (chunking samples into sentences with one token per sentence). We denote this variant as *xRAG w/ Two-Step PT + FT*.

Results on downstream tasks show that prompt compression methods can further benefit from sentence-level encoding, particularly for datasets with focused, concise contexts like HotpotQA*, where low-level facts are needed for reasoning (Table 6). We see 23% relative improvement over the baseline (in EM, from 0.38 for *xRAG (Reproduced) Cont.* to 0.47 for *xRAG w/ Two-Step PT + FT Sent.*). We generally observe consistent improvements over the baseline with this model when the information is encoded at sentence-level (*Sent.* columns). Still, the model cannot fully process complex, noisy contexts, as evident in results for HotpotQA and TriviaQA, where the performance with one token per context drops (*Cont.* columns).

Even though we observe lower grounding for multi-token encoded texts, the reconstruction experiments show that our two-step pre-training on

data with different granularity results in significant improvements in terms of information preservation. Crucially, this effect is visible independently of the amount of encoding tokens used. On average, we can reconstruct 50% of entities from the original text encoded into multiple tokens (compared to 13% for the *xRAG (Reproduced)*) (Figure 2). While cardinal entities remain the most challenging ones to preserve, the two-step model retains significantly more of them (0.36 vs. 0.23 at context-level and 0.50 vs. 0.09 at sentence-level). Moreover, we observe even higher similarity scores for text encoded into multiple tokens (F1 of 0.63 vs. 0.19), showing that our strategy captures finer-grained information.

5.3 Qualitative Analysis

To understand the type of information preserved, we conducted a qualitative analysis of reconstructed examples, examining 10 samples of varying lengths generated using 20 different prompts. Our investigation focused on prompts from the xRAG paraphrase training phase. The analysis confirmed our hypothesis that the xRAG tokens primarily capture the general topic of the compressed content but fail to retain specific details. This behavior is expected, as xRAG relies on a dense retrieval model trained for document similarity. Depending on the topic, the LLM can reconstruct the main entity described in the text, aligning with its strong performance on the TriviaQA dataset. However, numerical entities such as dates and numbers are often lost. Examples of reconstructed contexts can be found in Table 24 in Appendix E.

5.4 Compression Rates

Table 10 presents the compression rates for each model (measured in tokens and including the system instructions, context, etc.). PISCO achieves very high compression rates without performance loss in any of the aspects investigated with our evaluation framework, establishing it as the state-of-the-art for prompt compression in long-context scenarios. The sentence-level xRAG retains a high compression rate with a 4x-12x reduction in the input size and a 2x higher compression rate than LLMingua. The compression rate depends on the lengths of the inputs but the compression granularity is a controllable parameter of the method.

6 Lessons Learned and Future Directions

So far, we have seen that many compression techniques fail to preserve detailed information, espe-

⁴Reconstructed samples are in Table 24 in Appendix E.

Method	HotpotQA	HotpotQA*	arXiv-sum.	QuAC	TriviaQA	GSM8K
Mistral-7B	1515	341	6424	995	840	1135
Mistral-7B (no cont.)	44	44	—	162	40	109
xRAG (1 per cont.)	65 (23.3)	65 (5.2)	27 (237.9)	186 (5.3)	61 (13.8)	130 (8.7)
xRAG (1 per sent.)	148 (10.2)	80 (4.3)	525 (12.2)	234 (4.3)	103 (8.2)	172 (6.6)
PISCO	73 (20.8)	73 (4.7)	35 (183.5)	194 (5.1)	69 (12.2)	138 (8.2)
LLMLingua	388 (3.9)	312 (1.1)	329 (19.5)	398 (2.5)	362 (2.3)	417 (2.7)

Table 10: Average prompt lengths (in tokens) after compression. The compression rate is shown in brackets. See Table 8 for compression computational efficiency.

cially in long-context scenarios and at high compression rates, leading to drops in performance and grounding. In line with this, context reconstruction experiments indicate that xRAG tokens primarily capture the general topic of the compressed content but fail to retain essential details for a task (numbers, dates, names, etc.). We propose modifications of xRAG and provide evidence that manipulating information granularity during training improves information retention in soft prompts, which translates to improved performance and grounding in downstream tasks. Notably, improvements occur regardless of the number of encoding tokens used.

Multi-Token Context Understanding Given the complexity and information density of text in information-intensive tasks, an effective alternative to encoding everything into a single soft prompt token is needed (Mu et al., 2023; Li et al., 2024b). Our experiments suggest that while increasing text granularity is promising, reasoning across information stored independently in soft prompt tokens remains challenging. PISCO addresses this by using sequence-level distillation, rather than token-level (Louis et al., 2025). Our results confirm that the training with multi-token samples implemented in PISCO significantly improves grounding and information preservation and is crucial for the ability to integrate information across separate units (Cheng et al., 2024).

Context-Aware Adaptive Compression Multi-token soft prompting is an important direction for compression methods. However, a task-independent *one-size-fits-all* strategy does not capture the fact that every input comes with different complexity and level of detail required for the target task (Chevalier et al., 2023; Mu et al., 2023; Jiang et al., 2023b; Pan et al., 2024b). Moreover, despite the advancements in prompt compression methods, grounding and information retention remain suboptimal, especially when encoding with multiple tokens. Dynamically adjusting the compression rate and the compression strategy can

improve contextual understanding on information spread in several tokens (Nagle et al., 2024).

Task-Specific Embeddings One of the main advantages of xRAG-like methods is the fact that context encoding can be done with a different model than the target LLM. This model can be smaller and more efficient, or larger and more capable than the target LLM. xRAG’s reliance on a dense retrieval model trained for document similarity explains its tendency to only capture high-level topic information while overlooking specific details. A promising direction is exploring alternative models, such as instruction-tuned embeddings (Su et al., 2023; BehnamGhader et al., 2024; Wang et al., 2024b,a), which can provide task-specific representations (conditioned on a set of instructions) and could better preserve entity-level information.

Hybrid Soft-Hard Prompting Preserving details, such as named entities, in compressed representations is crucial for improving performance on tasks that require reasoning. Hard compression (Li et al., 2023; Jiang et al., 2023b; Chuang et al., 2024) is inherently explainable and can explicitly retain these details. In contrast, soft prompting has limited representational capacity, making it challenging to encode certain information, such as cardinals, in highly compressed embeddings (Wallace et al., 2019; Jiang et al., 2020; Epure and Hennequin, 2022). A promising approach combines both: hard prompts preserve key details (e.g., numbers, names), while the rest of the context is encoded via soft prompts, reducing information loss during the compression.

7 Conclusions

In this paper, we propose a framework for assessing the quality of prompt compression methods on long-context generation tasks. We shed light on both the capabilities and limitations of the state-of-the-art hard (LLMLingua) and soft (xRAG and PISCO) compression approaches. Our findings reveal that some existing compression techniques fail to preserve detailed information, especially in long-context scenarios. We explore several directions to improve the xRAG soft compression method, showing an increase in model performance – up to +23% on downstream tasks, up to 8 BERTScore points in grounding, and preserving 2.7x more entities.⁵

⁵The code repository: <https://github.com/amazon-science/information-preservation-in-prompt-compression>.

Limitations

Our study has several limitations that should be considered when interpreting the results. First, all experiments are conducted using Mistral-7B as the target LLM. While this allows for consistent evaluation, exploring different model sizes and architectures is beyond the scope of this work and remains an area for future research. Second, our analysis focuses exclusively on English benchmarks. Findings may not generalize to multilingual settings, where language-specific characteristics could influence the effectiveness of prompt compression methods. We believe that our findings will not change across languages, as we outline fundamental issues with the compression frameworks. If anything, we expect the soft approaches to be even more shallow in capturing details in other languages. Third, we evaluate only one representative baseline from each category of prompt compression techniques. While this provides an initial comparison, a broader investigation using our evaluation framework is needed to fully understand the trade-offs across different methods. Finally, although we examine multiple generation tasks, there is still room for further exploration, particularly in long-form QA scenarios, where prompt compression may have a more significant impact on model performance.

References

- Marah Abdin, Jyoti Aneja, Harkirat Singh Behl, Sébastien Bubeck, Ronen Eldan, Suriya Gunasekar, Michael Harrison, Russell J. Hewett, Mojan Javaheripi, Piero Kauffmann, James R. Lee, Yin Tat Lee, Yuanzhi Li, Weishung Liu, Caio César Teodoro Mendes, Anh Nguyen, Eric Price, Gustavo de Rosa, Olli Saarikivi, and 8 others. 2024. [Phi-4 technical report](#). *ArXiv preprint*, abs/2412.08905.
- Marah Abdin, Jyoti Aneja, Sebastien Bubeck, Caio César Teodoro Mendes, Weizhu Chen, Allie Del Giorno, Ronen Eldan, Sivakanth Gopi, Suriya Gunasekar, Mojan Javaheripi, Piero Kauffmann, Yin Tat Lee, Yuanzhi Li, Anh Nguyen, Gustavo de Rosa, Olli Saarikivi, Adil Salim, Shital Shah, Michael Santacrose, and 7 others. 2023. [Phi-2: The surprising power of small language models](#). *Microsoft Research Blog*.
- . Amazon AGI. 2024. [The amazon nova family of models: Technical report and model card](#). *Amazon Technical Reports*.
- Anthropic. 2023. [Long context prompting for claude 2.1](#).
- Parishad BehnamGhader, Vaibhav Adlakha, Marius Mosbach, Dzmitry Bahdanau, Nicolas Chapados, and Siva Reddy. 2024. [LLM2vec: Large language models are secretly powerful text encoders](#). In *First Conference on Language Modeling*.
- Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, and 12 others. 2020. [Language models are few-shot learners](#). In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, virtual*.
- Kaiyan Chang, Songcheng Xu, Chenglong Wang, Yingfeng Luo, Tong Xiao, and Jingbo Zhu. 2024. [Efficient prompting methods for large language models: A survey](#). *ArXiv preprint*, abs/2404.01077.
- Jiuhai Chen, Lichang Chen, Chen Zhu, and Tianyi Zhou. 2023. [How many demonstrations do you need for in-context learning?](#) In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 11149–11159, Singapore.
- Xin Cheng, Xun Wang, Xingxing Zhang, Tao Ge, Si-Qing Chen, Furu Wei, Huishuai Zhang, and Dongyan Zhao. 2024. [xRAG: Extreme context compression for retrieval-augmented generation with one token](#). In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.
- Alexis Chevalier, Alexander Wettig, Anirudh Ajith, and Danqi Chen. 2023. [Adapting language models to compress contexts](#). In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 3829–3846, Singapore.
- Eunseong Choi, Sunkyung Lee, Minjin Choi, Jun Park, and Jongwuk Lee. 2024. [From reading to compressing: Exploring the multi-document reader for prompt compression](#). In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 14734–14754, Miami, Florida, USA. Association for Computational Linguistics.
- Eunsol Choi, He He, Mohit Iyyer, Mark Yatskar, Wentau Yih, Yejin Choi, Percy Liang, and Luke Zettlemoyer. 2018. [QuAC: Question answering in context](#). In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2174–2184, Brussels, Belgium.
- Yu-Neng Chuang, Tianwei Xing, Chia-Yuan Chang, Zirui Liu, Xun Chen, and Xia Hu. 2024. [Learning to compress prompt in natural language formats](#). In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 7756–7767, Mexico City, Mexico.

- Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. [Training verifiers to solve math word problems](#). *ArXiv preprint*, abs/2110.14168.
- Arman Cohan, Franck Dernoncourt, Doo Soon Kim, Trung Bui, Seokhwan Kim, Walter Chang, and Nazli Goharian. 2018. [A discourse-aware attention model for abstractive summarization of long documents](#). In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 615–621, New Orleans, Louisiana.
- Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, Anirudh Goyal, Anthony S. Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, and 510 others. 2024. [The llama 3 herd of models](#). *ArXiv preprint*, abs/2407.21783.
- Elena V. Epure and Romain Hennequin. 2022. [Probing pre-trained auto-regressive language models for named entity typing and recognition](#). In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 1408–1417, Marseille, France. European Language Resources Association.
- Shahul Es, Jithin James, Luis Espinosa Anke, and Steven Schockaert. 2024. [RAGAs: Automated evaluation of retrieval augmented generation](#). In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 150–158, St. Julians, Malta.
- Weizhi Fei, Xueyan Niu, Pingyi Zhou, Lu Hou, Bo Bai, Lei Deng, and Wei Han. 2024. [Extending context window of large language models via semantic compression](#). In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 5169–5181.
- Tao Ge, Hu Jing, Lei Wang, Xun Wang, Si-Qing Chen, and Furu Wei. 2024. [In-context autoencoder for context compression in a large language model](#). In *The Twelfth International Conference on Learning Representations*, ICLR '24, Vienna, Austria.
- . Gemini Team. 2024. [Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context](#). *ArXiv preprint*, abs/2403.05530.
- Shivanshu Gupta, Matt Gardner, and Sameer Singh. 2023. [Coverage-based example selection for in-context learning](#). In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 13924–13950, Singapore.
- Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, L elio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timoth ee Lacroix, and William El Sayed. 2023a. [Mistral 7b](#). *ArXiv preprint*, abs/2310.06825.
- Huiqiang Jiang, Qianhui Wu, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. 2023b. [LLMLingua: Compressing prompts for accelerated inference of large language models](#). In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 13358–13376, Singapore.
- 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.
- Zhengbao Jiang, Frank F. Xu, Jun Araki, and Graham Neubig. 2020. [How can we know what language models know?](#) *Transactions of the Association for Computational Linguistics*, 8:423–438.
- Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. [TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension](#). In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, Vancouver, Canada.
- Yekyung Kim, Yapei Chang, Marzena Karpinska, Aparna Garimella, Varun Manjunatha, Kyle Lo, Tanya Goyal, and Mohit Iyyer. 2024. [FABLES: Evaluating faithfulness and content selection in book-length summarization](#). In *First Conference on Language Modeling*.
- Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. [The power of scale for parameter-efficient prompt tuning](#). In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3045–3059, Online and Punta Cana, Dominican Republic.
- Quentin Lhoest, Albert Villanova del Moral, Yacine Jernite, Abhishek Thakur, Patrick von Platen, Suraj Patil, Julien Chaumond, Mariama Drame, Julien Plu, Lewis Tunstall, Joe Davison, Mario  aško, Gunjan Chhablani, Bhavitvya Malik, Simon Brandeis, Teven Le Scao, Victor Sanh, Canwen Xu, Nicolas Patry, and 13 others. 2021. [Datasets: A community library for natural language processing](#). In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 175–184, Online and Punta Cana, Dominican Republic.
- Yucheng Li. 2023. [Unlocking context constraints of llms: Enhancing context efficiency of llms with self-information-based content filtering](#). *ArXiv preprint*, abs/2304.12102.

- Yucheng Li, Bo Dong, Frank Guerin, and Chenghua Lin. 2023. [Compressing context to enhance inference efficiency of large language models](#). In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 6342–6353, Singapore.
- Zongqian Li, Yinhong Liu, Yixuan Su, and Nigel Collier. 2024a. [Prompt compression for large language models: A survey](#). *ArXiv preprint*, abs/2410.12388.
- Zongqian Li, Yixuan Su, and Nigel Collier. 2024b. [500xcompressor: Generalized prompt compression for large language models](#). *ArXiv preprint*, abs/2408.03094.
- Chin-Yew Lin. 2004. [ROUGE: A package for automatic evaluation of summaries](#). In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain.
- Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2024. [Lost in the middle: How language models use long contexts](#). *Transactions of the Association for Computational Linguistics*, 12:157–173.
- Maxime Louis, Hervé Déjean, and Stéphane Clinchant. 2025. [Pisco: Pretty simple compression for retrieval-augmented generation](#). *Preprint*, arXiv:2501.16075.
- Jesse Mu, Xiang Li, and Noah D. Goodman. 2023. [Learning to compress prompts with gist tokens](#). In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, Louisiana, USA*.
- Tsendsuren Munkhdalai, Manaal Faruqui, and Sidharth Gopal. 2024. [Leave no context behind: Efficient infinite context transformers with infinite attention](#). *ArXiv preprint*, abs/2404.07143.
- Alliot Nagle, Adway Girish, Marco Bondaschi, Michael Gastpar, Ashok Vardhan Makkuva, and Hyeji Kim. 2024. [Fundamental limits of prompt compression: A rate-distortion framework for black-box language models](#). In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.
- 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. 2024a. [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.
- 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. 2024b. [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.
- Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, and 2 others. 2019. [Pytorch: An imperative style, high-performance deep learning library](#). In *Advances in Neural Information Processing Systems*, volume 32.
- Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. [Neural machine translation of rare words with subword units](#). In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany.
- Freda Shi, Xinyun Chen, Kanishka Misra, Nathan Scales, David Dohan, Ed H. Chi, Nathanael Schärli, and Denny Zhou. 2023. [Large language models can be easily distracted by irrelevant context](#). In *International Conference on Machine Learning*, volume 202 of *ICML '23*, pages 31210–31227, Honolulu, Hawaii, USA.
- Hongjin Su, Weijia Shi, Jungo Kasai, Yizhong Wang, Yushi Hu, Mari Ostendorf, Wen-tau Yih, Noah A. Smith, Luke Zettlemoyer, and Tao Yu. 2023. [One embedder, any task: Instruction-finetuned text embeddings](#). In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 1102–1121, Toronto, Canada.
- Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurélien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. [Llama: Open and efficient foundation language models](#). *ArXiv preprint*, abs/2302.13971.
- Eric Wallace, Yizhong Wang, Sujian Li, Sameer Singh, and Matt Gardner. 2019. [Do NLP models know numbers? probing numeracy in embeddings](#). In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5307–5315, Hong Kong, China.
- Zhongwei Wan, Xin Wang, Che Liu, Samiul Alam, Yu Zheng, Jiachen Liu, Zhongnan Qu, Shen Yan, Yi Zhu, Quanlu Zhang, Mosharaf Chowdhury, and Mi Zhang. 2024. [Efficient large language models: A survey](#). *Transactions on Machine Learning Research*.
- Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. 2024a. [Improving text embeddings with large language models](#). In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11897–11916, Bangkok, Thailand.

- Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. 2024b. [Multilingual e5 text embeddings: A technical report](#). *ArXiv preprint*, abs/2402.05672.
- Xiangfeng Wang, Zaiyi Chen, Tong Xu, Zheyong Xie, Yongyi He, and Enhong Chen. 2024c. [In-context former: Lightning-fast compressing context for large language model](#). In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 2445–2460, Miami, Florida, USA. Association for Computational Linguistics.
- Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. [Chain-of-thought prompting elicits reasoning in large language models](#). In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA*.
- David Wingate, Mohammad Shoeybi, and Taylor Sorensen. 2022. [Prompt compression and contrastive conditioning for controllability and toxicity reduction in language models](#). In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 5621–5634, Abu Dhabi, United Arab Emirates.
- Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, and 3 others. 2020. [Transformers: State-of-the-art natural language processing](#). In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online.
- Fangyuan Xu, Weijia Shi, and Eunsol Choi. 2024. [RECOMP: Improving retrieval-augmented LMs with context compression and selective augmentation](#). In *The Twelfth International Conference on Learning Representations, ICLR '24, Vienna, Austria*.
- Qwen An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxin Yang, Jingren Zhou, Junyang Lin, and 25 others. 2024. [Qwen2.5 technical report](#). *ArXiv preprint*, abs/2412.15115.
- Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. [HotpotQA: A dataset for diverse, explainable multi-hop question answering](#). In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium.
- Bingsheng Yao, Guiming Chen, Ruishi Zou, Yuxuan Lu, Jiachen Li, Shao Zhang, Yisi Sang, Sijia Liu, James Hendler, and Dakuo Wang. 2024. [More samples or more prompts? exploring effective few-shot in-context learning for LLMs with in-context sampling](#). In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 1772–1790, Mexico City, Mexico.
- Fan Yin, Jesse Vig, Philippe Laban, Shafiq Joty, Caiming Xiong, and Chien-Sheng Wu. 2023. [Did you read the instructions? rethinking the effectiveness of task definitions in instruction learning](#). In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3063–3079, Toronto, Canada.
- Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. [BERTScore: Evaluating text generation with BERT](#). In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia*.

Appendix for paper “Understanding and Improving Information Preservation in Prompt Compression for LLMs”

A Implementation Details

Models In all the experiments we use Mistral-7B Instruct v0.2 (Jiang et al., 2023a)⁶ that contains approximately 7.24B parameters as a target LLM. We use the associated model checkpoint implemented using the PyTorch (Paszke et al., 2019) framework that is available on the HuggingFace Hub via the transformers (Wolf et al., 2020) library. We access microsoft/phi-2 (Abdin et al., 2023)⁷ for LLMLingua and the Mistral 7B based xRAG (Hannibal046/xrag-7b⁸) in the same way. We load the datasets from the HuggingFace using the Datasets library (Lhoest et al., 2021).

Training We train our models using two compute instances, each equipped with 8 NVIDIA A100 Tensor Core GPUs, 1152 GiB of memory, and 96 vCPUs. The training process for a new xRAG model variant typically takes around 16 hours. We adopt a batch size of 1 per GPU and leverage the Accelerate library with a gradient accumulation step of 4.⁹ The specific hyper-parameters used during training are detailed in Table 11.

The tuning dataset for xRAG models exhibits some inconsistencies. The combined datasets contain a total of 1,128,075 samples. However, only 628,667 samples include a context component, and after filtering out instances where the background text is shorter than three words, 628,003 samples remain. The remaining 499,408 samples originate from QA or fact-checking datasets that do not provide explicit context.

Evaluation Evaluation is conducted on a subset of the available datasets due to the high number of datasets and model combinations. To ensure consistency, samples are randomly selected using a fixed seed of 42. Response generation follows a greedy search decoding strategy, while context truncation varies depending on the model. For xRAG, the retriever tokenizer processes an input truncated to 16,000 tokens, but no truncation is applied to the prompt passed to the target language model. Mistral, on the other hand, limits input to 8,192 tokens,

Hyperparameter	Pre-training	Fine-tuning
Optimizer	AdamW	AdamW
Learning rate	6e-3	2e-5
LR scheduler type	linear	linear
Warmup ratio	0.03	0.03
Weight decay	0.0	0.0
Epochs	1	1
KL α	—	2.0
KL temperature	—	1.0
Flash attention	True	True
Batch size	1	1
Gradient accumulation steps	4	2
Num GPUs	8	8

Table 11: Hyper-parameters used during the pre-training and fine-tuning phases of the xRAG model and its variants.

a necessary constraint given that an Arxiv article tokenized with Mistral’s tokenizer can reach 35,208 tokens, leading to out-of-memory (OOM) errors if not truncated. LLMLingua processes the full input for compression and then truncates the compressed prompt to 8,000 tokens when required before generation. For generation settings, the target token count in LLMLingua is fixed at 350, aligning with the best-reported performance in prior work. Mistral employs a *max_new_tokens* setting of 500. The evaluation metric in the LLMLingua reproducibility study uses human-written summaries from the dataset as ground-truth references, allowing for an assessment of the generated responses against high-quality, manually curated summaries.

Data Processing We use the validation partition from the "rc" TriviaQA subset, retaining 9,533 samples after removing those without context. Each sample includes either Web search results or Wikipedia articles, but we use only the first Web search result to better simulate a realistic retrieval-augmented generation (RAG) scenario. To manage lengthy documents, which range from 1 to 8,246 sentences (averaging 124), we limit the context to a maximum of 50 sentences, resulting in 5,743 samples, while 7,479 samples remain within 100 sentences. For evaluation, we compute results on 863 out of 1,000 QuAC validation samples, discarding those with fewer than four question-answer pairs, as the fourth question serves as the model’s prompt. Additionally, in-context learning (ICL) demonstrations for GSM8K are sampled from its training partition.

⁶huggingface.co/mistralai/Mistral-7B-Instruct-v0.2

⁷<https://huggingface.co/microsoft/phi-2>

⁸<https://huggingface.co/Hannibal046/xrag-7b>

⁹<https://huggingface.co/docs/accelerate>

Reproducibility of our Experiments To ensure experiment reproducibility, we use a fixed pseudo-random seed when sampling evaluation examples, allowing for consistent dataset selection across runs. Additionally, we set the model’s temperature to 0 during generation, ensuring a fully deterministic response process. These measures eliminate variability in both data sampling and model outputs, enabling reliable comparisons and replication of results.

B Experimental Setup

B.1 Evaluation Metrics

We use the standard similarity metrics for evaluating the downstream performance: (i) for summarization and long-form QA we use ROUGE (Lin, 2004)¹⁰ and BERTScore (Zhang et al., 2020);¹¹ (ii) for short-form QA and mathematical reasoning, we use Exact Match (EM) with normalization of the generated outputs (splitting text into alpha-numeric tokens and non-whitespace tokens, ignoring capitalization and multi-lines).

B.2 Grounding Evaluation

As part of our preliminary analysis, we evaluated several grounding approaches. Our analysis showed that methods based on Natural Language Inference (NLI) models – classifying text as either entailed or not in the source – perform poorly when dealing with long source texts and answers requiring information synthesis or reasoning. Similarly, using LLM-as-a-Judge through the RAGAs framework (Es et al., 2024) yielded factual correctness scores skewed towards 0, making it difficult to establish a reasonable threshold. More promising results were obtained by directly prompting an LLM (Claude 3 Haiku)¹² to score the grounding of sentences from generated responses in source paragraphs, particularly for summarization tasks. However, these generated grounding scores tended to skew towards higher values. Given the lack of sensitivity of these approaches to grounding evaluation in our long-context scenarios, we adopted a state-of-the-art approach for automatic faithfulness evaluation (FABLES, Kim et al. (2024)), as it shows balanced scores and is able to handle long contexts.

¹⁰<https://huggingface.co/spaces/evaluate-metric/rouge>

¹¹<https://huggingface.co/spaces/evaluate-metric/bertscore>

¹²<https://www.anthropic.com/claude/haiku>

C Datasets

C.1 Training Dataset

Figure 1 presents the distribution of a number of sentences per training sample in xRAG pre-training data. The dataset contains more than 26M Wikipedia snippets from `enwiki-dec2021`.

C.2 Evaluation Datasets

HotpotQA is a question-answering dataset featuring natural, multi-hop questions. It requires reasoning over supporting facts and enables more explainable question-answering systems. It contains three types of questions:

- (i) single-hop questions that require reasoning over one of the paragraphs
- (ii) multi-hop questions that were correctly answered by the model but require reasoning over multiple documents
- (iii) hard multi-hop questions that were not correctly answered by the model and require reasoning over multiple documents

We generate answers for 1,000 examples randomly sampled from the development partition and use them for evaluation.

Arxiv-summarization is a long document summarization dataset. The test partition used in our experiments contains 6,440 arXiv articles divided into sections. We treat the abstracts of the articles as a proxy for a human-written ground-truth summary.

Question Answering in Context (QuAC) is a dataset for modeling, understanding, and participating in information-seeking dialog. Data samples are in the form of interactive dialogs between two crowd workers: (i) a student who poses a sequence of free-form questions to learn as much as possible about a given topic, and (ii) a teacher who answers the questions by providing short spans from the text of the Wikipedia text that is hidden from the student. Results are computed for 863 out of 1,000 samples from the QuAC validation partition. We discard samples that have fewer than 4 question-answer pairs (turns in the conversation). The fourth question is the question we prompt the model to answer.

Grade School Math 8K (GSM8K) is a dataset of 8.5K high-quality, linguistically diverse grade school math word problems, designed for evaluating question-answering models on basic mathematical tasks. Solving these problems requires multi-

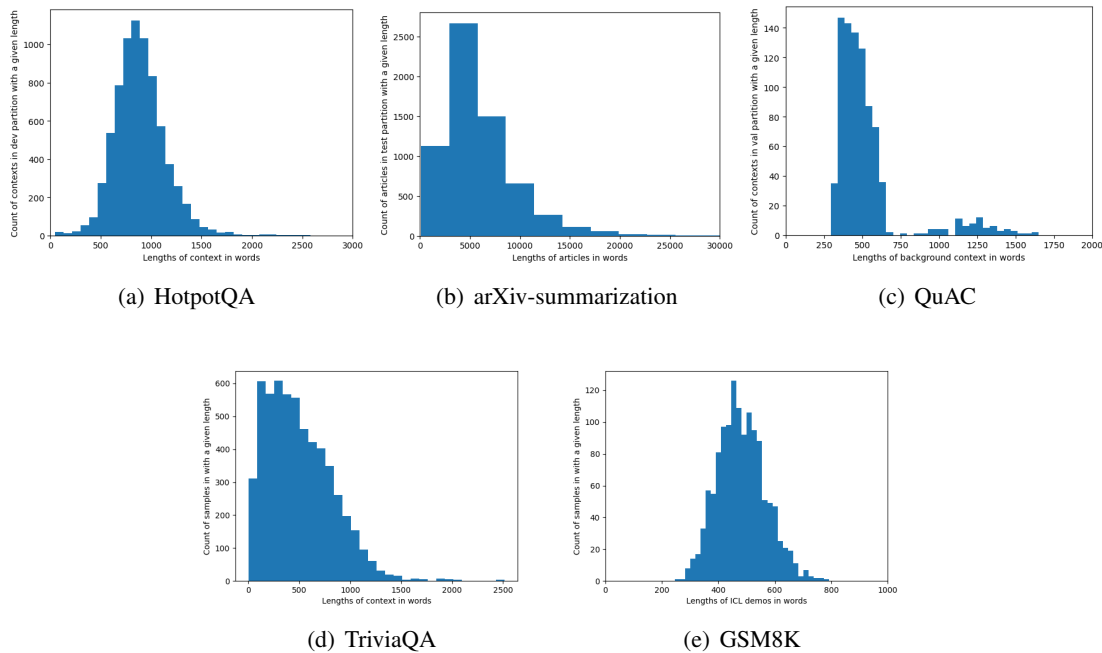


Figure 3: Histograms of context lengths in the different datasets we use for evaluation.

step reasoning, typically involving 2 to 8 steps of elementary arithmetic operations. The questions do not require concepts beyond early Algebra, and most can be solved without explicitly defining variables. Solutions are presented in natural language rather than pure mathematical expressions.

TriviaQA is a large-scale reading comprehension dataset containing over 650K question-answer-evidence triples, including 95K expert-authored question-answer pairs with independently collected evidence documents. The dataset features complex, compositional questions with high syntactic and lexical variability which require cross-sentence reasoning. This makes TriviaQA more challenging than other large-scale QA datasets.

D Reproducibility of Baselines

To ensure the reliability of our baseline implementations, we conduct a reproducibility study on xRAG and LLMLingua, evaluating them on the datasets used in their respective papers. Our goal is to verify that our implementations produce comparable results.

For xRAG, we assess performance on 1,000 randomly selected samples from the HotpotQA development set (see Table 12). We evaluate two versions: (i) the vanilla xRAG model released by the authors,⁸ and (ii) our reproduced version trained using the original data and code-

base (github.com/Hannibal046/xRAG). The reproduced model follows the same pre-training and fine-tuning setup, though we use approximately 50% of the fine-tuning samples with original context paragraphs instead of retrieved ones due to limitations in reproducing the original data pre-processing steps.

Our implementation of xRAG achieves slightly better results than those reported in the paper when using curated context and filtering out irrelevant, yet topically related, paragraphs. The performance gap between the vanilla and the reproduced xRAG is minimal. Additionally, Mistral-7B without prompt compression achieves results comparable to the best system on the HotpotQA leaderboard (EM of 0.775)¹³, confirming the competitiveness of our baseline.

We use the code provided by LLMLingua’s authors¹⁴ and verify our implementation on the [Arxiv-march-2023](#) dataset. Note, that we use the vanilla version of the dataset, which slightly differs from the one used in the LLMLingua paper, as the authors applied additional pre-processing steps to filter some examples. However, the paper lacks sufficient details for full reproducibility. Our results are close to, but slightly lower than, those reported in the original paper (see Table 13).

¹³<https://paperswithcode.com/sota/question-answering-on-hotpotqa>

¹⁴<https://github.com/microsoft/LLMLingua>

Method	Context	Exact match	
		dev_distractor_v1 (1000)	test-distractor
Mistral-7B (no compression)	All 10 context paragraphs	0.713	—
	Only paragraphs with supporting facts	0.787	—
xRAG (Vanilla)	All 10 context paragraphs	0.286	—
	Only paragraphs with supporting facts	0.382	—
xRAG (Ours, Reproduced)	All 10 context paragraphs	0.286	—
	Only paragraphs with supporting facts	0.375	—
xRAG (Paper, Cheng et al. (2024))	One retrieved document	—	0.340

Table 12: Reproducibility results for the xRAG approach. This table compares different xRAG variants: the *vanilla* version, based on the publicly available checkpoint (without modification); the *reproduced* version, which is our re-trained model using the original codebase (Appendix D); and xRAG (Paper), presenting the results reported in the xRAG paper.

Method	Test samples	BLEU	ROUGE-1	ROUGE2	ROUGE-L	BERTScore F1
LLMLingua (350 tokens) with Mistral-7B (Ours)	100 random	0.164	0.492	0.211	0.316	0.881
LLMLingua (Paper, Jiang et al. (2023b))	all (?)	0.232	0.542	0.327	0.427	0.903

Table 13: Reproducibility results of the LLMLingua approach. We compare the numbers reported in the LLMLingua paper (Jiang et al., 2023b) with the numbers that we obtain using the code provided by the authors (without modifications).

xRAG variant	HotpotQA			HotpotQA*			TriviaQA			QuAC		
	Cont.	Par.	Sent.	Cont.	Par.	Sent.	Cont.	Par.	Sent.	Cont.	Par.	Sent.
xRAG (Reproduced)	0.286	0.139	0.014	0.375	0.390	0.174	0.696	—	0.115	0.829	—	0.843
w/ PT + FT	0.248	0.188	0.158	0.362	0.376	0.372	0.680	—	0.557	0.842	—	0.861
w/ Sentence PT + FT	0.313	0.197	0.066	0.422	0.423	0.423	0.712	—	0.361	0.833	—	0.855
w/ Two-Step PT + FT	0.277	0.109	0.145	0.406	0.428	0.462	0.685	—	0.521	0.834	—	0.823

Table 14: Extended version of Table 6 with paragraph-level (*Par.*) scores. Performance of the different variants of the xRAG method on long-context datasets. Results are provided for context encoded into a different number of xRAG tokens. HotpotQA* indicates the scenario with only supporting documents taken as context.

xRAG variant	HotpotQA		HotpotQA*		arXiv-sum.		TriviaQA		QuAC		GSM8K	
	Cont.	Sent.	Cont.	Sent.	Cont.	Sent.	Cont.	Sent.	Cont.	Sent.	Cont.	Sent.
xRAG (Reproduced)	0.286	0.014	0.375	0.174	0.775	0.760	0.696	0.115	0.829	0.843	0.294	0.001
w/ PT + FT	0.248 (-13%)	0.158 (1029%)	0.362 (-3%)	0.372 (114%)	0.803 (4%)	0.758 (-0%)	0.680 (-2%)	0.557 (384%)	0.842 (2%)	0.861 (2%)	0.207 (-30%)	0.051 (50%)
w/ Sentence PT + FT	0.313 (9%)	0.066 (371%)	0.422 (13%)	0.423 (143%)	0.803 (3%)	0.739 (-3%)	0.712 (2%)	0.361 (214%)	0.833 (0%)	0.855 (1%)	0.231 (-21%)	0.030 (29%)
w/ Two-Step PT + FT	0.277 (-3%)	0.145 (936%)	0.406 (8%)	0.462 (166%)	0.785 (1%)	0.696 (-8%)	0.685 (-2%)	0.521 (353%)	0.834 (1%)	0.823 (-2%)	0.111 (-62%)	0.012 (11%)

Table 15: Performance of the different variants of the xRAG method. Results are provided for context encoded into a different number of xRAG tokens. Relative **improvement/drop** is calculated with respect to the reproduced xRAG.

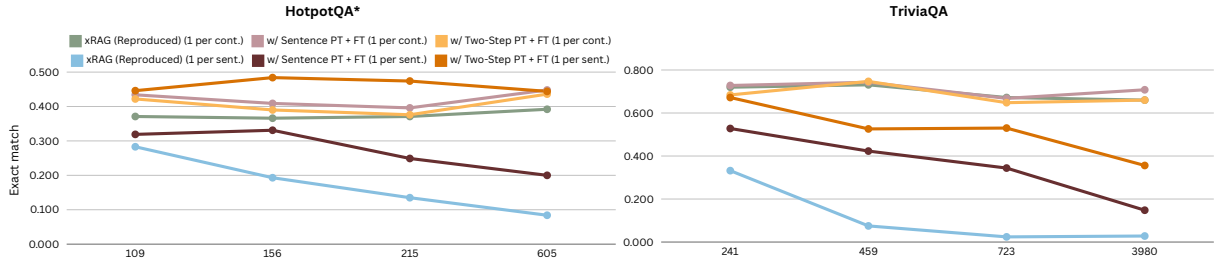


Figure 4: Downstream task performance of different variants of xRAG binned by the length of the input context. Both datasets are evaluated in terms of exact match. Each bucket contains approximately 250 samples.

E Additional Results and Analysis

E.1 Downstream Task Performance

Tables 14 and 15 present additional results for different variants of xRAG method on long-context datasets. Figure 4 presents downstream tasks performance of different variants of xRAG in terms of length of the context. The following xRAG variants are taken into account:

xRAG (Reproduced) The pre-training and fine-tuning procedures remain the same as in the original xRAG. We use the same pre-training data. For the fine-tuning step, we use $\sim 50\%$ of the samples they used with original context paragraphs instead of retrieved ones.¹⁵

xRAG w/ PT + FT (pre-training and fine-tuning with one token per sentence) We chunk each sample in pre-training and fine-tuning data into sentences and encode each sentence into a separate xRAG token.

xRAG w/ Sentence PT + FT (sentence-level pre-training + fine-tuning with one token per sentence) pre-training with samples containing single sentences,¹⁶ is followed by fine-tuning with samples chunked into sentences with one token per sentence.

xRAG w/ Two-Step PT + FT (sentence-level pre-training + pre-training and fine-tuning with one token per sentence) Two-step pre-training involves: (i) encoding one sentence per pre-training sample, and (ii) chunking samples into sentences with each sentence encoded separately. It is followed by fine-tuning with samples chunked into sentences with one token per sentence.

¹⁵This is due to limitations in the reproducibility of the pre-processing of the fine-tuning data originally used for xRAG.

¹⁶The pre-training step uses only one sentence from each pre-training sample and encodes it as a single xRAG token.

E.2 Grounding

Tables 16 and 17 report on grounding scores of the base compression methods and our xRAG modifications. In each table we present columns: *Ist* – grounding score of the first claim detected in the response, and *Avg* – the average grounding score for all the claims in the response. All the grounding scores for xRAG models are reported for the non-empty responses.

Statistics about the number of empty responses returned for each dataset can be found in Table 18. We can see that the reproduced xRAG model tends to return significantly more empty responses compared to other versions. This is yet another evidence that the newly proposed training regimes lead to more stable representations and better model performance.

E.3 Information Preservation

To better understand the information preservation in the xRAG compression method, we further extend our evaluation with ROUGE metrics (Tables 19 and 20). We use the same *seen* (during training, in-domain samples) and *unseen* (during training, out-of-domain samples) datasets as in Section 4.3. The low ROUGE scores (less than 0.5) indicate the models are not able to reconstruct the original content and instead, they are paraphrasing it, emphasizing the need for a metric that goes beyond lexical matching. Nevertheless, the ROUGE evaluations also show that our xRAG modification (*xRAG w/ Two-Step PT + FT*) increases more than 2x the ROUGE scores reaching ROUGE-1 values close to 0.4 on the sentence level (both on seen and unseen data). This shows that the new representations are able to capture more precisely the original content compared to the tokens from the reproduced xRAG model. In addition to entity preservation results for “*unseen*” samples discussed in Section 5, we provide results for “*seen*” samples

Method	HotpotQA		HotpotQA*		arXiv-sum.		QuAC		TriviaQA		GSM8K	
	1st	Avg	1st	Avg	1st	Avg	1st	Avg	1st	Avg	1st	Avg
Mistral-7B	0.86	0.80	0.80	0.75	1.00	0.97	0.94	0.93	0.81	0.78	0.58	0.50
xRAG	0.61	0.52	0.65	0.57	0.36	0.39	0.50	0.45	0.77	0.73	0.50	0.42
PISCO	0.62	0.59	0.79	0.76	0.84	0.74	0.65	0.63	0.86	0.84	0.56	0.48
LLMLingua	0.55	0.45	0.81	0.75	0.58	0.62	0.56	0.49	0.78	0.72	0.52	0.44

Table 16: Grounding scores with respect to the source document/context for responses generated with different methods. All scores are an average across 5 random sets of 100 samples from the original evaluation set (standard deviation (σ) between sets is < 0.03). “1st”: grounding score for the first claim detected in the response. “Avg”: average grounding score for all the claims in the response.

Method	xRAG Tokens	HotpotQA		HotpotQA*		arXiv-sum.		QuAC		TriviaQA	
		1st	Avg	1st	Avg	1st	Avg	1st	Avg	1st	Avg
xRAG (Reproduced)		0.55	0.50	0.66	0.58	0.40	0.38	0.51	0.43	0.81	0.72
w/ Sentence PT + FT	1 per cont.	0.57	0.52	0.69	0.62	0.42	0.44	0.56	0.49	0.81	0.73
w/ Two-Step PT + FT		0.53	0.40	0.54	0.45	0.43	0.35	0.50	0.42	0.76	0.68
xRAG (Reproduced)		0.31	0.31	0.51	0.47	0.00	0.26	0.48	0.47	0.76	0.71
w/ Sentence PT + FT	1 per sent	0.48	0.34	0.52	0.52	0.30	0.46	0.48	0.42	0.70	0.65
w/ Two-Step PT + FT		0.26	0.23	0.54	0.58	0.28	0.30	0.43	0.39	0.60	0.55

Table 17: Grounding scores with respect to the source document/context for responses generated with different methods. All scores are an average across 5 random sets of 100 samples from the original evaluation set ($\sigma < 0.02$). “1st”: grounding score for the first claim detected in the response. “Avg”: average grounding score for all the claims in the response.

Method	Encodings	HotpotQA	HotpotQA*	arXiv-sum.	QuAC	TriviaQA
xRAG	1 per cont.	0	0	10	0	0
xRAG (Reproduced)		0	0	65	0	0
w/ Sentence PT + FT	1 per cont.	0	0	0	0	0
w/ Two-Step PT + FT		0	0	5	0	0
xRAG (Reproduced)		56	6	98	27	54
w/ Sentence PT + FT	1 per sent	4	0	64	0	7
w/ Two-Step PT + FT		2	0	24	0	0
PISCO	8 per cont.	3	8	0	0	11

Table 18: Number of empty responses generated by soft prompting models.

Method	Encodings	Entire dataset		1-sent long samples		5-sents long samples		10-sents long samples	
		ROUGE-1	ROUGE-L	ROUGE-1	ROUGE-L	ROUGE-1	ROUGE-L	ROUGE-1	ROUGE-L
xRAG (Reproduced)		0.152	0.113	0.186	0.151	0.151	0.101	0.120	0.085
w/ Sentence PT + FT	1 per cont.	0.152	0.112	0.203	0.162	0.142	0.097	0.111	0.078
w/ Two-Step PT + FT		0.284	0.204	0.397	0.317	0.259	0.162	0.198	0.133
xRAG (Reproduced)		0.075	0.061	0.186	0.151	0.028	0.023	0.013	0.011
w/ Sentence PT + FT	1 per sent	0.091	0.072	0.202	0.162	0.043	0.032	0.027	0.021
w/ Two-Step PT + FT		0.384	0.273	0.397	0.317	0.434	0.294	0.320	0.209
PISCO	8 per cont.	0.509	0.356	0.525	0.439	0.547	0.351	0.454	0.277
	8 per sent.	0.555	0.409	0.509	0.415	0.608	0.437	0.549	0.374

Table 19: Results of information preservation by different variants of xRAG and PISCO on the **unseen (during training), out-of-domain examples**. Performance is reported in terms of ROUGE metrics computed between the original and reconstructed text. The examples are encoded into one token directly or split into multiple tokens, one per sentence.

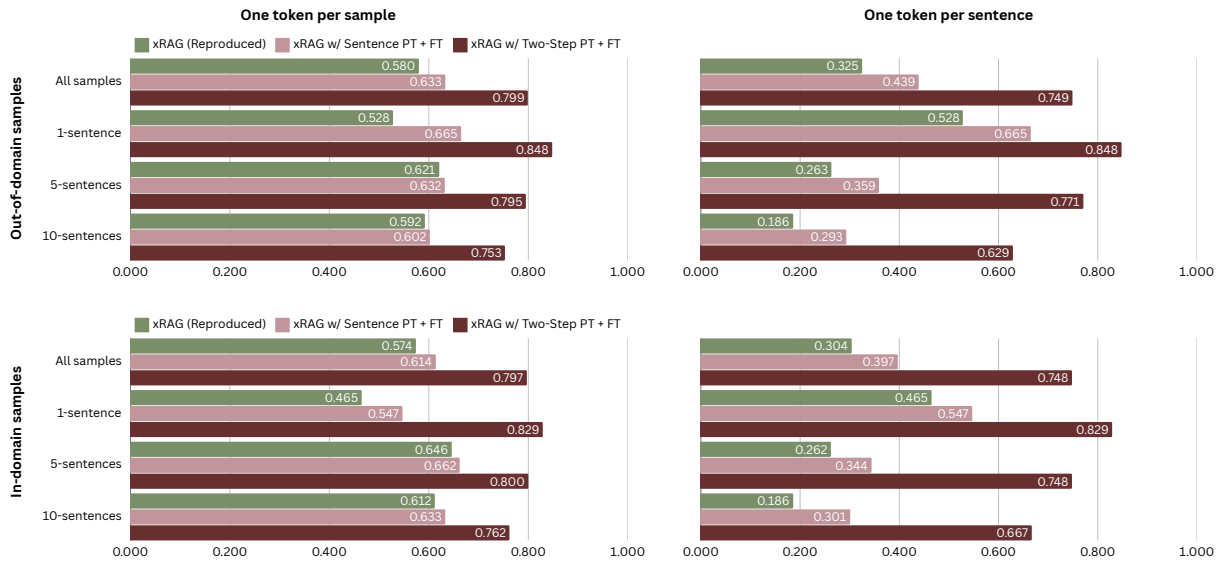


Figure 5: Results of information preservation by different variants of xRAG. Performance is reported in terms of the BERTScore F1 metric computed between the original and reconstructed sample. Results are presented for both in- and out-of-domain samples that are encoded into one token directly or split into sentences and then encoded in multiple tokens.

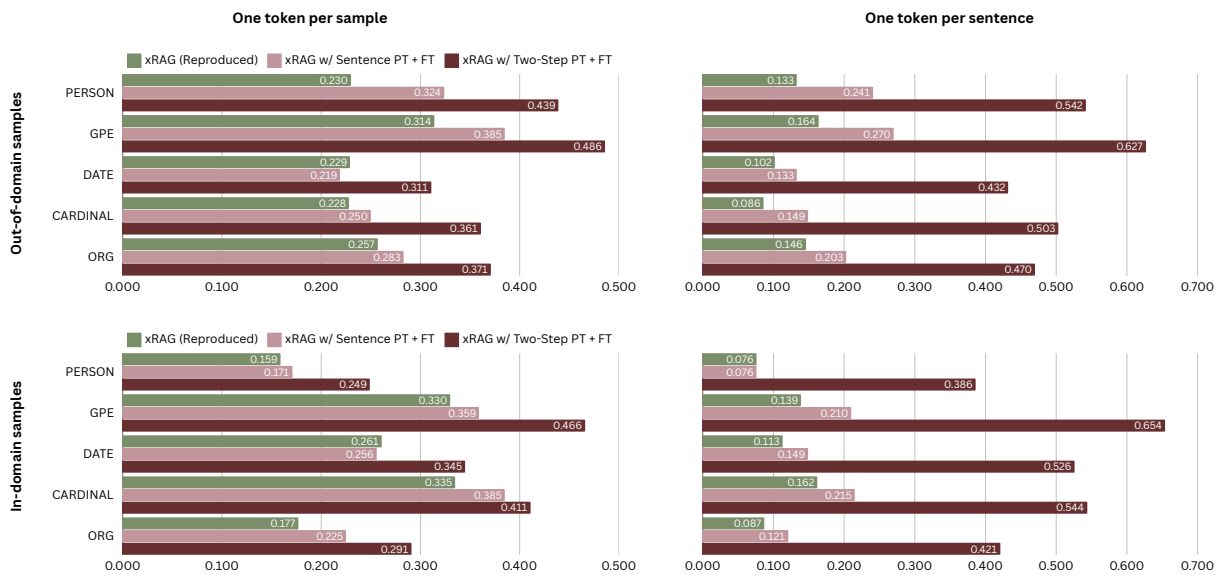


Figure 6: Results of entity preservation experiments. It is evaluated in terms of the exact match between entities and reconstructed text. Results are presented for both in- and out-of-domain samples that are encoded into one token directly or split into sentences and then encoded in multiple tokens.

in Figure 5. Both datasets contain 450 examples, out of which 150 samples are one-sentence long, 150 contain 5 sentences and 150 contain 10 sentences. Unseen examples are sampled from three evaluation datasets: HoptpotQA (Yang et al., 2018), QuAC (Choi et al., 2018), and TriviaQA (Joshi et al., 2017) (with 150 examples from each of these datasets).

Named Entity Preservation To quantify our observations from the qualitative analysis, we perform additional experiments to measure the amount of entity preservation after compression. We measure the fraction of text entities from the uncompressed input retained in the reconstructed version (see Figure 6). We observe particularly low scores for *dates* and *numerical* values. Additionally, entity preser-

Method	Encodings	Entire dataset		1-sent long samples		5-sents long samples		10-sents long samples	
		ROUGE-1	ROUGE-L	ROUGE-1	ROUGE-L	ROUGE-1	ROUGE-L	ROUGE-1	ROUGE-L
xRAG (Reproduced)	1 per cont.	0.153	0.115	0.144	0.127	0.171	0.119	0.144	0.100
w/ Sentence PT + FT		0.147	0.113	0.158	0.138	0.154	0.109	0.129	0.092
w/ Two-Step PT + FT		0.289	0.228	0.423	0.378	0.247	0.167	0.198	0.138
xRAG (Reproduced)	1 per sent	0.061	0.054	0.144	0.127	0.03	0.025	0.010	0.009
w/ Sentence PT + FT		0.080	0.066	0.158	0.138	0.048	0.035	0.034	0.026
w/ Two-Step PT + FT		0.422	0.327	0.423	0.378	0.451	0.315	0.391	0.287
PISCO	8 per cont.	0.475	0.341	0.410	0.361	0.569	0.376	0.446	0.286
	8 per sent.	0.553	0.421	0.420	0.371	0.633	0.460	0.606	0.431

Table 20: Results of information preservation by different variants of xRAG and PISCO on the **seen (during xRAG training), in-domain examples**. The ROUGE metrics are calculated in the same way as for the unseen set in Table 19.

vation for *people* is notably poor on in-domain samples, likely due to noise or ambiguous references in the pre-training data (e.g., “*Thomas Scott 1806–1816*” annotated as PERSON).¹⁷ In contrast, *geographical locations* are the most consistently preserved across all input types. Overall, our findings indicate that the model struggles with maintaining entities during prompt compression.

In general, the target LLM tends to hallucinate extensively when reconstructing the context, regardless of the amount of encoded information. For example, the following sentence from HotpotQA context: “*In May 1983, she married Nikos Karvelas, a composer, with whom she collaborated in 1975 and in November she gave birth to her daughter Sofia.*” is reconstructed as “*In May 1985, she married Nikos Karvelas, a composer and lyricist.*” introducing two factual errors (incorrect date and hallucinated information about the lyricist). We must note that different prompts produce widely varying content. Hereby, we do not recommend performing this analysis on a single prompt.

F Prompts Used in the Experiments

Table 21 contains prompts used in the context reconstruction experiments, Table 22 contains prompts used for grounding evaluation, and Table 23 contains prompts used for response generation for different tasks.

ID	Prompt
1	These two expressions are equivalent in essence:(1) <i>{token}</i> (2)
2	In other words, background: <i>{token}</i> is just another way of saying:
3	Background: <i>{token}</i> means the same as
4	<i>{token}</i> After unpacking the ideas in the background information above, we got:
5	<i>{token}</i> Please offer a restatement of the background sentences I’ve just read.

Table 21: Prompts used for reconstructing contexts encoded by soft prompt compression method.

¹⁷We use SpaCy (<https://spacy.io/>) for NER.

Prompt	
Claim detection	You are trying to verify the faithfulness of statements made in a given summary of an article against the actual text of the article. To do so, you first need to break the summary into a set of "atomic claims", each of which will then be passed to a human who will read the article and verify if the claim is true or not. Each atomic claim must be fully understandable without any other context from the summary (e.g., all entities must be referred to by name, not pronoun), and they must be situated within relevant temporal, location, and causal context whenever possible. Try to keep each atomic claim to a maximum of 2 sentences. Each atomic claim is separated with '-'. Summary: List of atomic claims:
Faithfulness evaluation	You are provided with a context and a statement. Your task is to carefully read the context and then determine whether the statement is true or false. Use the information given in the context to make your decision. Do not provide explanations. Context: Statement: Question: Based on the context provided, is the above statement True or False? Answer:

Table 22: Prompts used for grounding evaluation.

Task	Prompt
QA	[INST] Refer to the background document and answer the question. Provide only a short answer.\n\nBackground: {context}\n\nQuestion: {question} [/INST] The answer is:
Summarization	[INST] Briefly summarize this article:\n\nArticle: {context} [/INST] Summary:
Mathematical reasoning	[INST] Answer the math question by providing a numerical value. Precede the final answer with an explanation of the intermediate steps. Do not add any symbols to the final numerical answer.\n\n###\n Here are some examples:\n{icl_demos}\n###\n\nQuestion: {question} [/INST]
Conversational QA	[INST] Refer to the background document, as well as the conversational context and answer the question. Answer the question by extracting a specific span from the provided background.\n\nBackground: {context}\n\nConversational context: {conv_context}{question} [/INST]

Table 23: Prompts used for response generation for different tasks.

Method	Prompt	Original text	Reconstructed text
xRAG	<xRAG> Please offer a re-statement of the background sentences I've just read.	In May 1983, she married Nikos Karvelas, a composer, with whom she collaborated in 1975 and in November she gave birth to her daughter Sofia. After their marriage, she started a close collaboration with Karvelas. Since 1975, all her releases have become gold or platinum and have included songs by Karvelas.	In 1985, she married Nikos Karvelas, a composer and music producer. They collaborated on many projects, and she began to write lyrics for him.
	The token <xRAG> represents a compressed text. Please fully expand and reconstruct the original text.	Beginning in 1986, Motel 6 has advertised through radio commercials featuring the voice of writer and National Public Radio commentator Tom Bodett, with the tagline: We'll leave the light on for you.	The phrase "We'll leave the light on for you" is a common phrase used by motels to assure potential customers that they will be welcome. The phrase has been used in advertising for the Motel 6 chain since 1962.
	Reconstruct the content of this document: <xRAG>.	Question: A band's members each earn \$20 per gig. If there are 4 members and they've earned \$400, how many gigs have they played? Solution: The band earns \$80 per gig because $4 \times 20 = \ll4*20=80\gg80$ They played 5 gigs because $400 / 80 = \ll400/80=5\gg5$ ##### 5	$25 \times 40 = 10$
w/ PT + FT	Background: <xRAG> means the same as	The terrain is mostly flat and the transition from Borovets to the Black Rock takes just over an hour of peaceful walking.	the Black Rock Desert, but is much easier to walk on.
	Background: <xRAG> Can you put the above sentences in your own terms?	The terrain is mostly flat and the transition from Borovets to the Black Rock takes just over an hour of peaceful walking.	The walk from the car park to the Black Rock is a relatively flat and easy one, taking around 1 hour.
	Background: <xRAG> means the same as	Leblois acted unilaterally to send the 22nd Colonial Infantry Regiment to Termes to support the 3rd Colonial Division.	the 22nd Infantry Regiment.
w/ Sentence PT + FT	These two expressions are equivalent in essence:(i) <xRAG> (ii)	The letter was the first in a series of direct and indirect communications between the White House and the Kremlin throughout the remainder of the crisis.	The first of a series of letters between the White House and the Kremlin.
	These two expressions are equivalent in essence:(i) <xRAG> (ii)	The population was 1,871 at the 2010 census. The Flathead Valley, located in Northwestern Montana, is a region of the U.S. state of Montana.	2010 Census population of 1,871.
	These two expressions are equivalent in essence:(i) <xRAG> (ii)	The group's only album, Mary Star of the Sea, was released in 2003.	The Sea Mary is a 2003 album by the group.
w/ Two-Step PT + FT	<xRAG> After unpacking the ideas in the background information above, we got:	The letter was the first in a series of direct and indirect communications between the White House and the Kremlin throughout the remainder of the crisis.	the first letter in the series of correspondence between the White House and the Kremlin was the first of a series of letters that would be exchanged throughout the remainder of the crisis.
	In other words, background: <xRAG> is just another way of saying:	The population was 1,871 at the 2010 census. The Flathead Valley, located in Northwestern Montana, is a region of the U.S. state of Montana.	2 The population was 1,870 at the 2010 census. The Flathead Valley is located in the northwest corner of Montana, United States, and is the largest valley in the region.
	In other words, background: <xRAG> is just another way of saying:	The group's only album, Mary Star of the Sea, was released in 2003.	The Mary Star, the group's only album, was released in 2003.

Table 24: Results of reconstructing context using original xRAG model released by the authors of the paper, the xRAG model reproduced by us, and additional new variants of xRAG that we proposed in this paper.