500xCompressor: Generalized Prompt Compression for Large Language Models

Zongqian Li
University of Cambridge
zl510@cam.ac.uk

Yixuan Su
University of Cambridge
ys484@cam.ac.uk

Nigel Collier
University of Cambridge
nhc30@cam.ac.uk

Abstract

Prompt compression is important for large language models (LLMs) to increase inference speed, reduce costs, and improve user experience. However, current methods face challenges such as low compression ratios and potential training-test overlap during evaluation. To address these issues, we propose 500xCompressor, a method that compresses natural language contexts into a minimum of one special token and demonstrates strong generalization ability. The 500xCompressor introduces approximately 0.3% additional parameters and achieves compression ratios ranging from 6x to 500x, achieving 27-90% reduction in calculations and 55-83% memory savings when generating 100-400 tokens for new and reused prompts at 500x compression, while retaining 70-74% (F1) and 77-84% (Exact Match) of the LLM capabilities compared to using non-compressed prompts. It is designed to compress any text, answer various types of questions, and can be utilized by the original LLM without requiring fine-tuning. Initially, 500xCompressor was pretrained on the ArxivCorpus, followed by fine-tuning on the ArxivQA dataset, and subsequently evaluated on strictly unseen and cross-domain question answering (QA) datasets. This study shows that KV values outperform embeddings in preserving information at high compression ratios. The highly compressive nature of natural language prompts, even for detailed information, suggests potential for future applications and the development of a new LLM language. [1]

1 Introduction

Long prompts present several challenges in natural language processing applications, such as decreased inference speed, higher computation cost, and a negative influence on user experience. Additionally, the context length limit restricts model

[1] https://github.com/ZongqianLi/500xCompressor

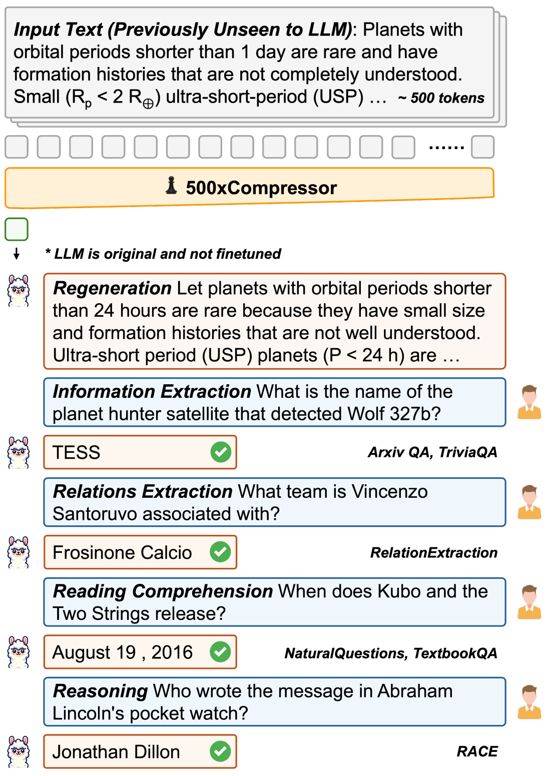
Input Text (Previously Unseen to LLM): Planets with orbital periods shorter than 1 day are rare and have formation histories that are not completely understood. Small (R_p < 2 $R_\oplus$) ultra-short-period (USP) … *~ 500 tokens*
500xCompressor
** LLM is original and not finetuned*
Regeneration Let planets with orbital periods shorter than 24 hours are rare because they have small size and formation histories that are not well understood. Ultra-short period (USP) planets (P < 24 h) are …
Information Extraction What is the name of the planet hunter satellite that detected Wolf 327b?
TESS
Arxiv QA, TriviaQA
Relations Extraction What team is Vincenzo Santoruvo associated with?
Frosinone Calcio
RelationExtraction
Reading Comprehension When does Kubo and the Two Strings release?
August 19 , 2016
NaturalQuestions, TextbookQA
Reasoning Who wrote the message in Abraham Lincoln's pocket watch?
Jonathan Dillon
RACE

Figure 1: The original text is compressed by 500xCompressor and utilized for downstream tasks.

performance and application scenarios, creating a strong demand for prompt length reduction.

Two primary methods for prompt compression have been proposed: hard prompt and soft prompt. Hard prompt methods, such as SelectiveSentence (Li et al., 2023) and LLMLingua (Jiang et al., 2023a), eliminate low-information sentences, words, or even tokens. On the other hand, soft prompt methods, including GIST (Mu et al., 2024), AutoCompressor (Chevalier et al., 2023), and ICAE (Ge et al., 2024), compress natural language tokens into a small number of special tokens. However, these methods have **problems** such as **low compression ratios** (low efficiency improvement), **unclear information loss**, and **potential**

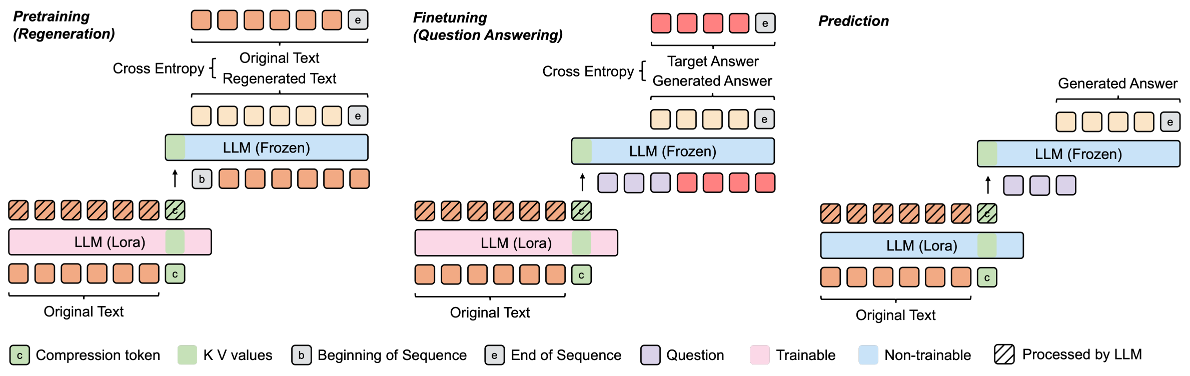

Figure 2: Process of pretraining (left), fine-tuning (middle), and prediction (right) with 500xCompressor.

training-test overlap during evaluation, as discussed in detail in Section 6. For instance, ICAE achieves compression ratios no higher than 15x, and the win rate evaluation metric cannot quantitatively measure the extent of information loss during compression. Additionally, evaluation texts sourced from the Wikipedia dataset might overlap with the training data for LLaMa series models (Grattafiori et al., 2024), raising questions that the generated content could be retrieved from the memory of the LLM rather than the compressed prompts.

To solve these problems, we propose 500xCompressor, illustrated in Figure 1. This method compresses prompts of approximately 500 tokens into a minimum of one token, allowing the compressed tokens to regenerate the original texts or be used for QA. Although trained on the ArxivCorpus and ArxivQA dataset, 500xCompressor could generalize to answer other types of questions. Analysis demonstrates that detailed information, such as proper nouns, special names, and numbers, could be accurately compressed and retrieved.

500xCompressor retains the **advantages** of previous methods and introduces several additional characteristics. Similar to previous soft prompt methods, 500xCompressor is **generalized** and **non-selective**, capable of compressing unseen texts across various topics for QA, demonstrating its generalization ability. Unlike selective compression methods, 500xCompressor aims to regenerate the entire original text, ensuring that all tokens in the original text contribute to the compression tokens. Moreover, the compressed prompts could be used to regenerate original texts or for QA **without requiring fine-tuning** of the LLM, preserving the LLM's original capabilities and improving the convenience of using compressed tokens.

In addition to these existing advantages, we provide **contributions** in three main areas:

- **High Compression Ratio:** This study evaluates the compression model with one and sixteen tokens to compress up to 500 tokens, achieving compression ratios up to 500x. These ratios significantly outperform previous studies, which reported ratios of less than 50x, fully testing the upper limit of prompt compression.
- **Strict Unseen Evaluation:** Using Arxiv abstracts published post-January 2024 ensures evaluation on content unseen by both the LLM and compression model, verifying that outputs are from compressed prompts rather than preexisting model knowledge.
- **Quantitative Analysis of Information Loss:** Through extractive QA with context-span answers, we realize direct quantitative comparison between compressed and uncompressed performance, providing precise measurements of compression-resulting information loss.

In this paper, the design of 500xCompressor is first introduced in Section 2, including how to train and use the compression model. After that, Section 3 explains the training and evaluation datasets, the baselines, and the evaluation metrics. The evaluation results for regeneration and QA are presented in Section 4, with ablation studies analyzing the variables influencing the compression models. This is followed by discussions in Section 5, and finally, the sections on related work and conclusions.

2 Methods

2.1 Training

The training process for the compression model is illustrated in Figure 2, including both pretraining and fine-tuning stages. The compression model comprises two components: an encoder and a decoder, which is similar to an autoencoder and comparable to ICAE. The encoder is the frozen LLM $\mathbf{\Theta}_{\text{LLM}}$ with trainable LoRA parameters $\mathbf{\Theta}_{\text{Lora}}$ (Hu et al., 2022), while the decoder is the original frozen LLM $\mathbf{\Theta}_{\text{LLM}}$. The encoder receives the original text tokens $\mathbf{T} = (t_1, t_2, \ldots, t_l)$ and the compression tokens $\mathbf{C} = (c_1, c_2, \ldots, c_k)$. Through layers, the information in the text is saved into the compression tokens, whose KV values in each layer of the LLM $\mathbf{H_C}$ are output and passed to the decoder.

During pretraining, the inputs of the decoder are the KV values of the compression tokens from the encoder, the beginning of sequence token, and the original text tokens $(\mathbf{H_C}, [\mathbf{BOS}], \mathbf{T})$. The LLM is trained to regenerate the original text based on the KV values, using the end of sequence token $[\mathbf{EOS}]$ to denote when to stop. The cross-entropy loss between the output of the deocder and the original text is calculated and used to train the LoRA parameters in the encoder:

$$\mathcal{L}_{\text{P}} = -\sum_{i=1}^{l} \log P(t_i | \mathbf{H_C}, [\mathbf{BOS}], t_{1:i-1}; \mathbf{\Theta}_{\text{LLM}}, \mathbf{\Theta}_{\text{Lora}}) \tag{1}$$

For instruction fine-tuning, the process is similar to pretraining. However, instead of the original texts, the decoder is provided with questions $\mathbf{Q} = (q_1, q_2, \ldots, q_m)$ and answers $\mathbf{A} = (a_1, a_2, \ldots, a_n)$, which are used to train the LLM to retrieve information from the KV values of the compression tokens and generate answers:

$$\mathcal{L}_{\text{F}} = -\sum_{j=1}^{n} \log P(a_j | \mathbf{H_C}, q_{1:m}, a_{1:j-1}; \mathbf{\Theta}_{\text{LLM}}, \mathbf{\Theta}_{\text{Lora}}) \tag{2}$$

The training process ensures no training-test overlap, as the original LLM parameters in both the encoder and decoder remain unchanged, and no additional parameters are introduced in the decoder. Thus, no information is saved in the decoder.

Main differences between 500xCompressor and ICAE: (1) The input of ICAE decoder is the output embeddings for the compression tokens, whereas 500xCompressor uses the KV values for the compression tokens. KV values could save more information and do not increase inference time. (2) In addition, this paper uses the $[\mathbf{BOS}]$ token to guide the LLM to regenerate the compressed texts, while ICAE creates a trainable new token.

2.2 Prediction

During prediction, all encoder and decoder parameters are frozen. The original text is fed into the encoder, which saves the information into compression tokens. These compression tokens' KV values are then input into the decoder, which regenerates the compressed text when guided by the $[\mathbf{BOS}]$ token or generates an answer based on a given question:

$$\hat{t}_i = \arg\max_{\hat{t}_i} P(\hat{t}_i | \mathbf{H_C}, [\mathbf{BOS}], \hat{t}_{1:i-1}; \mathbf{\Theta}_{\text{LLM}}) \tag{3}$$

$$\hat{a}_j = \arg\max_{\hat{a}_j} P(\hat{a}_j | \mathbf{H_C}, q_{1:m}, \hat{a}_{1:j-1}; \mathbf{\Theta}_{\text{LLM}}) \tag{4}$$

where $\hat{t}_i$ denotes the i-th token in the regenerated text, and $\hat{a}_j$ indicates the j-th token in the generated answer.

By replacing the original text tokens with compressed tokens, the speed of answering questions is increased. This is because, in inference, each token in the question or generated answer must attend to the previous tokens. Replacing a large number of original text tokens with a small number of compressed tokens reduces computational needs.

3 Experiments

3.1 Datasets

The **ArxivCorpus** was used to pretrain 500xCompressor, and the compression model was then fine-tuned on the **ArxivQA** dataset. After that, six benchmarks with different context lengths were used to evaluate the compression models for various abilities: **ArxivQA** and **TriviaQA** (Joshi et al., 2017) for information extraction, **RelationExtraction** (Levy et al., 2017) for relation extraction, **NaturalQuestions** (Kwiatkowski et al., 2019) and **TextbookQA** (Kembhavi et al., 2017) for reading comprehension, and **RACE** (Lai et al., 2017) for reasoning. Among these datasets, ArxivQA is introduced in this paper, while the others are classical QA datasets from MRQA (Fisch et al., 2019).

The ArxivCorpus comprises Arxiv paper abstracts published before April 2024, with pre-July 2023 papers forming the training set and post-January 2024 papers forming the development and test sets. Test set abstracts are selected by token lengths (96, 192, 288, 384, and 480) to evaluate the regeneration performance of prompt compression methods.

The ArxivCorpus was chosen for several reasons: (1) High-quality academic content with clear publication timestamps, (2) Verified temporal separation from LLaMa-3's March 2023 knowledge cutoff, ensuring the regenerated texts are based on the compressed prompts rather than the memory of the LLM, and (3) Official distribution through Cornell University, addressing copyright problems that influence datasets like Pile.

The ArxivQA dataset (more than **250k** QA pairs), derived from ArxivCorpus using LLaMa-3-70b-Instruct, contains extractive QA pairs with the number of QA pairs increasing proportionally with abstract length (starting with 5 pairs per 96-token abstract). Training and development QA pairs are generated from the training set abstracts, while test set pairs come from test set abstracts.

ArxivQA offers three main advantages: (1) Guaranteed LLM-unseen test contexts avoiding training-test overlap, (2) Extractive QA format allowing quantitative evaluation of information loss, and (3) Domain-specific questions generated by LLaMa-3-70b-Instruct based on ArxivCorpus ensuring both difficulty and quality.

3.2 Baselines and Gold Standard

Two baseline approaches are chosen: **LLMLingua-2** (Pan et al., 2024), exemplifying hard prompt compression through selective token elimination, and **ICAE**, utilizing soft prompt compression via continuous vectors. Both methods process the compressed context alongside questions for LLM inference. The gold standard provides the LLM with the complete combination of **instruction**, uncompressed context, and question.

3.3 Evaluation Metrices

For evaluating **text regeneration**, **Rouge-2-F** (based on bigram overlap) and **BLEU** (measuring n-gram precision) scores are used to assess the similarity between regenerated and original texts. For extractive **QA** tasks, **F1** score (the harmonic mean of precision and recall) and **Exact Match** (EM) are used to measure answer accuracy. Higher scores in all these metrics indicate better performance, with a maximum value of 100%.

3.4 Models

The encoder of 500xCompressor is frozen **LLaMA-3-8B-Instruct** with trainable LoRA parameters (rank=64) and the decoder is frozen LLaMA-3-8B-Instruct (Grattafiori et al., 2024).

4 Results

4.1 Efficiency Improvement

Table 1 demonstrates the importance of prompt compression for efficiency gains, showing improvements in both first-time processing (new prompt) and cached processing scenarios (reused prompt) at 500x compression. For new prompts, while compression introduces a minimal computational cost (+0.4%), the savings increase substantially with output length, reaching 49.10% reduction in computation at 400 tokens. Reused prompts demonstrate immediate computational benefits, achieving 90.64% reduction at 100 tokens output length. For memory usage of KV cache, reused prompts achieve 99.80% initial memory reduction, and both prompts show consistent memory savings from 83.16% to 55.33% as output length increases to 400 tokens. Given that real-world applications often involve batch processing and repeated access to the same content, these efficiency gains make 500xCompressor valuable in real-world scenarios.

Output	**Calculations**		**Memory**	
Length	New prompt	Reused prompt	New prompt	Reused prompt
0	+0.4	0	+0.2	**-99.80**
100	-27.39	**-90.64**	**-83.16**	-83.16
200	-40.47	-83.09	-71.28	-71.28
300	-46.56	-76.71	-62.37	-62.37
400	**-49.10**	-71.23	-55.33	-55.33

Table 1: Computation and memory savings (in percentage) achieved by 500xCompressor for different output lengths (token) at 500→1 compression. A new prompt refers to first-time processing of input, while a reused prompt denotes repeated processing that can utilize cached intermediate results.

4.2 Text Regeneration

The text regeneration capabilities of different prompt compression methods are evaluated on the strictly unseen test set of ArxivCorpus. Table 2 shows the performance across varying context lengths and compression ratios, measured by Rouge-2-F and BLEU scores. Our analysis examines the overall advantages, influencing variables, and stability of the improvements.

500xCompressor demonstrates consistent better performance over ICAE across all test scenarios. Our method outperforms ICAE on both Rouge-2-F and BLEU metrics for all context lengths and compression ratios tested. Quantitatively, the average improvement for Rouge-2-F scores increases by

Dataset	ArxivCorpus										Average	
Length	96		192		288		384		480			
Eval. Metrices	RG	BL	RG	BL	RG	BL	RG	BL	RG	BL	RG	BL
Ours$_{500\to16}$	**99.53**	**99.48**	**96.49**	**96.21**	**82.31**	**80.93**	**55.36**	**53.50**	**31.55**	**32.19**	**73.05**	**72.46**
ICAE$_{500\to16}$	83.52	81.85	58.21	55.90	40.96	38.37	34.28	32.03	29.71	29.61	49.33	47.55
Absolute Δ	16.02	17.62	38.28	40.31	41.35	42.56	21.07	21.46	1.84	2.58	23.71	24.91
Relative Δ	19.19%	21.53%	65.76%	72.12%	100.96%	110.92%	61.47%	66.98%	6.20%	8.71%	48.07%	52.38%
Ours$_{500\to1}$	**53.49**	**49.77**	**29.73**	**26.53**	**22.15**	**19.15**	**20.61**	**17.91**	**18.85**	**18.80**	**28.97**	**26.43**
ICAE$_{500\to1}$	30.29	24.18	18.21	13.94	13.89	10.36	13.36	9.92	12.28	11.68	17.61	14.02
Absolute Δ	23.19	25.59	11.51	12.59	8.25	8.79	7.25	7.99	6.56	7.11	11.36	12.41
Relative Δ	76.58%	105.84%	63.22%	90.33%	59.45%	84.81%	54.30%	80.48%	53.44%	60.86%	64.50%	88.56%

Table 2: Quantitative evaluation of **text regeneration** performance on the ArxivCorpus dataset with strictly unseen texts. RG and BL denote Rouge-2-F and BLEU scores respectively. The notation X→Y indicates compression from a maximum of X input tokens to Y compression tokens. Higher values between 500xCompressor (Ours) and ICAE baseline are shown in bold and their performance differences are shown by absolute and relative Δ. All improvements (shown in green) demonstrate the consistent better performance of 500xCompressor across varying context lengths and compression ratios.

23.71 points (48.07%) and 11.36 points (64.50%) at 31.25x and 500x.

The regeneration performance exhibits clear dependencies on both compression ratio and context length. Lower compression ratios and shorter contexts yield higher text precision, with Rouge-2-F and BLEU scores consistently exceeding 95% in optimal conditions. As compression ratios increase, the relative improvements over ICAE become more obvious, showing relative gains of 64.50% in Rouge-2-F and 88.56% in BLEU scores. While performance naturally decreases with longer contexts, the decrease rate shows a stable trend across different compression scenarios.

The method exhibits consistent stability in performance gains. Both absolute and relative improvements remain uniform across Rouge-2-F and BLEU metrics, indicating robust improvement in regeneration quality regardless of the evaluation criteria used.

While the results above demonstrate good text regeneration ability, the true performance of compression is better shown in downstream QA tasks.

4.3 Question Answering

Figure 3 shows the performance of different prompt compression methods across varying compression ratios on QA datasets. 500xCompressor consistently outperforms baseline methods at all compression ratios tested, confirming that KV values have advantages over embeddings (used in ICAE) in preserving information. Notably, as the compression ratio increases from 31.25x to 500x, 500xCompressor exhibits better performance retention, dropping from 74.53% to 70.60% (F1 score) and from 84.57% to 77.92% (Exact Match) of its uncompressed performance.

Tables 3 and 4 present evaluation results for

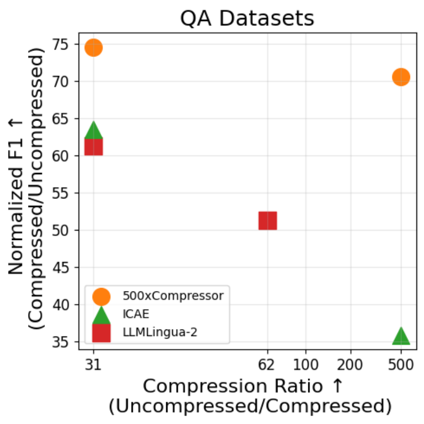

Figure 3: Performance comparison of prompt compression methods on **in-domain and cross-domain QA** datasets across varying compression ratios. Y-axis shows F1 scores normalized by uncompressed performance, while X-axis (log scale) denotes compression ratios defined as #maximum_uncompressed_tokens/#compression_tokens. ↑ indicates higher values are better.

500xCompressor on in-domain and cross-domain QA datasets. These results are analyzed from five aspects: overall performance, influencing variables, generalization capability, scalability, and stability.

Overall Performance 500xCompressor demonstrates higher performance across nearly all context lengths, compression ratios, and both in-domain and cross-domain datasets compared to ICAE and LLMLingua-2. In cross-domain evaluation, it achieves average improvements of 7.10 F1 and 7.61 EM points (19.94% and 37.66% relative) at 500→16 compression, with improvements increasing to 21.93 F1 and 16.64 EM points (107.70% and 161.58% relative) at 500→1 compression.

Performance variables The effectiveness of compression is influenced by both context length

Dataset	ArxivQA										Average	
Length	96		192		288		384		480			
Eval. Metrices	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM
Instruct	64.41	12.40	61.18	13.90	56.08	9.00	52.86	12.40	44.57	16.40	55.82	12.82
Lingua$_{500\to64}$	45.88	7.90	29.91	8.20	21.39	4.20	17.68	3.40	16.17	4.20	26.21	5.58
Lingua$_{500\to32}$	26.97	3.60	20.45	4.40	15.82	2.40	13.00	2.00	12.28	2.10	17.70	2.90
Ours$_{500\to16}$	**60.49**	**25.60**	**47.65**	**16.50**	**35.50**	**8.40**	**30.00**	7.10	**31.98**	**11.70**	**41.12**	**13.86**
ICAE$_{500\to16}$	57.95	23.20	44.41	15.10	33.88	7.70	28.06	**7.20**	29.72	10.60	38.80	12.76
Absolute Δ	2.54	2.40	3.23	1.40	1.62	0.70	1.94	0.10	2.25	1.10	2.31	1.10
Relative Δ	4.38%	10.34%	7.29%	9.27%	4.78%	9.09%	6.91%	1.38%	7.59%	10.37%	5.97%	8.62%
Ours$_{500\to1}$	**42.91**	**10.30**	**32.88**	**6.50**	**25.82**	**3.80**	**23.01**	**3.60**	**24.29**	**6.50**	**29.78**	**6.14**
ICAE$_{500\to1}$	26.87	3.50	21.76	2.30	20.34	2.20	17.35	1.70	17.72	3.60	20.81	2.66
Absolute Δ	16.04	6.80	11.12	4.20	5.47	1.60	5.65	1.90	6.56	2.90	8.97	3.48
Relative Δ	59.71%	194.28%	51.12%	182.60%	26.89%	72.72%	32.58%	111.76%	37.03%	80.55%	43.11%	130.82%

Table 3: **In-domain QA** evaluation results on the ArxivQA dataset with strictly unseen contexts. Length indicates the length of the context to be compressed. F1 and Exact Match (EM) scores are reported across varying context lengths. "Instruct" means full-context performance with instructions, while Lingua denotes LLMLingua-2 baseline. Performance deltas (Δ) between 500xCompressor (Ours) and ICAE baseline are shown in green (improvements) and red (decreases).

Dataset	RE		NaturalQ		RACE		TextbookQA		TriviaQA		Average	
Length	39 (553)		258 (2721)		369 (824)		729 (963)		955 (2124)			
Eval. Metrices	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM
Instruct	71.16	52.98	66.30	39.92	39.55	13.94	45.15	19.49	63.80	41.65	57.19	33.60
Lingua$_{500\to16}$	57.78	41.58	40.46	23.15	12.58	5.93	29.38	19.16	56.06	46.26	39.25	27.22
Lingua$_{500\to8}$	37.85	23.98	32.71	17.94	9.11	3.11	28.67	17.29	56.15	45.58	32.90	21.58
Ours$_{500\to16}$	**68.47**	**50.06**	45.53	**26.40**	**25.53**	**10.97**	**30.31**	**18.36**	**43.76**	**33.25**	**42.72**	**27.81**
ICAE$_{500\to16}$	66.03	44.60	**46.10**	25.18	24.32	9.64	13.27	4.79	28.36	16.78	35.62	20.20
Absolute Δ	2.43	5.46	0.57	1.21	1.20	1.33	17.03	13.57	15.40	16.46	7.10	7.61
Relative Δ	3.69%	12.24%	1.24%	4.82%	4.95%	13.84%	128.30%	283.33%	54.32%	98.08%	19.94%	37.66%
Ours$_{500\to1}$	**63.49**	**45.72**	**41.36**	**22.38**	**21.37**	**7.41**	**30.67**	**16.83**	**54.61**	**42.40**	**42.30**	**26.95**
ICAE$_{500\to1}$	44.49	27.27	26.65	11.21	14.24	4.45	6.19	2.19	10.24	6.37	20.36	10.30
Absolute Δ	18.99	18.45	14.70	11.16	7.12	2.96	24.48	14.63	44.37	36.03	21.93	16.64
Relative Δ	42.70%	67.66%	55.17%	99.52%	50.03%	66.41%	395.24%	666.14%	432.97%	565.32%	107.70%	161.58%

Table 4: **Cross-domain QA** evaluation results on diverse QA datasets including RelationExtraction (RE), NaturalQuestions (NaturalQ), RACE, TextbookQA, and TriviaQA. Context lengths are reported as average (maximum) token counts.

and compression ratio. While lower compression ratios and shorter contexts generally yield better performance, some cross-domain datasets exhibit interesting results. Notably, TextbookQA and TriviaQA show improved performance at 500→1 compared to 500→16 compression.

Generalization Capability The model's generalization ability is clearly shown in its cross-domain performance. The performance gap between 500xCompressor and ICAE widens at higher compression ratios across all QA datasets. Cross-domain improvements are consistently larger than in-domain gains, reaching up to 107.70% relative improvement in the average F1 at 500→1 compression.

Scalability and Robustness Context length influences performance differently across domains. For in-domain tasks, performance decrease stabilizes with increasing context length. In cross-domain scenarios, longer average context lengths relate with larger improvements, as proved by TextbookQA and TriviaQA showing substantial gains of 395.24% and 432.97% relative improvement respectively over ICAE at 500→1 compression. 500xCompressor demonstrates better robustness to increased compression ratios as well, with average F1 scores decreasing by only 0.42 points from 500→16 to 500→1, compared to ICAE's 15.26-point reduction.

Stability The performance improvements exhibit consistency in both absolute and relative gains across different compression ratios and datasets. This stability is observed in F1 and EM improvements and remains in both in-domain and cross-domain evaluations.

4.4 Case Study

Table 5 presents comparative examples of text regeneration and QA pairs among 500xCompressor and baselines. The examples verify previous findings that baselines demonstrate higher rates of information loss, mistakes, and hallucinations compared to 500xCompressor in both text regeneration and QA tasks.

While regeneration quality typically relates with QA performance, the examples reveal exceptions where accurate text regeneration leads to incorrect QA responses, and imperfect regeneration pro-

Original	500xCompressor (ours)	ICAE (baseline)
We show that every reciprocity sheaf gives rise to a cycle (pre)module in the sense of Rost over a perfect field. Over a perfect field of positive characteristic, we show that the first cohomology group of a logarithmic de Rham-Witt sheaf has a partial cycle module structure. As a consequence, we show that Kato complexes of logarithmic de Rham-Witt sheaves satisfy functoriality properties similar to Rost's cycle complexes.	We show that every reciprocity sheaf gives rise to a cycle (pre)module in the sense of Rost over a perfect field. Over a perfect field of positive characteristic, we show that the first cohomology group of a logarithmic de Rham-Witt cycle module has a partial cycle structure. As a consequence, we show that Kato modules of logarithmic de Rham-Witt complexes satisfy functorial properties similar to Rost's cycle complexes.	We show that every sheaf reciprocity gives rise to a cycle (pre)module *in the sense of Rost* over a *perfect field* Rost cycle. In the perfect field case, we show that over a positive characteristic field, the first logarithmic de Rham-*Witt sheaf* cohomology group of a Witt log-Witt cycle has a partial *cycle module structure* decomposition. As a consequence, we show that Kato's cycle complexes *of logarithmic de Rham-Witt sheaves* satisfy functoriality properties similar to Rost cycle complexes.
Q: What type of sheaf gives rise to a cycle premodule?		
A: Every reciprocity sheaf.	**A:** a reciprocity sheaf	**A:** a *reciprocity* sheaf of (logarithmic) differential forms
Q: Over what type of field do we show that Kato complexes satisfy functoriality properties?		
A: Over a perfect field of positive characteristic.	**A:** perfect fields of positive characteristic	**A:** a perfect field of *positive* characteristic zero
Q: What is the structure of the first cohomology group of a logarithmic de Rham-Witt sheaf?		
A: a partial cycle module structure	**A:** a partial cycle complex	**A:** a partial Kato cycle complex
Q: What type of complexes satisfy functoriality properties similar to Rost's cycle complexes?		
A: Kato complexes of logarithmic de Rham-Witt sheaves	**A:** Kato-Witt cycle complexes *of logarithmic de Rham-Witt sheaves*	**A:** Kato's complexes *of logarithmic de Rham-Witt sheaves*

Table 5: Examples of regenerated texts and QA pairs provided by 500xCompressor and ICAE. 96 tokens of the original text were compressed, which were then used for QA. Differences between the gold standard and the output include mistakes (red, containing incorrect text), information loss (*yellow and italic*, missing some text), hallucinations (blue, including text not present in the original), and paraphrasing (green, rephrasing the original text).

duces correct answers. This observation highlights the relationship between compression quality and task performance.

4.5 Ablation Studies

The performance of compression models is influenced by several variables, including the compression method (ICAE or 500xCompressor), task type (in-domain or cross-domain datasets), context length (length of text to be compressed), and compression ratio (number of compression tokens). The influences of these variables have been discussed in Section 4.3. To further analyze the influence of semantic information, we compare performance on original ArxivCorpus texts versus semantically meaningless texts.

Table 6 demonstrates that semantic understanding improves compression quality, with 500xCompressor achieving 99.48% BLEU score on ArxivCorpus texts compared to 11.77% on random texts. The performance gap keeps in semantically meaningless scenarios, where 500xCompressor maintains an advantage over ICAE (11.77% versus 2.06%), showing robust and improved preservation of both semantic and format-related information.

Dataset	**Arxiv**		**Random**	
Length	96	192	96	192
$\text{Ours}_{500\rightarrow16}$	**99.48**	**96.21**	**11.77**	**2.78**
$\text{ICAE}_{500\rightarrow16}$	81.85	55.90	2.06	0.84
Absolute Δ	17.62	40.31	9.70	1.93

Table 6: Text regeneration performance (BLEU scores) on semantic (ArxivCorpus) and non-semantic (Random) texts. Random texts are generated by increasing each token ID from the ArxivCorpus texts by one position. Arxiv is ArxivCorpus.

5 Discussions

The differences between 500xCompressor and ICAE could be better understood by comparing them to Prompt Tuning (Lester et al., 2021) and Prefix Tuning (Li and Liang, 2021). In Prompt Tuning, prefixed special tokens are trained to guide the model in completing specific downstream tasks.

Similarly, ICAE compresses contexts into prefixed special tokens for downstream tasks. Unlike Prompt Tuning, Prefix Tuning also trains the KV values associated with the prefixed special tokens. 500xCompressor, akin to Prefix Tuning, compresses texts into the KV values of prefixed special tokens. In Prompt Tuning or Prefix Tuning, the prefixed special tokens (and their KV values) only save the instruction for the downstream task. However, in ICAE and 500xCompressor, these tokens compress detailed information within the context in addition to the instruction.

There are three ways to understand the compressed tokens generated from natural language tokens: as memory, a new information format, and a new LLM language. Ge et al. (2024) associated text compression with working memory, viewing compressed tokens as an efficient way for LLMs to store knowledge. Cheng et al. (2024) interpreted text compression as a new information format, where compressed tokens, combined with natural language tokens, provide more information and have higher information richness. Jiang et al. (2023a) treated the compressed prompt as a new language for LLM. There are three elements that define a language: saving information, transmitting information, and adaptive evaluation. The compressed tokens could regenerate the original text, indicating that the information has been saved. Furthermore, these tokens can be used for downstream tasks and answer related questions, demonstrating their ability to transfer information. The ability of the compression models to process unseen texts further shows their generalization ability and adaptability. These characteristics make compressed tokens an efficient new language for LLMs.

6 Related Work

This work is related to prompt compression. There are two main approaches to reducing the number of prompt tokens: hard prompts and soft prompts.

Hard prompt methods identify and delete low-information content in the prompt. Li et al. (2023) proposed SelectiveSentence in 2023, which identifies rich-information content at the sentence or word level. Later, Jiang et al. (2023a) proved that LLMs could understand incomplete words or sentences, leading to the development of LLMLingua, LongLLMLingua, and LLMLingua-2, which delete useless tokens even if fluency is interrupted (Jiang et al., 2023a,b; Hu et al., 2022).

Soft prompt methods compress natural language tokens into a small number of special tokens. Wingate et al. (2022) optimized the difference between the answers generated by the original prompt and the compressed prompt, but this method lacked generalization, requiring training for each new prompt. Mu et al. (2024) solved this by proposing GIST tokens, but their limitations included the need for fine-tuning the original LLM and the short length of prompts to be compressed, typically less than thirty tokens. ICAE solved these issues by pretraining the compression model and avoiding additional parameters during decoding, allowing compression of texts up to around 500 tokens without changing the original LLM (Ge et al., 2024). However, the maximum compression ratio of ICAE is about 15x. To increase the text length for compression, Chevalier et al. (2023) proposed AutoCompressor, which progressively compresses the prompt but, like GIST tokens, is limited to fine-tuned LLMs and a complex training process. Other works analyze text compression within paragraphs (Ren et al., 2023). Soft prompts are also applied in RAG through xRAG and COCOM (Cheng et al., 2024; Rau et al., 2024).

It is worth noting that 500xCompressor is fundamentally a prompt compression method based on the soft prompt rather than a KV cache compression approach. While the KV values of compression tokens are used for inference, they remain unchanged throughout the process, with all compression processes done on the input prompts.

7 Conclusions

This paper proposes 500xCompressor, a prompt compression method capable of compressing any text and all tokens within it. 500xCompressor achieves a high compression ratio while retaining most capabilities of non-compressed prompts. This method proves that current prompts are highly compressible, developing further direction in compression applications.

Future work would involve applications such as in-context learning, personalization, and RAG. 500xCompressor has shown good generalization ability on cross-domain tasks and increasing the size and diversity of the training data is expected to make 500xCompressor be able to finish more tasks (for example, tasks requiring flexible formats and long outputs) and achieve better results.

Limitations

A consideration in our work was the careful selection of training data to avoid copyright issues. We chose to use the ArxivCorpus rather than datasets like Pile, as Arxiv papers are officially uploaded with clear copyright through Cornell University. Future development should also carefully consider copyright when using different datasets.

Ethics Statement

No ethical approval was required for this study.

Availability Statement

The codes related to this study have been uploaded to the open source community at https://github.com/ZongqianLi/500xCompressor.

References

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. *arXiv preprint arXiv:2405.13792*.

Alexis Chevalier, Alexander Wettig, Anirudh Ajith, and Danqi Chen. 2023. Adapting language models to compress contexts. In *The 2023 Conference on Empirical Methods in Natural Language Processing*.

Adam Fisch, Alon Talmor, Robin Jia, Minjoon Seo, Eunsol Choi, and Danqi Chen. 2019. MRQA 2019 shared task: Evaluating generalization in reading comprehension. In *Proceedings of 2nd Machine Reading for Reading Comprehension (MRQA) Workshop at EMNLP*.

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*.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, ..., and Zhiyu Ma. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Huiqiang Jiang, Qianhui Wu, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. 2023a. 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. Association for Computational Linguistics.

Huiqiang Jiang, Qianhui Wu, Xufang Luo, Dongsheng Li, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. 2023b. Longllmlingua: Accelerating and enhancing llms in long context scenarios via prompt compression. *arXiv preprint arXiv:2310.06839*.

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. Association for Computational Linguistics.

Aniruddha Kembhavi, Minjoon Seo, Dustin Schwenk, Jonghyun Choi, Ali Farhadi, and Hannaneh Hajishirzi. 2017. Are you smarter than a sixth grader? textbook question answering for multimodal machine comprehension. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5376–5384.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466.

Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. 2017. RACE: Large-scale ReAding comprehension dataset from examinations. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 785–794, Copenhagen, Denmark. Association for Computational Linguistics.

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. Association for Computational Linguistics.

Omer Levy, Minjoon Seo, Eunsol Choi, and Luke Zettlemoyer. 2017. Zero-shot relation extraction via reading comprehension. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 333–342, Vancouver, Canada. Association for Computational Linguistics.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597, Online. Association for Computational Linguistics.

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. Association for Computational Linguistics.

Jesse Mu, Xiang Lisa Li, and Noah Goodman. 2024. Learning to compress prompts with gist tokens. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, Red Hook, NY, USA. Curran Associates Inc.

Zhuoshi Pan, Qianhui Wu, Huiqiang Jiang, Menglin Xia, Xufang Luo, Jue Zhang, Qingwei Lin, Victor Rühle, Yuqing Yang, Chin-Yew Lin, H. Vicky Zhao, Lili Qiu, and Dongmei Zhang. 2024. LLMLingua-2: Data distillation for efficient and faithful task-agnostic prompt compression. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 963–981, Bangkok, Thailand. Association for Computational Linguistics.

David Rau, Shuai Wang, Hervé Déjean, and Stéphane Clinchant. 2024. Context embeddings for efficient answer generation in rag. *arXiv preprint arXiv:2407.09252*.

Siyu Ren, Qi Jia, and Kenny Q. Zhu. 2023. Context compression for auto-regressive transformers with sentinel tokens. In *The 2023 Conference on Empirical Methods in Natural Language Processing*.

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. Association for Computational Linguistics.

A Appendices

A.1 Model Training

The training parameters for 500xCompressor and ICAE are detailed in Table 7. Main packages are torch 2.3.1 and transformers 4.42.3, and the full environment can be got from Section 7. The evaluation losses for both models are illustrated in Figure 4. All models have successfully converged, with 500xCompressor demonstrating better performance compared to ICAE, as indicated by the evaluation loss.

	Pretraining		**Finetuning**	
	500→16	500→1	500→16	500→1
Total steps	42000	103800	20000	10000
Warm-up steps	300	300	300	300
Learning rate	1e-4	1e-4	5e-5	5e-5
Batch size	4	4	4	4
Optimizer	AdamW	AdamW	AdamW	AdamW

Table 7: Training parameters for 500xCompressor and ICAE.

A.2 ArxivCorpus and ArxivQA Dataset

Source of Arxiv abstracts in the ArxivCorpus: https://www.kaggle.com/datasets/Cornell-University/arxiv

The detailed information for ArxivCorpus and the ArxivQA dataset is shown in Table 8.

The prompt to generate the QA pairs:

```
context: {truncated_context}
task: design the {number} best extractive
 question answering pairs for the context
 to test information loss
requirement: the question should be
direct; the question should try to use
the same words in the context; the answer
 should directly appear in the context (a
 span of the context); the answer should
not be in the question; just output the
results in format and do not output other
 words
output json format: [{"id":1, "question":
 "", "answer": ""}, {"id":2, "question":
"", "answer": ""}, ...]
```

A.3 Question Answering

Prompt for QA (Instruct):

```
Please finish the extractive question
answering task. Just output the answer.
Context: {context} Question: {question}
Answer:
```

This paper and codes are helped with ChatGPT and Claude.

	ArxivCorpus			ArxivQA Dataset		
	Train	Development	Test	Train	Development	Test
Number of data records	2353924	3000	2500	250000	2500	5000
Knowledge cutoff	Pre 07/2023	01-04/2024	01-04/2024	Pre 07/2023	Pre 07/2023	01-04/2024
Source		Abstracts from Arxiv		Train set of ArxivCorpus		Test set of ArxivCorpus

Table 8: Detailed information about the ArxivCorpus and the ArxivQA dataset.

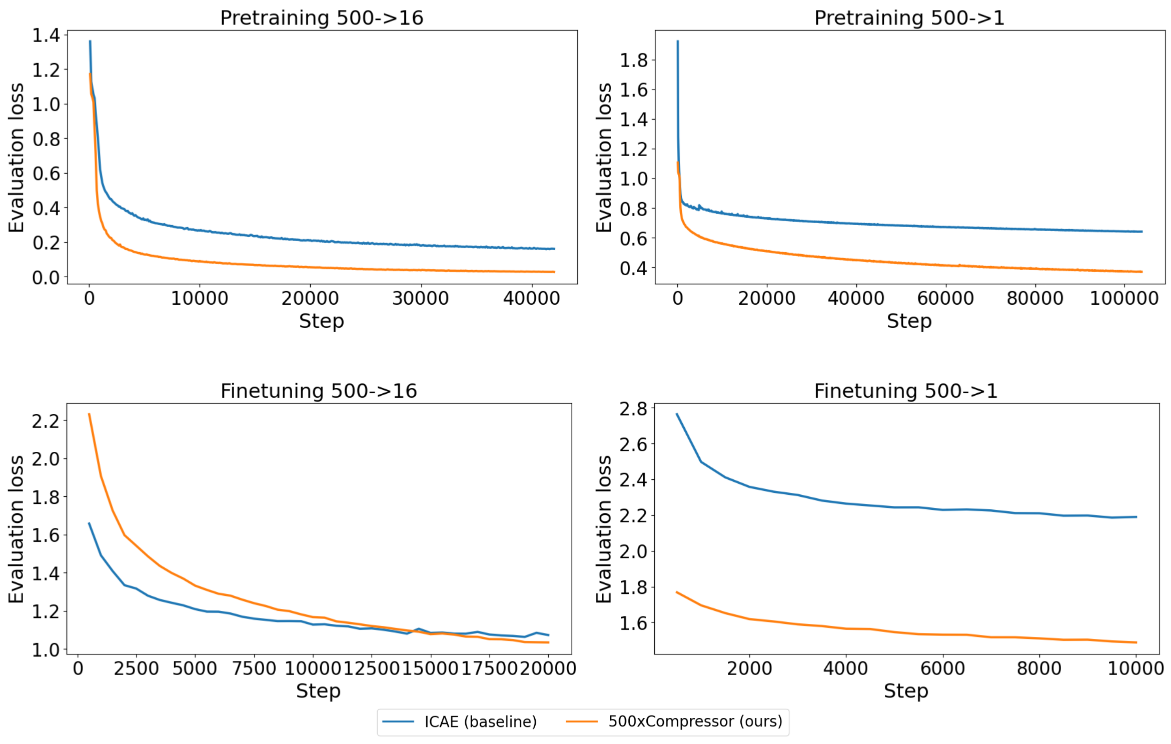

Figure 4: Evaluation loss for 500xCompressor and ICAE during pretraining and fine-tuning.