

Less Is More: Rule-Based Context Shortening Improves Small-Model RAG Accuracy

Teja Velagala
Founder, Researcher
teja@tealsquare.in

Ganesh F Bhovivaddar
Research Intern
ganeshfb2729@gmail.com

TealSquare Technologies Private Limited
<https://tealsquare.in>

Abstract

Retrieval-augmented generation (RAG) systems pass retrieved document chunks to a language model word for word. A substantial share of that text is grammatical scaffolding: articles, auxiliary verbs, prepositions, pronouns, and stock phrases such as “it is important to note that”. These words assist a human reader but carry little of the information needed to answer a question, and each one costs the model attention and time. Earlier work on prompt compression removes such material using a second neural model that scores or rewrites the context. This report examines a simpler question: what happens if the retrieved context is shortened with a fixed word list and a handful of regular expressions, without a model at all? We describe a three-pass rule-based shortener and compare it with verbatim context on three question-answering benchmarks that differ in domain and reasoning type, using a small open-weight generator running on a laptop. We measure answer accuracy, the amount of context removed, and end-to-end latency, examine where the method helps and where it performs poorly, and discuss what this implies for small-model RAG deployments. The complete pipeline and results are released under the Apache 2.0 licence.

1 Introduction

Language models answer more reliably when the prompt contains the relevant evidence, but adding more text does not continue to help. [Levy et al. \(2025\)](#) kept the total context length fixed and found that distributing the same evidence across more documents lowers accuracy for most models, while [Jin et al. \(2024\)](#) showed that answer quality initially improves and then declines as additional passages are retrieved. [Liu et al. \(2024\)](#) showed that accuracy is highest when the relevant passage appears at the start or end of a long prompt and drops sharply when it appears in the middle. [Paulsen \(2025\)](#) examined the context lengths at which models remain reliable and found that this practical limit is considerably lower than the advertised context window.

These findings have prompted work on *prompt compression*, which aims to reduce the number of tokens a model reads without removing the information required to answer. Existing approaches include learned token pruning ([Jiang et al., 2023](#); [Pan et al., 2024](#)), self-information scoring ([Li et al., 2023](#)), and trained extractive or abstractive compressors ([Hwang et al., 2025](#); [Xu et al., 2024](#)). Each of them requires either a reference language model to judge token importance or a dedicated compression model trained on distilled data.

We examine how effective a simple form of compression can be. Stop word removal dates back to Luhn’s common-word lookup table (Luhn, 1958) and was a standard initial step in information retrieval and text classification for decades (Manning et al., 2008; Sarica and Luo, 2021; Silva and Ribeiro, 2003). These methods require no model or training data and execute in microseconds. We are not aware of any study that measures how these steps affect RAG accuracy when applied to the retrieved context immediately before it reaches the generator.

This report presents that measurement. We evaluate **full context**, where the retrieved chunks are used verbatim, against **shortened context**, where stop words, auxiliary verbs, adverbs, filler phrases, and pronouns are removed, across three benchmarks spanning factual, numerical, and scientific question answering. All experiments use a 4B-parameter Qwen3.5 generator on a single laptop.

Our contributions are:

1. A measurement of how rule-based context shortening changes RAG accuracy for a 4B model on SQuAD 2.0, DROP, and Qasper, with a breakdown of where it helps and where it performs poorly.
2. A timing analysis that distinguishes the cost of the shortening step from the change in generation latency.

2 Related Work

Prompt compression. LLMLingua (Jiang et al., 2023) scores each token’s perplexity using a small reference model and removes the low-perplexity tokens, reaching up to $20\times$ compression with a minor loss in task performance. LLMLingua-2 (Pan et al., 2024) reframes compression as token classification, trained on labels distilled from GPT-4, and reports 2 to $5\times$ compression. LongLLMLingua (Jiang et al., 2024) ranks documents according to how well they predict the question before pruning tokens, allowing the method to scale to long contexts. Selective Context (Li et al., 2023) prunes lexical units using self-information, defined as the negative log probability under a base model. All of these approaches are *model-based*, because a reference model must process the input before it reaches the main model. Li et al. (2025) survey the area and divide it into hard compression, which selects or rewrites discrete text, and soft compression, which learns compact embeddings. Our method represents hard compression in its simplest form, using a fixed word list and no model.

Compressing retrieved context for RAG. RECOMP (Xu et al., 2024) trains extractive and abstractive compressors that condense retrieved documents before they are inserted into the prompt. EXIT (Hwang et al., 2025) uses context-aware extractive compression to retain the sentences that matter for a given query. AutoCompressors (Chevalier et al., 2023) learn to fold long contexts into summary vectors, while the In-Context Autoencoder (Ge et al., 2024) maps spans of text to a small number of memory slots. These approaches require additional model capacity in exchange for improved compression quality. We examine whether a zero-cost rule-based alternative can match or outperform verbatim context for a small generator.

Noise in retrieved context. Jin et al. (2024) trace the accuracy decline from longer retrieval lists to hard negatives, passages that appear relevant but are not. Cuconasu et al. (2024) show that retrieved passages that appear relevant but do not contain the answer reduce accuracy, and Yoran et al. (2024) fine-tune the generator on mixed relevant and irrelevant passages so that it learns to ignore the distractors. The common thread is that material that does not help answer the question can still interfere with finding the material that does. Our method operates at the

word level rather than the passage level, removing syntactic noise within relevant passages instead of discarding entire passages.

Stop words in NLP. Stop word removal has been part of information retrieval since [Luhn \(1958\)](#) deleted pronouns, prepositions, and articles by table lookup, and it remained standard in classical retrieval systems even though modern web engines have largely dropped it ([Manning et al., 2008](#)). [Silva and Ribeiro \(2003\)](#) reported that stop word removal has a notable effect on the recall of text categorisers. [Sarica and Luo \(2021\)](#) developed a domain-specific stop word list for technical language processing. We have found no prior work that examines whether a language model can still read and answer questions from text with its grammatical scaffolding removed.

3 Method

3.1 Overview

For each question, we first retrieve the top three chunks using hybrid search (FAISS dense retrieval plus BM25) and then assess them with two strategies:

- **Strategy A, full context:** concatenate the three chunks verbatim and provide them to the model as a prompt.
- **Strategy B, shortened context:** apply the rule-based shortener to each chunk, concatenate the shortened chunks, and prompt the same model using the same question.

Both strategies use the identical retrieved chunks, the same model, and the same prompt template. The only difference is whether the context words are filtered.

3.2 Rule-based shortening

The shortener performs three passes over each chunk.

Pass 1: filler phrase replacement. A list of more than forty multi-word phrases is matched using case-insensitive regular expressions and replaced with a shorter equivalent or removed (Table 1).

Table 1: Examples of filler phrase substitutions. The full list contains more than forty entries.

Filler phrase	Replacement
in order to	to
due to the fact that	because
it is important to note that	(removed)
a large number of	many
with regard to	regarding
there is / there are	(removed)

Pass 2: word-level stop word removal. Each remaining word is converted to lowercase, stripped of punctuation, and checked against a set of more than 180 stop words grouped as *articles* (a, an, the), *auxiliary verbs* (is, are, was, were, has, have, had, do, does, did, and the modals), *prepositions* (of, in, on, at, to, for, by, from, with, and so on), *conjunctions* (and, or, but, so), *pronouns* (it, this, that, they, he, she, and so on), *adverbs* (very, really, quite, just, also, approximately, significantly, and so on), and *determiners* (some, any, each, every). Words outside the set are retained with their original casing and punctuation.

Pass 3: cleanup. Orphaned punctuation is reattached (“word , next” becomes “word, next”), sequences of spaces are collapsed, and leading punctuation is removed.

Example.

Full: “The Berlin Wall was a guarded concrete barrier that divided Berlin from 1961 to 1989. The wall fell on November 9, 1989.”

Shortened: “Berlin Wall guarded concrete barrier divided Berlin 1961 1989. wall fell November 9, 1989.”

The output retains content words (nouns, main verbs, numbers, proper nouns) and sentence-final punctuation. It reads like a telegram, but all the facts remain intact.

3.3 Evaluation

Every question is scored; none is filtered out. Accuracy is calculated as the fraction of correct answers, with a 0.8 B Qwen3.5 model serving as the judge (Es et al., 2024; Zheng et al., 2023) to determine whether the predicted answer is semantically equivalent to the gold answer. F1 and ROUGE-L are presented as secondary metrics. We also record per-question generation time, shortening time, and context length measured in words and tokens.

4 Experimental Setup

4.1 Datasets

We evaluate three benchmarks selected to provide variation in domain, context length, and reasoning type (Table 2).

Table 2: Evaluation datasets, with 200 questions each.

Dataset	Domain	N	Reasoning type
SQuAD 2.0 (Rajpurkar et al., 2018)	Wikipedia	200	Extractive span
DROP (Dua et al., 2019)	Mixed	200	Numerical / discrete
Qasper (Dasigi et al., 2021)	NLP papers	200	Scientific information
Total		600	

SQuAD 2.0 pairs Wikipedia paragraphs with extractive answers and includes questions that cannot be answered from the passage. DROP requires discrete reasoning such as counting, sorting, and arithmetic over narrative paragraphs. Qasper contains information-seeking questions written over full NLP papers, with answers linked to evidence spans in the paper. Together they cover compact factual text (SQuAD, 411 words per retrieved chunk set on average), longer narrative passages (DROP, 488 words), and dense scholarly prose (Qasper, 209 words).

4.2 Model stack

All models used in the pipeline are Apache 2.0 licensed (Qwen Team, 2026; Zhang et al., 2025) and are executed locally through Ollama (Ollama Inc., 2026) (Table 3).

The experiments were conducted on an Apple M3 Pro laptop with 36 GB of unified memory, 12 cores, and Metal acceleration. Generation used a temperature of 0.1, a 128-token limit, and an instruction to answer in one to five words. Retrieval employed a hybrid of FAISS HNSW (Douze et al., 2024; Malkov and Yashunin, 2020) and BM25 (Robertson and Zaragoza, 2009) with $\alpha = 0.5$ and $k = 3$.

Table 3: Models used in the pipeline. All models are Apache 2.0 licensed.

Role	Model	Parameters
Answer generation	Qwen3.5	4 B
Document embedding	Qwen3-Embedding	0.6 B
Answer judge	Qwen3.5	0.8 B

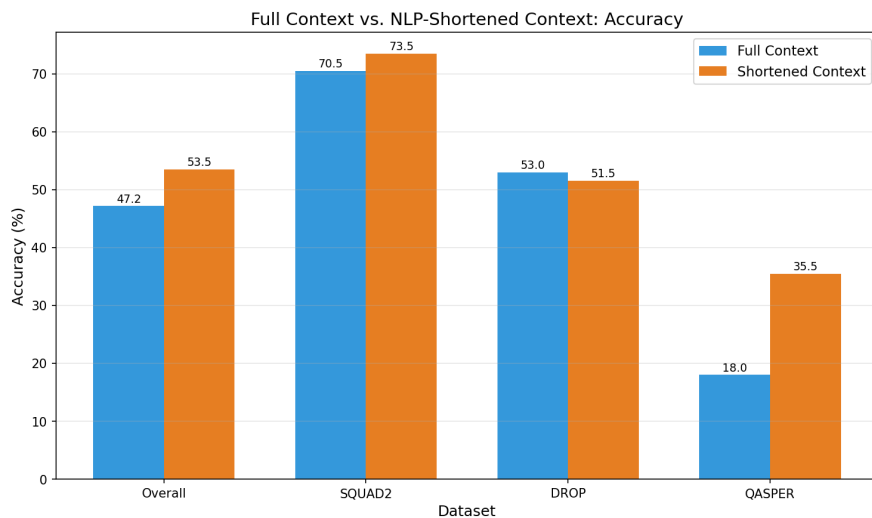
5 Results

5.1 Overall accuracy

Table 4 presents the complete results. Shortened context increases overall accuracy from 47.2% to 53.5%, representing a gain of 6.3 percentage points (13.3% relative). The trend remains the same on two of the three datasets, and the Qasper gain drives the overall result.

Table 4: Full versus shortened context: accuracy, context length, and generation time per question.

Dataset	Accuracy (%)		Avg. words		Avg. time (s)	
	Full	Short	Full	Short	Full	Short
SQuAD 2.0	70.5	73.5	411	260	1.53	1.24
DROP	53.0	51.5	488	315	1.79	1.42
Qasper	18.0	35.5	209	139	1.09	0.99
Overall	47.2	53.5	369	238	1.47	1.22

**Figure 1:** Accuracy by dataset. Shortened context matches or beats full context everywhere except DROP.

5.2 Per-dataset analysis

SQuAD 2.0 (+3.0 points). Wikipedia paragraphs are somewhat wordy. Shortening removes 35.7% of the words, mainly articles, auxiliary verbs, and prepositions, while entity names, dates, and factual claims remain. The 4 B model identifies the answer more reliably in the shortened text. We interpret this as the answer span being closer to the surrounding content words, leaving the model with less information to connect.

DROP (−1.5 points). DROP requires counting, comparison, and arithmetic over narrative passages, for example “How many touchdowns were scored in the first half?”. Shortening removes 35.4% of the words, and some are function words that mark structure on which the reasoning depends: “in the first half” becomes “first half”. The small decrease suggests that some syntactic structure carries useful information for reasoning tasks, which a fixed word list cannot reliably distinguish from noise.

Qasper (+17.5 points). This represents the largest improvement in the study. Qasper contexts are sections of NLP papers, filled with academic filler such as “it is worth noting that”, “in the context of”, and “we observe that”. Shortening removes only 20.9% of the words here, less than on the other datasets because academic text contains a higher share of content words, but the words it removes are the least informative ones. At full length the 4 B model struggles with these passages. Once the filler is removed, the facts the model needs are much easier to locate.

5.3 Context reduction

Across all 600 questions, shortening removes an average of 30.7% of the words (median 35.4%) and 22.9% of the tokens (Figure 2). The word count decreases more than the token count because most stop words correspond to single tokens, while many content words are divided into multiple tokens. DROP shows the largest absolute reduction (488 to 315 words) because narrative text contains more function words than technical text.

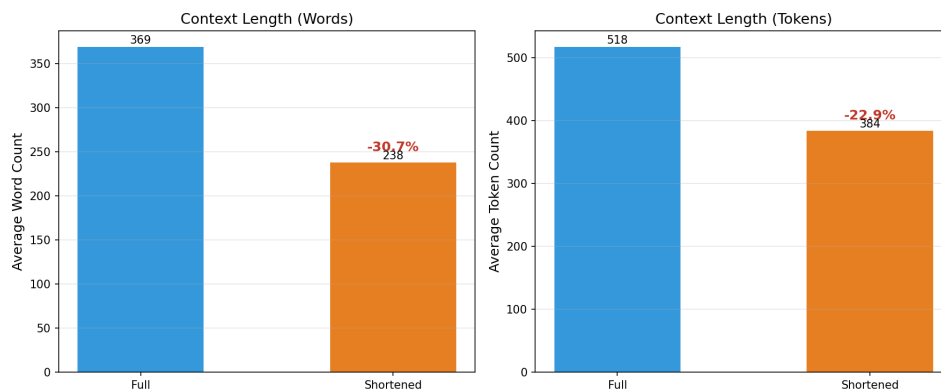


Figure 2: Average context length before and after shortening.

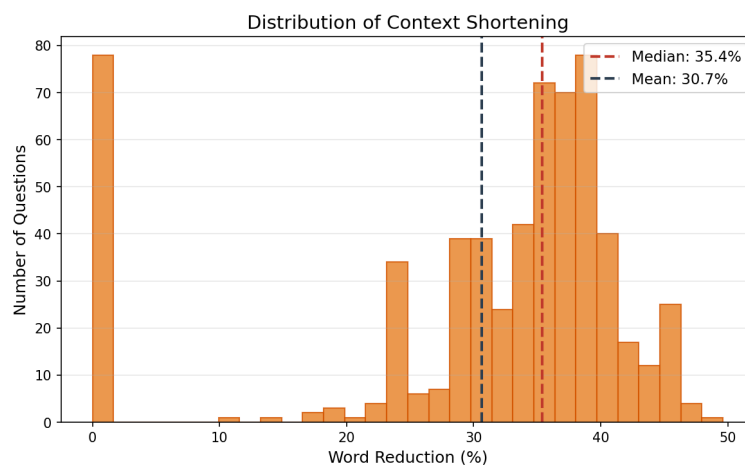


Figure 3: Distribution of word reduction among the 600 questions.

5.4 Timing

The shortening step takes an average of 3.1 ms per question, which is negligible compared with generation (Figure 4). Generation time decreases from 1.47s with full context to 1.22s with shortened context, representing a 17.2% reduction. The speedup follows the token reduction: fewer input tokens mean fewer key-value cache entries and less attention work at each decoding step. The shortened pipeline is therefore more accurate and faster from end to end.

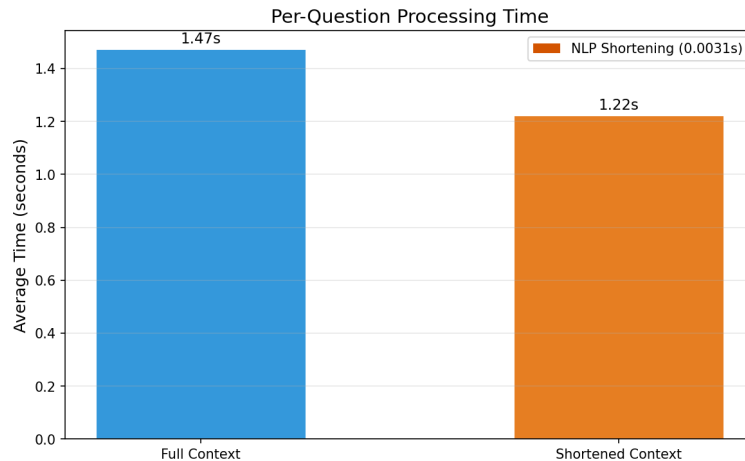


Figure 4: Processing time per question. Shortening adds about 3 ms and reduces generation time by 17%.

6 Discussion

Why does shortening help? We identify two mechanisms that likely work together. The first is *noise reduction*. Function words and filler phrases consume attention without contributing to the answer. A 4 B model has limited capacity to distribute across a long prompt, and removing the noise leaves more of it for the content words. This agrees with the observations of [Levy et al. \(2025\)](#) and [Liu et al. \(2024\)](#) that models have trouble extracting signal from long, cluttered contexts. The second mechanism is *answer proximity*: shortening brings the answer tokens closer to the surrounding content words, reducing the distance the model needs to span. This matters most for small models whose positional handling degrades over long spans.

Why does DROP lose a little? Numerical reasoning relies on structure. “In the first half” encodes a temporal constraint that “first half” alone expresses less clearly. Prepositions and articles that are noise for fact lookup can provide signal for discrete reasoning. Keeping function words near numbers and time expressions could therefore help a shortener reduce this gap.

Comparison with model-based compression. LLMingua achieves up to $20\times$ compression with a drop of about 1.5 points in exact match on GSM8K ([Jiang et al., 2023](#)), at the cost of running a reference model over every prompt. Our rule-based approach achieves only about $1.4\times$ compression, but it improves accuracy, adds 3 ms of latency, and requires no model at all. These approaches can be combined, with rule-based shortening first providing a cleaner and shorter context for a model-based compressor.

Practical implications. For deployments where compute is limited, such as edge devices, air-gapped systems, or applications that pay per token, rule-based shortening is nearly free: it improves accuracy, lowers latency, and requires no additional infrastructure. It fits into any RAG pipeline as a text-processing step between retrieval and generation.

7 Limitations

1. We evaluated a single 4 B generator. Larger models with greater attention capacity may gain less from noise reduction, or may be affected by the loss of syntax.
2. The stop word list is fixed and English-only. Morphologically rich languages require different lists, and domain-specific stop words (Sarica and Luo, 2021) may perform better than a general one.
3. The same shortening rules were applied to each dataset. Adaptive rules, such as keeping prepositions near numbers, could eliminate the DROP regression.
4. Correctness is judged by a 0.8 B model. A larger judge or human raters could change the absolute accuracy numbers.
5. The one-to-five-word answer format disadvantages datasets whose gold answers are longer.

8 Conclusion

Removing stop words, auxiliary verbs, adverbs, and filler phrases from retrieved context increased RAG accuracy by 6.3 percentage points for a 4 B model across three benchmarks, while reducing context length by 31% and generation time by 17%. The shortening step takes about 3 ms, so it adds no measurable cost to a pipeline. The gain is greatest on verbose scientific text (Qasper, +17.5 points) and becomes a small loss on numerical reasoning (DROP, −1.5 points).

For a small model, the useful information in retrieved text is mainly conveyed by its content words, while the surrounding grammar can interfere more than it helps. Next steps include adaptive shortening that preserves structural cues for reasoning tasks, tests on larger models to determine where the benefit fades, and combining rule-based preprocessing with model-based compression for deeper reductions.

References

- Alexis Chevalier, Alexander Wettig, Anirudh Ajith, and Danqi Chen. Adapting language models to compress contexts. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3829–3846, 2023. doi: 10.18653/v1/2023.emnlp-main.232. URL <https://aclanthology.org/2023.emnlp-main.232/>. arXiv:2305.14788.
- Florin Cuconasu, Giovanni Trappolini, Federico Siciliano, Simone Filice, Cesare Campagnano, Yoelle Maarek, Nicola Tonellotto, and Fabrizio Silvestri. The power of noise: Redefining retrieval for RAG systems. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 719–729. ACM, 2024. doi: 10.1145/3626772.3657834. arXiv:2401.14887.
- Pradeep Dasigi, Kyle Lo, Iz Beltagy, Arman Cohan, Noah A. Smith, and Matt Gardner. A dataset of information-seeking questions and answers anchored in research papers. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4599–4610, 2021. doi: 10.18653/v1/2021.naacl-main.365. URL <https://aclanthology.org/2021.naacl-main.365/>.
- Matthijs Douze, Alexandr Guzhva, Chengqi Deng, Jeff Johnson, Gergely Szilvasy, Pierre-Emmanuel Mazaré, Maria Lomeli, Lucas Hosseini, and Hervé Jégou. The Faiss library. *arXiv preprint arXiv:2401.08281*, 2024. URL <https://arxiv.org/abs/2401.08281>.
- Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for*

- Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2368–2378, 2019. doi: 10.18653/v1/N19-1246. URL <https://aclanthology.org/N19-1246/>.
- Shahul Es, Jithin James, Luis Espinosa Anke, and Steven Schockaert. 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, 2024. doi: 10.18653/v1/2024.eacl-demo.16. URL <https://aclanthology.org/2024.eacl-demo.16/>.
- Tao Ge, Jing Hu, Lei Wang, Xun Wang, Si-Qing Chen, and Furu Wei. In-context autoencoder for context compression in a large language model. In *Proceedings of the Twelfth International Conference on Learning Representations (ICLR)*, 2024. URL <https://openreview.net/forum?id=uREj4ZuGJE>. arXiv:2307.06945.
- Taeho Hwang, Sukmin Cho, Soyeong Jeong, Hoyun Song, SeungYoon Han, and Jong C. Park. EXIT: Context-aware extractive compression for enhancing retrieval-augmented generation. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 4895–4924, 2025. doi: 10.18653/v1/2025.findings-acl.253. URL <https://aclanthology.org/2025.findings-acl.253/>. arXiv:2412.12559.
- Huiqiang Jiang, Qianhui Wu, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. LLMingua: Compressing prompts for accelerated inference of large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 13358–13376, 2023. doi: 10.18653/v1/2023.emnlp-main.825. URL <https://aclanthology.org/2023.emnlp-main.825/>. arXiv:2310.05736.
- Huiqiang Jiang, Qianhui Wu, Xufang Luo, Dongsheng Li, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. LongLLMingua: 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, 2024. doi: 10.18653/v1/2024.acl-long.91. URL <https://aclanthology.org/2024.acl-long.91/>. arXiv:2310.06839.
- Bowen Jin, Jinsung Yoon, Jiawei Han, and Sercan O. Arik. Long-context LLMs meet RAG: Overcoming challenges for long inputs in RAG. *arXiv preprint arXiv:2410.05983*, 2024. URL <https://arxiv.org/abs/2410.05983>.
- Shahar Levy, Nir Mazon, Lihi Shalmon, Michael Hassid, and Gabriel Stanovsky. More documents, same length: Isolating the challenge of multiple documents in RAG. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 19539–19547, 2025. doi: 10.18653/v1/2025.findings-emnlp.1064. URL <https://aclanthology.org/2025.findings-emnlp.1064/>. arXiv:2503.04388.
- Yucheng Li, Bo Dong, Frank Guerin, and Chenghua Lin. Compressing context to enhance inference efficiency of large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6342–6353, 2023. doi: 10.18653/v1/2023.emnlp-main.391. URL <https://aclanthology.org/2023.emnlp-main.391/>. arXiv:2310.06201.
- Zongqian Li, Yinhong Liu, Yixuan Su, and Nigel Collier. Prompt compression for large language models: A survey. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 7182–7195, 2025. doi: 10.18653/v1/2025.naacl-long.368. URL <https://aclanthology.org/2025.naacl-long.368/>. arXiv:2410.12388.

- Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173, 2024. doi: 10.1162/tacl_a_00638. URL <https://aclanthology.org/2024.tacl-1.9/>. arXiv:2307.03172.
- Hans Peter Luhn. The automatic creation of literature abstracts. *IBM Journal of Research and Development*, 2(2):159–165, 1958. doi: 10.1147/rd.22.0159.
- Yu. A. Malkov and D. A. Yashunin. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(4):824–836, 2020. doi: 10.1109/TPAMI.2018.2889473. arXiv:1603.09320.
- Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, Cambridge, UK, 2008. doi: 10.1017/CBO9780511809071. URL <https://nlp.stanford.edu/IR-book/>.
- Ollama Inc. Ollama. Software, 2026. URL <https://ollama.com>. Source: <https://github.com/ollama/ollama>, MIT licence. Accessed September 2026.
- 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. LLMingua-2: Data distillation for efficient and faithful task-agnostic prompt compression. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 963–981, 2024. doi: 10.18653/v1/2024.findings-acl.57. URL <https://aclanthology.org/2024.findings-acl.57/>. arXiv:2403.12968.
- Norman Paulsen. Context is what you need: The maximum effective context window for real world limits of LLMs. *arXiv preprint arXiv:2509.21361*, 2025. URL <https://arxiv.org/abs/2509.21361>.
- Qwen Team. Qwen3.5: Towards native multimodal agents. Blog post and model release, February 2026. URL <https://qwen.ai/blog?id=qwen3.5>. Model weights: <https://huggingface.co/Qwen/Qwen3.5-4B> and <https://huggingface.co/Qwen/Qwen3.5-0.8B>, Apache 2.0.
- Pranav Rajpurkar, Robin Jia, and Percy Liang. Know what you don’t know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789, 2018. doi: 10.18653/v1/P18-2124. URL <https://aclanthology.org/P18-2124/>.
- Stephen Robertson and Hugo Zaragoza. The probabilistic relevance framework: BM25 and beyond. *Foundations and Trends in Information Retrieval*, 3(4):333–389, 2009. doi: 10.1561/15000000019.
- Serhad Sarica and Jianxi Luo. Stopwords in technical language processing. *PLOS ONE*, 16(8): e0254937, 2021. doi: 10.1371/journal.pone.0254937.
- Catarina Silva and Bernardete Ribeiro. The importance of stop word removal on recall values in text categorization. In *Proceedings of the International Joint Conference on Neural Networks (IJCNN)*, volume 3, pages 1661–1666, 2003. doi: 10.1109/IJCNN.2003.1223656.
- Fangyuan Xu, Weijia Shi, and Eunsol Choi. RECOMP: Improving retrieval-augmented LMs with context compression and selective augmentation. In *Proceedings of the Twelfth International Conference on Learning Representations (ICLR)*, 2024. URL <https://openreview.net/forum?id=mlJLVigNHp>. arXiv:2310.04408.

- Ori Yoran, Tomer Wolfson, Ori Ram, and Jonathan Berant. Making retrieval-augmented language models robust to irrelevant context. In *Proceedings of the Twelfth International Conference on Learning Representations (ICLR)*, 2024. URL <https://openreview.net/forum?id=ZS4m74kZpH>. arXiv:2310.01558.
- Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, Fei Huang, and Jingren Zhou. Qwen3 embedding: Advancing text embedding and reranking through foundation models. *arXiv preprint arXiv:2506.05176*, 2025. URL <https://arxiv.org/abs/2506.05176>.
- Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging LLM-as-a-judge with MT-bench and chatbot arena. In *Advances in Neural Information Processing Systems*, volume 36, pages 46595–46623. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/91f18a1287b398d378ef22505bf41832-Abstract-Datasets_and_Benchmarks.html. NeurIPS 2023 Datasets and Benchmarks Track. arXiv:2306.05685.