A Systematic Analysis of Chunking Strategies for Reliable Question Answering

Sofia Bennani[A] **and Charles Moslonka**[A,B]
[A]Artefact Research Center, Paris, France
[B]MICS, CentraleSupélec, Université Paris-Saclay, Gif-sur-Yvette, France

We study how document chunking choices impact the reliability of Retrieval-Augmented Generation (RAG) systems in industry. While practice often relies on heuristics, our end-to-end evaluation on Natural Questions systematically varies chunking method (token, sentence, semantic, code), chunk size, overlap, and context length. We use a standard industrial setup: SPLADE retrieval and a Mistral-8B generator. We derive actionable lessons for cost-efficient deployment: (i) overlap provides no measurable benefit and increases indexing cost; (ii) sentence chunking is the most cost-effective method, matching semantic chunking up to ~ 5k tokens; (iii) a "context cliff" reduces quality beyond ~ 2.5k tokens; and (iv) optimal context depends on the goal (semantic quality peaks at small contexts; exact match at larger ones).

1. Introduction

Enterprises increasingly deploy RAG systems for knowledge access and user support. In industrial settings, these systems operate under strict constraints regarding latency, storage costs, and maintainability. While agentic workflows are the ultimate goal, reliability hinges on the foundational retrieval layer — particularly how source documents are chunked. Despite ample work on retrieval [1, 2, 3] and LLMs, chunking is often left to rules of thumb. This paper reports an end-to-end, data-driven study conducted to standardize our production defaults. Our contributions are:

- A systematic evaluation of chunking method, size, overlap, and context length for agentic question answering (QA) on Natural Questions (NQ).
- Compact, deployable guidance: avoid overlap; prefer sentence chunking; select context size by task; beware the "context cliff".
- A reliability view that includes abstention ("NONE") rates alongside semantic and exact-match metrics.

2. Experimental Setup

We evaluate a standard two-stage RAG pipeline (Figure 1): a sparse retriever indexes chunked documents; the top-ranked chunks are passed to an instruction-tuned LLM prompted to answer strictly from context and output "NONE" otherwise.

Task and corpus. We use the Natural Questions [4] short-answer subset (open-domain QA). The underlying corpus is English Wikipedia; documents are ingested and chunked according to each strategy below, then indexed.

Retrieval and context budgeting. We use SPLADE [5, 6] (pretrained weights [7]) to build a sparse index over all chunks. At query time, we retrieve the top-ranked chunks and fill a token budget C (context length) using the generator's tokenizer, appending in rank order until the budget is reached. This "fill-to-budget" policy ensures fair comparison across chunk sizes and methods (no fixed-K bias).

Published at the European Conference on Information Retrieval (ECIR) 2026
Corresponding author(s): charles.moslonka@artefact.com
Artefact Research Center, Paris, France

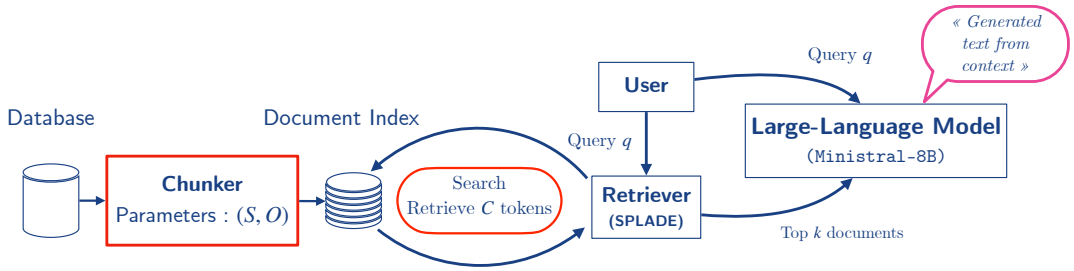


Figure 1 | RAG pipeline architecture with parameters S, O and C.

Chunking strategies.

- Token: fixed-size sliding windows of target size S with optional token overlap O.
- Sentence: respects sentence boundaries; no sentence is split.
- Semantic: sentence-preserving; adjacent sentences are merged if cosine similarity (`all-MiniLM-L12-v2`) exceeds 0.5, up to the target size S.
- Code: structure-aware parsing (e.g., functions/classes) for source code, focused on markdown; included for completeness though NQ is text-centric.

Generation and abstention. We use `Ministral-8B-Instruct-2410` [8] with low-temperature decoding ($T = 0.1$). The prompt enforces grounded generation and explicit abstention: "Answer only using the provided context. If the context is insufficient, output 'NONE'." We cap output length to short answers.

Parameters varied. We evaluate four **methods** (Token, Sentence, Semantic, Code) across a grid of sizes. We test **chunk sizes** S from 50 to 500 (step 50), with **overlaps** O of 0% or 20%. Finally, we retrieve into a **context budget** C of {500, 1k, 2.5k, 5k, 10k} tokens.

Metrics and protocol. We report:

- BERTScore [9] (semantic quality vs. reference answer).
- Exact Match (EM) with standard normalization (lowercasing; stripping punctuation/articles).
- None Ratio: fraction of queries where the model outputs "NONE".

We compute 95% bootstrap confidence intervals over questions; claims of "no measurable difference" indicate paired deltas within CIs (e.g., $|\Delta$ BERTScore$| \leq 0.004$, EM differences ≤ 0.001). We intentionally do not use rerankers or LLM-as-reranker to isolate the effect of chunking and context budgeting.

3. Findings

We report the end-to-end effects that were most consistent and actionable across settings, along with brief mechanisms and deployment implications.

F1. Overlap adds cost without measurable gains. Across paired configurations, adding 10–20% overlap did not improve BERTScore or EM (e.g., $|\Delta$BERTScore$| \leq 0.004$; EM differences ≤ 0.001).

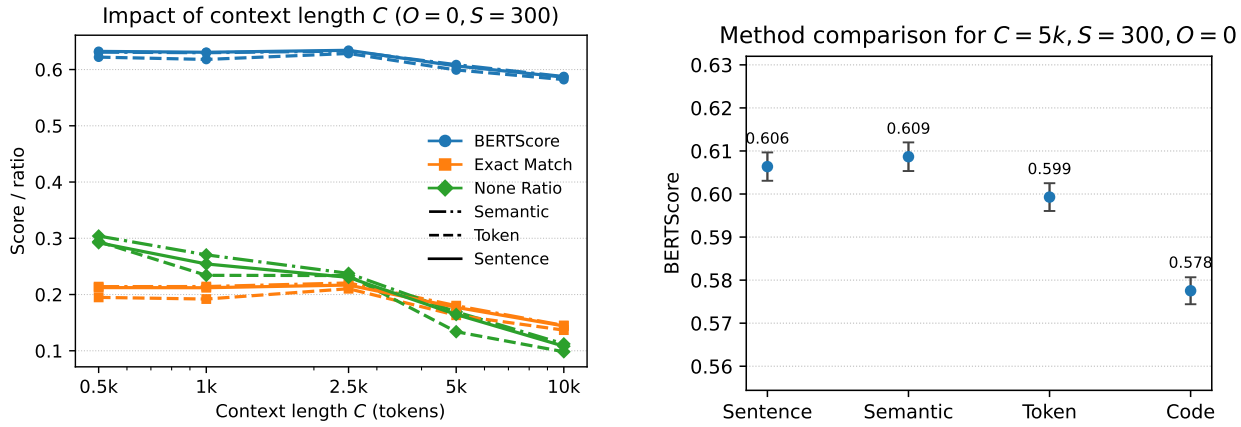


Figure 2 | Left: Effect of context length C on metrics for different chunking methods (Sentence, Semantic, Token). Right: Chunking method comparison at fixed $C = 5000$ tokens and $S = 300$, $O = 0$. Dots show means; bars denote 95% bootstrap CIs.

Mechanism: with a sentence-aware pipeline and a sparse retriever, boundary spillover rarely changes the top-C content; overlap mostly introduces near-duplicates. Cost: for overlap ratio r, chunk count (and index size) inflates by a factor $1/(1-r)$ (e.g., $r = 0.2$ leads to $1.25\times$ more chunks), increasing ingestion time and storage. Recommendation: use $O = 0$ unless you have evidence your retriever benefits from boundary redundancy.

F2. Method "tier list": sentence ≈ semantic > token ≫ code (for text). Sentence and semantic chunking were statistically tied up to ~5k tokens; token chunking lagged; code chunking was not competitive on this text task (Fig. 2). Mechanism: sentence-preserving methods keep topical coherence and reduce cross-sentence fragmentation, improving both retrieval precision and LLM grounding. Semantic merging helps when very large contexts are used (slight edge $C > 5$k), likely by packing semantically contiguous text. Recommendation: default to sentence; consider semantic only for very large C or highly discursive documents.

F3. The "context cliff": more is not always better. Performance improved from small to moderate contexts but dropped beyond ~2.5k tokens. For sentence chunking ($O = 0$, $S = 300$), BERTScore was stable between 0.5k–2.5k tokens and then declined by ~ 4–5% relatively at 10k tokens. Mechanism: long-context LLMs can suffer from distraction and redundancy; retrieval at large budgets introduces overlapping or off-topic chunks, diluting signal. Recommendation: identify and enforce a sweet spot for C; in our setup, $C \approx 2.5$k was a strong default for QA. Note that the exact drop-off point is model-dependent; our values reflect `Ministral-8B-Instruct-2410` and should be re-tuned per LLM. However, the existence of a performance plateau or decline with excessive context is a consistent phenomenon in RAG.

F4. Goal-driven tuning: small C for semantic quality; larger C for factual accuracy; abstention is tunable. BERTScore tended to peak at small, focused contexts (~ 500 tokens), whereas EM peaked at larger contexts (~ 2.5k). None Ratio fell with larger C (e.g., from ~ 30% at 0.5k to ~ 11% at 10k) and rose with larger S. Mechanism: small C concentrates the most relevant evidence (good for semantic faithfulness), while larger C increases recall across disparate mentions (good for EM). Larger chunks reduce the number of distinct contexts retrieved, increasing abstention when narrow

evidence is missed. Recommendation: for summaries/explanations, keep C small; for factoid QA, use $C \approx 2.5$k. To reduce "NONE", increase C and use smaller S.

Table 1 | Practical defaults for agentic QA (text documents).

Choice	Default	Rationale
Overlap O	0%	No measurable benefit; reduces cost/complexity
Chunker	Sentence	Matches semantic up to ~5k tokens; cheaper
Chunk size S	150–300	Balances recall vs. abstention
Context C (QA)	~2.5k	Avoids context cliff; boosts EM
Context C (Summ.)	~500	Maximizes semantic faithfulness
When $C > 5$k	Consider Semantic	Slight edge at very large contexts

Limitations and Future Work. Our study focuses on optimizing the *first-stage* retrieval index, a critical step for latency-sensitive industrial applications. We intentionally excluded rerankers and late-interaction models (e.g., ColBERT) to isolate the effects of chunking on the base retriever; while these methods often improve precision, they incur higher storage and latency costs that must be weighed against their benefits in future work. Furthermore, our results on Natural Questions are most representative of general text-centric corpora. While code chunking remains the clear choice for source code, these findings should be validated on specialized enterprise domains (e.g., legal or technical documentation). Finally, we used low temperature ($T = 0.1$) to minimize generation variance, though key trends persisted under bootstrap resampling.

4. Conclusion

Chunking is a first-order design choice for reliable, cost-effective RAG agents. Our study provides compact, deployable guidance: avoid overlap; default to sentence chunking; tune context to the task; beware the context cliff beyond ~2.5k tokens; and use S and C to control abstention. These defaults have improved the robustness of client-facing agents in practice and offer a baseline for future IR-for-agents evaluations.

Acknowledgments This work was done as part of the ArGiMi project, funded by BPIFrance under the France2030 national effort towards numerical common goods.

References

[1] Manuel Faysse, Hugues Sibille, Tony Wu, Bilel Omrani, Gautier Viaud, Céline Hudelot, and Pierre Colombo. ColPali: Efficient Document Retrieval with Vision Language Models, October 2024.

[2] Keshav Santhanam, Omar Khattab, Jon Saad-Falcon, Christopher Potts, and Matei Zaharia. ColBERTv2: Effective and Efficient Retrieval via Lightweight Late Interaction, July 2022.

[3] Thibault Formal, Carlos Lassance, Benjamin Piwowarski, and Stéphane Clinchant. Towards Effective and Efficient Sparse Neural Information Retrieval. ACM Transactions on Information Systems, 42(5):1–46, September 2024.

[4] 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. Natural Questions: A Benchmark for Question Answering Research. Transactions of the Association for Computational Linguistics, 7:453–466, November 2019.

[5] Thibault Formal, Benjamin Piwowarski, and Stéphane Clinchant. SPLADE: Sparse Lexical and Expansion Model for First Stage Ranking, July 2021.

[6] Thibault Formal, Carlos Lassance, Benjamin Piwowarski, and Stéphane Clinchant. SPLADE v2: Sparse Lexical and Expansion Model for Information Retrieval, September 2021.

[7] Weize Kong, Jeffrey M. Dudek, Cheng Li, Mingyang Zhang, and Michael Bendersky. SparseEmbed: Learning Sparse Lexical Representations with Contextual Embeddings for Retrieval. In Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 2399–2403, Taipei Taiwan, July 2023. ACM.

[8] Mistral AI Team. Un Ministral, des Ministraux | Mistral AI. https://mistral.ai/fr/news/ministraux.

[9] Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. BERTScore: Evaluating Text Generation with BERT. ArXiv, April 2019.