Is Semantic Chunking Worth the Computational Cost?

Renyi Qu
Vectara, Inc.
renyi@vectara.com

Ruixuan Tu
University of Wisconsin–Madison
turx2003@gmail.com

Forrest Bao
Vectara, Inc.
forrest@vectara.com

Abstract

Recent advances in Retrieval-Augmented Generation (RAG) systems have popularized **semantic chunking**, which aims to improve retrieval performance by dividing documents into semantically coherent segments. Despite its growing adoption, the actual benefits over simpler **fixed-size chunking**, where documents are split into consecutive, fixed-size segments, remain unclear. This study systematically evaluates the effectiveness of semantic chunking using three common retrieval-related tasks: document retrieval, evidence retrieval, and retrieval-based answer generation. The results show that the computational costs associated with semantic chunking are not justified by consistent performance gains. These findings challenge the previous assumptions about semantic chunking and highlight the need for more efficient chunking strategies in RAG systems.

1 Introduction

In Retrieval-Augmented Generation (RAG) systems, cutting documents into smaller units called "chunks" has a crucial effect on the quality of both retrieval and generation tasks (Chen et al., 2023; Wadhwa et al., 2024; Shi et al., 2023; Yu et al., 2023). By retrieving the most relevant chunks for a given query and feeding them into a generative language model, these systems aim to produce accurate and contextually appropriate answers. However, the effectiveness of chunking strategies remains a significant challenge in optimizing retrieval quality and computational efficiency (Lewis et al., 2020; Finardi et al., 2024).

Known as **fixed-size chunking**, the traditional way to chunk is to cut documents into chunks of a fixed length such as 200 tokens (Gao et al., 2023). While computationally simple, this approach can fragment semantically related content across multiple chunks, leading to suboptimal retrieval performance. Recently, there has been a surge of interest in **semantic chunking**, where documents are segmented based on semantic similarity, with some industry applications suggesting promising improvements in performance (LangChain, 2024; LlamaIndex, 2024; McCormick, 2024). However, there is no systematic evidence that semantic chunking yields a performance gain in downstream tasks, and if there is, the gain is significant enough to justify the computational overhead than fixed-size chunking.

Such a systematic evaluation is not trivial due to the lack of data that can be directly used to compare chunking strategies. Therefore, we design an indirect evaluation using three proxy tasks: (1) document retrieval, measuring the ability to identify relevant documents; (2) evidence retrieval, measuring the ability to locate ground-truth evidence; and (3) answer generation, testing the quality of answers produced by a generative model using retrieved chunks. Our findings challenge prevailing assumptions about the benefits of semantic chunking, suggesting that its advantages are highly task-dependent and often insufficient to justify the added computational costs. This study lays the groundwork for future exploration of more efficient and adaptive chunking strategies in RAG systems.

In general, our contributions are:

- We present a novel, large-scale evaluation framework comparing semantic and fixed-size chunking across diverse tasks.
- We demonstrate that while semantic chunking shows some benefits in certain scenarios, these are inconsistent and often insufficient to justify the computational cost.

2 Chunking Strategies

In this paper, a document is first split into sentences which are then grouped into chunks. We evaluate three chunking strategies, hereafter referred to as

Findings of the Association for Computational Linguistics:
NAACL 2025, pages 2155–2177

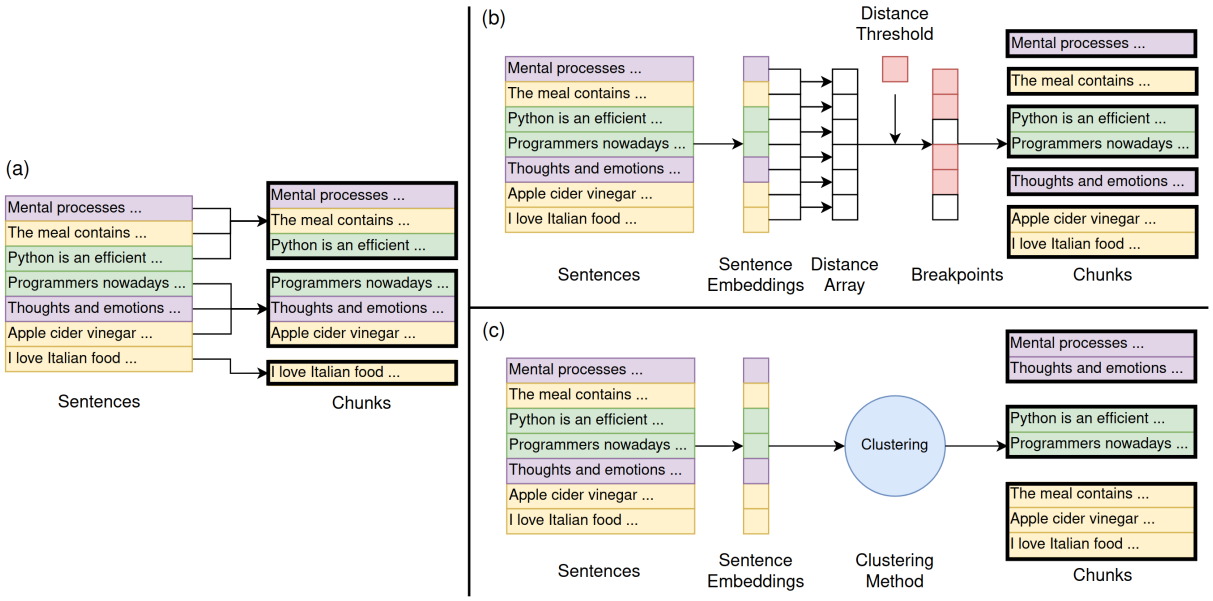

Figure 1: Illustration of the three chunkers tested in this study. Colored segments represent different topics within the sample document: Purple for psychology, Green for programming, and Yellow for food. Red blocks mark chunk breakpoints. (a) Fixed-size Chunker splits the document into consecutive, uniform chunks without considering semantic content. (b) Breakpoint-based Semantic Chunker segments the text by detecting semantic distance thresholds between consecutive sentences to maintain coherence. (c) Clustering-based Semantic Chunker groups semantically similar sentences, potentially combining non-consecutive text to form topic-based chunks.

"chunkers."

Fixed-size Chunker This is our baseline chunker that splits a document sequentially into fixed-size chunks, based on a predefined or user-specified number of sentences per chunk.

Although this approach is simple and computationally efficient, it may separate contextually related sentences, leading to potential degradation in retrieval quality (Lewis et al., 2020; Finardi et al., 2024; Gao et al., 2023). To alleviate this, we use overlapping sentences between consecutive chunks, a common practice to maintain some degree of contextual continuity.

Breakpoint-based Semantic Chunker A breakpoint-based chunker scans over the sequence of sentences and decide where to insert a breakpoint to separate sentences before and after it into two chunks. A breakpoint is inserted if the semantic distance between two consecutive sentences exceeds a thredshold, meaning a significant topic change.

We tested four relative thresholds for determining breakpoints, as proposed by LangChain (2024).Additionally, we tested two absolute thresholds, which use predetermined values to determine chunk boundaries, reducing computational overhead.

However, the breakpoint-based chunkers make decisions using only two sentences each time. This strategy maybe locally greedy. To chunk with more information at a bigger scope, we propose a new type of semantic chunkers next.

Clustering-based Semantic Chunker This type of chunkers leverage clustering algorithms to group sentences together semantically, capturing global relationships and allowing for non-sequential sentence groupings. However, it risks losing losing contextual information hidden in the proximity of sentences. To mitigate this, we defined a new distance measure that combines positional and semantic distances. Specifically, we calculate a weighted sum between the positional distance (i.e., the sentence index difference) and the cosine distance between two sentence $\mathbf{x}_a$ and $\mathbf{x}_b$:

$$d(\mathbf{x}_a, \mathbf{x}_b) = \lambda d_{\text{pos}}(\mathbf{x}_a, \mathbf{x}_b) + (1-\lambda) d_{\cos}(\mathbf{x}_a, \mathbf{x}_b) \quad (1)$$

$$d_{\text{pos}}(\mathbf{x}_a, \mathbf{x}_b) = \frac{|a-b|}{n} \quad (2)$$

$$d_{\cos}(\mathbf{x}_a, \mathbf{x}_b) = 1 - \max(\cos(\texttt{emb}(\mathbf{x}_a), \texttt{emb}(\mathbf{x}_b)), 0) \quad (3)$$

where n is the total number of sentences in the document, $\texttt{emb}(\cdot)$ is the embedding function, and λ is a hyperparameter. When $\lambda = 0$, the chunker operates purely based on semantic similarity; when $\lambda = 1$, it mirrors the Fixed-size Chunker. In Eq. (3), a cosine similarity of 0 indicates orthogonal (unrelated) sentence embeddings, while negative cosine

similarity values are treated as 0, as they do not aid in retrieval or generation.

Without losing generality, we employed single-linkage agglomerative clustering and DBSCAN (Ester et al., 1996) as representatives of clustering algorithms. Further details on these methods an their adjustments during experimentation are provided in Appendix A.

3 Experiments

In the absence of ground-truth chunk data, we designed three experiments to indirectly assess the quality of each chunker: document retrieval, evidence retrieval, and answer generation. Different datasets and evaluation metrics were used for each experiment to align with the specific task requirements. All documents were first split into sentences using SpaCy's `en_core_web_sm` model (Explosion, 2024) before being embedded and chunked. We tested three embedding models selected to represent a range of performances based on their rankings on the MTEB Leaderboard (Muennighoff et al., 2022). See Appendix E.2 for details.

3.1 Document Retrieval

This experiment assessed the effectiveness of chunkers in retrieving relevant documents for a given query. We used 10 datasets, shown in Tables 1 and 4. Most documents on the BEIR benchmark (Thakur et al., 2021) are too short for chunking to be effective. To address this, we synthesized longer documents by stitching short documents from six datasets where documents are too short (see Appendix C for details). We randomly sampled 100 queries from each dataset and retrieved the top k chunks, where $k \in [1, 3, 5, 10]$. Each retrieved chunk was mapped to its source document, and the retrieved documents were evaluated by comparing them to a set of relevant documents for each query.

3.2 Evidence Retrieval

Here we evaluate chunkers at a finer granularity than the previous experiment by measuring their abilities to locate evidence sentences. We selected additional datasets from RAGBench (Friel et al., 2024), shown in Tables 2 and 5, because few datasets contain long documents with ground-truth evidence sentences. We measured the number of ground-truth evidence sentences present in the retrieved top-k chunks.

3.3 Answer Generation

This experiment measured how chunkers impacted the quality of LLM-generated answers. We used `gpt-4o-mini` as the generative model. The top-5 retrieved chunks were used as input for the LLM, and generated answers were compared to ground-truth responses using semantic similarity measures. We reused the datasets from Section 3.2, as they included long documents, evidence, and reference answers.

4 Results

4.1 Measuring and reporting performances

As mentioned earlier, we used three proxy tasks the study chunking. We cannot directly assess the quality of retrieval at the chunk level due to the lack of ground-truth at the chunk level. Instead, each retrieved chunk is mapped back to either the source document or the included evidence sentences.

Since the number of relevant documents or evidence sentences is not fixed (unlike the k value for retrieved chunks), traditional metrics such as Recall@k and NDCG@k are not suitable. F1 provides a balanced measure that accounts for both precision and recall under these circumstances. Therefore, we use **F1@5** as the metric. For further details, see Appendix D.

For each dataset, results are reported based on the best hyperparameter configuration for each chunker, determined by the average F1 score across all k values. All results to be reported below are obtained using `dunzhang/stella_en_1.5B_v5` as the embedder for being the best among those tested.

In the following subsections, **Bold** values indicate the best performance on the respective dataset. The results for Answer Generation closely matched those of Evidence Retrieval and are discussed in Appendix E.1. Additional analysis of hyperparameters is provided in Appendix B. Inspection of the outputs of different chunkers is provided in Appendix E.4.

4.2 Document Retrieval

Table 1 shows varied chunker performance, with Fixed-size Chunker excelling on non-stitched datasets and Semantic Chunkers performing better on stitched datasets.

As described in Appendix C, stitched documents, averaging 100 sentences, were formed by combining short documents (fewer than 10 sentences)

Dataset	Fixed-size	Breakpoint	Clustering
Miracl*	69.45	**81.89**	67.35
NQ*	43.79	**63.93**	41.01
Scidocs*	16.82	17.60	**19.87**
Scifact*	35.27	**36.27**	35.70
BioASQ*	61.86	61.87	**62.49**
NFCorpus*	21.36	21.07	**22.12**
HotpotQA	**90.59**	87.37	84.79
MSMARCO	**93.58**	92.23	93.18
ConditionalQA	**68.11**	64.44	65.94
Qasper	**90.99**	89.27	90.77

Table 1: F1@5 for Document Retrieval (%). Datasets marked with * are stitched. Rows are sorted by the average number of sentences per document (before stitching) in ascending order for easier comparison.

from datasets like Miracl and NQ, leading to high topic diversity. In such cases, Breakpoint-based Semantic Chunker outperformed others by better preserving topic integrity, splitting sentences based on semantic dissimilarity to form chunks similar to the original documents. In contrast, Fixed-size and Clustering-based Chunkers often mixed sentences from different documents, increasing noise and lowering retrieval quality.

As document length increased, fewer documents were stitched together, reducing topic diversity. This diminished the advantage of Breakpoint-based Semantic Chunker, while Clustering-based Semantic Chunker improved. The gap between semantic and fixed-size chunkers narrowed, with Fixed-size Chunker benefiting from higher topic integrity.

These results suggest that in real life, the topics in a document may not be as diverse as in our artificially noisy, stitched data, and hence semantic chunkers may not have an edge over fixed-size chunker there.

4.3 Evidence Retrieval

As shown in Table 2, Fixed-size Chunker performed best on 3 out of 5 datasets, indicating a slight edge in capturing core evidence sentences. However, the performance differences between the Fixed-size Chunker and the two semantic chunkers were minimal, suggesting no clear advantage for any specific chunking strategy. See Appendix B for more details.

Further inspection revealed that despite variations in chunking methods, the top-k retrieved chunks frequently contained the same evidence sentences, explaining the minimal performance differences. This suggests that adding semantic information did not significantly enhance performance, as the benefits of semantic grouping were often redundant when core evidence was already captured by sentence positions. These findings indicate that the performance of the chunkers largely depends on how effectively the embedding models capture the semantic richness of individual sentences, rather than the chunking strategy itself.

Dataset	Fixed-size	Breakpoint	Clustering
ExpertQA	**47.11**	47.08	46.87
DelucionQA	43.05	43.24	**43.36**
TechQA	**28.98**	28.49	27.96
ConditionalQA	18.23	**19.83**	19.14
Qasper	**8.66**	8.16	8.50

Table 2: F1@5 for Evidence Retrieval (%). Rows are sorted by the average number of sentences per document in ascending order for easier comparison.

4.4 Results for Answer Generation

As shown in Tables 3, Semantic Chunkers performed slightly better than Fixed-size Chunker based on BERTScore, but the differences are minimal, making it difficult to draw any definitive conclusions.

Dataset	Fixed-size	Breakpoint	Clustering
ExpertQA	0.65	0.65	**0.65**
DelucionQA	0.76	**0.76**	0.76
TechQA	0.68	0.68	**0.68**
ConditionalQA	0.42	**0.43**	0.43
Qasper	0.49	0.49	**0.50**

Table 3: BERTScore for Answer Generation.

5 Conclusion

In this paper, we evaluated semantic and fixed-size chunking strategies in RAG systems across document retrieval, evidence retrieval, and answer generation. Semantic chunking occasionally improved performance, particularly on stitched datasets with high topic diversity. However, these benefits were highly context-dependent and did not consistently justify the additional computational cost. On non-synthetic datasets that better reflect real-world documents, fixed-size chunking often performed better. Overall, our results suggest that fixed-size chunking remains a more efficient and reliable choice for practical RAG applications. The impact of chunking strategy was often overshadowed by other factors, such as the quality of embeddings, especially when computational resources are limited or when working with standard document structures.

Limitations

Sentence-level Chunking Our study focuses on sentence-level chunking, where documents are split into individual sentences, and each sentence is treated as a segment for grouping. This approach results in sentence embeddings that lack contextual information. While we attempted to address this by overlapping sentences in Fixed-size Chunker and incorporating positional distance in Semantic Chunker (global), the embeddings themselves remained context-free. Further exploration of contextual embeddings is necessary before definitively concluding the limitations of semantic chunking.

Lack of Chunk Quality Measures As noted in Section 4, while the output chunks differed between methods, retrieval and generation performances were similar across chunkers. In addition to the influence of embedding models, the absence of direct chunk quality metrics likely contributed to this issue. Having ground-truth query-chunk relevance scores would provide more accurate evaluations than relying solely on document or evidence mapping.

Lack of Suitable Datasets Despite testing multiple datasets, our selection was constrained by a lack of comprehensive datasets. An ideal dataset would include long documents representative of real-world use cases, diverse query types, human-generated answers, query-document relevance scores, and human-labeled evidence sentences. Our synthetic documents had artificially high topic diversity due to random stitching, potentially leading to unreliable results. Additionally, the answer sets in RAGBench (Friel et al., 2024) were generated by LLMs, which may not accurately assess chunk quality. A dataset containing all these elements is needed for a more thorough evaluation of chunking strategies.

References

Valeriia Bolotova-Baranova, Vladislav Blinov, Sofya Filippova, Falk Scholer, and Mark Sanderson. 2023. Wikihowqa: A comprehensive benchmark for multi-document non-factoid question answering. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5291–5314.

Vera Boteva, Demian Gholipour, Artem Sokolov, and Stefan Riezler. 2016. A full-text learning to rank dataset for medical information retrieval. In *Advances in Information Retrieval: 38th European Conference on IR Research, ECIR 2016, Padua, Italy, March 20–23, 2016. Proceedings 38*, pages 716–722. Springer.

Vittorio Castelli, Rishav Chakravarti, Saswati Dana, Anthony Ferritto, Radu Florian, Martin Franz, Dinesh Garg, Dinesh Khandelwal, Scott McCarley, Mike McCawley, et al. 2019. The techqa dataset. *arXiv preprint arXiv:1911.02984*.

Jianlv Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. 2024. Bge m3-embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. *arXiv preprint arXiv:2402.03216*.

Tong Chen, Hongwei Wang, Sihao Chen, Wenhao Yu, Kaixin Ma, Xinran Zhao, Dong Yu, and Hongming Zhang. 2023. Dense x retrieval: What retrieval granularity should we use? *arXiv preprint arXiv:2312.06648*.

Arman Cohan, Sergey Feldman, Iz Beltagy, Doug Downey, and Daniel S Weld. 2020. Specter: Document-level representation learning using citation-informed transformers. *arXiv preprint arXiv:2004.07180*.

Pradeep Dasigi, Kyle Lo, Iz Beltagy, Arman Cohan, Noah A Smith, and Matt Gardner. 2021. A dataset of information-seeking questions and answers anchored in research papers. *arXiv preprint arXiv:2105.03011*.

Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. In *kdd*, volume 96, pages 226–231.

Explosion. 2024. English • spacy models documentation.

Paulo Finardi, Leonardo Avila, Rodrigo Castaldoni, Pedro Gengo, Celio Larcher, Marcos Piau, Pablo Costa, and Vinicius Caridá. 2024. The chronicles of rag: The retriever, the chunk and the generator. *arXiv preprint arXiv:2401.07883*.

Robert Friel, Masha Belyi, and Atindriyo Sanyal. 2024. Ragbench: Explainable benchmark for retrieval-augmented generation systems. *arXiv preprint arXiv:2407.11005*.

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, and Haofen Wang. 2023. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*.

Greg Kamradt. 2024. 5 levels of text splitting.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466.

LangChain. 2024. How to split text based on semantic similarity.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474.

LlamaIndex. 2024. Semantic chunker.

Chaitanya Malaviya, Subin Lee, Sihao Chen, Elizabeth Sieber, Mark Yatskar, and Dan Roth. 2023. Expertqa: Expert-curated questions and attributed answers. *arXiv preprint arXiv:2309.07852*.

Zach McCormick. 2024. Solving the out-of-context chunk problem for rag.

Niklas Muennighoff, Nouamane Tazi, Loïc Magne, and Nils Reimers. 2022. Mteb: Massive text embedding benchmark. *arXiv preprint arXiv:2210.07316*.

T Nguyen. 2016. Ms marco: A human generated machine reading comprehension dataset. *arXiv preprint arXiv:1611.09268*.

Mobashir Sadat, Zhengyu Zhou, Lukas Lange, Jun Araki, Arsalan Gundroo, Bingqing Wang, Rakesh R Menon, Md Rizwan Parvez, and Zhe Feng. 2023. Delucionqa: Detecting hallucinations in domain-specific question answering. *arXiv preprint arXiv:2312.05200*.

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*, pages 31210–31227. PMLR.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2020. Mpnet: Masked and permuted pre-training for language understanding. *Advances in neural information processing systems*, 33:16857–16867.

Haitian Sun, William W Cohen, and Ruslan Salakhutdinov. 2021. Conditionalqa: A complex reading comprehension dataset with conditional answers. *arXiv preprint arXiv:2110.06884*.

Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. Beir: A heterogenous benchmark for zero-shot evaluation of information retrieval models. *arXiv preprint arXiv:2104.08663*.

George Tsatsaronis, Michael Schroeder, Georgios Paliouras, Yannis Almirantis, Ion Androutsopoulos, Eric Gaussier, Patrick Gallinari, Thierry Artieres, Michael R Alvers, Matthias Zschunke, et al. 2012. Bioasq: A challenge on large-scale biomedical semantic indexing and question answering. In *2012 AAAI Fall Symposium Series*.

David Wadden, Shanchuan Lin, Kyle Lo, Lucy Lu Wang, Madeleine van Zuylen, Arman Cohan, and Hannaneh Hajishirzi. 2020. Fact or fiction: Verifying scientific claims. *arXiv preprint arXiv:2004.14974*.

Hitesh Wadhwa, Rahul Seetharaman, Somyaa Aggarwal, Reshmi Ghosh, Samyadeep Basu, Soundararajan Srinivasan, Wenlong Zhao, Shreyas Chaudhari, and Ehsan Aghazadeh. 2024. From rags to rich parameters: Probing how language models utilize external knowledge over parametric information for factual queries. *arXiv preprint arXiv:2406.12824*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W Cohen, Ruslan Salakhutdinov, and Christopher D Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. *arXiv preprint arXiv:1809.09600*.

Wenhao Yu, Hongming Zhang, Xiaoman Pan, Kaixin Ma, Hongwei Wang, and Dong Yu. 2023. Chain-of-note: Enhancing robustness in retrieval-augmented language models. *arXiv preprint arXiv:2311.09210*.

Dun Zhang. 2024. Stella.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*.

Xinyu Zhang, Nandan Thakur, Odunayo Ogundepo, Ehsan Kamalloo, David Alfonso-Hermelo, Xiaoguang Li, Qun Liu, Mehdi Rezagholizadeh, and Jimmy Lin. 2023. Miracl: A multilingual retrieval dataset covering 18 diverse languages. *Transactions of the Association for Computational Linguistics*, 11:1114–1131.

Andrew Zhu, Alyssa Hwang, Liam Dugan, and Chris Callison-Burch. 2024. Fanoutqa: a multi-hop, multi-document question answering benchmark for large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 18–37.

A Clustering methods for Clustering-based Semantic Chunker

We applied single-linkage agglomerative clustering to sentence embeddings in two stages. First, we computed a distance matrix where each entry represents the distance between pairs of sentences in the document. Second, we iteratively formed clusters by merging sentence pairs with the smallest distances, ensuring that the resulting cluster does not exceed a predefined maximum chunk size. This process continued until all distances had been processed, after which we relabeled the merged clusters.

To address challenges encountered during experimentation, we implemented the following adjustments:

Chunk Size Constraint Without a size constraint, this chunker tends to form one large chunk while leaving a few isolated sentences as individual chunks. To avoid this, we imposed a maximum chunk size threshold that directly depends on the number of chunks and the total number of sentences in the input document.

Distance Threshold for Stopping To prevent isolated sentences from being grouped arbitrarily, we introduced a distance threshold. Once this threshold is exceeded, clustering stops, and any remaining sentences are left ungrouped. In this paper, the threshold was set to be 0.5.

A limitation of the single-linkage method is its requirement to specify the number of clusters, which can be difficult without prior knowledge. To mitigate this, we also experimented with DBSCAN (Ester et al., 1996), a density-based clustering method that adjusts the number of clusters dynamically based on the density of sentence embeddings. DBSCAN follows the same initial steps as single-linkage clustering but replaces the merging process with density-based clustering.

B Hyperparameters

Fixed-size Chunker We tested two hyperparameters: the number of chunks and the number of overlapping sentences between consecutive chunks. For the number of chunks, we tested integer values between 2 and 10 to observe performance changes with different chunk sizes. For the overlapping sentences, we tested two settings: 0 or 1. If set to 1, one sentence overlaps between consecutive chunks; if set to 0, there is no overlap.

Breakpoint-based Semantic Chunker We tested two hyperparameters: the type of breakpoint threshold and the threshold amount. Sentences were split into chunks when the distance between consecutive sentences exceeded a predefined threshold. We evaluated four relative threshold types from (Kamradt, 2024):

- **Percentile**: The nth percentile of the linear interpolation of the distance array. We tested [10, 30, 50, 70, 90].
- **Standard deviation**: The mean of the linear interpolation plus a fraction of the standard deviation. We tested [1, 1.5, 2, 2.5, 3].
- **Interquartile**: The mean of the linear interpolation plus a fraction of the interquartile range. We tested [0.5, 0.75, 1, 1.25, 1.5].
- **Gradient**: The nth percentile of the second-order accurate difference in the distance array. We tested [10, 30, 50, 70, 90].

Additionally, we tested two absolute versions of "Percentile" and "Gradient":

- **Distance**: A cosine distance threshold value. We tested [0.1, 0.2, 0.3, 0.4, 0.5] based on empirical distance values.
- **Gradient**: A threshold value based on the second-order accurate difference. We tested [0.01, 0.05, 0.1, 0.15, 0.2] based on empirical gradient values.

Note that the number of chunks or chunk size is not tunable in the Breakpoint-based Semantic Chunker.

Clustering-based Semantic Chunker For the single-linkage chunker, we tested two hyperparameters: λ, which controls the weight of the positional distance in the overall distance calculation, and the number of chunks, as in the Fixed-size Chunker. We tested [0, 0.25, 0.5, 0.75, 1] for λ.

For the DBSCAN chunker, we evaluated three hyperparameters: λ, similar to single-linkage; EPS, the maximum distance between two samples for them to be considered part of the same neighborhood; and "min_samples", the minimum number of samples required in a neighborhood for a point to be classified as a core point. For EPS, we tested [0.1, 0.2, 0.3, 0.4, 0.5]. For "min_samples", we tested [1, 2, 3, 4, 5].

Dataset	Type	Split	#D	S/D(*)	S/D	D/Q
Miracl (Zhang et al., 2023)	Stitched	train	1184	102	4	3
NQ (Kwiatkowski et al., 2019)	Stitched	test	488	88	5	1
Scidocs (Cohan et al., 2020)	Stitched	test	1692	88	8	5
Scifact (Wadden et al., 2020)	Stitched	test	420	99	8	1
BioASQ (Tsatsaronis et al., 2012)	Stitched	train	2368	93	9	6
NFCorpus (Boteva et al., 2016)	Stitched	test	364	118	12	37
HotpotQA (Yang et al., 2018)	Original	test	800	20	20	2
MSMARCO (Nguyen, 2016)	Original	dev	398	64	64	1
ConditionalQA (Sun et al., 2021)	Original	dev	652	120	120	1
Qasper (Dasigi et al., 2021)	Original	test	416	130	130	1

Table 4: Datasets for Document Retrieval. "#D" means the number of selected long documents. "S/D" means the average number of sentences per document (before stitching). "S/D(*)" means the average number of sentences per long document (after stitching). "D/Q" means the average number of relevant long documents per query. The synthesized datasets are labeled as "Synthetic".

Dataset	Split	#D	S/D	E/Q
ExpertQA (Malaviya et al., 2023)	test	777	20	12
DelucionQA (Sadat et al., 2023)	test	235	23	9
TechQA (Castelli et al., 2019)	test	648	49	15
ConditionalQA (Sun et al., 2021)	dev	652	120	5
Qasper (Dasigi et al., 2021)	test	416	130	4

Table 5: Datasets for Evidence Retrieval and Answer Generation. "#D" means the number of selected long documents. "S/D" means the average number of sentences per long document. "E/Q" means the average number of evidence sentences per query.

Name	Rank	Model Size (millions)
`dunzhang/stella_en_1.5B_v5` (Zhang, 2024)	3	1543
`BAAI/bge-large-en-v1.5` (Chen et al., 2024)	36	335
`all-mpnet-base-v2` (Song et al., 2020)	105	110

Table 6: Embedding models used in the experiments. "Rank" represents the rank of the model on the MTEB Leaderboard (Muennighoff et al., 2022). "Model Size" represents the number of parameters in the embedding model.

C Document Stitching and Dataset Choices

Most document retrieval datasets consist of short documents (fewer than 20 sentences), which are inadequate for effectively evaluating chunkers. Initially, we experimented with datasets from BEIR (Thakur et al., 2021), but the short length of these documents showed no performance improvement with chunking. Short documents lack the complexity required to assess how chunkers manage context and semantic coherence across longer spans of text.

To overcome this limitation, we created long documents by stitching shorter documents from existing datasets. Each stitched document contains approximately 100 sentences, better reflecting real-world long-document retrieval scenarios. In this setup, if a short document is relevant to a query, the corresponding stitched long document is considered relevant. This creates a coarser granularity for document retrieval and motivated the need for the evidence retrieval experiment, which offers a finer level of evaluation.

We selected datasets based on diversity in document topics and query types. Keyword-specific queries tend to favor lexical search, which can degrade the performance of semantic search methods. For the document retrieval task, we used the datasets listed in Table 4, including NFCorpus, NQ, HotpotQA, Scidocs, and Scifacts from BEIR (Thakur et al., 2021).

For evidence retrieval and answer generation, we used the datasets listed in Table 5. No stitched document was used.

D Choice of Evaluation Metrics

Document Retrieval Retrieval can be viewed as two tasks: classification and ranking. In this paper, a document is considered retrieved if any chunk from it is retrieved, irrespective of the query-chunk relevance score. This approach shifts the focus from query-chunk relevance to query-document evaluation, reducing the influence of ranking metrics such as NDCG, MAP, or MRR.

- **Recall@k**: Fraction of relevant documents retrieved within the top-k chunks, over all relevant documents.
- **Precision@k**: Fraction of relevant documents retrieved within the top-k chunks, over all retrieved documents.
- **F1@k**: The harmonic mean of precision and recall.

In typical retrieval experiments, recall is often the primary metric. However, our setup requires balancing recall with precision and F1 score. Since the number of retrieved chunks is fixed but the number of retrieved documents varies, precision and F1 are crucial. For instance, if five chunks are retrieved for a query with only one relevant document, retrieving all five chunks from this document would result in 100% recall and precision. However, if only one chunk is relevant and the rest are from irrelevant documents, the recall remains 100%, but precision drops, leading to a different quality of retrieval. In such cases, the F1 score better captures this trade-off by balancing recall and precision.

Evidence Retrieval In evidence retrieval, recall and precision are sensitive to chunk size when considered separately. Larger chunks tend to have higher recall, as they are more likely to contain evidence sentences, but also lower precision, as they may include more irrelevant sentences. Larger chunks are often less desirable as they introduce more noise. For example, "No Chunker" will consistently have the highest recall and lowest precision, as it treats entire documents as single chunks. The F1 score helps balance these biases, providing a better indicator of whether the chunker produces appropriately sized chunks that capture relevant evidence. Therefore, we focus on F1 scores in our analysis.

- **Recall@k**: Fraction of retrieved evidence sentences over all evidence sentences.
- **Precision@k**: Fraction of retrieved evidence sentences over all retrieved sentences.
- **F1@k**: The harmonic mean of precision and recall.

Answer Generation Generated answers were assessed using BERTScore for semantic similarity between generated and actual answers, and cosine similarity between the queries and generated answers.

- **BERTScore** (Zhang et al., 2019): A measure of the semantic similarity between generated answers and reference answers using contextual embeddings. We used the best model `microsoft/deberta-xlarge-mnli` for calculating this score.
- **QA Similarity**: The cosine similarity between the query and generated answer, providing a measure of consistency and correctness in relation to the original query.

E Additional Results and Analyses

We present full results and analyses that are not reported in Section 4 in this section. See Table 20 for F1 scores at all k values for document retrieval. See Table 21 for F1 scores at all k values for evidence retrieval.

E.1 Results for Answer Generation

As shown in Table 7, semantic chunkers performed slightly better than Fixed-size Chunker in terms of QA cosine similarity. However, the differences are minimal, making it difficult to draw any definitive conclusions from the results.

Dataset	Fixed-size	Breakpoint	Clustering
ExpertQA	0.81	**0.82**	0.81
DelucionQA	0.82	0.82	**0.82**
TechQA	**0.89**	0.88	0.89
ConditionalQA	0.36	**0.36**	0.36
Qasper	0.44	0.44	**0.44**

Table 7: QA Cosine Similarity for Answer Generation.

E.2 Impact of Embedding Models

The choice of embedding model significantly affected retrieval performance (See Table 6 for tested models). In the Evidence Retrieval experiment, `BAAI/bge-large-en-v1.5` outperformed `all-mpnet-base-v2` by 1.06% on F1@1 and 1.32% on F1@10, both statistically significant at the 5% level. `dunzhang/stella_en_1.5B_v5` showed an average improvement of 7.44% over `BAAI/bge-large-en-v1.5` across all F1 values. This result was statistically significant with $p = 1.59 \times 10^{-5}$, highlighting the critical role of embedding models in retrieval tasks. See Tables 21-23 for full F1 scores from the three embedding models on evidence retrieval.

E.3 Hyperparameter Analysis

For Figures 2-5, all scores are normalized and averaged across datasets and k values. We aimed to identify chunker configurations that perform well across various datasets and k values, making it logical to average the results. The title of each plot row indicates the chunker and experiment being analyzed, while each subplot title specifies the fixed hyperparameter. The y-axis shows the metric score, and the x-axis represents the hyperparameter being analyzed. Blue lines denote recall, orange lines represent precision, and green lines indicate the F1 score.

Clustering-based Semantic Chunker (Single-linkage) As `n_clusters` increases, the average chunk size decreases. This has little effect on document retrieval since chunks are mapped to their source documents regardless of size. However, Figure 2 shows that while recall remains steady, precision rises significantly as chunk size decreases, even when $\lambda = 1$ (the Fixed-size Chunker case). This occurs due to a drop in the number of retrieved documents as smaller chunks from the same document are retrieved.

No clear trend for λ was observed, indicating that shifting the weight between semantic and positional information does not significantly affect document retrieval. This suggests two possibilities: (1) Sentences close in position are often semantically similar; (2) Chunks with non-contiguous, yet semantically similar sentences do not enhance document retrieval.

In Figure 2, evidence retrieval shows an inverse trend. As chunk size decreases, fewer sentences are retrieved, lowering the chance of retrieving evidence sentences and causing a sharp decline in recall. Thus, the F1 score remains relatively unchanged.

In addition, Figure 2 shows that as λ approaches 1 (representing the Fixed-size Chunker), the F1 score (green line) gradually increases, indicating that positional information contributed more to retrieval performance than semantic similarity, likely because core evidence sentences were often located close together.

Clustering-based Semantic Chunker (DBSCAN) As `EPS` increases, the threshold for grouping samples into the same cluster loosens, increasing average chunk size. As seen in Figure 3, this leads to a decrease in precision and an increase in recall for document and evidence retrieval, respectively, similar to the single-linkage case.

Breakpoint-based Semantic Chunker As the distance threshold between consecutive sentences increases, fewer breakpoints appear, resulting in larger chunks. Regardless of the threshold type, it ultimately determines chunk size. In Figure 4, we observe similar trends to Figure 2 and 3: as chunk size increases, precision decreases in both retrieval tasks, while recall increases sharply for evidence retrieval. The rise in standard deviation is expected, as values from standard deviation-based

thresholds are generally higher than those from percentiles or interquartile ranges.

Fixed-size Chunker Figure 5 shows results for the Fixed-size Chunker. The trends mirror those seen in other chunkers. Adding one overlapping sentence between chunks does not notably improve performance, indicating that a single overlapping sentence is insufficient to significantly boost contextual coherence.

E.4 Chunk Inspection

We examined the output chunks to (1) confirm that different chunkers were functioning as intended, and (2) investigate the reasons behind performance differences. BEIR's HotpotQA dataset (Thakur et al., 2021; Yang et al., 2018) was selected for its reasonably sized documents. We randomly sampled five documents, stitching the first four together to form a stitched document (Figure 6), and keeping the fifth as a normal document (Figure 7. The document IDs are:

- Stitched: 44547136, 14115210, 5580754, 54045118.
- Normal: 30214079.

Inspection on Stitched Documents In Figure 6, Documents 1 and 3 have four sentences each, while Documents 2 and 4 contain three and five sentences, respectively. The Fixed-size Chunker, which ignores semantic relationships and document structure, frequently misassigned sentences, leading to errors that propagated through subsequent chunks. For instance, a sentence from Document 3 was appended to Document 2, illustrating the limitations of Fixed-size Chunking with stitched documents containing numerous short segments. This explains its poor performance under such conditions. However, simply splitting the document into structured sections before applying fixed-size chunking will solve this issue.

In contrast, both semantic chunkers performed better on stitched documents, but still had issues. The Clustering-based Chunker made one error by grouping Sentence 16 (the last sentence of Document 4) into Chunk 2. This happened because, despite the large positional distance, the semantic similarity was high, causing the sentence to be incorrectly included. Without considering positional structure like the Fixed-size and Breakpoint-based Chunkers, the Clustering-based Chunker often mixed sentences from different documents. While this might be useful for multi-document tasks (Bolotova-Baranova et al., 2023; Zhu et al., 2024), it was problematic here, leading to worse performance when many short documents were stitched together.

The Breakpoint-based Chunker also made errors. It could, like the Fixed-size Chunker, group a sentence with a different chunk due to low semantic similarity with neighboring sentences, as seen with Sentence 4 being moved to Chunk 2. This shows the advantage of the joint distance measure in Equation 1, which prevented this error for the Clustering-based Chunker. Moreover, controlling chunk size was challenging; higher thresholds led to overly large chunks, while lower thresholds resulted in single-sentence chunks lacking contextual information, such as Chunk 4's meaningless "Name binding" phrase.

Inspection on Normal Documents In Figure 7, the document about "Interact Home Computer" was naturally divided into four sections, though this structure was not provided to the chunkers. The Fixed-size Chunker repeated its issue from stitched documents, occasionally grouping sentences from adjacent sections into the same chunk, and this error could be easily fixed by splitting the document by sections beforehand.

Although this example did not fully highlight the Clustering-based Chunker's limitations, it still demonstrated the downsides of relying solely on semantic similarity. Sentences 8-9, though belonging to Chunk 3, were grouped into Chunk 2 due to high semantic similarity. This showed that even with added positional information, semantic-based chunking could misgroup content that shared context, as these sentences were clearly about the sales of Interact Home Computer.

For the Breakpoint-based Chunker, errors seen in stitched documents were even more pronounced. Despite using the optimal configuration for each chunker (minimizing errors), Breakpoint-based Chunker still produced chunks containing only a single sentence, such as Chunk 3 and 5. Additionally, separating Sentences 5 and 6, which both discussed "Interact Electronics Inc," was an especially poor decision. These examples underscore that semantic similarity alone is not a reliable measure for effective chunking, and it may be less useful than straightforward positional information.

E.5 Impact of embedders on Document Retrieval and Answer Generation

In addition to `dunzhang/stella_en_1.5B_v5`, we experimented on 2 other embedding models on the MTEB benchmark: `BAAI/bge-large-en-v1.5` and `all-mpnet-base-v2`. The three chunking strategies are denoted as: Fixed-size, Breakpoint, and Clustering.

E.5.1 Document Retrieval

We present F1@k scores (%) from the 2 embedding models on Document Retrieval. A "stitched" dataset originally contained short documents, which we stitched into longer ones to enhance test set diversity.

- **Observation:** Breakpoint-based chunking has a clear edge on datasets that originally contained short documents which were stitched together to form longer documents by us. In contrast, fixed-size chunking performs better on datasets with natively long documents.
- **Interpretation:** Short documents stitched by us are likely to be of unrelated topics. Hence, breakpoint-based chunking, which was given high hope by the LangChain and LlamaIndex communities, do not have an advantage for natively long documents.
- **Recommendation:** Just use fixed-size chunking in practice.

Using `BAAI/bge-large-en-v1.5` as embedder

Dataset	Fixed-size	Breakpoint	Clustering
Miracl (stitched)	59.81	**62.27**	61.98
NQ (stitched)	81.59	**85.62**	81.60
Scidocs (stitched)	6.37	7.62	**8.03**
Scifact (stitched)	44.02	**51.98**	46.58
BioASQ (stitched)	48.36	**52.56**	45.83
NFCorpus (stitched)	10.35	11.02	**11.43**
HotpotQA	**66.67**	**66.67**	**66.67**
MSMARCO	**95.00**	**95.00**	**95.00**
ConditionalQA	**76.38**	74.54	76.01
Qasper	**92.45**	90.84	90.84

Table 8: F1@1 (%)

Dataset	Fixed-size	Breakpoint	Clustering
Miracl (stitched)	70.10	**78.87**	69.45
NQ (stitched)	55.03	**66.53**	52.36
Scidocs (stitched)	14.13	16.39	**16.80**
Scifact (stitched)	37.98	**52.82**	38.38
BioASQ (stitched)	61.82	**62.26**	61.53
NFCorpus (stitched)	16.36	17.56	**18.53**
HotpotQA	**91.43**	90.87	90.46
MSMARCO	93.26	**93.56**	92.64
ConditionalQA	**70.75**	68.78	67.83
Qasper	**89.18**	87.62	88.65

Table 9: F1@2 (%)

Dataset	Fixed-size	Breakpoint	Clustering
Miracl (stitched)	63.78	**76.72**	60.96
NQ (stitched)	48.34	**57.05**	45.95
Scidocs (stitched)	18.06	18.47	**20.49**
Scifact (stitched)	29.24	**45.00**	30.36
BioASQ (stitched)	61.66	**66.36**	61.39
NFCorpus (stitched)	18.77	**19.79**	19.75
HotpotQA	**87.36**	76.05	78.18
MSMARCO	**90.93**	90.09	89.51
ConditionalQA	**54.75**	54.45	54.59
Qasper	**81.03**	76.33	77.15

Table 10: F1@5 (%)

Dataset	Fixed-size	Breakpoint	Clustering
Miracl (stitched)	45.19	**61.50**	41.94
NQ (stitched)	21.34	**30.81**	20.19
Scidocs (stitched)	17.19	**19.89**	19.52
Scifact (stitched)	19.57	**31.27**	19.91
BioASQ (stitched)	54.22	**62.65**	53.35
NFCorpus (stitched)	21.73	21.73	**22.27**
HotpotQA	**54.24**	43.43	43.80
MSMARCO	**78.90**	76.65	68.90
ConditionalQA	**34.73**	33.24	33.55
Qasper	**58.67**	46.86	42.97

Table 11: F1@10 (%)

Using `SentenceBert/all-mpnet-base-v2` as embedder

Dataset	Fixed-size	Breakpoint	Clustering
Miracl (stitched)	60.13	**62.97**	61.08
NQ (stitched)	81.53	**88.90**	81.16
Scidocs (stitched)	**5.40**	5.00	5.33
Scifact (stitched)	42.68	**50.70**	45.36
BioASQ (stitched)	40.68	**44.00**	38.75
NFCorpus (stitched)	7.21	**8.21**	6.71
HotpotQA	**62.67**	62.00	**62.67**
MSMARCO	**98.00**	94.00	**98.00**
ConditionalQA	**72.69**	71.96	71.59
Qasper	**88.95**	85.98	87.60

Table 12: F1@1 (%)

Dataset	Fixed-size	Breakpoint	Clustering
Miracl (stitched)	64.61	**73.95**	67.21
NQ (stitched)	53.30	**65.94**	50.86
Scidocs (stitched)	13.97	14.36	**16.42**
Scifact (stitched)	36.18	**51.07**	36.59
BioASQ (stitched)	49.54	**52.62**	48.14
NFCorpus (stitched)	12.39	13.55	**14.40**
HotpotQA	**78.69**	76.53	75.40
MSMARCO	**94.65**	93.19	94.28
ConditionalQA	**72.63**	69.00	68.69
Qasper	**87.75**	85.25	86.97

Table 13: F1@3 (%)

Dataset	Fixed-size	Breakpoint	Clustering
Miracl (stitched)	60.20	**69.93**	59.72
NQ (stitched)	37.05	**49.57**	36.60
Scidocs (stitched)	15.18	15.68	**17.85**
Scifact (stitched)	27.74	**43.76**	28.99
BioASQ (stitched)	47.42	**56.07**	47.72
NFCorpus (stitched)	14.71	**16.34**	16.15
HotpotQA	**71.73**	65.09	65.84
MSMARCO	**91.46**	89.88	90.05
ConditionalQA	**60.46**	56.10	60.44
Qasper	**79.47**	78.99	79.46

Table 14: F1@5 (%)

Dataset	Fixed-size	Breakpoint	Clustering
Miracl (stitched)	43.07	**51.27**	41.25
NQ (stitched)	21.49	**31.18**	20.83
Scidocs (stitched)	16.82	18.96	**19.50**
Scifact (stitched)	18.48	**29.28**	18.78
BioASQ (stitched)	42.58	**54.47**	41.28
NFCorpus (stitched)	18.16	**19.81**	19.14
HotpotQA	**47.21**	40.56	40.06
MSMARCO	**79.51**	78.62	67.78
ConditionalQA	**38.04**	36.63	36.44
Qasper	**61.31**	47.64	51.62

Table 15: F1@10 (%)

E.5.2 Answer Generation

We present BERTScore and QA Similarity scores from the 2 embedding models on Answer Generation.

- **Takeaways:** Just use fixed-size chunking in practice. Breakpoint-based and clustering-based have no advantage.

Using `BAAI/bge-large-en-v1.5` as embedder

Dataset	Fixed-size	Breakpoint	Clustering
ExpertQA	**0.68**	0.67	0.67
DelucionQA	0.76	**0.78**	0.76
TechQA	**0.69**	0.67	0.68
ConditionalQA	**0.40**	0.39	0.39
Qasper	**0.50**	0.50	0.49

Table 16: BERTScore

Dataset	Fixed-size	Breakpoint	Clustering
ExpertQA	**0.86**	0.86	0.85
DelucionQA	0.82	0.83	**0.83**
TechQA	**0.90**	0.90	0.89
ConditionalQA	0.35	**0.36**	0.36
Qasper	**0.46**	0.46	0.45

Table 17: QA Similarity

Using `SentenceBert/all-mpnet-base-v2` as embedder

Dataset	Fixed-size	Breakpoint	Clustering
ExpertQA	**0.67**	0.66	0.66
DelucionQA	0.76	**0.76**	0.76
TechQA	**0.68**	0.68	0.68
ConditionalQA	0.40	0.40	**0.41**
Qasper	**0.50**	0.50	0.50

Table 18: BERTScore

Dataset	Fixed-size	Breakpoint	Clustering
ExpertQA	**0.84**	0.84	0.84
DelucionQA	0.81	0.82	**0.83**
TechQA	0.86	**0.87**	0.87
ConditionalQA	0.34	**0.35**	0.34
Qasper	**0.45**	0.41	0.40

Table 19: QA Similarity

Metric	F1@1			F1@3			F1@5			F1@10		
Chunker	Fixed-size	Breakpoint	Clustering	Fixed-size	Breakpoint	Clustering	Fixed-size	Breakpoint	Clustering	Fixed-size	Breakpoint	Clustering
Miracl*	67.55	**69.73**	68.61	76.03	**83.55**	75.24	69.45	**81.89**	67.35	49.89	**67.83**	46.59
NQ*	**92.92**	92.36	88.63	62.29	**83.37**	60.29	43.79	**63.93**	41.01	24.02	**36.18**	22.56
Scidocs*	7.60	7.73	**10.40**	15.16	14.92	**18.93**	16.82	16.60	**19.87**	16.96	16.88	**19.94**
Scifact*	55.07	53.38	**55.09**	43.97	**52.91**	46.60	35.27	**36.27**	35.70	22.33	**27.59**	22.32
BioASQ*	53.09	**55.95**	53.14	61.92	**70.74**	61.84	61.86	61.87	**62.49**	54.37	**56.82**	55.44
NFCorpus*	11.41	**12.49**	11.42	19.00	19.10	**20.24**	21.36	21.07	**22.12**	22.95	23.48	**24.09**
HotpotQA	66.00	66.00	**66.67**	92.06	91.83	**92.33**	**90.59**	87.37	84.79	**61.34**	52.22	51.30
MSMARCO	**99.00**	97.00	98.00	**95.35**	94.92	94.73	**93.58**	92.23	93.18	**85.75**	84.34	77.57
ConditionalQA	**83.03**	79.70	79.34	**78.67**	74.63	76.09	**68.11**	64.44	65.94	**44.66**	40.37	39.35
Qasper	**96.50**	93.53	95.96	**95.21**	92.20	95.14	**90.99**	89.27	90.77	68.86	**69.59**	62.41

Table 20: F1 scores for all k values for Document Retrieval (%). Datasets marked with * are stitched. Rows are sorted by the average number of sentences per document (before stitching) in ascending order for easier comparison.

Metric	F1@1			F1@3			F1@5			F1@10		
Chunker	Fixed-size	Breakpoint	Clustering	Fixed-size	Breakpoint	Clustering	Fixed-size	Breakpoint	Clustering	Fixed-size	Breakpoint	Clustering
ExpertQA	**42.43**	35.25	40.83	48.67	48.49	**48.73**	**47.11**	47.08	46.87	33.18	**36.53**	33.92
DelucionQA	**39.40**	28.12	34.60	44.18	**45.43**	44.05	43.05	43.24	**43.36**	**37.29**	36.16	36.32
TechQA	**39.38**	29.27	31.68	**28.98**	28.49	27.96	**28.98**	28.49	27.96	**16.92**	16.76	14.51
ConditionalQA	23.14	**23.61**	22.15	19.81	**22.01**	17.32	18.23	**19.83**	19.14	14.56	**15.41**	15.25
Qasper	8.22	**8.58**	8.36	**9.67**	8.83	8.75	**8.66**	8.16	8.50	**6.99**	6.78	6.52

Table 21: F1 scores for all k values for Evidence Retrieval (%), from `dunzhang/stella_en_1.5B_v5`.

Metric	F1@1			F1@3			F1@5			F1@10		
Chunker	Fixed-size	Breakpoint	Clustering	Fixed-size	Breakpoint	Clustering	Fixed-size	Breakpoint	Clustering	Fixed-size	Breakpoint	Clustering
ExpertQA	**40.34**	33.39	38.31	**44.60**	43.71	44.36	42.25	41.46	**43.03**	29.54	**31.89**	29.28
DelucionQA	**33.88**	27.02	32.73	42.10	**43.58**	40.79	40.85	40.89	**41.22**	**37.29**	36.16	35.99
TechQA	**34.90**	28.25	29.57	23.09	**23.92**	22.24	19.82	**20.90**	19.27	13.00	**13.25**	12.92
ConditionalQA	**20.09**	**20.09**	19.40	**18.24**	16.93	14.27	**14.83**	13.89	10.73	**10.72**	9.24	6.39
Qasper	**7.80**	6.20	5.34	**6.88**	6.83	6.76	**6.59**	6.43	5.70	**4.99**	4.71	4.34

Table 22: F1 scores for all k values for Evidence Retrieval (%), from `BAAI/bge-large-en-v1.5`.

Metric	F1@1			F1@3			F1@5			F1@10		
Chunker	Fixed-size	Breakpoint	Clustering	Fixed-size	Breakpoint	Clustering	Fixed-size	Breakpoint	Clustering	Fixed-size	Breakpoint	Clustering
ExpertQA	**40.96**	32.75	38.83	42.76	**43.39**	43.38	41.78	41.56	**42.07**	**31.82**	29.64	31.37
DelucionQA	**38.02**	32.22	31.67	39.78	**42.22**	39.68	**41.31**	35.34	41.04	35.94	27.77	**36.11**
TechQA	**31.04**	23.30	27.24	**24.62**	23.41	24.60	19.42	**21.24**	19.56	**16.56**	14.07	12.21
ConditionalQA	18.01	**20.87**	17.73	14.73	**18.65**	14.24	11.67	**16.09**	11.05	7.25	**12.95**	7.11
Qasper	**8.09**	6.92	6.98	**6.97**	6.23	6.67	6.56	5.98	6.24	**4.23**	4.12	3.62

Table 23: F1 scores for all k values for Evidence Retrieval (%), from `all-mpnet-base-v2`.

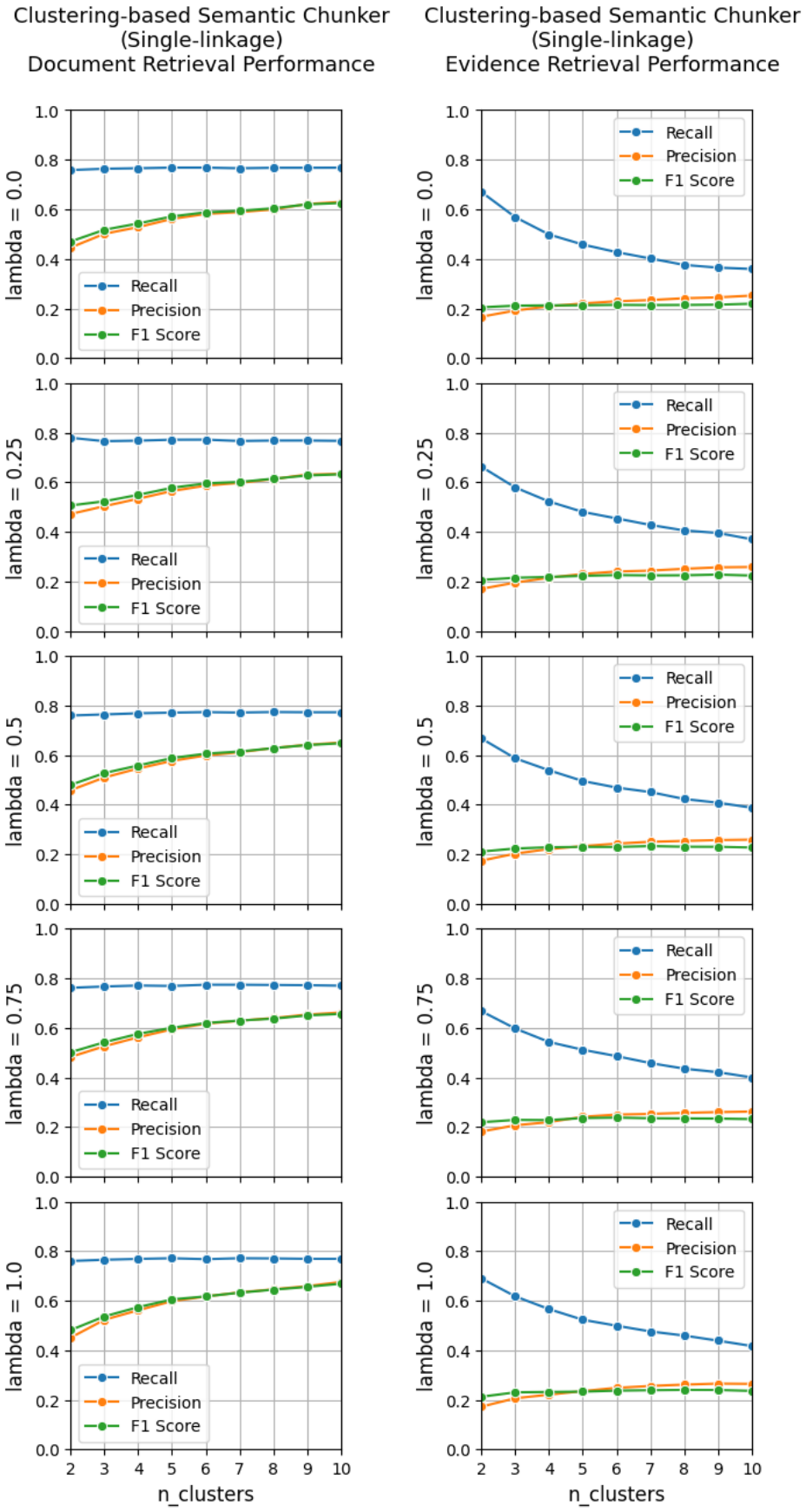

Figure 2: Performance vs. hyperparameter values for Clustering-based Semantic Chunker (Single-linkage). Left: Document Retrieval; Right: Evidence Retrieval. The x-axis shows `n_clusters`, and the y-axis shows the metric value. Each subplot's y-label indicates the fixed hyperparameter value, with λ increasing from top to bottom.

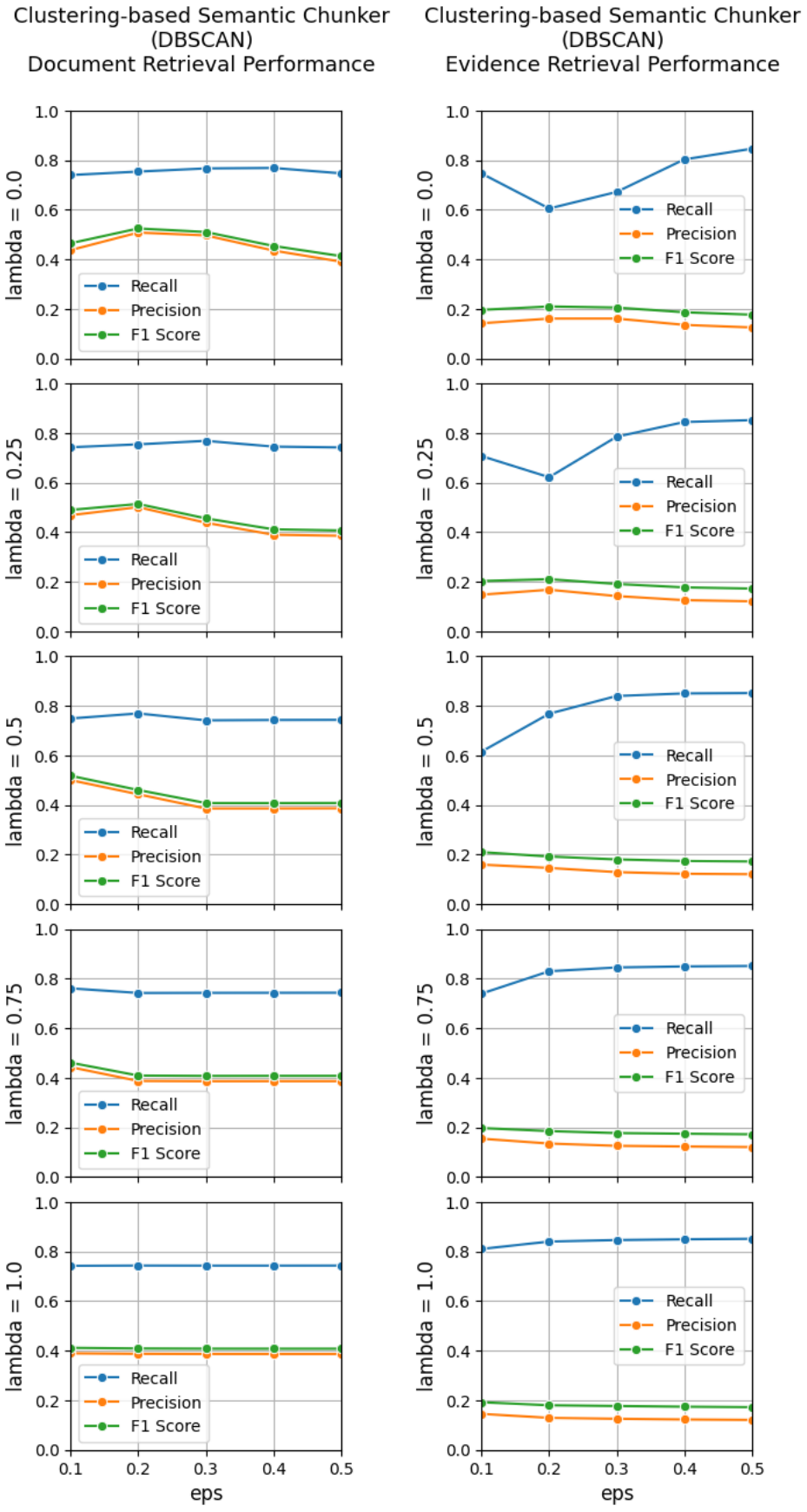

Figure 3: Performance vs. hyperparameter values for Clustering-based Semantic Chunker (DBSCAN). Left: Document Retrieval; Right: Evidence Retrieval. The x-axis shows eps, and the y-axis shows the metric value. Each subplot's y-label indicates the fixed hyperparameter value, with λ increasing from top to bottom.

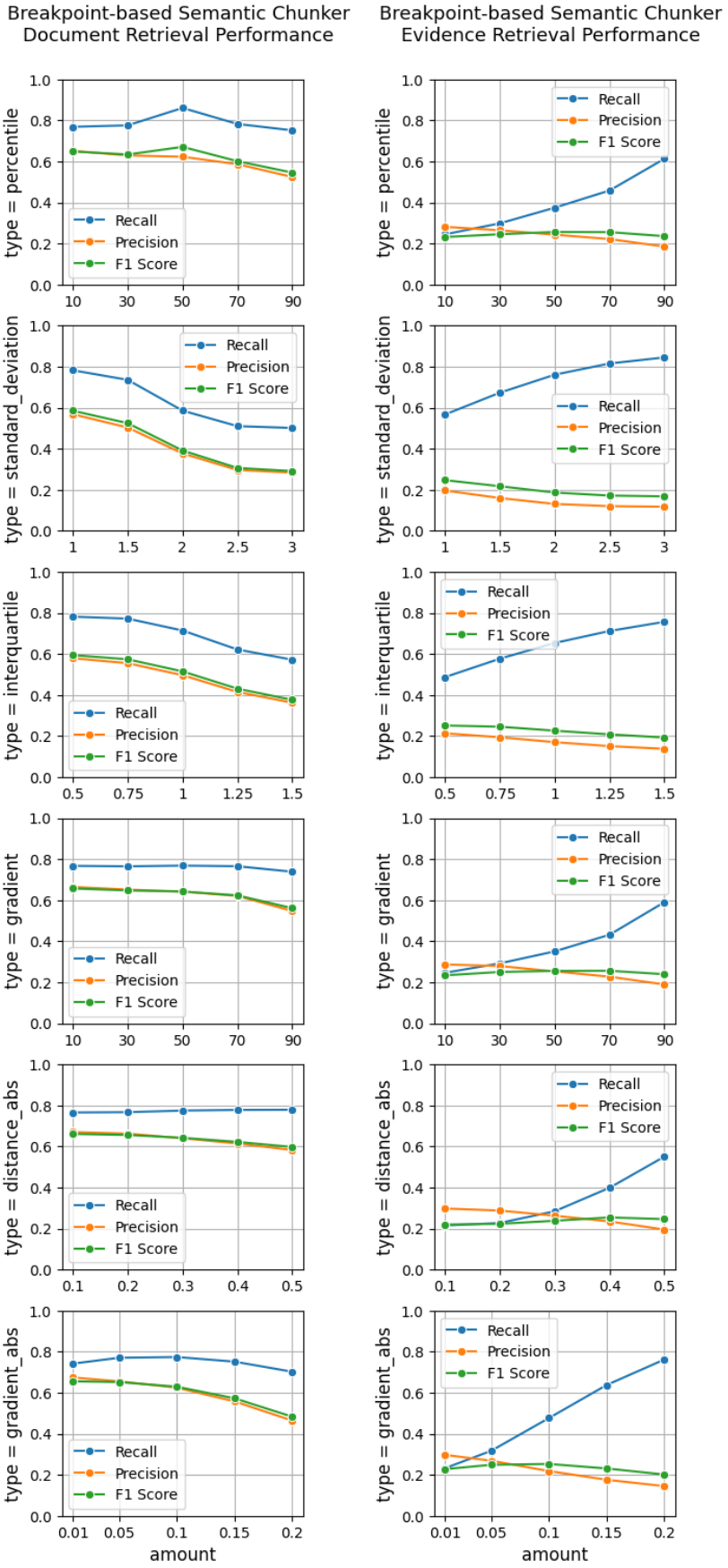

Figure 4: Performance vs. hyperparameter values for Breakpoint-based Semantic Chunker. Left: Document Retrieval; Right: Evidence Retrieval. The x-axis shows `n_clusters`, and the y-axis shows the metric value. Each subplot's y-label indicates the breakpoint threshold type.

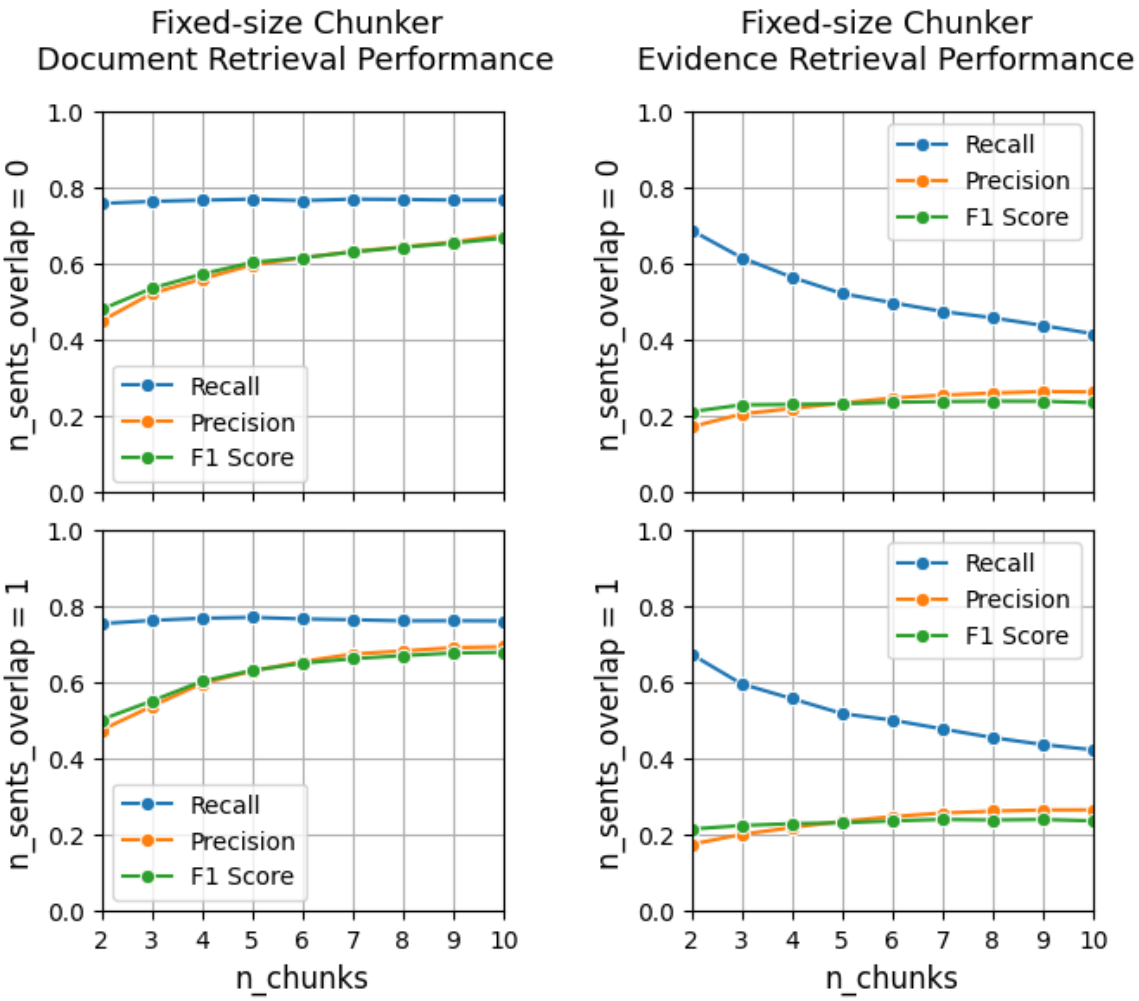

Figure 5: Performance vs. hyperparameter values for Fixed-size Chunker. Left: Document Retrieval; Right: Evidence Retrieval. The x-axis shows `n_chunks`, and the y-axis shows the metric value. Each subplot's y-label indicates the fixed hyperparameter value, with `n_sents_overlap` increasing from top to bottom.

Stitched Document:

```
Document 1:
 1 Tom Muecke
 2 Thomas Warren Muecke, Jr. (pronounced Mickey) (August 20, 1963 – ...
 3 He played college football at Baylor University and attended Angleton ...
 4 He was also a member of the Houston Oilers and Calgary Stampeders.

Document 2:
 5 Jonathan Losos
 6 Jonathan B. Losos is an American biologist, focusing in organismic and ...
 7 His most cited papers are 819, 800, 729 and 665 and was especially ...

Document 3:
 8 1883 Boston Beaneaters season
 9 The 1883 Boston Beaneaters season was the thirteenth season of the ...
10 The Beaneaters won their third National League pennant, their third in ...
11 This is also generally recognized as the year during which the team's ...

Document 4:
12 Name binding
13 In programming languages, name binding is the association of entities ...
14 An identifier bound to an object is said to reference that object.
15 Machine languages have no built-in notion of identifiers, but name ...
16 Binding is intimately connected with scoping, as scope determines ...
```

Fixed-size Chunker (n_sents_per_chunk = 4):

```
Chunk 1:
 1 Tom Muecke
 2 Thomas Warren Muecke, Jr. (pronounced Mickey) (August 20, 1963 – ...
 3 He played college football at Baylor University and attended Angleton ...
 4 He was also a member of the Houston Oilers and Calgary Stampeders.

Chunk 2:
 5 Jonathan Losos
 6 Jonathan B. Losos is an American biologist, focusing in organismic and ...
 7 His most cited papers are 819, 800, 729 and 665 and was especially ...
 8 1883 Boston Beaneaters season

Chunk 3:
 9 The 1883 Boston Beaneaters season was the thirteenth season of the ...
10 The Beaneaters won their third National League pennant, their third in ...
11 This is also generally recognized as the year during which the team's ...
12 Name binding

Chunk 4:
13 In programming languages, name binding is the association of entities ...
14 An identifier bound to an object is said to reference that object.
15 Machine languages have no built-in notion of identifiers, but name ...
16 Binding is intimately connected with scoping, as scope determines ...
```

Clustering-based Chunker (Single-linkage, lambda = 0.5, n_chunks = 4):

```
Chunk 1:
 1 Tom Muecke
 2 Thomas Warren Muecke, Jr. (pronounced Mickey) (August 20, 1963 - ...
 3 He played college football at Baylor University and attended Angleton ...
 4 He was also a member of the Houston Oilers and Calgary Stampeders.

Chunk 2:
 5 Jonathan Losos
 6 Jonathan B. Losos is an American biologist, focusing in organismic and ...
 7 His most cited papers are 819, 800, 729 and 665 and was especially ...
16 Binding is intimately connected with scoping, as scope determines ...

Chunk 3:
 8 1883 Boston Beaneaters season
 9 The 1883 Boston Beaneaters season was the thirteenth season of the ...
10 The Beaneaters won their third National League pennant, their third in ...
11 This is also generally recognized as the year during which the team's ...

Chunk 4:
12 Name binding
13 In programming languages, name binding is the association of entities ...
14 An identifier bound to an object is said to reference that object.
15 Machine languages have no built-in notion of identifiers, but name ...
```

Cleavage-based Chunker (breakpoint_threshold_type = “percentile”, amount = 70):

```
Chunk 1:
 1 Tom Muecke
 2 Thomas Warren Muecke, Jr. (pronounced Mickey) (August 20, 1963 - ...
 3 He played college football at Baylor University and attended Angleton ...

Chunk 2:
 4 He was also a member of the Houston Oilers and Calgary Stampeders.
 5 Jonathan Losos
 6 Jonathan B. Losos is an American biologist, focusing in organismic and ...
 7 His most cited papers are 819, 800, 729 and 665 and was especially ...

Chunk 3:
 8 1883 Boston Beaneaters season
 9 The 1883 Boston Beaneaters season was the thirteenth season of the ...
10 The Beaneaters won their third National League pennant, their third in ...
11 This is also generally recognized as the year during which the team's ...

Chunk 4:
12 Name binding

Chunk 5:
13 In programming languages, name binding is the association of entities ...
14 An identifier bound to an object is said to reference that object.
15 Machine languages have no built-in notion of identifiers, but name ...
16 Binding is intimately connected with scoping, as scope determines ...
```

Figure 6: Example of chunking a stitched document using different chunkers. Each line shows a sentence and its original index in the document. Bold red lines indicate errors where a sentence is incorrectly assigned to a chunk. The configuration listed next to each chunker name represents the optimal setup for minimizing errors.

Original Document:

```
Section 1: Overview
 1 Interact Home Computer
 2 The Interact Home Computer is a rare, very early (1978) American ...
 3 It sold under the name "interact Model One home computer".
 4 The original Interact Model One computer was designed by Rick Barnich ...

Section 2: Company Details
 5 Interact Electronics Inc was a privately held company that was funded ...
 6 The President/Founder of Interact Electronics Inc was Ken Lochner, who ...
 7 Ken had started Interact Electronics Inc after a successful startup ...

Section 3: Sales
 8 Only a few thousand Interacts were sold before the company went bankrupt.
 9 Most were sold by the liquidator "Protecto Enterprizes" of Barrington, ...
10 The Interact Model One Home Computer debuted at the Consumer ...
11 The majority of sales were thru Mail Order houses and you could buy it ...

Section 4: Use Cases
12 Probably the most successful application available for the Interace ...
13 With it, a store could type in whatever message they wanted to appear ...
14 Although it was mostly a Game machine at the time with games such as ...
15 Customers began hooking up Interact to control everything from lights ...
```

Fixed-size Chunker (n_sents_per_chunk = 4):

```
Chunk 1:
 1 Interact Home Computer
 2 The Interact Home Computer is a rare, very early (1978) American ...
 3 It sold under the name "interact Model One home computer".
 4 The original Interact Model One computer was designed by Rick Barnich ...

Chunk 2:
 5 Interact Electronics Inc was a privately held company that was funded ...
 6 The President/Founder of Interact Electronics Inc was Ken Lochner, who ...
 7 Ken had started Interact Electronics Inc after a successful startup ...
 8 Only a few thousand Interacts were sold before the company went bankrupt.

Chunk 3:
 9 Most were sold by the liquidator "Protecto Enterprizes" of Barrington, ...
10 The Interact Model One Home Computer debuted at the Consumer ...
11 The majority of sales were thru Mail Order houses and you could buy it ...
12 Probably the most successful application available for the Interace ...

Chunk 4:
13 With it, a store could type in whatever message they wanted to appear ...
14 Although it was mostly a Game machine at the time with games such as ...
15 Customers began hooking up Interact to control everything from lights ...
```

Clustering-based Chunker (Single-linkage, lambda = 0.5, n_chunks = 4):

```
Chunk 1:
 1 Interact Home Computer
 2 The Interact Home Computer is a rare, very early (1978) American ...
 3 It sold under the name "interact Model One home computer".
 4 The original Interact Model One computer was designed by Rick Barnich ...

Chunk 2:
 5 Interact Electronics Inc was a privately held company that was funded ...
 6 The President/Founder of Interact Electronics Inc was Ken Lochner, who ...
 7 Ken had started Interact Electronics Inc after a successful startup ...
 8 Only a few thousand Interacts were sold before the company went bankrupt.
 9 Most were sold by the liquidator "Protecto Enterprizes" of Barrington, ...

Chunk 3:
10 The Interact Model One Home Computer debuted at the Consumer ...
11 The majority of sales were thru Mail Order houses and you could buy it ...

Chunk 4:
12 Probably the most successful application available for the Interace ...
13 With it, a store could type in whatever message they wanted to appear ...
14 Although it was mostly a Game machine at the time with games such as ...
15 Customers began hooking up Interact to control everything from lights ...
```

Cleavage-based Chunker (breakpoint_threshold_type = “percentile”, amount = 70):

```
Chunk 1:
 1 Interact Home Computer
 2 The Interact Home Computer is a rare, very early (1978) American ...
 3 It sold under the name "interact Model One home computer".
 4 The original Interact Model One computer was designed by Rick Barnich ...
 5 Interact Electronics Inc was a privately held company that was funded ...

Chunk 2:
 6 The President/Founder of Interact Electronics Inc was Ken Lochner, who ...
 7 Ken had started Interact Electronics Inc after a successful startup ...

Chunk 3:
 8 Only a few thousand Interacts were sold before the company went bankrupt.

Chunk 4:
 9 Most were sold by the liquidator "Protecto Enterprizes" of Barrington, ...
10 The Interact Model One Home Computer debuted at the Consumer ...
11 The majority of sales were thru Mail Order houses and you could buy it ...

Chunk 5:
12 Probably the most successful application available for the Interace ...

Chunk 6:
13 With it, a store could type in whatever message they wanted to appear ...
14 Although it was mostly a Game machine at the time with games such as ...
15 Customers began hooking up Interact to control everything from lights ...
```

Figure 7: Example of chunking a normal document using different chunkers. Each line shows a sentence and its original index in the document. Bold red lines indicate errors where a sentence is incorrectly assigned to a chunk. The configuration listed next to each chunker name represents the optimal setup for minimizing errors.