Semantic Outlier Removal with Embedding Models and LLMs

Eren Akbiyik, João Almeida, Rik Melis,
Ritu Sriram, Viviana Petrescu, Vilhjálmur Vilhjálmsson

TripleLift
Zürich, Switzerland

Correspondence: eakbiyik@triplelift.com

Abstract

Modern text processing pipelines demand robust methods to remove extraneous content while preserving a document's core message. Traditional approaches—such as HTML boilerplate extraction or keyword filters—often fail in multilingual settings and struggle with context-sensitive nuances, whereas Large Language Models (LLMs) offer improved quality at high computational cost. We introduce SORE (Semantic Outlier Removal), a cost-effective, transparent method that leverages multilingual sentence embeddings and approximate nearest-neighbor search to identify and excise unwanted text segments. By first identifying core content via metadata embedding and then flagging segments that either closely match predefined outlier groups or deviate significantly from the core, SORE achieves near-LLM extraction precision at a fraction of the cost. Experiments on HTML datasets demonstrate that SORE outperforms structural methods and yield high precision in diverse scenarios. Our system is currently deployed in production, processing millions of documents daily across multiple languages while maintaining both efficiency and accuracy. To facilitate further research, we will publicly release our implementation and evaluation datasets.

1 Introduction

Effective content extraction from web pages is a critical component in many modern NLP pipelines, enabling cleaner inputs for downstream tasks such as summarization, classification, and information retrieval. However, web documents typically contain significant amounts of extraneous content—navigation elements, advertisements, legal disclaimers, related article recommendations, and other non-essential text—that can degrade the performance of these tasks.

Traditional approaches to this problem include HTML-structure-based methods like Readability.js (rea) and Boilerpipe (Kohlschütter et al., 2010), which leverage DOM and formatting patterns to identify main content. While efficient, these methods often fail when faced with diverse HTML structures, especially across multiple languages and website designs. They also struggle to distinguish semantically irrelevant text that shares structural characteristics with the main content.

More recently, Large Language Models (LLMs) have demonstrated impressive capabilities in content extraction (Brown et al., 2020), as they can understand the semantic meaning and context of text. However, deploying LLMs at scale incurs substantial computational costs, introducing latency and budget concerns for production systems processing millions of documents. Additionally, LLMs may introduce hallucinations or unpredictable behaviors that compromise reliability.

To address these limitations, we introduce SORE (Semantic Outlier Removal), a system that bridges the gap between traditional structure-based methods and LLMs by utilizing multilingual embedding models. SORE leverages semantic similarity to identify core content by measuring similarity to document metadata, detect outliers by measuring distance to predefined outlier categories, and remove unwanted content while providing transparent justification.

Our approach offers several key advantages for industrial applications. First, SORE operates in a language-agnostic manner, enabling effective content extraction across diverse languages without requiring language-specific rules. Second, it provides transparency with clear explanations for why specific text segments are removed, facilitating debugging and continuous improvement. Third, SORE achieves near-LLM quality extraction at a fraction of the computational cost—a critical factor for production systems processing millions of documents. Finally, its implementation using approximate nearest neighbor search ensures scalability even with large document volumes.

Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 6: Industry Track), pages 826–835
July 28-30, 2025 ©2025 Association for Computational Linguistics

This paper describes the SORE algorithm, its implementation details optimized for production deployment, and comprehensive experiments demonstrating its effectiveness compared to both traditional methods and LLM-based approaches. We also provide a detailed cost analysis, highlighting the significant efficiency gains achieved by our approach. To promote reproducibility and facilitate further research, we will make our implementation and evaluation datasets publicly available.

2 Related Work

2.1 HTML Boilerplate Removal

Extracting main content from HTML documents remains challenging in web information retrieval. Kohlschütter et al. (2010) introduced text density features to identify boilerplate content, while Readability.js (rea) employs heuristic rules based on HTML structure. Despite their efficiency, these approaches struggle with complex layouts and multilingual content.

2.2 Embedding Models for Text Similarity

Dense vector representations have transformed NLP by capturing semantic relationships between texts. Evolving from word embeddings (Mikolov et al., 2013; Pennington et al., 2014) to sentence representations, models like Sentence-BERT (Reimers and Gurevych, 2019) adapted transformer architectures for similarity tasks. Multilingual embedding models (Artetxe and Schwenk, 2019; Wang et al., 2024) now enable cross-lingual applications, with commercial services like Cohere (coh) and AWS Titan offering production-ready solutions.

2.3 LLMs for Content Extraction

LLMs demonstrate strong capabilities in understanding contextual meaning (Brown et al., 2020; Scao et al., 2023), making them promising for content extraction. However, they require significant computational resources and may produce inconsistent outputs (Bender et al., 2021). Their effectiveness varies across languages, particularly for lower-resource ones (Nguyen et al., 2023).

2.4 Outlier Detection in Text

Text outlier detection approaches include density-based methods (Taleb Sereshki et al., 2023) and embedding space analysis (Hämmerl et al., 2023). Most work focuses on document-level detection rather than identifying outlier segments within documents.

Our work bridges these areas by leveraging embedding-based similarity with efficient nearest-neighbor search for multilingual outlier content identification, balancing traditional methods' efficiency with LLMs' semantic understanding.

3 SORE: System Design and Implementation

We introduce SORE (Semantic Outlier Removal), a method for removing unwanted text segments from documents based on semantic similarity. SORE identifies and removes text segments that match known patterns of boilerplate content or semantically diverge from the document's theme.

3.1 Algorithm Overview

SORE operates through four sequential steps that transform raw HTML content into clean content:

Step 1: Segmentation and Embedding. The document is first split into segments (sentences or paragraphs) using an HTML parser that preserves the document structure. Each segment is then converted into a fixed-length dense vector representation using a multilingual embedding model. The document's metadata (e.g., title and description) is also embedded into a vector $\mathbf{w}_m$, which serves as a representation of the document's core theme.

Step 2: Core Identification. We compute the cosine distance between each segment's embedding and the metadata embedding $\mathbf{w}_m$. The segments with the smallest distances (highest similarities) to $\mathbf{w}_m$ are selected as the document's "core content". Specifically, we select the top $k\%$ of segments, where k is a configurable parameter that controls the strictness of core content selection.

Step 3: Outlier Detection. We define "outlier groups" by embedding phrases indicative of unwanted content types (e.g., advertisements, legal disclaimers, navigation). For each non-core segment, we compute its distance to the closest outlier group and its distance to the core content set. A segment is flagged for removal if either it is too close to an outlier group or it is too distant from the core content (distance above threshold d), where d is a configurable distance cutoff parameter.

Step 4: Segment Removal. Flagged segments are removed from the document, and the removal

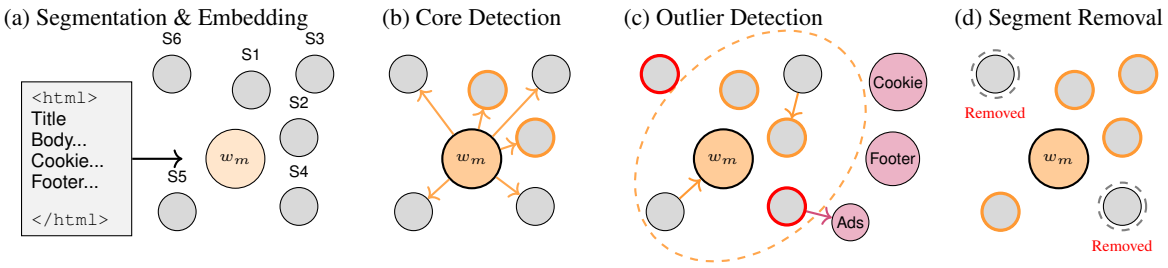

Figure 1: **Overall pipeline of SORE. (a) Segmentation & Embedding:** We split the HTML into segments (S1–S6) and embed them along with a metadata vector $\mathbf{w}_m$. **(b) Core Identification:** Compute similarity of each segment to $\mathbf{w}_m$ and select the top $k\%$ (orange outlines). **(c) Outlier Detection:** Embed predefined outlier groups (purple). For each non-core segment, check distance to the core region (dashed circle) and outlier groups. Flag segments that are too distant from the core or too close to outliers. **(d) Segment Removal:** Remove flagged segments (dashed/gray), keeping the remaining set as the cleaned content.

reason is recorded (e.g., "matched *disclaimers*" or "too irrelevant"). This explanation provides transparency and aids in system refinement.

Figure 1 illustrates these four steps, showing how segments are embedded, core content is identified, outliers are detected and removed. Figure 2 provides an overview of the system architecture, highlighting the key components and data flow.

3.2 Implementation Optimizations

For processing millions of documents daily in production, computational efficiency is critical. We optimized SORE through several techniques:

Approximate Nearest Neighbor Search. Computing cosine distances at scale between large numbers of high-dimensional vectors can be computationally expensive. We leveraged Voyager[1], an approximate nearest neighbor (ANN) implementation that uses HNSW (Hierarchical Navigable Small World) under the hood. This provides significant efficiency gains with high accuracy.

Precomputed Indices. During initialization, we create an ANN index and add the outlier group embeddings to it, generating a byte dump of this index. For each document to be cleaned, we load this precomputed index, add the newly computed core content and metadata embeddings, and query for nearest neighbors. This approach avoids rebuilding the entire index for each document.

Optimized Distance Calculations. Since modern embedding models typically produce normalized vectors, we use inner product distance (1 - dot product) rather than full cosine distance computation, reducing computational overhead.

[1] https://github.com/spotify/voyager

Batched Processing. Embedding computation is performed in batches to maximize throughput when processing multiple documents, optimizing API usage and reducing per-document latency.

In our production Java implementation, the cleanup of each HTML file takes an average of 200 milliseconds, with the external API call for embedding computation accounting for most of the duration (over 100 ms). This performance enables SORE to process millions of documents daily within reasonable time and cost constraints.

3.3 Key Design Decisions

3.3.1 Balancing Efficiency and Semantic Understanding

SORE addresses three key challenges for industrial deployment: (1) **Cost efficiency**: LLM inference costs approximately 25× more than our embedding-based approach, saving hundreds of thousands of dollars monthly at scale; (2) **Latency**: SORE processes documents in 200ms compared to LLMs' 2500ms, meeting strict production constraints; and (3) **Determinism**: Unlike LLMs that may produce inconsistent results, SORE provides transparent, deterministic explanations for content removal decisions.

3.3.2 Core Content Identification Strategy

We chose **metadata similarity** as our approach for identifying core content, using document metadata as a semantic anchor. This offers several advantages: it typically reflects the document's main theme, is available for most web documents, operates language-agnostically, and establishes a semantic "north star" for identifying relevant content. Empirical testing showed that selecting the top $k\%$ of segments most similar to metadata pro-

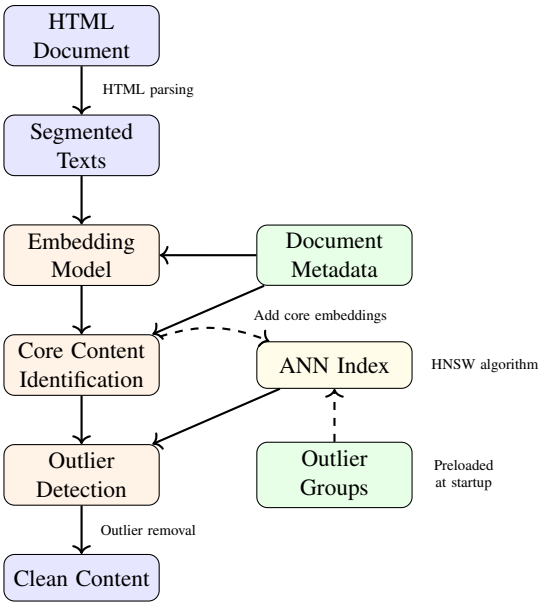

Figure 2: SORE architecture showing the optimized processing pipeline. The system parses HTML documents, segments text, and processes content through an embedding model. Core content is identified using metadata similarity, then an ANN index enables efficient outlier detection by comparing with preloaded outlier groups. This efficient architecture processes millions of documents daily with minimal latency.

vides a reliable core content identification mechanism across diverse document types.

3.3.3 Outlier Group Development

Our outlier groups were developed through iterative analysis combining data analysis and domain expertise. We implemented **semantic clustering** to represent outlier groups as clusters in the embedding space, allowing flexible matching of semantically similar content even when exact phrases differ. Each outlier group was tuned through **precision-recall balancing**, and our production system enables **continuous refinement** by logging removal decisions for ongoing improvement. The set of outliers used in this study, together with the performance analysis that SORE enables in choosing these keywords, is provided in Appendix A.

4 Experimental Evaluation

4.1 Datasets and Evaluation Setup

We evaluated SORE using two in-house HTML datasets representing real-world content cleaning challenges:

SORE-SMALL This dataset contains approximately 200 samples with hand-extracted main content from various websites across multiple languages and domains. The manually extracted content serves as a high-quality ground truth for evaluating extraction precision and recall.

Method	F-score	Precision	Recall
LLM (tag-depth)	**0.765**	0.895	0.711
LLM (raw html)	0.690	0.865	0.637
LLM (raw text)	0.583	0.795	0.520
SORE (cohere, c=0.5, k=10%)	*0.732*	0.700	0.840
ReadabilityJS	0.678	0.569	0.936

Table 1: Performance comparison on SORE-SMALL across different content extraction methods. SORE achieves near-LLM performance at a fraction of the computational cost. Precision measures the proportion of extracted text that belongs to the ground truth, while recall measures the proportion of ground truth text that was successfully extracted. F-score is the harmonic mean of precision and recall.

SORE-LARGE This dataset comprises approximately 20,000 samples with automatically extracted ground truth using a combination of ReadabilityJS and n-gram–based content cleanup. It focuses on high precision, removing groups of characters that appear on multiple pages across the web in a multi-million document corpus (e.g., legal disclaimers that appear on every page of a given domain).

For evaluation, we compared SORE against several baseline approaches:

ReadabilityJS A popular open-source HTML content extractor based on structural heuristics, widely used in industry.

LLM Variants We tested three LLM-based approaches: (1) LLM (raw HTML) providing the entire HTML content to the LLM for extraction; (2) LLM (raw text) extracting the complete text content from HTML as input; and (3) LLM (tag-depth) a hybrid approach supplying text content along with HTML tag information and tree depth. The relevant LLM prompts and additional discussions can be found in Appendix B.

4.2 Performance Comparison

4.2.1 Extraction Quality

Table 1 compares SORE against other content extraction methods on SORE-SMALL. SORE achieved a near-LLM level F-score with significantly lower computational requirements.

The results demonstrate that SORE achieves 96% of the best LLM approach's F-score (0.732

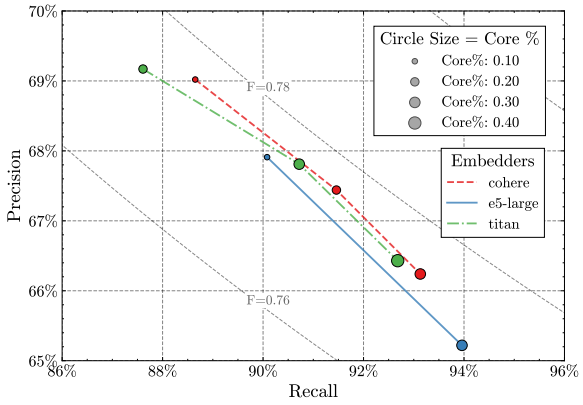

Figure 3: Precision-recall trade-offs for different embedding models and SORE parameter settings on SORE-LARGE. AWS Titan (1024d) with core=20% and cutoff=0.8 provides the best balance of precision, recall, and cost. Each point on the curves represents different parameter configurations.

vs. 0.765) while offering significant advantages in computational efficiency. Notably, SORE outperforms ReadabilityJS by 7.9% in F-score, with substantially higher precision (0.700 vs. 0.569) while maintaining strong recall.

4.2.2 Embedding Model Comparison and Parameter Tuning

Figure 3 shows the precision-recall trade-offs for various embedding models and parameter configurations on SORE-LARGE. Each point represents a different combination of core percentage (k) and embedder type, with the best distance cutoff (d) parameters per model family. We compare two commercial solutions, Cohere and AWS Titan, with the open source multilingual embedding model e5-large (Wang et al., 2024).

For this dataset, ReadabilityJS scores 0.596 precision and 0.988 recall, while LLM (tag-depth) achieves 0.885 precision and 0.718 recall (both outside the graph). AWS Titan emerged as the most cost-effective choice ($\sim$ 200 CHF/month), with comparable performance to more expensive solutions ($\sim$ 1200 CHF/month for Cohere). The optimal parameters for the AWS Titan-based SORE were found to be 1024-dimensional embeddings, 0.8 distance cutoff, and 0.2 core percentage.

The parameter tuning experiments revealed that higher values of distance cutoff (d) increase precision but reduce recall, lower values of core percentage (k) make the system more selective but may miss relevant content, and higher-dimensional embeddings generally perform better. These findings enabled us to select parameters that balanced performance and cost for our production deployment.

4.3 Multilingual Capability and Case Studies

4.3.1 Multilingual Performance

A key advantage of SORE is its language-agnostic operation. Table 2 presents examples of text segments removed by SORE across multiple languages, demonstrating the system's multilingual capabilities and semantic understanding.

Unlike traditional approaches that rely on language-specific patterns or rules, SORE leverages multilingual embedding models that capture semantic relationships across languages. This enables effective content extraction for documents in Chinese, French, Spanish, and other languages without requiring separate models or rule sets.

5 Industrial Impact and Cost Analysis

5.1 Production Deployment

SORE is currently deployed in a production environment, processing millions of documents daily across multiple languages. The system is implemented as a scalable service that integrates with existing data processing pipelines, providing cleaned content for downstream tasks such as classification and information retrieval.

Our production deployment focuses on four key aspects: (1) **Horizontal scaling** with multiple instances processing documents in parallel; (2) **Comprehensive monitoring** capturing performance metrics and removal decisions for continuous improvement; (3) **Fallback mechanisms** that revert to more conservative extraction when SORE removes unexpectedly large portions of a document; and (4) **Configurable parameters** that can be adjusted based on specific use cases and language requirements. To promote reproducibility and further research, we will make our implementation and evaluation datasets publicly available.

5.2 Cost and Efficiency Comparison

A key advantage of SORE over LLM-based approaches is its significantly lower computational cost. Table 3 compares the cost and performance characteristics of different approaches.

SORE achieves near-LLM performance at a fraction of the cost, with 12.5$\times$ lower latency (200ms vs. 2500ms) and 25$\times$ lower cost ($600 vs. $15,000 per million documents) when using AWS Titan embeddings. For our produc-

URL	Title	Removed Text	Reason
huffpost.com/...	10 Things Guests Notice Most About Your Home	SolStock via Getty Images	Source
foodsguy.com/...	Coconut Sugar Vs Brown Sugar	*This post may contain affiliate links. Please see my disclosure to learn more.	Affiliate Disclosure
buzzfeed.com/...	This Black Widow Moment...	03:27 PM - 29 Apr 2019	Last updated
dealmoon.com/...	Dyson V12 Detect Slim 激光探测无绳吸尘器翻新 $349.99	点击购买>>	Buy
blog-rct.com/...	Melvyn Jaminet fait passer un message...	A lire ci-dessous :	Also read
lapatilla.com/...		¡Únete al club ahora! Suscríbete al boletín más importante de Venezuela	Subscribe for free
cleanmyspace.com/...	Bathroom Cleaning: 10 Things...	Learn More About The 3 Wave Cleaning System	[too irrelevant]
jagranjosh.com/...	Only People With 20/20 Vision Can Spot...	Your Way Of Clenching Your Fist Reveals Your Hidden Personality Traits	[too irrelevant]

Table 2: Examples of text removed by SORE. The first three rows show examples of removed text with specific reasons. The next three rows demonstrate the system's multilingual capabilities (Chinese, French, Spanish). The last two rows show text removed because it was semantically too distant from the core content.

Method	F-score	Avg. Latency	Cost per 1M docs
LLM (tag-depth)	0.793	2500 ms	$15,000
ReadabilityJS	0.743	50 ms	$7
SORE (AWS Titan)	0.776	200 ms	$600
SORE (Cohere)	0.777	250 ms	$3,600

Table 3: Cost and performance comparison using SORE-LARGE. SORE with AWS Titan provides the best balance of performance and cost, with a latency 12.5× lower than LLMs and cost 25× lower per million documents.

tion system processing over 30 million documents monthly, SORE saves approximately $432,000 annually compared to an LLM-based approach while delivering comparable quality. This substantial cost reduction has made advanced semantic content cleaning viable at scale.

6 Conclusion

We introduced SORE (Semantic Outlier Removal), a cost-effective, transparent method for removing unwanted content from web documents while preserving their core message. By leveraging multilingual sentence embeddings and approximate nearest-neighbor search, SORE achieves performance comparable to LLM-based approaches at a fraction of the computational cost.

Our experiments demonstrate that SORE outperforms traditional structure-based methods while maintaining high precision across diverse multilingual scenarios. The system's transparency—providing clear reasons for why specific content is removed—facilitates debugging and continuous improvement.

SORE is currently deployed in production, processing millions of documents daily across multiple languages. Its efficiency and effectiveness make it a practical solution for large-scale content extraction and cleaning in industrial settings. To promote reproducibility and further research in this area, we will make our implementation and evaluation datasets publicly available.

Future work will explore integrating SORE with domain-specific knowledge bases, refining outlier group definitions based on ongoing accuracy analysis, and extending its application to more nuanced tasks such as sentiment-based filtering.

Ethics Statement

SORE is designed to extract main content from web pages while respecting copyright and terms of service. The system does not alter the meaning of content but rather removes extraneous elements. We acknowledge the potential risk that in some cases, SORE might remove content that some users consider important. To mitigate this risk, our implementation includes detailed logging of removal reasons and fallback mechanisms when excessive content is removed.

References

Cohere. https://cohere.ai. Accessed: 2023-11-15.

Readability.js. https://github.com/mozilla/readability. Accessed: 2023-11-15.

Mikel Artetxe and Holger Schwenk. 2019. Massively multilingual sentence embeddings for zero-shot cross-lingual transfer and beyond. *Transactions of the Association for Computational Linguistics*, 7:597–610.

Emily M. Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. 2021. On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, pages 610–623.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33:1877–1901.

Katharina Hämmerl, Alina Fastowski, Jindřich Libovický, and Alexander Fraser. 2023. Exploring anisotropy and outliers in multilingual language models for cross-lingual semantic sentence similarity. *Preprint*, arXiv:2306.00458.

Christian Kohlschütter, Peter Fankhauser, and Wolfgang Nejdl. 2010. Boilerpipe: A boilerplate removal and fulltext extraction library. In *Proceedings of the 19th International Conference on World Wide Web*, pages 661–662. ACM.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, pages 3111–3119.

Thuat Nguyen, Hao Chi, Long Pham, Nigel Tran, Kafai Tran, Wei Xie, Mona Abdulhai, Dimitri Semenov, Alim Khaddaj, Jón Gudmundsson Einarsson, et al. 2023. CulturaX: A cleaned, enormous, and multilingual dataset for large language models in 167 languages. *Preprint*, arXiv:2309.09400.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 1532–1543.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, pages 3982–3992.

Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, Matthias Galliard, et al. 2023. BLOOM: A 176b-parameter open-access multilingual language model. *Preprint*, arXiv:2211.05100.

Mahnaz Taleb Sereshki, Morteza Mohammadi Zanjireh, and Mahdi Bahaghighat. 2023. Textual outlier detection with an unsupervised method using text similarity and density peak. *Acta Univ. Sapientiae, Informatica*, 15(1):91–110.

Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. 2024. Multilingual e5 text embeddings: A technical report. *arXiv preprint arXiv:2402.05672*.

A Outlier Groups

SORE uses a carefully curated set of outlier groups to identify and remove unwanted content. These groups were developed through extensive analysis of web content patterns and iteratively refined based on performance metrics. Each group represents a category of content typically not part of the main article text.

A.1 Outlier Group Performance

We analyzed the accuracy of removal for different outlier keywords. Table 4 shows the least accurate keywords from our analysis.

Phrase	Occurrence	Accuracy
Home	9777	0.510
Frequently asked questions	822	0.540
Similar	1559	0.543
dd/mm/yyyy	117	0.556
Not found	532	0.564
21.02.2023	2219	0.591
Order	1996	0.599
Error	2600	0.600
URL	1177	0.601
404	3391	0.602

Table 4: Removal accuracy for the 10 least accurate outlier keywords. Even the least accurate keywords exhibit accuracy above 0.5, with most outlier groups performing significantly better.

The results indicate that some ambiguous terms like "Home" have relatively lower accuracy due to their context-dependent nature—they may appear in both navigation elements and legitimate main content. However, even these challenging outlier groups achieve better than random performance, and the system's overall accuracy benefits from the combination of multiple outlier detection signals.

A.2 Outlier Group Keywords

The outlier groups are represented as sets of phrases and patterns that, when embedded, create semantic clusters in the embedding space. The following list shows our production outlier groups organized by category:

A.2.1 Date-time Related Content

"Date", "21.02.2023", "21.02.2024", "21.02.2025", "Published at", "Last updated", "Time", "Published", "Updated", "dd/mm/yyyy", "mm/dd/yyyy", "yyyy-mm-dd", "dd.mm.yy"

A.2.2 Authorship Information

"Author", "Writer", "Contributor", "Editor", "Posts", "Written by"

A.2.3 Comment Sections

"Comment", "Reply", "Feedback", "Discussion", "Leave a comment"

A.2.4 Source Attribution

"Source", "Website", "Publisher", "URL", "Link"

A.2.5 Related Content Links

"Related", "Read more", "Look:", "Similar", "See also", "Also read", "Read next", "Get more", "Frequently asked questions"

A.2.6 Calls to Action

"CTA", "Buy", "Shop", "Order", "Click here", "Check out", "View more", "Visit", "Let me know", "Download", "Subscribe", "Sign up", "Contact us", "Receive notifications"

A.2.7 Navigation Elements

"Breadcrumbs", "Home >", "Home > About", "Navigation", "Home", "About"

A.2.8 Contact Information

"Contact", "Email", "Phone", "Address", "Contact us"

A.2.9 Social Media Elements

"Social", "Facebook", "Twitter", "Instagram", "LinkedIn", "TikTok", "Share", "Like", "Follow", "3425 views"

A.2.10 Legal Content

"Legal", "Terms", "Privacy", "Policy", "Disclaimer", "Cookie", "Accept", "Policy", "Settings"

A.2.11 Page Infrastructure

"Footer", "Copyright", "All rights reserved", "Search", "Find", "Look for", "Explore", "Error", "404", "Not found", "Page not found", "Error", "Try again later"

A.2.12 Commercial Content

"Advertisement", "Sponsored", "Promotion", "Sponsor", "Subscription", "Subscribe", "Newsletter", "Membership", "Join", "Affiliate", "Affiliate links", "Disclosure", "Affiliate Disclosure"

A.2.13 Miscellaneous Boilerplate

"Refresh this page", "Login required", "License", "Enter your email", "Thank you for reading", "Subscribe for free"

B LLM Prompts

For the LLM baseline comparisons, we systematically developed and tested several prompting strategies. Through empirical evaluation, we found that providing structured context about HTML tags and their depth in the document tree ("tag-depth" approach) yielded the best results, as it strikes a balance between:

1. Providing sufficient structural context that pure text approaches lack
2. Avoiding overwhelming the model with full HTML markup
3. Creating a constrained output format (line numbers) that prevents hallucination.

The tag-depth approach also significantly outperformed both raw HTML and raw text approaches in our experiments, as shown in Table 1. Below are the three prompting strategies we evaluated:

Raw HTML Prompt

Analyze the given HTML and extract only the main article/post/discussion content, ensuring that the extracted content meets the criteria for a perfect extraction as defined below.
1. Include All Core Content: - Extract the complete core content of the main article, which are exclusively: - Title - Headings and Subheadings - All paragraphs that form the continuous, coherent text of the article
2. Exclude All Irrelevant Elements: - Do not include any peripheral or irrelevant elements such as: - Headers, footers, navigation bars, sidebars - Comments, author bios, blog names, date stamps, author names, etc. - Advertisements (e.g., "Buy now") - Breadcrumbs (e.g., "Home > Category > Subcategory") - Promotional teasers (e.g., "Sign up for our newsletter") - Navigation links (e.g., "Go to the next article") - Irrelevant image captions (e.g., "Source: Getty Images") - Calls-to-action (e.g., "Join our group") - Recommendations for other articles (e.g., "See related article: ...") - Contact information (e.g., "Reach us at...") - Social media links (e.g., "Connect with @...") - Disclaimers or cookie notices
3. Output Format: - Provide only the main article content without any additional text or commentary. - Do not include any formatting tags or metadata.
Input HTML: text
Output format: text

Raw Text Prompt

Analyze the given text and extract ONLY the main article content:
1. Identify the core article content, focusing on continuous, coherent text that with a clear title. 2. Ignore all peripheral content: headers, footers, navigation, sidebars, comments, author bios, blog names, date stamps, author names, etc, but do not ignore the content that is included in the main article. 3. Output the main article content.
Input text: text
Output format: text

Tag-depth Prompt (Best Performing)

For the given numbered lines of text from an HTML with their parent tags and the tag depths in the HTML tree, extract the core content (like ReadabilityJS).
1. IDENTIFY CORE CONTENT - Each page has a main content, which can be an article, blog post, forum thread, etc. - Extract the main content, which includes the title, headings, paragraphs, and any other relevant text. - Exclude all peripheral content: headers, footers, navigation, sidebars, comments, author bios, blog names, date stamps, author names, etc.
2. EXCLUDE IF ANY OF THESE ARE TRUE: - Appears in site navigation sections - Contains ANY of these patterns: * Social media handles or URLs * Date stamps or bylines * Copyright notices * Contact information * Newsletter signup text * "Related article" references * Adver-

tisement markers * Image credits or captions * Tags or categories * Call-to-action phrases * Navigation instructions * Comment section markers * Share button text * Footer content - Some examples are: 'Related: you will not believe what happened next' or 'Sign up to our newsletter' or 'Source: Getty Images' or 'Contact us via Instagram' or 'Date: 2022-01-01'"
3. VALIDATE SELECTION - Verify selected lines form a coherent narrative - Check that no essential context is lost - Confirm removal of ALL peripheral content
Input: text
Output format: [comma-separated list of line numbers containing ONLY the essential content]
Notes: - Include ONLY numbers in the output, no explanations - If a line contains mixed content, exclude it entirely - When in doubt about a line, exclude it - Aim for maximum precision over recall
Example output: 1,2,3,5,8,...