Team JLP at PAN 2026: Span-Based Hybrid Source Retrieval for Generative Plagiarism Detection

Notebook for the PAN Lab at CLEF 2026

Liam Kaan Cody[1,†], Peter Felix Panzitt[1,†] and James Ernst Cox[1,†]

[1]Graz University of Technology, Graz, Austria

Abstract

The PAN 2026 generative plagiarism detection task asks systems to recover the source papers used to produce a scientific article written by a large language model (LLM). We describe Team JLP's source retrieval system. The main difficulty is length: a suspicious article can draw from several sources, and a full-document query often washes out local evidence. We therefore treat the article as a collection of sentence-level retrieval queries. Each sentence searches a full-document BM25 index and a dense index over structure-aware source-document chunks. We fuse the sparse and dense rankings with Reciprocal Rank Fusion (RRF), then combine the sentence-level scores with CombSUM to obtain one source-document ranking. Since no labeled development set was released, we created a 200-query synthetic set from the source corpus and used it to choose the span and fusion settings. On this synthetic set, the selected hybrid run reached normalized discounted cumulative gain at rank 10 (nDCG@10) of 0.9893 and Recall@10 = 0.9950. On the official test set of 200 suspicious documents, our primary submission achieved Recall@10 = 0.8542 and nDCG@10 = 0.7994, ranking first among submitted systems. These results indicate that local sentence-level evidence can recover most sources in generated scientific text when it is aggregated across the whole document.

Keywords

Generative plagiarism detection, source retrieval, hybrid retrieval, reciprocal rank fusion, query-by-document retrieval

1. Introduction

The academic integrity of published documents is becoming increasingly difficult to guarantee. As large language models (LLMs) make fluent, accurate-sounding documents easier to produce, we face a different problem from the one that typical plagiarism detection mechanisms were built to address. In generated plagiarism, source material can be reused while the surface wording is changed through paraphrase; what remains may be shared ideas, terminology, and claims rather than copied sentences. For this kind of reuse, the problem moves from near-duplicate matching to source attribution [1]. At the same time, major venues have updated policies for AI writing assistance, with venue-specific disclosure rules and restrictions [2, 3, 4], and AAAI announced an LLM-assisted peer-review pilot for AAAI-26 [5], which makes the question of where generated text comes from a current, practical concern [1].

This setting is the main target for the PAN 2026 generative plagiarism detection task. The previous 2025 edition introduced plagiarism by replacing individual paragraphs in otherwise genuine publications, which made the task manageable for naive embedding-similarity methods. The task overview reports recall up to about 0.8 from such baselines, but poor generalization to older datasets [6]. To address this, the 2026 edition generates suspicious documents from the ground up using two or more source articles, restoring the retrieval aspect of plagiarism detection [1, 7, 8]. Participants in the source retrieval task receive an LLM-generated scientific article and must rank the source documents in a fixed corpus that contributed to it. In this structure, there are no passage-level alignments, the relevant sources are not known in advance, and a single suspicious document may draw from several.

CLEF 2026 Working Notes, 21 – 24 September 2026, Jena, Germany

[†]These authors contributed equally.

lcody@student.tugraz.at (L. K. Cody); peter.panzitt@student.tugraz.at (P. F. Panzitt); james.cox@student.tugraz.at (J. E. Cox)

0009-0002-9221-7655 (L. K. Cody)

We approach the task as source retrieval with long, multi-source queries. Because a suspicious document draws from several sources at once, treating the full text as a single query dilutes source-specific signals: different sources pull the representation in different directions, and BERT-based rankers cannot process the full length regardless [9, 10]. We therefore split each suspicious document into sentence-level spans and retrieve independently per span, combining a BM25 index over full source documents with a dense index over structure-aware source chunks. The two rankings are fused per span with Reciprocal Rank Fusion (RRF) and aggregated into a single document ranking with CombSUM.

On the official test set of 200 suspicious documents, our submission placed first among systems submitted to the PAN 2026 subtask. The pipeline uses only off-the-shelf retrieval components and requires no task-specific training data, yet it recovers most sources even when the generated text is heavily paraphrased. Because the method indexes source documents rather than learning generation-specific features, it can be applied to new corpora without retraining and does not need to be updated when the generation model changes.

1.1. Contributions

Our contributions are as follows:

- We formulate source retrieval for generated plagiarism as evidence aggregation over short spans rather than retrieval using a single document-length query. The system retrieves each sentence independently and combines the resulting rankings, preserving source-specific evidence that may be diluted in a whole-document representation.
- We present a training-free hybrid retrieval pipeline for this setting. It combines full-document BM25 with dense retrieval over structure-aware chunks, applies RRF at sentence level, and uses CombSUM to aggregate sentence rankings into the final source-document ranking.

2. Related Work

In PAN 2026, generative plagiarism detection is a document retrieval task. The system receives an LLM-written scientific document and must rank the source articles in a fixed corpus that likely contributed to it. Those sources are not known in advance, and a single suspicious document can draw from several articles. A suspicious document is also far longer than a normal keyword query [1]. The setting is therefore close to query-by-document (QBD) retrieval, where a full document is the query and systems must account for the lengths of both the query and the candidate documents [10].

While full documents express richer information needs, they exceed the input length of standard BERT-based rankers [9], and long-context Transformer variants have not shown high effectiveness in QBD tasks [11, 10]. One recent QBD approach avoids scoring the full query-document pair as a single sequence. Askari et al. split queries and candidate documents into sentences, compare sentence embeddings, and show that their sentence-level relevance model benefits from covering the full length of both texts [10].

Earlier work on plagiarism source retrieval follows a related multi-query pattern. Williams et al. partition suspicious documents into five-sentence passages, generate several keyword queries from each passage, and combine the top search results for re-ranking before candidate documents are downloaded [12]. Kong et al. focus on this aggregation problem and propose AggSR, a learning-to-rank model that incorporates search-result aggregation caused by correlations among queries from the same suspicious document [13]. This line of work motivates our decision to decompose each suspicious document into shorter spans before retrieval.

After decomposition, each span still has to be matched against a large scientific corpus. Hybrid retrieval commonly combines BM25 [14] with dense retrieval to merge exact lexical overlap with learned semantic similarity [15]. While BM25 provides strong zero-shot generalization [16], combining it with dense retrieval has been shown to improve out-of-domain and adversarial robustness [17]. We assume generated scientific text preserves some lexical evidence from its sources, for example,

technical terminology, entity names, or recurring scientific phrasing. At the same time, the surrounding content may be substantially paraphrased. We therefore combine BM25 with dense retrieval. To avoid calibrating BM25 and dense similarity scores directly, we fuse the two ranked lists with RRF [18], a simple rank-based method for combining retrieval results.

Together, these observations motivate the three main design choices in our method: span-level querying, complementary sparse and dense backends, and rank-based fusion.

3. Data

We use three data resources: the organizer-provided source corpus [7], a synthetic development set built from the same source collection, and the official unlabeled test queries. The source corpus D contains about 87,000 scientific articles and forms the candidate source set against which both official and synthetic queries are ranked.

For development, we use a synthetic labeled set with 200 suspicious documents and document-level source labels. The labels identify which source articles were used to generate each synthetic query, but they do not provide passage-level alignments. We therefore use this set to compare retrieval backends and tune span decomposition and rank fusion at the document-ranking level. The generation procedure itself is described in the Method section.

The official test set contains 200 suspicious documents generated for the task. We use this dataset for the submitted runs after selecting the pipeline on the synthetic development set.

4. Method

Following the task formulation and related work, our method treats generative plagiarism detection as source retrieval with long, multi-source queries. We describe three parts of the method: the synthetic development setup, corpus representations for sparse and dense retrieval, and the inference pipeline for ranking candidate sources.

4.1. Synthetic Dataset Generation

Because no organizer-provided development dataset exists, we build a synthetic dataset from the same corpus documents to evaluate our pipeline and tune hyperparameters. Let D denote the corpus of legitimate source documents. For each synthetic instance, we sample a subset of source documents $S \subset D$, where $|S| = n$, and prompt an LLM to generate a survey article based on the provided documents. To produce a varied dataset, we control three axes per query: the number of source articles n, the target token length τ, and the structural paraphrase severity ρ. Given a generation prompt P, we define the resulting plagiarized query q as

$$q = LLM(P, \tau, \rho, S).$$

The paraphrase severity ρ is represented with three levels:

Light Paraphrasing "Stay fairly close to the source wording and structure where natural."

Rewrite "Modify sentence structure, reword phrases, and incorporate elements of general knowledge to ensure coherence. The less token overlap, the better."

Full Rephrase "Completely rephrase in your own words. Feel free to incorporate elements from general knowledge to ensure coherence, flow, and better understanding."

These levels were chosen to mirror the paraphrasing prompts used in the 2025 PAN Generative Plagiarism Detection task [19]. We sampled the number of source articles n from a clipped Gaussian distribution $\mathcal{N}(\mu = 2.0, \sigma = 1.5)$ restricted to $n \in [2, 4]$. Target lengths τ were drawn uniformly

from $\{500, 750, 1000, 1250, 1500, 1750, 2000\}$ tokens. The paraphrase severity ρ was drawn from the three levels with probabilities 0.6 (light), 0.3 (rewrite), and 0.1 (full rephrase). For text generation, we used `gemma-4-31b-it` via the Google AI Studio API with seed 461 and `max_output_tokens` set to τ [20]. The generation prompt required the output to include standard Markdown sections: Abstract, Introduction, Methodology, Experiment, Discussion, and Conclusion.

The generated query q is paired with its source set S as ground-truth relevance labels for the development experiments in Section 5.

4.2. Corpus Representation and Indexing

We use two complementary representations of the source corpus. Let D denote the source corpus, with each source document $d \in D$. These representations feed the two retrieval branches in the inference pipeline.

For lexical retrieval, each source document d is treated as a single full document and indexed with BM25 [14]. This branch captures exact-matching signals such as domain-specific terminology, formulas, and names. We implemented the sparse branch using the `bm25s` library and its Lucene-style BM25 variant [21].

4.2.1. Dense Structure-Aware Indexing

For dense retrieval, each source document d is instead split into a set of chunks

$$C_d = \{c_1, \ldots, c_k\},$$

and each chunk is encoded with a pre-trained bi-encoder E. We used the General Text Embeddings base model (GTE-Base, `Alibaba-NLP/gte-base-en-v1.5`) [22, 23] via `sentence-transformers` [24] to encode all chunks before retrieval. The retriever can then compare passages by semantic similarity. When the generated text paraphrases a source heavily, lexical overlap is sparse, so semantic matching carries more of the signal.

Encoding each source article as a single dense representation is a poor fit for document-to-document retrieval: Dense Passage Retrieval (DPR)-style models have limited input length, and paragraph-level aggregation has been shown to outperform document-level dense retrieval for long documents [25]. Therefore we split documents using a structure-aware heuristic rather than fixed-length windows, aiming for blocks of 220 to 320 words.

We first divide each document into paragraph blocks by blank lines. Motivated by the use of multi-paragraph passages as retrieval units [26], this chunking strategy keeps paragraphs and section boundaries mostly intact before applying word-limit splits. Heading boundaries (detected by regex matching Roman numerals, decimals, common academic section titles, and short title-case blocks) mark section breaks. Consecutive blocks under the same heading are merged until they reach the maximum word limit. A block that exceeds the limit is split further: at sentence boundaries first, then via an overlapping sliding window of 280 words with a 40-word overlap if a sentence remains too long.

During retrieval, we map chunk-level evidence back to source documents by max-pooling over C_d [27].

4.3. Pipeline

At inference time, the system applies the two corpus representations to a suspicious document and returns a ranked list of candidate source documents. It first splits the query into spans, retrieves sparse and dense evidence for each span, and then merges the span-level rankings. Figure 1 summarizes the full retrieval pipeline.

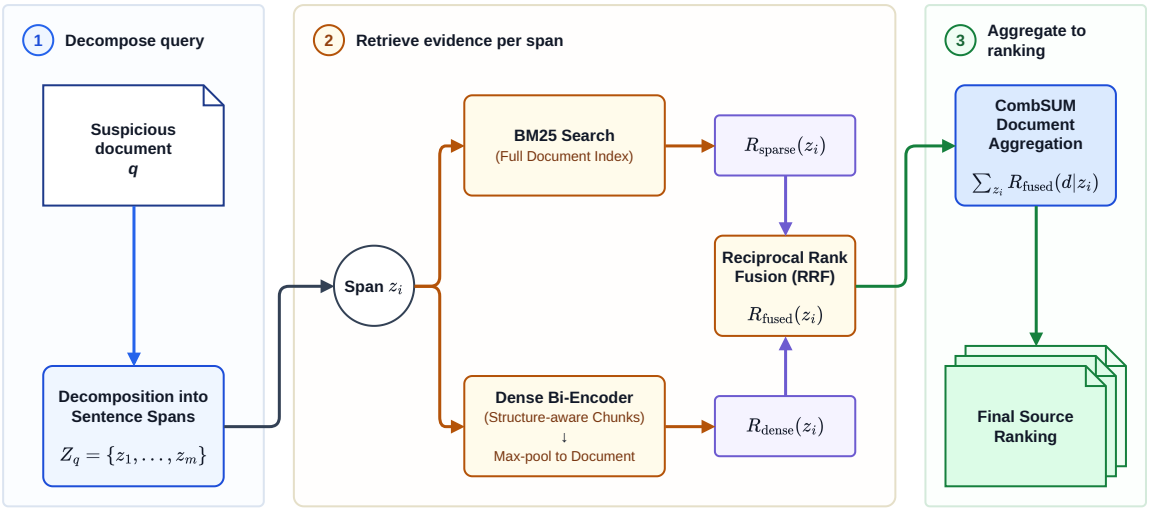

Figure 1: Span-based hybrid retrieval pipeline with per-span sparse–dense rank fusion and CombSUM document aggregation.

4.3.1. Query Decomposition

Following the span-based retrieval motivation in Section 2, each suspicious document q is decomposed into a set of atomic text spans

$$Z_q = \{z_1, \ldots, z_m\}.$$

Each span z_i is treated as an independent query. In our implementation, we use individual sentences as these atomic spans (span size $= 1$). This helps retrieval focus on localized evidence of plagiarism by separating potentially plagiarized regions from surrounding benign text. Because a generated query may combine content from multiple sources, representing the full document as a single query can produce a diffuse representation that averages over different source signals and may not match any individual source well [28, 10]. Smaller spans are more likely to focus on a single topic or source document.

4.3.2. Hybrid Retrieval

Lexical Retrieval with BM25 For lexical retrieval, each query span $z \in Z_q$ is submitted to the previously built full-document BM25 index. The output is a ranked list $R_{sparse}(z)$ of source documents based on exact token overlap, weighted by term frequency, inverse document frequency, and document-length normalization.

Dense Retrieval For dense retrieval, each query span $z \in Z_q$ is encoded using the same bi-encoder E as the chunk index described in Section 4.2.1. We compare the query embedding against the pre-computed embeddings of all source chunks $\bigcup_{d \in D} C_d$. The task requires source-document rankings, so we aggregate chunk-level scores at document level by taking the maximum score over C_d for each document d. The dense branch returns the ranking $R_{dense}(z)$.

Backend Fusion: BM25 + Dense Retrieval For each span $z \in Z_q$, the lexical BM25 ranking $R_{sparse}(z)$ and the dense ranking $R_{dense}(z)$ are combined into a single ranking $R_{fused}(z)$. The raw BM25 and dense similarity scores are not directly comparable, so we fuse rank positions rather than scores using RRF [18], implemented through `ranx` [29]:

$$R_{fused}(d \mid z) = \frac{1}{K + rank_{sparse}(d \mid z)} + \frac{1}{K + rank_{dense}(d \mid z)}$$

Table 1
Development-set baselines on the synthetic dataset.

System	Query	Span fusion	nDCG@10	Recall@10	MRR	nDCG@100	MAP
BM25	full	–	0.8009	0.8267	0.9189	0.8325	0.7365
GTE-Base	full	–	0.5279	0.5858	0.6871	0.5783	0.4366
BM25 + GTE-Base RRF	full	–	0.7744	0.8113	0.9015	0.8093	0.7016
BM25	sentences	CombSUM	0.9343	0.9525	0.9804	0.9492	0.9027
GTE-Base	sentences	CombSUM	0.8462	0.8900	0.9152	0.8753	0.7926
BM25 + GTE-Base RRF (JLP)	sentences	CombSUM	**0.9893**	0.9950	**1.0000**	**0.9900**	**0.9803**
BM25 + GTE-Base RRF (JLP-2)	sentences	CombSUM-minmax	0.9887	**0.9975**	0.9975	0.9887	0.9782

Here, K is a smoothing constant (set to $K = 1$ in our runs). Documents near the top of either ranked list receive higher scores, and documents retrieved by both branches receive contributions from both.

4.3.3. Intra-document span aggregation

For a suspicious document q with spans $Z_q = \{z_1, \ldots, z_m\}$, the previous step produces one fused backend ranking $R_{fused}(z_i)$ for each span z_i. We merge these span rankings into a single source-document ranking. Our main aggregation uses the fused span scores directly. Documents not retrieved for a span receive a score of zero for that span. We compute the final document score for query q with CombSUM [30]:

$$S(d \mid q) = \sum_{z_i \in Z_q} R_{fused}(d \mid z_i)$$

Generic spans tend to produce weak matches across many documents, while source-specific spans produce more concentrated scores. Direct summation lets those stronger local signals carry more weight. This is the version submitted as `jlp`. The second submitted run, `jlp-2`, uses the same CombSUM aggregation after min-max normalizing the fused document scores within each span. With this normalization, each span contributes on the same score range before the scores are summed. We sort source documents by the resulting score in descending order to produce the final ranked list.

5. Experimental Setup

Since official relevance labels are unavailable during development, we used the synthetic development set described in Section 3 to compare retrieval variants and select the retrieval settings for the submitted runs. For efficiency, we capped the development runs at 100 candidate source documents per suspicious document. Preliminary runs on the synthetic set showed that relevant sources were usually already within this range, so deeper rankings were unlikely to affect the comparison between retrieval variants.

5.1. Evaluation Protocol

The task measures known during development were nDCG@10, Recall@10, and Recall@100. We selected runs by nDCG@10, since it rewards systems that rank relevant sources early; the recall measures only check whether those sources appear somewhere within the cutoff. We also inspected Recall@10, mean reciprocal rank (MRR), nDCG@100, mean average precision (MAP), and Precision@10 before choosing the submitted settings. Runs, relevance judgments (qrels), and evaluation metrics are handled with `ranx` [31].

5.2. Baselines

We compared full-query and span-based variants of the sparse, dense, and hybrid retrieval backends. The full-query runs provide a document-level retrieval baseline, while the single-backend span runs

Table 2
Search spaces used for development-set tuning.

Parameter	Values
Query decomposition	full query; sentence windows of 1, 2, 3, 4, 5, 7, or 10 sentences
Span fusion	CombSUM, CombMNZ
Backend RRF K	1–10, 20, 30, 60, 90, 120, 180, 240

isolate the effect of query decomposition before backend fusion is added. Table 1 summarizes the main development-set baselines.

The full-query baselines suggest that submitting the entire generated document as a single query is not the best fit for the task. BM25 is the stronger of the two single backends on this development set, and it remains the harder baseline after adding sentence-level decomposition. The selected hybrid configuration improves over the best BM25-only span run on the main metric and on the secondary metrics.

5.3. Hyperparameter Search

We tuned the retrieval choices most directly tied to the method: query decomposition, span fusion, and the RRF constant for the hybrid backend.

Hyperparameter search was performed using Optuna [32] with a Tree-structured Parzen Estimator (TPE) sampler and a fixed seed of 42. We ran three studies, each with up to 100 trials: BM25-only, GTE-Base-only, and BM25 + GTE-Base fused with RRF. The BM25-only and GTE-Base-only spaces contain 16 nominal configurations, while the hybrid space contains 272 combinations after adding the RRF constant. A 100-trial budget, therefore, gives full practical coverage of the smaller studies and a broad, though not exhaustive, sample of the hybrid space. The BM25 and GTE-Base studies tuned the query decomposition mode and the span fusion method (CombSUM and CombMNZ [30]). The hybrid study tuned those parameters, plus the RRF constant used to merge BM25 and dense rankings for each span. During the hyperparameter search, score normalization was fixed to none; the min-max step was added only after the retrieval settings had been selected.

The search spaces were refined after preliminary runs on the synthetic development set. Those runs suggested that shorter sentence windows and smaller RRF constants were more promising, so we sampled these regions more densely while still retaining larger windows and higher K values for comparison.

The best hybrid runs favored short spans and aggressive backend fusion. The tuning results also suggested that CombSUM is a better fit for the task than CombMNZ. The selected retrieval configuration uses single-sentence spans, CombSUM for the final span aggregation, and backend RRF with $K = 1$. On the synthetic development set, the raw-score version achieved $\mathrm{nDCG@10} = 0.9893$, $\mathrm{Recall@10} = 0.9950$, $\mathrm{MRR} = 1.0000$, $\mathrm{nDCG@100} = 0.9900$, and $\mathrm{MAP} = 0.9803$. We then used the same retrieval settings for our two official submissions (`jlp` and `jlp-2`) on the test set, increasing the output depth from 100 to 1000 candidate documents per query.

6. Results

Table 3 shows our two submissions on the official PAN 2026 test set of 200 suspicious documents, alongside every best submission from other teams and the organizer-provided BM25 baseline [7]. Our primary submission, `jlp`, achieves the highest score on Recall@10, nDCG@10, nDCG@100, and MRR. The variant `jlp-2` places second on the same four metrics and first on Recall@100. The margin to the third-ranked run (`plain-dingo` by team `engstf`) is roughly 0.18 on Recall@10 and 0.16 on nDCG@10, and the margins to the BM25 baseline are roughly 0.28 on Recall@10 and 0.25 on nDCG@10.

Table 3
Results on the official PAN 2026 test set.

Team	Run	Recall@10	Recall@100	nDCG@10	nDCG@100	MRR
jlp	jlp	**0.8542**	0.9629	**0.7994**	**0.8326**	**0.9760**
jlp	jlp-2	0.8375	**0.9646**	0.7794	0.8170	0.9609
engstf	plain-dingo	0.6708	0.8404	0.6422	0.6871	0.9076
fsu-aigc-detector	amiable-report	0.6567	0.6567	0.6427	0.6427	0.8923
fosuai	oscillating-list	0.6300	0.6300	0.6263	0.6263	0.9350
baseline	BM25	0.5767	0.7825	0.5532	0.6068	0.8318
bedratiuk-lab	fuchsia-grouse	0.3862	0.3862	0.3854	0.3854	0.6191
nlp-de	nlp-de-run-01	0.0979	0.2250	0.0761	0.1114	0.1371

6.1. Discussion

While there is a clear drop in pipeline performance on the official test set compared to the development set, the relative ordering remains the same. Development nDCG@10 was 0.9893 on the hybrid configuration; on the test set, it was 0.7994. The synthetic queries were likely easier than the official test set, where generation details are unknown and may include heavier paraphrasing or domain shifts. The development dataset still served its purpose as a relative-ranking tool for hyperparameter selection, with the hybrid configuration beating each single-backend variant in our internal evaluation, and remaining the strongest submission on the test set.

The BM25 baseline also shows a drop from the development set to the test set: it reached $\mathrm{nDCG@10} = 0.8009$ on our synthetic queries, but 0.5532 on the official test set. This suggests that the official generation process preserved less directly usable lexical evidence than our own prompting setup. This is where the hybrid design helps: BM25 can exploit surviving technical terms, entities, formulas, and recurring phrasing, while the dense branch can contribute when the same source evidence is expressed with different wording.

On the test set, `jlp` is ahead on every metric except Recall@100, where `jlp-2` wins by 0.0017. Raw RRF scores preserve the sparse–dense rank evidence within a span: documents ranked highly by BM25, dense retrieval, or both receive larger contributions, while documents that appear only lower in the fused rankings receive smaller ones. This difference gets compressed by min–max normalization, which gives every span the same contribution range, so less specific spans can compete more strongly with source-specific ones. As seen with the Recall@100 result, flattening helps weakly matched true sources reach the top 100, but results in fewer strongly matched true sources reaching the top 10.

7. Conclusion

We presented Team JLP's system for the PAN 2026 generative plagiarism detection task. The approach treats generative plagiarism detection as source retrieval: it splits each suspicious document into sentence-level spans, searches with both BM25 and dense chunk retrieval, and merges the evidence into a document ranking. On our synthetic development set, the span-based hybrid configuration outperformed the full-query baselines and the single-backend variants. On the official test set, the same configuration achieved Recall@10 = 0.8542 and nDCG@10 = 0.7994, placing first on both metrics among submitted systems, and confirming that the relative ordering established on the synthetic data carried over to the official evaluation. The remaining gap suggests that source retrieval for generated articles is not only a question of matching text more effectively. It also requires a better understanding of where source evidence survives after generation: technical definitions, method descriptions, named entities, and experimental details may behave very differently from background or connective text. Future systems could also treat a suspicious document as a mixture of source-specific evidence rather than a flat set of sentence queries. Instead of summing all span scores, an aggregator could group mutually supporting spans, separate evidence for different sources, and reward candidates that explain coherent

parts of the generated article.

Declaration on Generative AI

During the preparation of this work, the authors used Gemini and Grammarly to help with grammar, spell-checking, and phrasing. After using these tools/services, the authors reviewed and edited the content as needed and take full responsibility for the publication's content.

References

[1] J. Bevendorff, M. Fröbe, A. Greiner-Petter, A. Jakoby, M. Mayerl, P. Nakov, H. Plutz, M. Potthast, B. Stein, M. N. Ta, Y. Wang, E. Zangerle, Overview of PAN 2026: Voight-Kampff Generative AI Detection, Text Watermarking, Multi-Author Writing Style Analysis, Generative Plagiarism Detection, and Reasoning Trajectory Detection, in: M. Hagen, M. Potthast, B. Stein, P. Schaer, E. Zangerle, A. Barrón-Cedeño, A. G. S. de Herrera, E. S. Salido, S. MacAvaney, J. M. Struß (Eds.), Experimental IR Meets Multilinguality, Multimodality, and Interaction. Proceedings of the Seventeenth International Conference of the CLEF Association (CLEF 2026), Lecture Notes in Computer Science, Springer, Berlin Heidelberg New York, 2026.

[2] Association for the Advancement of Artificial Intelligence, AAAI Publication Policies & Guidelines, https://aaai.org/aaai-publications/aaai-publication-policies-guidelines/, 2025.

[3] J. Boyd-Graber, N. Okazaki, A. Rogers, ACL 2023 policy on AI writing assistance, https://2023.aclweb.org/blog/ACL-2023-policy/, 2023.

[4] E. Brunskill, K. Cho, B. Engelhardt, Clarification on large language model policy, https://icml.cc/Conferences/2023/llm-policy, 2023. ICML 2023.

[5] Association for the Advancement of Artificial Intelligence, AAAI launches AI-powered peer review assessment system, https://aaai.org/aaai-launches-ai-powered-peer-review-assessment-system/, 2025. Press release.

[6] A. Greiner-Petter, M. Fröbe, J. P. Wahle, T. Ruas, B. Gipp, A. Aizawa, M. Potthast, Overview of the plagiarism detection task at PAN 2025, in: Working Notes of CLEF 2025, 2025. URL: https://arxiv.org/abs/2510.06805.

[7] A. Greiner-Petter, Y.-A. Wu, Y. Kitamura, K. W. Kim, M. Fröbe, B. Gipp, A. Aizawa, M. Potthast, Overview of the Plagiarism Detection Task at PAN 2026, in: A. Barrón-Cedeño, A. G. S. de Herrera, E. S. Salido, S. MacAvaney, J. M. Struß (Eds.), Working Notes of CLEF 2026 – Conference and Labs of the Evaluation Forum, CEUR Workshop Proceedings, CEUR-WS.org, 2026.

[8] M. Fröbe, M. Wiegmann, N. Kolyada, B. Grahm, T. Elstner, F. Loebe, M. Hagen, B. Stein, M. Potthast, Continuous Integration for Reproducible Shared Tasks with TIRA.io, in: J. Kamps, L. Goeuriot, F. Crestani, M. Maistro, H. Joho, B. Davis, C. Gurrin, U. Kruschwitz, A. Caputo (Eds.), Advances in Information Retrieval. 45th European Conference on IR Research (ECIR 2023), Lecture Notes in Computer Science, Springer, Berlin Heidelberg New York, 2023, pp. 236–241. URL: https://link.springer.com/chapter/10.1007/978-3-031-28241-6_20. doi:10.1007/978-3-031-28241-6_20.

[9] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, BERT: Pre-training of deep bidirectional transformers for language understanding, in: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), Association for Computational Linguistics, Minneapolis, Minnesota, 2019, pp. 4171–4186. URL: https://aclanthology.org/N19-1423/. doi:10.18653/v1/N19-1423.

[10] A. Askari, S. Verberne, A. Abolghasemi, W. Kraaij, G. Pasi, Retrieval for extremely long queries and documents with rprs: A highly efficient and effective transformer-based re-ranker, ACM Trans. Inf. Syst. 42 (2024). URL: https://doi.org/10.1145/3631938. doi:10.1145/3631938.

[11] A. Abolghasemi, S. Verberne, L. Azzopardi, Improving bert-based query-by-document retrieval with multi-task optimization, in: Advances in Information Retrieval: 44th European Conference on IR Research, ECIR 2022, Stavanger, Norway, April 10–14, 2022, Proceedings, Part II, Springer-

Verlag, Berlin, Heidelberg, 2022, p. 3–12. URL: https://doi.org/10.1007/978-3-030-99739-7_1. doi:10.1007/978-3-030-99739-7_1.

[12] K. Williams, H. Chen, S. R. Choudhury, C. L. Giles, Unsupervised ranking for plagiarism source retrieval, in: Working Notes for CLEF 2013 Conference, CEUR-WS, 2013.

[13] L.-L. Kong, Z.-Y. Han, H.-L. Qi, M.-Y. Yang, Source Retrieval Model Focused on Aggregation for Plagiarism Detection, Information Sciences 503 (2019) 336–350. doi:10.1016/j.ins.2019.07.015.

[14] S. E. Robertson, H. Zaragoza, The probabilistic relevance framework: BM25 and beyond, Found. Trends Inf. Retr. 3 (2009) 333–389. URL: https://doi.org/10.1561/1500000019. doi:10.1561/1500000019.

[15] V. Karpukhin, B. Oguz, S. Min, P. Lewis, L. Wu, S. Edunov, D. Chen, W.-t. Yih, Dense passage retrieval for open-domain question answering, in: B. Webber, T. Cohn, Y. He, Y. Liu (Eds.), Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), Association for Computational Linguistics, Online, 2020, pp. 6769–6781. URL: https://aclanthology.org/2020.emnlp-main.550/. doi:10.18653/v1/2020.emnlp-main.550.

[16] N. Thakur, N. Reimers, A. Rücklé, A. Srivastava, I. Gurevych, BEIR: A heterogeneous benchmark for zero-shot evaluation of information retrieval models, in: Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks, 2021. URL: https://arxiv.org/abs/2104.08663.

[17] M. Luo, S. Jain, A. Gupta, A. Einolghozati, B. Oguz, D. Chatterjee, X. Chen, C. Baral, P. Heidari, A study on the efficiency and generalization of light hybrid retrievers, in: Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), Association for Computational Linguistics, 2023, pp. 1617–1626. URL: https://aclanthology.org/2023.acl-short.139/. doi:10.18653/v1/2023.acl-short.139.

[18] G. V. Cormack, C. L. A. Clarke, S. Buettcher, Reciprocal rank fusion outperforms condorcet and individual rank learning methods, in: Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '09, Association for Computing Machinery, New York, NY, USA, 2009, p. 758–759. URL: https://doi.org/10.1145/1571941.1572114. doi:10.1145/1571941.1572114.

[19] J. Bevendorff, D. Dementieva, M. Fröbe, B. Gipp, A. Greiner-Petter, J. Karlgren, M. Mayerl, P. Nakov, A. Panchenko, M. Potthast, A. Shelmanov, E. Stamatatos, B. Stein, Y. Wang, M. Wiegmann, E. Zangerle, Overview of PAN 2025: Voight-Kampff Generative AI Detection, Multilingual Text Detoxification, Multi-Author Writing Style Analysis, and Generative Plagiarism Detection, in: J. C. de Albornoz, J. Gonzalo, L. Plaza, A. G. S. Herrera, J. Mothe, F. Piroi, P. Rosso, D. Spina, G. Faggioli, N. Ferro (Eds.), Experimental IR Meets Multilinguality, Multimodality, and Interaction. Proceedings of the Sixteenth International Conference of the CLEF Association (CLEF 2025), volume 16089 of *Lecture Notes in Computer Science*, Springer, Berlin Heidelberg New York, 2025, pp. 388–411. doi:10.1007/978-3-032-04354-2_21.

[20] Google DeepMind, Gemma 4 model card, https://ai.google.dev/gemma/docs/core/model_card_4, 2026. Model release for Gemma 4 31B IT.

[21] X. H. Lù, Bm25s: Orders of magnitude faster lexical search via eager sparse scoring, 2024. URL: https://arxiv.org/abs/2407.03618. arXiv:2407.03618.

[22] Z. Li, X. Zhang, Y. Zhang, D. Long, P. Xie, M. Zhang, Towards general text embeddings with multi-stage contrastive learning, 2023. URL: https://arxiv.org/abs/2308.03281. arXiv:2308.03281.

[23] X. Zhang, Y. Zhang, D. Long, W. Xie, Z. Dai, J. Tang, H. Lin, B. Yang, P. Xie, F. Huang, M. Zhang, W. Li, M. Zhang, mgte: Generalized long-context text representation and reranking models for multilingual text retrieval, 2024. URL: https://arxiv.org/abs/2407.19669. arXiv:2407.19669.

[24] N. Reimers, I. Gurevych, Sentence-bert: Sentence embeddings using siamese bert-networks, in: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, 2019. URL: https://arxiv.org/abs/1908.10084.

[25] S. Althammer, S. Hofstätter, M. Sertkan, S. Verberne, A. Hanbury, PARM: A paragraph aggregation retrieval model for dense document-to-document retrieval, in: Advances in Infor-

mation Retrieval, Springer, 2022, pp. 19–34. URL: https://doi.org/10.1007/978-3-030-99736-6_2. doi:10.1007/978-3-030-99736-6_2.

[26] M. A. Hearst, Text tiling: Segmenting text into multi-paragraph subtopic passages, Computational Linguistics 23 (1997) 33–64. URL: https://aclanthology.org/J97-1003/.

[27] Z. Dai, J. Callan, Deeper text understanding for IR with contextual neural language modeling, in: Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, 2019, pp. 65–74. URL: https://doi.org/10.1145/3331184.3331269.

[28] O. Khattab, M. Zaharia, Colbert: Efficient and effective passage search via contextualized late interaction over bert, in: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '20, Association for Computing Machinery, New York, NY, USA, 2020, p. 39–48. URL: https://doi.org/10.1145/3397271.3401075. doi:10.1145/3397271.3401075.

[29] E. Bassani, L. Romelli, ranx.fuse: A python library for metasearch, in: CIKM, ACM, 2022, pp. 4808–4812. doi:10.1145/3511808.3557207.

[30] E. A. Fox, J. A. Shaw, Combination of multiple searches, in: D. K. Harman (Ed.), The Second Text REtrieval Conference (TREC-2), volume 500-215 of *NIST Special Publication*, National Institute of Standards and Technology (NIST), 1993, pp. 243–252. URL: https://trec.nist.gov/pubs/trec2/t2_proceedings.html.

[31] E. Bassani, ranx: A blazing-fast python library for ranking evaluation and comparison, in: European Conference on Information Retrieval, Springer, 2022, pp. 259–264.

[32] T. Akiba, S. Sano, T. Yanase, T. Ohta, M. Koyama, Optuna: A next-generation hyperparameter optimization framework, in: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2019.