FELLA at CheckThat! 2026: From BM25 to Stella, a Multilingual Pipeline for Scientific Source Retrieval

Notebook for the CheckThat! Lab at CLEF 2026

Filippo Corradi[1], Enrico De Lorenzo Buratta[1], Luca Fantin[1], Alessandro Sartor[1], Lorenzo Tamburi[1] and Nicola Ferro[1]

[1]*University of Padua, Italy*

Abstract

Social media posts often discuss scientific findings without explicitly linking the publication they refer to, which makes verifying such claims difficult. Task 1 of the CLEF 2026 CheckThat! Lab addresses this problem by asking systems to retrieve, for a given post written in English, German, or French, the implicitly cited paper from a corpus of 10,000 English scientific publications. This paper presents the system developed by Team FELLA for the task. We build a multi-stage retrieve-and-rerank pipeline: the lexical foundation is a BM25 ranker (Apache Lucene) with a custom analyzer, and non-English queries are translated into English via LibreTranslate before retrieval. On top of this, we experiment with dense retrieval using SPECTER2 and combine the two via Reciprocal Rank Fusion, then explore progressively stronger bi-encoder models (BGE-large, GTE-large, Stella-1.5B-v5). A cross-encoder is used as a second-stage re-ranker over the top-50 candidates. Our best configuration, Stella-1.5B-v5 with cross-encoder re-ranking, reaches MRR@5 of 0.61, 0.53, and 0.63 on English, German, and French respectively on the development set. These results show that a retrieval-tuned bi-encoder combined with query translation can bridge most of the multilingual gap on consumer hardware, offering a practical building block for the automated verification of scientific claims on social media.

Keywords

Information Retrieval, CheckThat! 2026, Cross-Language Information Retrieval, Bi-Encoders, Cross-Encoders, Re-Ranking

1. Introduction

The goal of the work is to implement an *Information Retrieval (IR)* system for Task 1 [1] of the CLEF 2026 CheckThat! Lab [2] that matches tweets in three different languages (English, German, and French) with a collection of scientific documents available only in English. Such a system is particularly valuable for analyzing the communication gap between public discourse and scientific articles, allowing for an assessment of how closely multilingual conversations on social media are in line with research. The implementation faces two fundamental challenges. First, the *linguistic gap*: the queries are multilingual, while the target index is monolingual, requiring an effective *Cross Language Information Retrieval (CLIR)* strategy. Second, the *data noise*: tweets are characterized by their brevity, informal language, and the presence of social metadata (such as hashtags and mentions) that can introduce significant semantic noise. The objective of this project is thus to develop a pipeline that maximizes retrieval precision while maintaining high recall across all three languages.

The paper is organized as follows: Section 3 describes our approach; Section 4 explains our experimental setup; Section 5 discusses our main findings; finally, Section 6 draws some conclusions and outlooks for future work.

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

filippo.corradi@studenti.unipd.it (F. Corradi); enrico.delorenzoburatta@studenti.unipd.it (E. De Lorenzo Buratta); luca.fantin.1@studenti.unipd.it (L. Fantin); alessandro.sartor.3@studenti.unipd.it (A. Sartor); lorenzo.tamburi@studenti.unipd.it (L. Tamburi); nicola.ferro@unipd.it (N. Ferro)

https://www.dei.unipd.it/~ferro/ (N. Ferro)

0000-0001-9219-6239 (N. Ferro)

© 2026 Copyright for this paper by its authors.
Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

2. Related Work

Lexical ranking models rooted in the probabilistic ranking principle [3] remain the standard starting point for retrieval systems, with BM25 [4, 5] still being the most widely adopted weighting scheme, typically through the Apache Lucene library [6]. A long line of work has tried to mitigate the vocabulary mismatch problem that affects lexical matching, from stemming algorithms such as Porter [7], KStem [8], and Snowball [9], to query expansion through pseudo-relevance feedback; the latter is known to be fragile on short, noisy queries, where expansion terms can easily cause query drift.

In CLIR, where queries and documents are written in different languages, the two classical strategies are document translation and query translation [10]. Query translation is usually preferred when the collection is large and the queries are short, since only the query side needs to be processed; the growing availability of open machine translation services such as LibreTranslate [11] makes this approach cheap and easy to deploy.

More recently, dense retrieval has emerged as a strong alternative to lexical matching: a bi-encoder maps queries and documents into a shared embedding space, and retrieval reduces to a nearest-neighbour search, made efficient by libraries such as FAISS [12]. In the scientific domain, SPECTER2 [13] learns paper representations from citation graphs, placing co-cited papers close in embedding space. General-purpose sentence encoders trained on large-scale retrieval supervision have also progressed quickly, and public benchmarks such as MTEB now track a rapidly evolving family of models of increasing size, from compact encoders like MiniLM to billion-parameter, retrieval-tuned models.

Since bi-encoders score query and document independently, they cannot model fine-grained interactions between the two texts. Cross-encoder re-rankers [14] address this by jointly encoding the query-document pair with full cross-attention, at a much higher computational cost; the standard compromise is the retrieve-and-rerank pattern, in which a fast first stage produces a small candidate pool and the cross-encoder only re-orders it. Complementary to re-ranking, rank fusion techniques such as Reciprocal Rank Fusion [15] combine the outputs of heterogeneous systems, exploiting the observation that ranking errors of different retrieval paradigms need not coincide [16]. Finally, large language models have been used to bridge the query-document gap at query time: HyDE [17] generates a hypothetical relevant document from the query and retrieves with its embedding, which has proven effective on lexically poor queries.

Our system combines these building blocks — BM25 with query translation, dense bi-encoder retrieval, cross-encoder re-ranking, rank fusion, and HyDE-style expansion — and evaluates how each of them behaves on the specific tweet-to-paper matching problem posed by the CheckThat! task.

3. Methodology

This section describes our approach, organized around the two challenges introduced in Section 1. Section 3.1 presents the lexical pipeline and the text cleaning steps that address the noise of social media queries; Section 3.2 tackles the linguistic gap through query translation; Section 3.3 compares sparse, dense, and hybrid strategies for the first-stage ranking; Section 3.4 describes the second-stage neural re-ranking; finally, Section 3.5 covers the additional strategies we explored to push the system beyond the bi-encoder ceiling.

Figure 1 summarizes the overall architecture of our retrieval pipeline. Following the figure from left to right: the scientific collection and the tweets are parsed and cleaned by the `CheckThatDocumentParser` and `CheckThatTopicParser` components[1] respectively (Section 3.1); non-English topics are translated into English by the `QueryTranslator` (Section 3.2); the first stage retrieves a pool of candidate documents, either from the Lucene inverted index or from a dense bi-encoder index (Section 3.3); a cross-encoder then re-ranks the top candidates to produce the final run (Section 3.4).

[1]The implementation of all the components mentioned in this section is available in our repository: https://bitbucket.org/upd-dei-stud-prj/seupd2526-fella/src/master/.

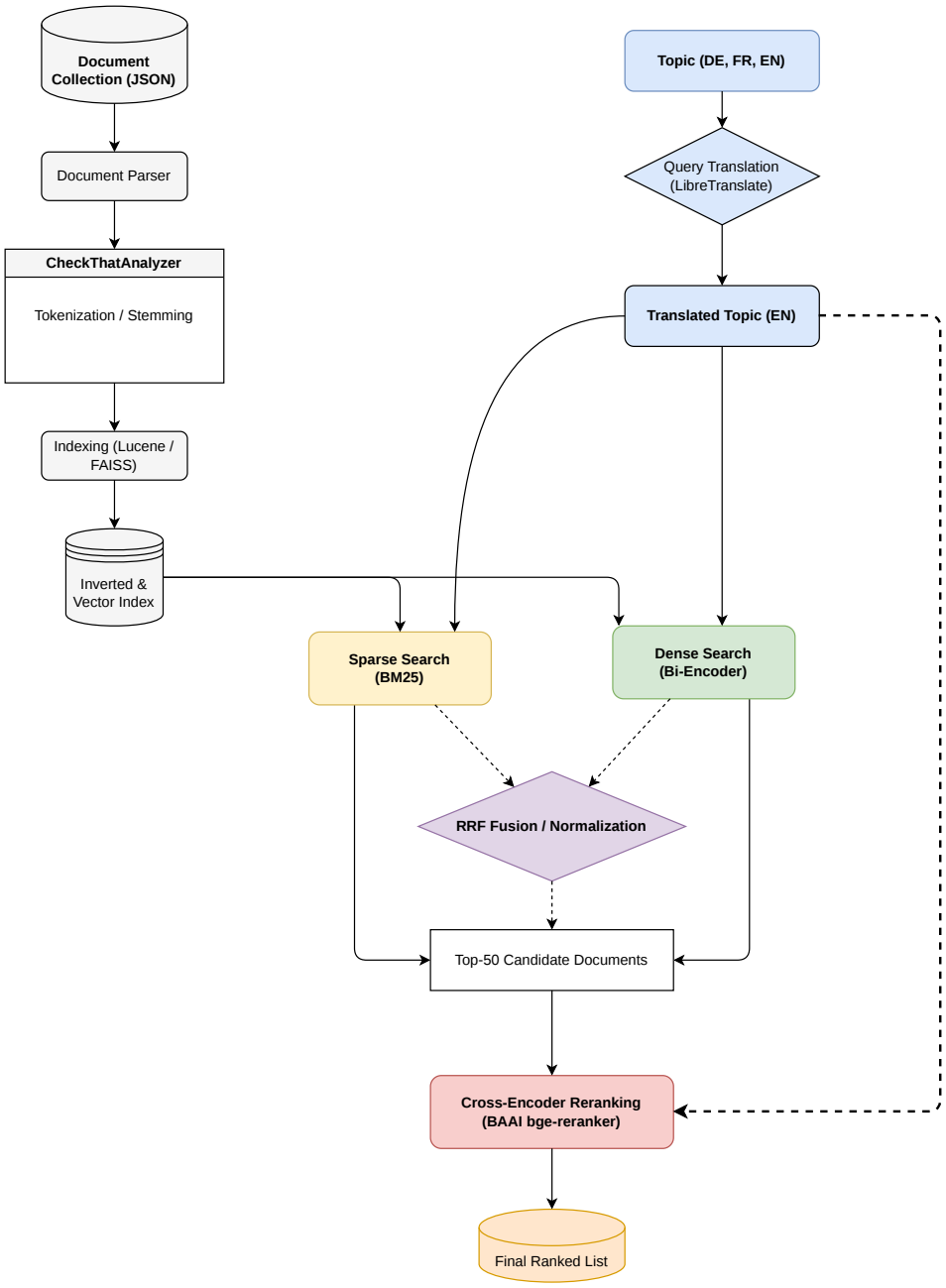


Figure 1: Overview of the proposed multi-stage retrieval pipeline.

The fundamental backbone of our retrieval system is a lexical pipeline based on the BM25 ranking function, implemented using the Apache Lucene library. The architecture follows a classic IR pipeline: parsing, analysis, indexing, and searching.

3.1. Lexical Baseline and Text Processing

Parsing and Data Cleaning The system uses two distinct parsing strategies to handle the heterogeneity of the input data. The `CheckThatDocumentParser` is employed to process the scientific collection, extracting key fields such as the title and abstract. In the same way, the `CheckThatTopicParser` is dedicated to the tweets. Given the noisy nature of social media text, we implemented a custom cleaning function that removes user mentions (e.g., `@username`) and hashtags. By deleting these tokens, we reduce the risk of the retrieval model assigning excessive weight to frequent but non-informative social tags, in order to improve the signal-to-noise ratio of the query.

Text Analysis and Indexing To handle the linguistic variability of the input, we developed the `CheckThatAnalyzer`. This component performs a sequence of operations: tokenization, stop-word removal, and stemming. We adopted several stemming algorithms, namely Snowball [9], KStem [8], and Porter [7], for English, German, and French, which allows the system to map different variants of a word to a single root form. This is very important for reducing the vocabulary gap between the informal style of tweets and the formal terminology of scientific papers.

Searching The processed documents are then stored in an inverted index via the `DirectoryIndexer`, which enables the `DirectorySearcher` to perform efficient keyword-based retrieval. This lexical baseline provides the necessary benchmark to evaluate the impact of more advanced strategies, such as the translation-based approach described in Section 3.2 and the neural re-ranking analyzed in Section 3.4.

3.2. Overlapping of linguistic difference

The difference between the language of the queries (EN, DE, FR) and the language of the document collection (EN) constitutes a common problem in CLIR. We evaluated several strategies to overcome the linguistic barrier, including for example the use of native cross-lingual analyzers. However, native lexical matching (where DE/FR terms are matched against an EN index) suffers from a strict lack of overlap in the vocabulary, leading to poor recall. To try to resolve this, we implemented a *Query Translation* strategy.

Technical Implementation via Containerization The translation process is managed by the `QueryTranslator` component, which uses a translation model deployed via a Docker container (specifically the `libretranslate/libretranslate` image). The service exposes a REST API, enabling the `QueryTranslator` to send non-English queries via HTTP requests and receive their English equivalents. This modular approach simplifies the deployment and allows the translation model to be updated or scaled independently without affecting the main IR pipeline.

3.3. First Stage Ranking: Sparse vs Dense vs Hybrid

Sparse Retrieval via Pseudo-Relevance Feedback To improve the performance of the lexical baseline, we applied Pseudo-Relevance Feedback (PRF) using Rocchio expansion (Exp K). The goal was to enrich the initial queries by extracting relevant terms from the top-k retrieved documents ($k = 7$). We selected $m = 5$ expansion terms and assigned them a marginal weight of $\alpha = 0.15$ to mitigate the risk of introducing noisy words that could negatively influence the query. To further constrain this issue, two strict filters were applied: a minimum IDF threshold (IDF > 4) for term selection, and a relevance score threshold (> 6) for the initial candidate documents.

Despite these constraints, the experiment yielded disappointing results, degrading the baseline MRR@5 scores, with a particularly significant drop for the French and German languages. This degradation is largely attributable to the severe *vocabulary mismatch* between the informal, concise language of the source tweets and the dense, academic text of the scientific publications. Ultimately, relying on a purely statistical surface-form approach for expansion inadvertently associated the queries with entirely out-of-context terms, triggering a severe query drift.

Dense Retrieval with SPECTER2 General-purpose sentence encoders are not trained to capture citation proximity: two papers may be semantically related yet far apart in embedding space if they were never co-cited. To address this, we used `allenai/specter2_base` with its `specter2_proximity` task adapter [13], a model specifically pre-trained on citation graphs to place co-cited papers close in embedding space.

Each document is encoded as its title and abstract joined by the model's separator token (`[title] [SEP] [abstract]`). The CLS-token representation from the final Transformer layer is extracted and L2-normalised, so that inner-product search is equivalent to cosine similarity. Query encoding follows the same pipeline on the cleaned tweet text (after removing @mentions, #hashtags, and Boolean operators); for German and French topics, the pre-translated English versions are used, consistent with Section 3.2. Exp L is therefore not a native cross-lingual system: it relies on the same machine-translated input as the lexical baseline.

Document embeddings are computed once, cached to disk, and reused across all languages and splits. Retrieval uses a FAISS [12] `IndexFlatIP` for exact inner-product search, returning the top-50 candidates per query.

Hybrid Retrieval via Reciprocal Rank Fusion BM25 and SPECTER2 operate on different signals — surface-form overlap versus semantic proximity — so their ranking errors need not coincide. To test whether combining them recovers documents that neither retrieves alone, we fused the Exp H (BM25) and Exp L (SPECTER2) ranked lists per query using Reciprocal Rank Fusion (RRF) [15].

The fused score for a document d is:

$$\text{score}(d) = \frac{1}{k + r_{\text{BM25}}(d)} + \frac{1}{k + r_{\text{dense}}(d)} \tag{1}$$

where $r_s(d)$ is the rank of d in system s. Documents absent from one list receive a phantom rank equal to the length of that list plus one, ensuring that single-system candidates still contribute to the combined score. The top-50 documents by fused score are retained.

The smoothing constant k controls how sharply the score drops off with rank: a smaller k makes the function steeper, giving more weight to documents that both systems rank highly. We swept $k \in \{10, 20, 30, 60\}$ on the development set; $k = 10$ achieved the best MRR@5 across all three languages and was adopted for the final run (Exp M), suggesting that the task rewards strong agreement between the two retrievers over a more uniform combination. Exp M also serves as the first-stage candidate pool for the neural re-ranking step described in Section 3.4.

Bi-Encoder Retrieval with General-Purpose Sentence Encoders After observing that SPECTER2 was trained for paper-to-paper similarity rather than the tweet-to-paper task we needed, we decided to test more general dense retrievers. The idea was to see if a different family of pretrained encoders could close the language gap between the informal tweets and the scientific abstracts more effectively. The setup follows the same logic as Exp L: documents are encoded as the concatenation of title and abstract, embeddings are L2-normalised and stored in a FAISS index, and queries are passed through the same translation step used for the BM25 baseline.

We started from a small general-purpose model, `all-MiniLM-L6-v2` (Exp Q), as a first sanity check; despite not being specifically tuned for retrieval, it already produced better results than SPECTER2 on English and French. We then moved to progressively larger models in search of a more significant improvement: first `BAAI/bge-large-en-v1.5` (Exp R) and then `Alibaba-NLP/gte-large-en-v1.5` (Exp S). Both gave only marginal gains over MiniLM, suggesting that simply scaling up a general retriever was not enough on its own. The last model we tried is `dunzhang/stella_en_1.5B_v5` (Exp T), a much larger encoder specifically finetuned for retrieval, which sits among the strongest open models on the MTEB benchmark for its size class.

Stella turned out to be the strongest first-stage retriever in our pipeline, with a clear margin over BGE-large and GTE-large on every language and a consistent gap over SPECTER2. From this point

onwards, all the second-stage and fusion experiments described in Section 3.4 use Stella's top-50 as their candidate pool.

3.4. Second Stage Neural Re-Ranking

Cross-Encoder Architecture To refine the initial ranking retrieved by the first stage, we introduced a second-stage neural re-ranking utilizing a Cross-Encoder architecture in the `neural_reranker.py`. Specifically, we adopted the `BAAI/bge-reranker-base`[14] model (along with experiments on its larger variants, like `BAAI/bge-reranker-v2-m3`). Unlike Bi-Encoders, which map queries and documents into separated vector spaces, a Cross-Encoder takes as input the concatenated pair of the query and the candidate document text. This permits the model to compute deep cross-attention over all the tokens of the pair, capturing complex semantic dependencies that simple dot products do not detect. To adapt to the model's memory constraints during GPU inference, we truncated the document body to the first 2000 characters, which in any case contained the most informative content (title and abstract).

Score Normalization and Fusion The first-stage candidate generation (e.g., BM25 or bi-encoder model) and the neural Cross-Encoder produce scores that belong to completely different mathematical scales. To correctly aggregate them, we implemented a Min-Max normalization to project both score distributions into a $[0, 1]$ range for each individual query. After the normalization, the final ranking was computed using a linear interpolation formula:

$$Score_{final} = \alpha \cdot Score_{first\_stage} + (1 - \alpha) \cdot Score_{neural} \tag{2}$$

By sweeping the α parameter (testing values from 0.1 to 0.9), it was possible to find the optimal trade-off. This step is important to retain the exact keyword matching capabilities of the lexical baseline while injecting the deep understanding of the neural model.

The Importance of Candidate Depth (Top-K) A fundamental aspect of our reranking phase was tuning the depth of the candidate pool passed to the second stage. We conducted a series of specific experiments to test different cut-offs: Top-10, Top-30, Top-50, and Top-100. More details about the analysis of results can be found in Subsection 5.4.

3.5. Improving the Bi-Encoder performance

Since Stella's first-stage results stalled around 0.59 MRR@5 on English, we ran a series of additional experiments aimed at pushing the system further. None of these attempts produced a real improvement, but the negative results are still informative and show some of the limits of our approach.

Stronger Cross-Encoder on Stella's Top-50 We replaced the default `bge-reranker-base` reranker with the larger `BAAI/bge-reranker-v2-m3`, expecting that a more powerful cross-encoder would help reorder the candidates more accurately. Using the same alpha-sweep formulation of Section 3.4, the best per-language alphas turned out to be $\alpha = 0.7$ on English and French and $\alpha = 0.5$ on German. The improvement over Stella alone, however, was minimal (about +0.02 MRR points on average). When the candidate set produced by the first stage is already mostly correct, the cross-encoder has very little room left to reorder, and a stronger reranker does not really help.

Multi-Source RRF Beyond BM25 + Dense Building on Exp M, we extended the RRF fusion to combine multiple ranked lists at the same time, in the hope that more diverse signals would compensate for the limits of each individual retriever. We tried both a four-way fusion combining BM25, BM25 with neural boosts, Stella alone and Stella + cross-encoder, and a simpler two-way fusion of BM25 and Stella + cross-encoder. Neither performed better than the strongest single source. The four-way version

suffered from too much redundancy, because multiple variants of the same family of retrievers ended up carrying overlapping information; the two-way version, on the other hand, dragged the stronger ranking down towards the weaker one. RRF tends to work well when its inputs are roughly comparable in quality and complementary in their errors, and in our case neither condition was satisfied.

Hypothetical Document Embeddings (HyDE) The last attempt was a HyDE-style query expansion [17]. For each tweet, a small instruction-tuned LLM generates a short hypothetical scientific paragraph that should resemble the kind of passage that would be relevant to the claim, and the generated text is then encoded with Stella to perform the actual retrieval. We tested two variants: one in which the original tweet is concatenated with the generated paragraph before encoding, and one in which only the LLM output is encoded. Both ended up below Stella alone on every language. Looking at the generated paragraphs, they were grammatically fine but generic, and in several cases the LLM hallucinated specific citations or numerical results that do not appear anywhere in the corpus. Our reading is that the tweets in this dataset already contain enough scientific terminology to align well with Stella's embedding space; the LLM-generated text adds some context but also dilutes the original keyword signal, and any hallucination actively pulls the embedding away from the right document.

Taken together, these three negative results suggest that on this task the main bottleneck is not the retriever or the reranker, but rather the inherent ambiguity of mapping short, noisy social-media queries to highly specific scientific abstracts.

4. Experimental Setup

4.1. Dataset

For our experiments, we utilized the official dataset provided by the CLEF CheckThat! Lab 2026 for Task 1: *Source Retrieval for Scientific Web Claims* [1].

The main corpus is provided in a single JSON file (`collection_data.json`) containing exactly 10,000 scientific publications. Each document record provides a unique identifier (`pubkey`), the title, the abstract, the publication venue, and the list of authors. The queries represent the social media posts and are distributed across three different languages: English (EN), German (DE), and French (FR). For each language, the organizers provided a `train` and a `dev` split. Every query record contains an internal index, the raw text of the post, and the target `pubkey`, which serves as the ground truth for evaluating our IR system.

4.2. Evaluation metrics

The official evaluation metric for the task is the Mean Reciprocal Rank at cutoff 5 (MRR@5).

The reciprocal rank for a single query is defined as:

$$RR@5 = \begin{cases} \frac{1}{rank_i} & \text{if } rank_i \leq 5 \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

where $rank_i$ is the position of the first relevant document for query i. If the relevant document does not appear in the top-5 results, the contribution is 0. The Mean Reciprocal Rank at cutoff 5 is then computed as:

$$MRR@5 = \frac{1}{|Q|} \sum_{i=1}^{|Q|} RR@5_i \tag{4}$$

This metric measures, for each query, the rank position of the first relevant document and computes its reciprocal value. Then, the final score is obtained by averaging these reciprocal ranks over all the queries. In this setting, since we are dealing with a known-item search task (each query has exactly one

correct document), MRR@5 is particularly suitable because it directly rewards systems that rank the correct paper as high as possible in the top-5 results.

In addition to the official metric, we found it useful to also consider two other evaluation measures.

The first one is Recall@5, which checks whether the correct document is present in the top-5 retrieved ones. This metric is especially important for evaluating the first-stage ranking, because if the relevant document is not retrieved at this stage, the neural re-ranker cannot recover it later.

Finally, we also report Precision@1, which measures how often the correct document is ranked in the first position. This metric gives a strict evaluation of the system, highlighting its ability to place the relevant document exactly at the top of the ranking.

4.3. Hardware

The lexical part of the pipeline (BM25 indexing, query parsing, RRF fusion) only needs CPU and runs on any of our laptops. All the neural components, the bi-encoders, the cross-encoders, and the LLM used for query expansion, were instead run locally on an NVIDIA GeForce RTX 4060 Laptop GPU with 8 GB of VRAM, on a machine equipped with an AMD Ryzen 7 8845HS and 32 GB of system RAM. Each Python script in `python_dense/` and `python_reranker/` picks the device automatically through `torch.cuda.is_available()`, using CUDA when available and falling back to CPU only as a safety net.

The 8 GB VRAM limit drove several practical choices. The smaller models (`all-MiniLM-L6-v2`, `bge-large-en-v1.5`, `bge-reranker-base`) fit comfortably in fp32. The larger ones (`stella_en_1.5B_v5`, `Qwen2.5-3B-Instruct`, `bge-reranker-v2-m3`) had to be loaded in fp16 through the `model_kwargs` argument, otherwise they would not fit. Document and corpus embeddings are computed once and stored to disk under `cache/`, so that re-running a variant of the same experiment only requires encoding the queries again a few seconds instead of several minutes.

4.4. URL to git repository and its organisation

The source code of our IR system is available online at the following address: https://bitbucket.org/upd-dei-stud-prj/seupd2526-fella/src/master/.

5. Results and Discussion

5.1. Overall Performance

The best results of the project were obtained by replacing the BM25 first stage with a strong dense retriever based on `stella_en_1.5B_v5` (Exp T) and reranking its top-50 candidates with the larger `bge-reranker-v2-m3` cross-encoder (Exp U). This combination reached an MRR@5 of about 0.61 on English, 0.53 on German, and 0.63 on French, the highest scores observed on all three languages. End-to-end, the full pipeline takes around 10 minutes to run on the development set, of which roughly a minute and a half is spent by Stella encoding the queries and retrieving the top-50 candidates, while the remainder is dominated by the cross-encoder reranking step.

Table 1 reports the milestone runs of the project, i.e., the experiments in which a new model or pipeline component was introduced. Intermediate parameter variations (field boosts, analyzer variants, candidate-pool depths, alpha values) are omitted for brevity; the complete experiment log is available in the project repository.

5.2. Impact of translation

The effectiveness of the translation strategy is clearly evident when comparing the native lexical baseline with the translation-based pipeline. In the initial experiments (Exp A, B, and C), using native analyzers for German and French resulted in poor performance, with MRR@5 values fixing around 0.17 for German and 0.16 for French. These results confirm that basically applying language-specific

Table 1
Milestone experimental runs (MRR@5) on the development set for English (EN), German (DE), and French (FR). Each row introduces a new model or component; intermediate parameter variations are omitted. Exp A onwards includes the social-metadata cleaner (removal of @mentions and hashtags).

RunID	New model / component	EN	DE	FR	avg
seupd2526-fella-bm25baseline	BM25, standard Lucene analyzers	0.4685	0.0863	0.0869	0.2139
seupd2526-fella-french-german-analyzers	language-specific DE/FR analyzers	0.4685	0.1733	0.1638	0.2685
seupd2526-fella-expA-body-native	social-metadata cleaner	0.5145	0.1682	0.1786	0.2871
seupd2526-fella-expH-translation-multifield	DE/FR query translation	0.5145	0.3556	0.4915	0.4539
seupd2526-fella-expI-translation-neural-equalboosts	cross-encoder re-ranking (bge-reranker-base)	0.5541	0.3853	0.5497	0.4964
seupd2526-fella-expK-prf-body-only	pseudo-relevance feedback (Rocchio)	0.5145	0.1682	0.1786	0.2871
seupd2526-fella-expL-specter2	dense retrieval (SPECTER2)	0.3604	0.2486	0.3425	0.3172
seupd2526-fella-expM-hybrid-rrf-k10	hybrid RRF fusion (BM25 + SPECTER2)	0.5181	0.3811	0.5108	0.4700
seupd2526-fella-expQ-biencoder-minilm	bi-encoder (all-MiniLM-L6-v2)	0.4262	0.2968	0.4306	0.3845
seupd2526-fella-expR-biencoder-bge-large	bi-encoder (bge-large-en-v1.5)	0.5083	0.4091	0.5366	0.4847
seupd2526-fella-expS-biencoder-gte-large	bi-encoder (gte-large-en-v1.5)	0.5108	0.4193	0.5240	0.4847
seupd2526-fella-expT-biencoder-stella-1.5b	bi-encoder (stella_en_1.5B_v5)	0.5940	0.5237	0.6129	0.5769
seupd2526-fella-expT-stella-rerank-base-top50	Stella + cross-encoder (bge-reranker-base)	0.6000	0.5236	0.6206	0.5814
seupd2526-fella-expU-stella-rerank-v2m3-best	Stella + cross-encoder (bge-reranker-v2-m3)	0.6108	0.5311	0.6314	0.5911
seupd2526-fella-expV-rrf-4way-k10	multi-source RRF fusion	0.5998	0.4835	0.6157	0.5663
seupd2526-fella-expW-hyde-stella-combined	HyDE query expansion (Qwen2.5-3B)	0.5710	0.4976	0.6047	0.5578

stemming and stop-word removal is insufficient to overcome the linguistic gap. A significant turning point was observed in Experiment H, where the translation of DE and FR queries into English was introduced. As shown in our results, the MRR@5 for German increased from 0.1682 to 0.3556, and for French from 0.1786 to 0.4915. This relevant improvement proves that translating the query is far more effective than attempting to match non-English terms against an English index. Due to the significant improvement, the translation pipeline was adopted as the standard preprocessing step for all subsequent neural re-ranking experiments.

5.2.1. Cross-Lingual Fails

In our initial attempts to manage the multilingual nature of the dataset (Experiments C and E), we explored a native cross-lingual approach using specific analyzers within the BM25 framework. The objective was to align the German and French queries with the predominantly English documents in the corpus.

However, this methodology proved entirely ineffective. The application of these analyzers, particularly in combination with a multi-field search configuration, triggered severe *query drift*. Lacking the deep semantic understanding required to accurately map concepts across different languages, the system ultimately amplified noisy, out-of-context terms.

This resulted in a drastic collapse in retrieval performance, as evidenced by Experiment E: the MRR@5 score plummeted to 0.039 for English queries, while remaining at critical levels for German (0.097) and French (0.085). These results unequivocally demonstrated that relying solely on cross-lingual lexical analyzers is insufficient to overcome the language barrier within traditional statistical models.

5.3. First-Stage Retrieval Analysis

Table 2 compares the first-stage retrieval strategies on the development set.

BM25 vs. Dense Retrieval Comparing Exp A (BM25, native queries) and Exp L (SPECTER2) reveals a language-dependent trade-off. On English queries, BM25 outperforms SPECTER2 by a large margin (MRR@5 0.5145 vs. 0.3604): English tweets share enough vocabulary with paper abstracts for surface-form matching to dominate. On German and French, the situation reverses: SPECTER2 achieves MRR@5 of 0.2486 and 0.3425, compared to 0.1682 and 0.1786 for native BM25. Without translation,

Table 2
First-stage retrieval: MRR@5 and Recall@5 on the development set. Exp A = BM25 (native, no translation); Exp H = BM25 + translation; Exp L = SPECTER2 dense; Exp M = Hybrid RRF ($k = 10$). Best per column in **bold**.

System	MRR@5				Recall@5			
	EN	DE	FR	avg	EN	DE	FR	avg
Exp A (BM25, native)	0.5145	0.1682	0.1786	0.2871	0.5869	0.2124	0.2251	0.3415
Exp H (BM25 + trans.)	0.5145	0.3556	0.4915	0.4539	0.5869	0.4275	0.5684	0.5276
Exp L (SPECTER2)	0.3604	0.2486	0.3425	0.3172	0.4471	0.3238	0.4359	0.4023
Exp M (Hybrid RRF)	**0.5181**	**0.3811**	**0.5108**	**0.4700**	**0.6225**	**0.4922**	**0.6054**	**0.5734**

BM25 cannot bridge the language gap; SPECTER2, trained on a multilingual citation graph, provides a partial cross-lingual signal that lexical matching entirely lacks.

Once translation is applied (Exp H), however, BM25 reclaims the lead on all three languages (MRR@5 0.3556 and 0.4915 on DE/FR vs. 0.2486 and 0.3425 for Exp L), suggesting that the primary bottleneck for SPECTER2 is not the language barrier but the domain mismatch between informal tweets and formal scientific abstracts. SPECTER2 was pre-trained on paper-to-paper citation proximity; its embedding space is not aligned with the tweet-to-paper retrieval task.

Hybrid RRF Since BM25 and SPECTER2 operate on complementary signals, their error sets are largely disjoint: a paper missed by the lexical ranker due to vocabulary mismatch may still be retrieved by the dense model, and vice versa. Exp M exploits this by fusing both ranked lists via RRF (see Eq. 1). The improvement over Exp H is most pronounced in Recall@5, which rises from 0.5276 to 0.5734 on average (+0.046), at a modest MRR@5 gain of +0.016. This broader candidate pool is precisely what makes the subsequent neural re-ranking effective, as discussed in Section 5.4.

Bi-Encoder Dense Retrieval The bi-encoder experiments described in Section 3.3 test if a more general-purpose dense retriever can do better than SPECTER2 on this task. The smallest model, `all-MiniLM-L6-v2` (Exp Q), already beats Exp L on English (0.4262 vs. 0.3604) and French (0.4306 vs. 0.3425), confirming that SPECTER2's specialization on paper-to-paper similarity is not well aligned with the tweet-to-paper task. Moving to the larger `bge-large-en-v1.5` (Exp R) and `gte-large-en-v1.5` (Exp S) raises the scores to roughly 0.51 on English, 0.41 on German and 0.53 on French, which is already comparable to BM25 + translation (Exp H) on English and clearly above it on the other two languages.

The real jump, however, comes from `stella_en_1.5B_v5` (Exp T), which reaches 0.5940 on English, 0.5237 on German and 0.6129 on French. Stella outperforms not only SPECTER2 and the smaller bi-encoders, but also the BM25 + translation pipeline on all three languages, becoming the strongest first-stage retriever of our system. This is also what motivated us to adopt Stella's top-50 as the candidate pool for the second-stage and fusion experiments discussed in Section 3.5.

5.4. Second-Stage Re-Ranking Analysis

Our experiments demonstrated that setting the cut-off too strictly (e.g., Top-10 in experiment O and Top-30 in experiment P) chokes the maximum achievable recall, as the reranker is deprived of potentially relevant documents that were ranked slightly lower by the first stage.

On the other hand, processing a very deep pool (Top-100, in experiment N) not only slows down the time of computation significantly (doubles with respect to Top-50), but also introduces semantic noise, causing the Cross-Encoder to occasionally promote irrelevant but semantically similar documents.

The Top-50 (experiment I) cut-off emerged as the "ideal" cut-off, balancing the initial recall of the first stage with the final precision of the neural re-ranker. This ideal cut-off was evaluated with a First-Stage

Ranking with BM25, after the translation of French and German topics (see Subsection 3.2) and with equal boosts assigned to the fields.

In experiment J, using Top-50, we tried to boost the Body field, but the results were not successful. This happened probably because the Body field already dominates the representation of the document, and increasing its weight reduces the contribution of more informative fields such as title and abstract. As a consequence, the ranking becomes more noisy and less focused on the core semantic match between query and document.

To further substantiate the top-50 choice for the dense first stage, Figure 2 shows Stella's Recall@K per language for $K \in \{1, 5, 10, 20, 50, 100\}$ on the test set. The recall never truly saturates: it keeps growing roughly logarithmically in K, reaching about 0.85 at $K = 50$ and 0.89 at $K = 100$. The gain per additional candidate, however, drops sharply: moving from 20 to 50 candidates adds about 6 recall points, while doubling the pool from 50 to 100 adds only 4 more, at twice the re-ranking cost. Moreover, our earlier candidate-depth sweep showed that deeper pools actively hurt the final MRR@5, because the cross-encoder starts promoting semantically similar but incorrect documents. The top-50 cutoff is therefore not a recall saturation point but a cost–precision trade-off: it captures most of the attainable recall while keeping the candidate pool clean enough for the re-ranker.

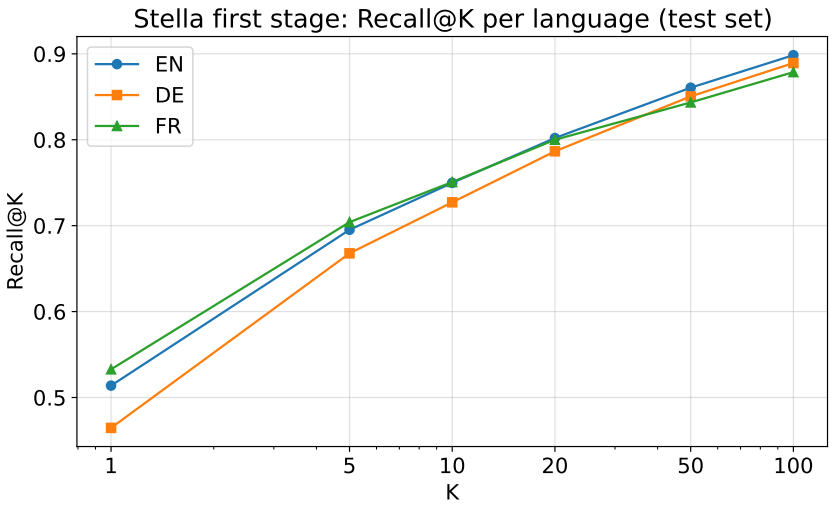


Figure 2: Recall@K of the Stella first stage per language on the test set, for $K \in \{1, 5, 10, 20, 50, 100\}$ (logarithmic x-axis). Recall grows roughly logarithmically in K: beyond the top-50 cutoff, doubling the pool buys about 4 extra recall points at twice the re-ranking cost.

Overall, these results highlight that the effectiveness of the neural reranking stage is strongly dependent on the quality and diversity of the candidate pool generated in the first stage. Even a powerful Cross-Encoder does not compensate for a weak initial recall, confirming that the two stages must be balanced (and for this reason we used the parameter α).

A different perspective on the second-stage step came from our dense retrieval pipeline (Section 3.3, Exp T). On top of Stella's top-50 we ran the larger `bge-reranker-v2-m3` cross-encoder (Exp U), again with the same α-sweep formulation. The optimal alpha turned out to depend on the language: $\alpha = 0.7$ on English and French, $\alpha = 0.5$ on German, suggesting that the right balance between the dense and the cross-encoder signals is not universal. The combination achieved the best overall scores of the project (0.6108 on English, 0.5311 on German, 0.6314 on French), but the improvement over Stella alone is only of about $+0.02$ MRR points. This confirms a clear saturation effect: when the first-stage candidate set produced by a strong dense retriever is already mostly correct, even a much larger cross-encoder has very little room left to reorder the documents. In other words, the neural re-ranking stage is most effective when the first stage has good recall but moderate precision.

5.5. Statistical Analysis

In this section, we analyze the retrieval performance of five of our experiments across the three evaluated languages (English, German, and French). To evaluate the systems, we begin with a visual and descriptive analysis of the performance distributions, followed by a breakdown of the document ranking success, and finally a formal statistical validation.

A note on the data split: all the tuning and model selection decisions discussed so far were made exclusively on the *development* set. The analysis in this section is instead computed on the official *test* set, whose ground truth was released by the organizers after the submission deadline. Since the test labels were never used for any tuning decision, this poses no leakage concern; using the test set here simply provides a larger and independent sample (6,076 English, 876 German, and 1,220 French queries) for the statistical comparison. This is why Figures 3–5 refer to the test set while the previous subsections discuss development-set scores.

5.5.1. Performance Distribution Across Languages

To understand the distribution of the *Reciprocal Rank (RR)* at 5 (RR@5) scores across different queries and languages, we generated the box plots shown in Figure 3. The figure is divided into three parallel panels, one for each language (EN, DE, FR), displaying the five systems under evaluation.

For each box, the whiskers extend from the minimum to the maximum possible values (0.0 to 1.0). The solid orange line represents the median score, while the dashed green line indicates the mean value, which corresponds exactly to the *Mean Reciprocal Rank (MRR)* at 5 (MRR@5) of the system for that specific language.

It is important to note the explicitly "squashed" shape of the boxes. This is not a visual artifact, but rather a direct consequence of the mathematical properties of the RR@5 metric. Since we are evaluating the top 5 positions, the RR@5 can only assume six discrete values: 1, 0.5, 0.33, 0.25, 0.2, and 0.

Analyzing the results, we can observe a monotonic growth in the MRR@5 (the green dashed line) across all three languages following the system progression: $\mathrm{A} \rightarrow \mathrm{H} \rightarrow \mathrm{I} \rightarrow \mathrm{T} \rightarrow \mathrm{U}$. A major breakthrough is visible when transitioning to the dense models (T_Stella and U_Stella_CE). For these two systems, the median (orange line) reaches the maximum value of 1.0. This is a highly significant result: it means that for more than 50% of the queries, the systems are able to rank the relevant document at the absolute first position (Rank 1).

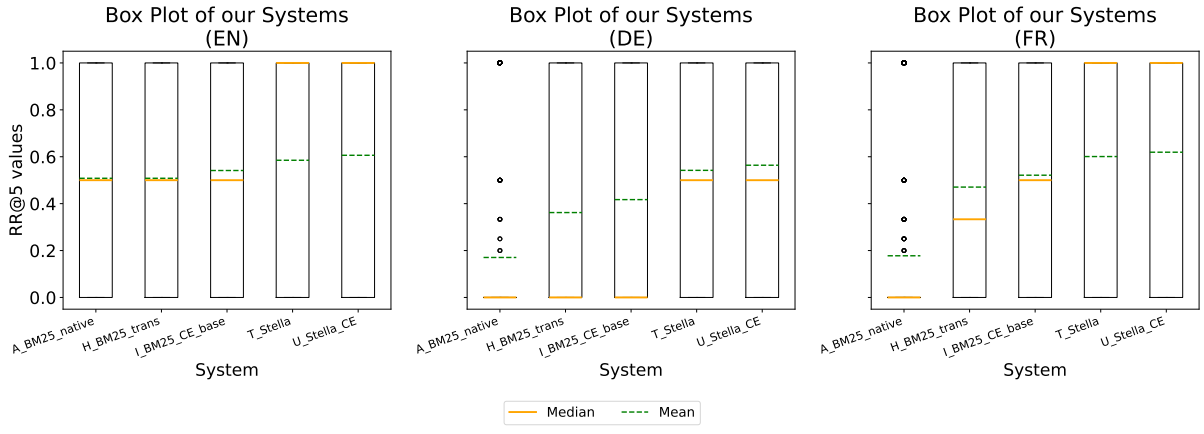


Figure 3: Box plots of the RR@5 values for the five systems on the test set, divided by language (EN, DE, FR). The orange line indicates the median, while the dashed green line indicates the mean (MRR@5).

5.5.2. Rank Distribution Analysis

To further investigate how the relevant documents are positioned within the top-5 retrieved results, we pooled the queries across the three languages and generated a stacked bar chart, presented in Figure 4.

This visualization provides a clear summary of the precision and the retrieval success of our pipeline.

Each column represents a system, and the total height represents 100% of the queries. The column is divided into three colored bands:

- **Green band:** Percentage of queries where the relevant document is found at Rank 1. This is effectively the Precision at 1 (P@1).
- **Orange band:** Percentage of queries where the relevant document is found between Rank 2 and Rank 5 ($0 < RR < 1$).
- **Gray band:** Percentage of queries where the relevant document is not found within the top 5 positions ($RR = 0$), representing the failure rate of the system.

The MRR@5 value is annotated on top of each bar for quick reference. The chart clearly illustrates how the pipeline becomes increasingly precise at the very top of the ranking. The green band (P@1) grows visibly and consistently from system A_BM25_native to U_Stella_CE. Concurrently, the gray band shrinks significantly, demonstrating that the more advanced systems fail far less frequently.

Finally, it is worth noting that the sum of the green and orange bands represents the Success Rate at 5 (which acts as a proxy for Recall@5 in this single-relevant-document scenario). This combined area also increases steadily, confirming that the algorithmic enhancements not only push relevant documents to higher ranks but also retrieve documents that were previously completely missed by the baseline systems.

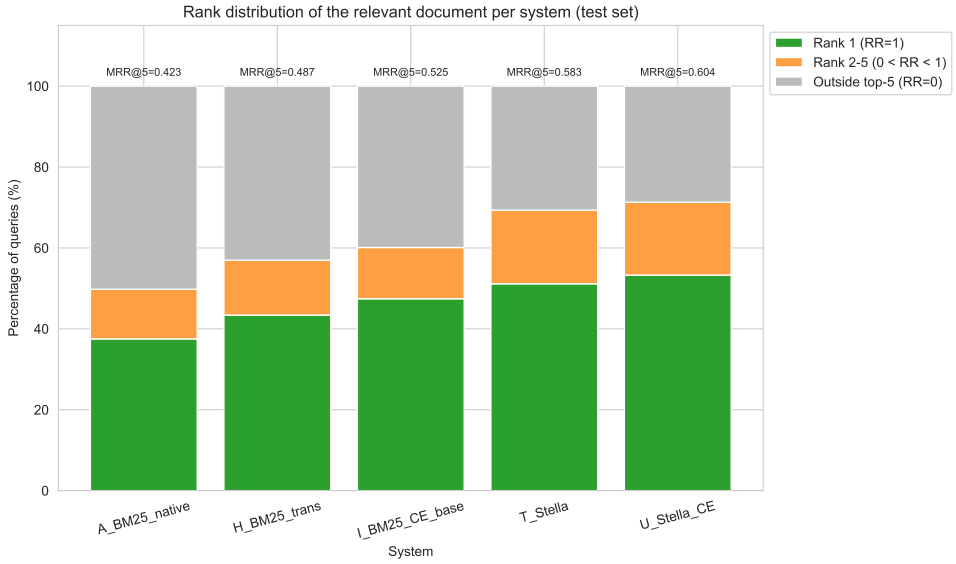


Figure 4: Rank distribution of the relevant documents per system (pooled across languages) on the test set. The annotations on top show the pooled MRR@5.

5.5.3. Statistical Validation

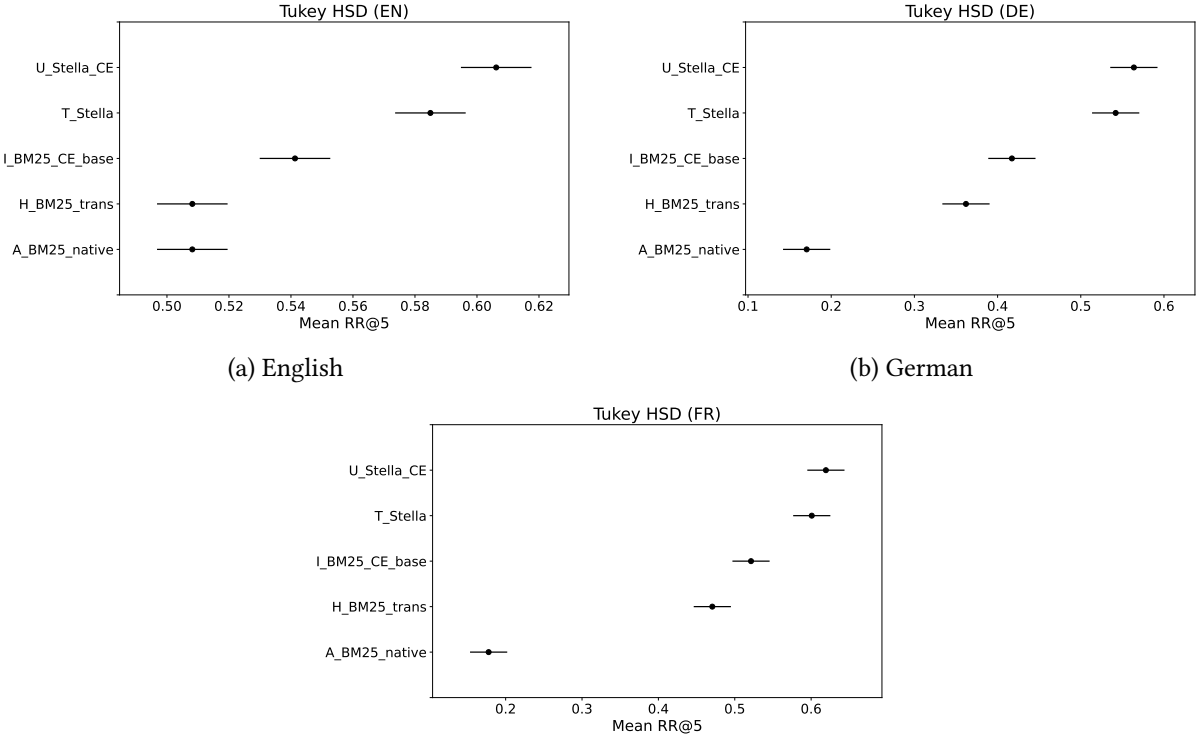
We conducted three separate Two-Way ANOVAs—one for each language—to assess the effect of the retrieval system on the MRR@5 scores. This approach inherently resolves any potential issues related to unbalanced topic distributions across the multilingual dataset.

For all three languages, the system factor showed a highly significant effect ($p < 0.001$), with F-statistics of $F = 255.18$ for English, $F = 277.62$ for German, and $F = 472.34$ for French. This confirms that the choice of architecture fundamentally impacts performance regardless of the query language.

To inspect the localized differences between specific configurations, we applied the Tukey HSD post-hoc test ($\alpha = 0.05$) independently for each language. In the corresponding plots (Figure 5), overlapping segments indicate that the performance difference between two runs is not statistically significant.

The tests highlight three main architectural insights:

- **Translation works (Run A vs. Run H):** As logically expected, the intervals for the native baseline and the translated baseline are perfectly identical in English (where translation has no effect). However, they are completely disjoint in German and French. This statistically validates that translating the queries effectively bridges the cross-lingual gap.
- **Sparse vs. Dense (Runs A/H/I vs. Runs T/U):** Across all three languages, there is a clear gap with no overlap between the lexical models and the dense models. This confirms that moving to semantic embeddings gave us the biggest and most consistent performance boost of the entire project.
- **Cross-Encoder Saturation (Run T vs. Run U):** In all three languages, the intervals for the Stella bi-encoder and the Stella + Cross-Encoder pipeline overlap. Even though the absolute MRR@5 increased slightly, the difference is never statistically significant. This confirms the saturation effect discussed in Section 5.4: the bi-encoder already retrieved very good candidates, leaving little room for the cross-encoder to make a meaningful difference.



(a) English

(b) German

(c) French

Figure 5: Tukey HSD simultaneous pairwise comparisons at a 95% confidence level ($\alpha = 0.05$) separated by language.

6. Conclusions and Future Work

Our system improved progressively through a series of experiments that combined classical and neural retrieval techniques. The initial BM25 baseline, while reasonable for English queries, performed poorly on German and French because of the strong language gap with the predominantly English corpus. The first major improvement came after translating the German and French queries into English, combined with a tuned BM25 setup (Exp H), which raised the MRR@5 substantially across all three languages and made the multilingual setting viable. Adding a cross-encoder reranker on top of the BM25 candidates further pushed the results upward (Exp I), confirming that even a relatively small neural model can refine the lexical ranking effectively when applied to a high-quality first-stage pool.

The largest improvements, however, came from replacing the BM25 first stage with a strong dense retriever. Among the bi-encoders we tested, `stella_en_1.5B_v5` (Exp T) clearly was the best, beating both the BM25 + translation pipeline and the smaller bi-encoders on every language. Combining Stella with a stronger cross-encoder reranker (Exp U) gave us the best overall results of the project, reaching an MRR@5 of about 0.61 on English, 0.53 on German, and 0.63 on French.

Beyond this point, additional attempts such as multi-source RRF fusions and HyDE-style query expansion did not provide further gains, suggesting that we had reached a natural plateau given the dense models we were able to run on consumer hardware. With more time and more powerful resources, the most natural next step would be to fine-tune a retrieval model directly on the training pairs of the task: this would teach the encoder a more direct alignment between tweets and scientific abstracts than what a general-purpose model can offer. Listwise reranking with a larger LLM, where the model rereads the top-K candidates and produces a global ranking, would be another promising direction worth exploring.

Declaration on Generative AI

During the preparation of this work, we used generative AI assistants (Claude, Gemini) in order to: support the prototyping and debugging of the experimental Python scripts, perform grammar and spelling checks, and improve the readability of parts of the text. After using these tools, we reviewed and edited the content as needed and take full responsibility for the publication's content.

References

[1] S. Schellhammer, S. Hafid, Y. S. Kartal, K. Boland, D. Dimitrov, K. Todorov, S. Dietze, Overview of the CLEF-2026 CheckThat! lab task 1 on source retrieval for scientific web claims, in: E. S. Salido, A. Barrón-Cedeño, Alba García Seco de Herrera, S. MacAvaney, J. M. Struß (Eds.), CLEF 2026 Working Notes, CLEF 2026, Jena, Germany, 2026.

[2] J. M. Struß, S. Schellhammer, S. Dietze, V. V., V. Setty, T. Chakraborty, P. Nakov, A. Anand, P. Chungkham, S. Hafid, D. Sahnan, K. Todorov, Overview of the CLEF-2026 CheckThat! lab: Advancing multilingual fact-checking, in: Experimental IR Meets Multilinguality, Multimodality, and Interaction. Proceedings of the Seventeenth International Conference of the CLEF Association (CLEF 2026), Springer, 2026.

[3] S. E. Robertson, The probability ranking principle, Journal of Documentation 33 (1977) 294–304.

[4] S. E. Robertson, The probabilistic character of relevance, Information Processing & Management 13 (1977) 247–251.

[5] W. B. Croft, D. J. Harper, Using probabilistic models of document retrieval without relevance information, Journal of Documentation 35 (1979) 285–295.

[6] A. Sharma, Practical Apache Lucene 8. Uncover the Search Capabilities of Your Application, Apress Media, New York, USA, 2020.

[7] M. F. Porter, An algorithm for suffix stripping, Program 14 (1980) 130–137.

[8] R. Krovetz, Viewing Morphology as an Inference Process, in: R. Korfhage, E. Rasmussen, P. Willett (Eds.), Proc. 16th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 1993), ACM Press, New York, USA, 1993, pp. 191–202.

[9] M. F. Porter, Snowball: A Language for Stemming Algorithms, http://snowball.tartarus.org/texts/introduction.html, 2001.

[10] A. Hosseinzadeh Vahid, P. Arora, Q. Liu, G. J. Jones, A comparative study of online translation services for cross language information retrieval, in: Proceedings of the 24th International Conference on World Wide Web, WWW '15 Companion, International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 2015, pp. 859–864. URL: http://dx.doi.org/10.1145/2740908.2743008. doi:10.1145/2740908.2743008.

[11] LibreTranslate, libre translate, https://libretranslate.com/, 2023.

[12] J. Johnson, M. Douze, H. Jégou, Billion-scale similarity search with GPUs, IEEE Transactions on Big Data 7 (2021) 535–547.
[13] A. Singh, M. D'Arcy, A. Cohan, D. Downey, S. Feldman, SciRepEval: A multi-format benchmark for scientific document representations, in: Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Singapore, 2023, pp. 4548–4577.
[14] BAAI, Bge reranker models, https://huggingface.co/BAAI/bge-reranker-base, 2023.
[15] 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, 2009, pp. 758–759.
[16] K. Hofmann, S. Whiteson, A. Schuth, M. de Rijke, Learning to Rank for Information Retrieval fromUser Interactions, ACM SIGWEB Newsletter (2014) 5:1–5:7.
[17] L. Gao, X. Ma, J. Lin, J. Callan, Precise zero-shot dense retrieval without relevance labels, in: Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (ACL 2023), 2023, pp. 1762–1777.