DIS Project 1: Document Retrieval

Team Datanauts
https://www.kaggle.com/code/saidgurbuz/bm25-chunks

Stefan Krsteski
stefan.krsteski@epfl.ch

Said Gürbüz
said.gurbuz@epfl.ch

Matea Tashkovska
matea.tashkovska@epfl.ch

1 Introduction

As the volume and diversity of digital documents increase, the challenge of effectively retrieving relevant information from multilingual text corpora becomes more and more important. This project addresses the need for robust document retrieval techniques that can handle multiple languages.

Our approach leveraged both statistical methods and transformer-based models to optimize retrieval across a corpus of documents spanning seven languages. We found that BM25, when fine-tuned with optimal parameters and supported by a data preprocessing approach including a chunking strategy, provided the best performance for this task.

2 Methods

The corpus consists of 268,022 documents covering seven languages: English, French, German, Italian, Spanish, Arabic, and Korean. The dataset further contains training and evaluation sets, each containing a query, one positive document, and negative documents, as well as a test set including 2,000 queries for which we aim to retrieve documents. To identify the most relevant documents, we utilized statistical methods such as TF-IDF and BM25 (Robertson and Zaragoza, 2009), as well as transformer-based models, including gte-multilingual-base (Zhang et al., 2024) and e5-large-instruct (Wang et al., 2024). We calculated retrieval score based on cosine similarity.

2.1 Data Preprocessing

Before employing the statistical-based models, we began with data preprocessing for each document. To tokenize the documents, which is necessary for the TF-IDF and BM25 algorithms, we used NLTK's [1] word-based tokenizer for each language separately. The tokens were then converted to lowercase to eliminate discrepancies caused by case differences. In addition, we remove stopwords and punctuation from each language to minimize noise and achieve accurate frequency measurements. For Arabic, we applied normalization techniques to simplify script variations and ensure consistent token forms, using `PyArabic` [2]. For German documents, we applied stemming to map different forms of the same word to a normalized form, which is especially beneficial in morphologically rich languages (Braschler and Ripplinger, 2004). This pipeline was not used for the transformer-based models, as they are inherently designed to capture the semantic meaning of sentences.

We discovered that the corpus documents were of significant length, where the average number of words per document was around 4500. To handle long documents that exceed the model context windows we implemented a chunking approach. For transformer-based model with small window size (e5-large-instruct), this approach is particularly crucial, as it allowed us to process documents that would otherwise exceed its context window. For BM25, while not constrained by context length, we applied the same chunking strategy to investigate its effect on retrieval performance and maintain consistency in our experimental setup.

Each document was split into non-overlapping chunks of 500 tokens. During retrieval, we processed each chunk independently and then aggregated the results at the document level to get unique documents. Specifically, when retrieving the top-10 documents for a query, we:

1. Split each document D_i into chunks $\{c_{i1}, c_{i2}, \ldots, c_{in}\}$ of 500 tokens each.
2. Compute relevance scores between the query and all chunks across all documents.
3. For each chunk c_{ij} that appears in top results, map it back to its parent document D_i.
4. Aggregate these results to determine the final top-10 most relevant documents.

2.2 Models

TF-IDF. It is a traditional statistical method for representing documents as vectors based on the significance of each word within a document and the corpus as a whole. TF-IDF consists of two main components: The term frequency (TF), which quantifies how frequently a word appears in a specific document, and the inverse document frequency (IDF), which adjust this frequency based on how commonly the word appears across all documents in the corpus.

[1] https://www.nltk.org

[2] https://pypi.org/project/PyArabic/

BM25. This algorithm extends TF-IDF by incorporating a modified term frequency that considers saturation effects, thereby preventing the overemphasis of frequently occurring terms. It utilizes parameters k_1 and b to regulate the influence of term frequency and document length on scoring, effectively reducing bias towards longer documents and over-represented terms. Additionally, we conducted a grid-search hyperparameter tuning to identify optimal values for k_1 and b, testing k_1 from 0.7 to 2.0 in increments of 0.3, and b from 0.3 to 0.9 in increments of 0.2. To accommodate the variations of different languages, we modified the approach by using per-language average document length and an IDF matrix specific to each language.

e5-large-instruct. It is a transformer-based encoder model pre-trained on multilingual data to support semantic retrieval across languages. It has been trained on a diverse dataset containing 1-billion multilingual text pairs using contrastive learning. As one of the top performing models on the MTEB (Muennighoff et al., 2023), e5-large-instruct is particularly suitable for multilingual retrieval tasks due to its large parameter count (560 million) and its high dimensional embeddings space (1024). The context length of e5-large-instruct is 512 and supports more than 100 languages which makes it suitable for our task.

gte-multilingual-base. GTE-multilingual-base is another encoder only model that was pretrained on multilingual data that covers 70 languages. A distinguishing feature of this model is its ability to process text sequences up to 8192 tokens, which allows it to do full-document retrieval by capturing and representing the entirety of longer documents without the need for chunking. The model comprises 305 million parameters and generates embeddings of dimension 768.

For fine-tuning the transformers models, we employed contrastive loss as our primary objective, with the aim of refining the model's ability to distinguish relevant documents from irrelevant ones, by leveraging the provided training dataset. This approach optimizes the model to bring embedding of relevant (positive) documents closer to the query embedding, while pushing embeddings of irrelevant (negative) documents further away in the vector space. During training, the model learns to embed positive documents close to the query representation by minimizing their distance in the embedding space, which increases the retrieval precision. More details are provided in the Appendix B.1.

3 Results

In the Table 1, we compare performance of the different information retrieval methods using Recall@10 as the evaluation metric on the test set. Recall@10 measures the percentage of queries for which the relevant (positive) document appears within the top 10 retrieved results. BM25 performed the best among the models, especially when documents were divided into 500-token chunks, with optimized hyperparameters $k1 = 1.1$ and $b = 0.3$. This chunked BM25 configuration achieved Recall@10 of 0.8193, which significantly outperformed both the traditional TF-IDF method and transformers-based models. The results also show that fine-tuning the sentence transformers modestly improved their results. Furthermore, we can see that chunking was an effective strategy for both statistical and transformer based methods. Table 2 presents language-specific performance metrics on the validation set, achieved by the best performing BM25 retrieval method. We can observe that there are differences in the models ability to retrieve the top document between languages, with Korean being the worst.

Table 2: Performance metrics for different languages on the validation set by the best BM25 retrieval method.

Language	**Recall@1 (↑)**	**Recall@10 (↑)**
English	0.695	0.875
French	0.840	0.925
German	0.565	0.740
Spanish	0.850	0.965
Italian	0.670	0.830
Korean	0.525	0.680
Arabic	0.550	0.770

3.1 Analysis

In analyzing the results of our retrieval systems, key observations include the superior performance of the statistical-based BM25 method over transformer-based models. We believe BM25's strong performance is due to its highly adjustable parameters, which can be tuned to match the specific distribution of terms in the corpus. This makes it great at handling queries focused on specific keywords (entities) that are also present in the corpus.

In contrast, sentence transformers showed suboptimal performance, a result that surprised us, considering they are pretrained on large multilingual datasets. Despite this, fine-tuning yielded modest improvements. We suspect the initial poor performance of sentence transformers may stem from their limitations in processing dense information within large texts (Chen et al., 2022). When searching for a specific keyword within a document, these models compress all information into a single vector, which can dilute a specific keywords' significance.

Chunking documents into 500-token segments allowed our system to capture local relevance that full-document method might miss. Also, by ensuring every part of the document is captured, this method prevents the loss of information, proving more effective than simple truncation.

Finally, we suspect that applying a generalized preprocessing pipeline led to performance drop. Although

Table 1: Performance comparison of different retrieval methods on 40% of the test set.

Method	Configuration	Recall@10 (↑)
TF-IDF	Full document	0.5086
BM25	Full document (k_1=1.2, b=0.7)	0.7562
	Chunked-500 (k_1=1.2, b=0.7)	0.8045
	Chunked-500 (k_1=1.1, b=0.3)	**0.8193**
E5-large-instruct	Full document (truncated)	0.4641
	Chunked-500	0.6980
GTE-multilingual-base	Pretrained (Full doc)	0.5693
	Pretrained (Chunked-500)	0.6710
	Finetuned (Full doc)	0.6188

For BM25, k_1 controls term frequency saturation (higher values mean slower saturation), while b controls document length normalization (0.0 = no normalization, 1.0 = full normalization).

we implemented custom processing for a few languages-such as stemming for German and normalization for Arabic, not all languages received the same level of customization. We believe extending these language-specific adjustments across all languages in the data could significantly improve the results, especially for languages with unique tokenization needs (e.g. Korean with its morpheme-based structure).

4 Discussion

This project evaluated various text retrieval methods, showcasing strengths and limitations of each. Our results show dominant performance of the BM25 method, which outperformed transformer-based models.

Despite being state of the art, transformers struggled with this retrieval task. The chunking strategy proved beneficial, improving the capture of local textual relevance that full-document analysis often misses.

Future work could investigate a hybrid retrieval approach, such as SPLADE (Formal et al., 2021) adapted for multilingual tasks, which merges BM25-like sparse representations with dense sentence representations from transformers to enhance retrieval performance.

References

Martin Braschler and Bärbel Ripplinger. 2004. How effective is stemming and decompounding for german text retrieval? *Inf. Retr.*, 7:291–316.

Junying Chen, Qingcai Chen, Dongfang Li, and Yutao Huang. 2022. Sedr: Segment representation learning for long documents dense retrieval.

Thibault Formal, Benjamin Piwowarski, and Stéphane Clinchant. 2021. Splade: Sparse lexical and expansion model for first stage ranking. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2288–2292.

Niklas Muennighoff, Nouamane Tazi, Loïc Magne, and Nils Reimers. 2023. Mteb: Massive text embedding benchmark.

Stephen Robertson and Hugo Zaragoza. 2009. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends in Information Retrieval*, 3:333–389.

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

Xin Zhang, Yanzhao Zhang, Dingkun Long, Wen Xie, Ziqi Dai, Jialong Tang, Huan Lin, Baosong Yang, Pengjun Xie, Fei Huang, Meishan Zhang, Wenjie Li, and Min Zhang. 2024. mgte: Generalized long-context text representation and reranking models for multilingual text retrieval.

A Code

Final submission achieved with BM25. https://www.kaggle.com/code/saidgurbuz/bm25-chunks
BM25 notebook with our final approach that can be used to achieve our best performing score in under 5 minutes.

GitHub Repository https://github.com/Stefanstud/Infomation-retrieval
Just in case, our GitHub repository that contains every approach and supporting scripts. This also includes some alternative methods that we do not mention in main text, such as BGE-M3 and learning to rank.

B Appendix

B.1 Contrastive Loss

In this project, we employ a contrastive loss function to fine-tune our transformer-based model. This loss encourages the model to bring embeddings of similar pairs closer while pushing embeddings of dissimilar pairs apart. Specifically, each query sample q_i is associated with one positive sample p_i and multiple negative samples $n_{i,j}$. For each query q_i, we compute similarity scores with its positive and (M) negative samples using a similarity function $\text{sim}(\cdot, \cdot)$ (cosine similarity). The loss for a single query-positive-negative set is defined as:

$$L_i = -\log \frac{\exp(\gamma \cdot \text{sim}(e_{q_i}, e_{p_i}))}{\exp(\gamma \cdot \text{sim}(e_{q_i}, e_{p_i})) + \sum_{j=1}^{M} \exp(\gamma \cdot \text{sim}(e_{q_i}, e_{n_{i,j}}))} \tag{1}$$

where e_{q_i}, e_{p_i}, and $e_{n_{i,j}}$ are the embeddings of the query, positive, and negative samples, respectively, and γ is a scaling factor that controls the sharpness of the probability distribution. The total loss over a batch of N samples is the average of the individual losses. This formulation ensures that the similarity between each query q_i and its positive p_i is maximized, while the similarities between the query q_i and its negatives $n_{i,j}$ are minimized. By doing so, the model learns to distinguish the positive sample from the negatives, which enhances its retrieval performance.

B.2 Pretrained Models

All of the pretrained models used in this project were purely for comparison, including, gte-multilingual and e5-large-instruct. These models were used as a baseline to gauge the effectiveness of chunking and fine-tuning. It is worth mentioning that fine-tuning models were not used with chunking approach, as we had not ground truths for the chunked documents.

B.3 BGE-M3

The BGE-M3 model supports sparse, dense, and ColBERT representations, making it suited for hybrid retrieval with cross-encoder reranking as the optimal approach. However, due to the high computational requirements to run this model, its use was considered out of scope for our main experimentation and is discussed in the Appendix. Our experiments focused on using only dense vector representations of this model, which achieved a Recall@10 score of 0.73 without any fine-tuning. We believe that utilizing all three representations would likely yield the best overall performance, as the combination of sparse and dense vectors would capture both semantic meaning and keyword-specific features. Moreover, this unified representation approach can be fine-tuned using the training data to yield an even better model.