Long Documents Formater for Information Retrieval (LDF-IR): a BM25 Enhancer

Christophe Rodrigues, Richard Goudelin, Oscar Pastural

April 2024

1 Abstract

Despite the significant advancements in Natural Language Processing (NLP), supervised models need huge trainings to understand the subtle arguments of a query. In the context of Information Retrieval[8] (documents retrieval), an unsupervised algorithm, the BM25 algorithm remains a highly effective tool, often surpassing complex NLP models. In this paper, we propose an innovating preprocessing technique that enhances the BM25 algorithm by incorporating semantic understanding from pre-trained text embedders. Our approach focuses on refining the representation of documents through selective word duplication and omission, which improves the rarity and relevance of terms. This method was applied to the NF Corpus dataset[9], part of BEIR Benchmark[10][1], and achieved a Normalized Discounted Cumulative Gain (NDCG@10) score of 0.347, surpassing SPLADE's model score of 0.345 and the baseline BM25's score of 0.322. This enhancement retains the efficiency of BM25 in terms of memory usage and execution speed, making it suitable for text-based database searches and real-time applications.

2 Introduction

The field of Information Retrieval (IR) is a core component of computer science, tasked with the extraction of relevant documents from extensive databases in response to user queries. The primary objective of such systems is to sift through vast quantities of text, pinpointing documents of interest while concurrently discarding those that lack relevance.

In the corporate environment, the need for efficient IR systems is paramount. Businesses are often repositories of vast swathes of unlinked, text-heavy documents. Companies must keep their data private but also be need to navigate through their system's documents. Our research is aimed at developing an IR system tailored to the corporate context, one that can adeptly navigate through a company's data, retrieving the most pertinent documents in response to specific queries.

Historically, IR systems have been reliant on term-matching approaches, where the frequency and presence of words from the query within documents govern retrieval efficacy. However, this method grapples with issues of polysemy, synonymy, and lexical gaps that can hinder accuracy.

Nowadays a lot has improve and make the research in retrieval information promising because of:

- The surge in computational capabilities
- The proliferation of large, annotated datasets

- Innovations in deep learning methodologies like RNNs, CNNs, transfer learning, and advanced pre-training strategies
- The integration of sophisticated pre-trained language models (e.g., GPT-2 and BERT), which enhance semantic and contextual understanding

Despite their potential, these advanced techniques are often synonymous with high data and computational resource requirements. As such, there is an active pursuit within the academic and research communities for the development of algorithms that conserve computational resources without compromising on the quality of NLP tasks.

Our contribution to this field lies in presenting an innovative model that amalgamates the reliable aspects of traditional IR systems with modern technological advantages. This hybrid model aims to reduce computational demand while enhancing the accuracy. Through this, we aspire to contribute to the evolution of IR systems that are both powerful and pragmatic for real-world applications.

This article would be structured like this: in the First Section, we will talk about Preliminaries studies that led to our method. In the Second Section, we will talk about our method. In the third section, we will evaluate our model on NF Corpus and compare them to results made by other Information Retrieval Models. In our Fourth Section, we will discuss the results and limitation, and finally our future works.

3 Preliminaries studies

3.1 The choice of the dataset

The NFcorpus dataset is a collection of data used in the field of information retrieval, and it poses an interesting challenge for several reasons. Firstly, NFcorpus consists of documents in the French language, which sets it apart from the numerous English-language corpora available in the field. Given that the French language has unique syntactic and semantic characteristics, processing and analyzing these documents require specific techniques tailored to this language.

Furthermore, NFcorpus is composed of documents from various sources such as news articles, online forums, blogs, and more. This diversity of sources introduces significant variations in writing style, formality level, and text quality. Consequently, it becomes more challenging to find generalized patterns and methods for effective information analysis and retrieval within this corpus.

Another challenge associated with NFcorpus is the presence of noise and ambiguity in the texts. The documents may contain typographical errors, abbreviations, incomplete or poorly structured sentences, making automated natural language processing tasks more complex. Traditional information retrieval techniques may not be as effective when applied to such noisy and non-standardized data.

Finally, when evaluating the overall performance of different information retrieval (IR) systems on this dataset, the highest achieved score is a modest 0.345. Therefore, considering these factors, we deliberately selected this complex dataset where semantic understanding plays a significant role in scoring due to the usage of specific technical terminology.

BM25: A Strong Baseline in Information Retrieval

Understanding BM25[11] is crucial as it is our main contender to beat and also our chosen algorithm to improve. First, it is important to recognize why BM25 is a muscular challenge today despite being an algorithm from the 1980s:

As highlighted in the BEIR paper, the global performances of BM25 compared to other computationally heavy and slow systems are really impressive. In fact, the efficiency and effectiveness of BM25 in various contexts have made it a dominant choice in information retrieval systems:

- Traditional Dominance: BM25 has been widely used in IR systems as a lexical retrieval approach based on TF-IDF principles.
- Robust Baseline: Despite the advancements in neural models, BM25 serves as a strong baseline for zero shot-text retrieval, providing reasonable performance without specific training on the task or dataset.
- Computational Efficiency: Compared to computationally expensive neural models, BM25 demonstrates computational efficiency while still delivering satisfactory retrieval performance.
- Performance Gains: While newer neural models have shown significant performance gains over BM25 in specific contexts, BM25 remains a reliable option for various retrieval tasks.
- Generalization Capabilities: Although BM25 performs well in many scenarios, there is still room for improvement in its generalization capabilities, especially when dealing with out-of-distribution data or different text domains.

BM25 Scoring Function

The BM25 score of a document (D) for a query (Q) is calculated as follows:

$$\text{score}(D, Q) = \sum_{i=1}^{n} \text{IDF}(q_i) \cdot \frac{f(q_i, D) \cdot (k_1 + 1)}{f(q_i, D) + k_1 \cdot \left(1 - b + b \cdot \frac{|D|}{\text{avgdl}}\right)} \tag{1}$$

Given a query term q_i and a document D, let $f(q_i, D)$ denote the frequency of q_i in D, and $|D|$ represent the length of the document. The average document length in the collection is denoted as avgdl. Parameters k_1 and b are tuning parameters in the model, typically set within $k_1 \in [1.2, 2.0]$ and $b = 0.75$, respectively.

The Inverse Document Frequency (IDF) of term q_i, denoted as $\text{IDF}(q_i)$, is computed using the equation:

$$\text{IDF}(q_i) = \log\left(\frac{N - n(q_i) + 0.5}{n(q_i) + 0.5} + 1\right) \tag{2}$$

where N is the total number of documents in the collection, and $n(q_i)$ is the number of documents that contain the term q_i.

Dataset ↓	**BM25 flat**	**DeepCT**	**SPARTA**	**docT5query**	**DPR**	**ANCE**	**GenQ**	**ColBERT**	**TAS-B**	**uniCOIL**	**SPLADE**
NFCorpus	0.322	0.283	0.301	0.328	0.189	0.237	0.319	0.305	0.324	0.333	0.345

Table 1: Performance of various models on the NFCorpus dataset, with data sourced from BEIR.

BM25 continues to be a strong contender in the field of information retrieval due to its impressive overall performance, fast execution time, and low memory usage. However, one limitation of BM25 is apparent: it does not consider the semantic relationships between words. Consequently, BM25 may struggle to identify closely related documents that contain similar meaning but different lexical forms.

3.2 Pattern mining of documents

To begin our exploration into effective document retrieval, we first sought to discern the attributes that define a high-score document within our dataset. This entailed visualizing the relationship between user queries and the most relevant documents via dimensional plotting methods (UMAP). By representing both queries and documents in vectorized form, we observed in two-dimensional and three-dimensional spaces the emergence of discernible patterns.

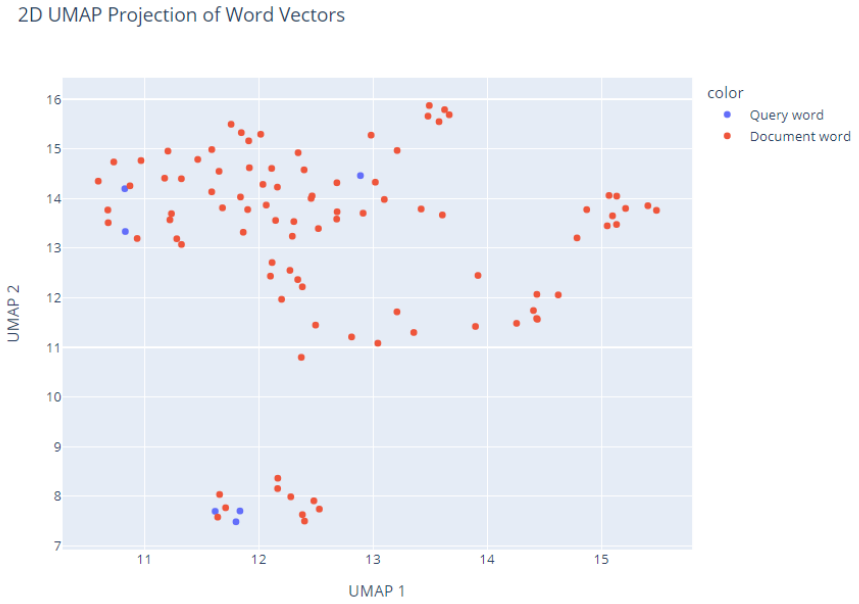

Figure 1: 2d projection of the vectorized words of the query over the vectorized words of the top 3 document for that query according to the dataset

Our initial findings revealed that the query words often formed distinct clusters, frequently positioned at a relatively large distance from the central aggregation of document words. This clustering suggested that words located near the center of the vector space might hold less discriminating power in representing document relevance.

From this insight, we hypothesized that the central words within a document may not be as critical for its representation, particularly when considering retrieval algorithms like BM25 that evaluate terms from a lexical perspective. This prompted us to consider a preprocessing step that acknowledges the limited utility of these central terms.

By integrating a preprocessing approach that filters out less informative central words, we aimed to enhance the inherent strengths of BM25—namely, its speed and modest memory requirements. Our goal was to refine the system's overall performance, leveraging BM25's efficiency while mitigating its potential overemphasis on less representative terms.

4 Improving the representation of a document according to BM25 using a pre-trained text embedder

4.1 General Pipeline

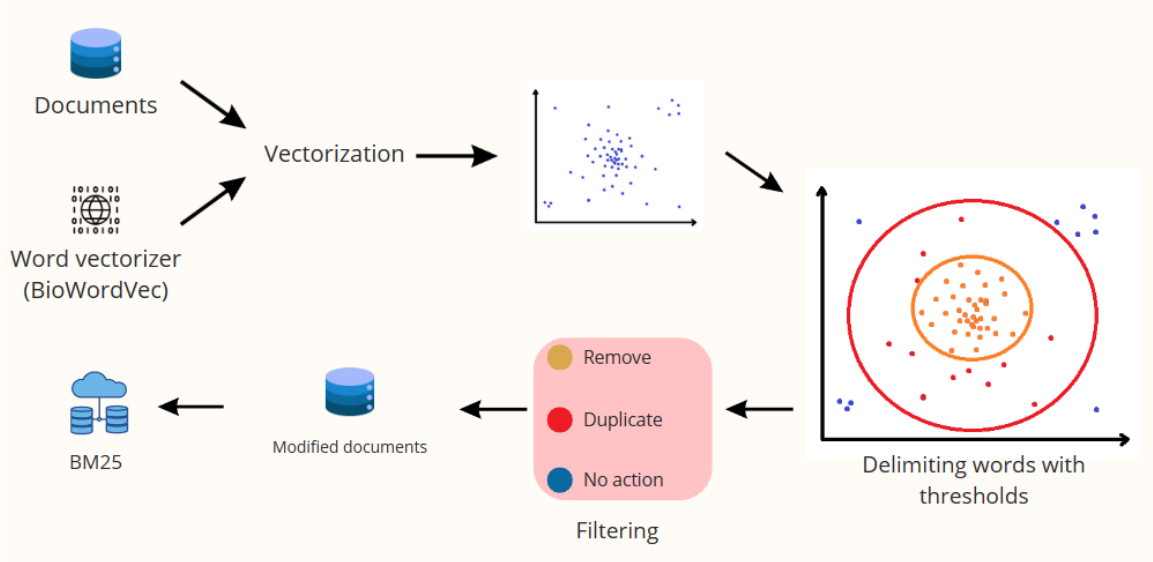

Figure 2: General pipeline of the pre-processing

a) Vectorization: Documents are processed through BioWordVec to transform words into vectors, capturing their semantic meaning.

b) Threshold Application: Vectors are then evaluated, and words are classified using thresholds to decide their relevance.

c) Filtering: Based on relevance, words are either removed, duplicated, or left as is to optimize document uniqueness.

d) BM25 Processing: The modified documents are fed into the BM25 algorithm for enhanced information retrieval.

4.2 First idea : duplicate and delete words

This pipeline does not describe our first idea. Our first idea is different from our final approach but it remains important to introduce because it is what lead us to promising results. Here is the logic, instead of duplicating middle rare word

4.3 Refining the Term Selection Process

In the context of the BM25 algorithm, enhancing the retrieval quality involves refining the term selection process. One approach is to eliminate "middle words" that are too common. These are words that fall below a certain threshold, defined as a lower percentage x times the maximum distance of a word from the center of the document vector space. This threshold helps to identify and remove words that contribute less to the document's uniqueness:

$$\text{threshold} = x \cdot \max(\text{distances to center}) \tag{3}$$

By omitting these words, the algorithm reduces noise and focuses on more distinctive terms that carry a higher Inverse Document Frequency (IDF) weight, thus improving the search accuracy and relevance of the results.

4.4 Enhancing the Uniqueness of Rare Words

To emphasize the distinctiveness of truly rare and specific words—those above the upper threshold—a strategy is employed to adjust the frequency of borderline rare words within the document. Words that are moderately infrequent and fall between the upper and lower thresholds are identified and duplicated. This action does not directly alter the rarity of the most uncommon words but instead modifies the term frequency (TF) values within the document, leading to an indirect increase in the IDF scores of the truly rare words. The adjustment of term frequencies results in a more nuanced representation of the document's content according to the BM25 model. With these modifications, the BM25 algorithm can provide a more accurate measure of term significance, which leads to improved document retrieval performance.

4.5 Evaluating the performances

Method	**Score**	**Relative Score (%)**
BM25 Flat	0.322	-0.92
BM25 Beir	0.325	0
uniCOIL	0.333	2.46
SPLADE[13]	0.345	6.15
TAS-B	0.324	-0.31
Contriever	0.328	0.92
Duplicate&delete	0.32844	1.76
Stem+duplicate&delete	0.3474	7.63

Table 2: Comparison of Methods by Score and Relative Score

Duplicate&delete obtained a score of that improve of 1.76% compared to BM25 using our pre-processing method over non stemminized and just tokenized corpus, this suggests that the technique it employs adds value likely by manipulating document representations in a way that better aligns with user queries. Stem+duplicate&delete stands out with a relative score improvement of 7.63%, marking a substantial enhancement over BM25, this was possible with stemminized word with our pre-processing method.

4.6 Ablation study : roles of our differents parameters

4.6.1 Pre-processing

In text preprocessing, we undertake several steps to clean and standardize the input data before any further processing or analysis. First, we convert all text to lowercase to ensure uniformity across different forms of the same word, which helps in reducing the complexity of subsequent processing steps. Additionally, we often implement what is commonly referred to as flat preprocessing by incorporating the document's title into the main text body. This approach enriches the context and enhances the overall comprehension of the document's content.

In our preprocessing routine, we begin by removing various special characters and punctuation marks which could lead to the creation of non-informative tokens. Concurrently, we filter out stopwords—these are frequently occurring words that typically do not carry significant meaning and could skew the analysis.

After that we tried two types of preprocessing:

- We begin our text preprocessing by tokenizing the text, which involves breaking it down into individual words or elements. This foundational method of tokenization yielded an NDCG@10 score of 0.322.

  Following tokenization, we enhance our preprocessing approach by removing common words and selectively duplicating semi-rare words. This tailored adjustment led to an improved NDCG@10 score of 0.3294. This outcome is particularly notable when compared to the benchmarks provided in Table 1.

- We applied stemming using the Porter Stemmer algorithm, which reduces words to their root forms. This crucial step enables us to treat different forms of a word—such as plurals or various tense structures—as a single entity, which is particularly relevant given the term-matching logic of the BM25 algorithm. This seemingly modest preprocessing enhancement significantly improved our score, elevating it to 0.344 NDCG@10.

  Following the application of our comprehensive preprocessing methodology, the score further increased to 0.3474 NDCG@10. This demonstrates the effectiveness of integrating stemming into our preprocessing routines, optimizing the performance of the BM25 algorithm in handling diverse linguistic variations.

4.6.2 Word embedder

To effectively vectorize words, we utilize BioWordVec, an expansive 13GB word embedding model renowned for its robust representation of word semantics in the domain of biomedical. This model successfully vectorizes nearly every word encountered. However, to optimize our BM25 scoring, we introduced stemming in the pre-processing phase. Stemming normalizes words by trimming them to their root forms, which can lead to a subset of words—approximately 10%—that BioWordVec is unable to vectorize due to their rarity or specialized nature.

We observed that these non-vectorized stemmed words are really rare, to the extent that they were not included in the training corpus of BioWordVec. Consequently, these terms, if vectorized, would inherently fall above our established upper rarity threshold. Therefore, they are would have been preserved in their original form within the document representation. By maintaining these rare stemmed words unaltered, we inherently boost the document's representational uniqueness in the context of the BM25 algorithm.

After extensive experimentation with BioWordVec, we opted to switch to a more lightweight model, approximately 4GB in size—three times lighter than our previous choice. We observed that our enhancement techniques remained effective, with a slight decrease in performance. Specifically, using the stemming method, our score decreased marginally from 0.3474 to 0.3471 NDCG@10.

This transition to a lighter model demonstrates that it is feasible to achieve robust performance even with resource constraints, making it a viable option for individuals or organizations with limited computational resources.

4.6.3 Stability of the method, variating upper and lower thresholds

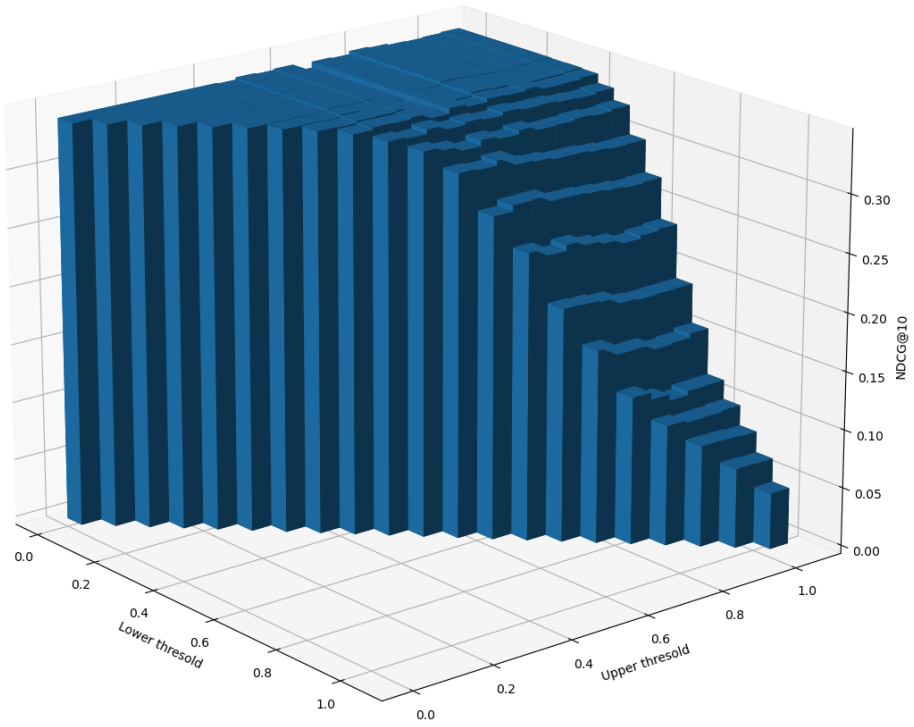

Figure 3: Grid search for the upper and lower thresholds

If you take reasonable threshold you outperform BM25 in almost any cases. We wanted to find a way to find the best threshold for a given dataset. So we decided to do this using a supervised learning method. In fact out of the 323 query that are available on NF Corpus we did a gridsearch that showed two things:

- We can find goods parameters using a grid search over 30 queries
- The methods is stable,the stability of the method show that it is a robust method and we are not under over-fitting

The results from the grid search suggest that not only is the method consistent across different queries, but also that a smaller, curated set of queries can be effective for tuning the parameters. This streamlined approach to threshold optimization allows for both efficient computation and high-accuracy outcomes in enhancing BM25's retrieval capabilities.

4.7 Time evaluation

Models need to be able to compare a query to highly numerous corpuses. A high speed for retrieving the right documents is strongly recommended especially in our use case. For our comparison, since the preprocess is based on long documents, we chose to compare the speed of BM25 and our model on NF Corpus on two different setups. The scores in the table are average scores done from doing the experiment of measuring the time of calculations for all the corpus, and the measures were taken with the time library in python and expressed in seconds.

Time for all corpus	**Average Time per Query**
2.041	0.00632
2.129	0.00659

Table 3: GPU on Google Colab V100 (in seconds)

Time for all corpus	**Average Time per Query**
1.427	0.00442
1.288	0.00399

Table 4: CPU on local Jupyter Notebook (Intel64 Family 6 Model 158 Stepping 9, GenuineIntel) (in seconds)

We notice that our model is slightly faster than BM25. That can be explained by the removal of common words that consist a big part of the data and still out represent less common words, even if their presence is doubled.

4.8 Memory Usage

The size taken by our model is really light: it only contains functions to re-write documents before enacting a BM25 on them. To get a view on what it takes in terms of memory, we simulated the memory used by BM25 and our method (average memory used for 30 tries with re-initialization of the memory at each iteration). Memory was measured with psutil library in python. Both measurements were taken in the same context:

Method	**Average Memory usage**
BM25	1638.4
Enhanced BM25	273.07

Table 5: Average Memory usage in octets

As we can see, our method reduces the memory used by the algorithm. This can be explain by the variety of words in native documents and documents re-written.

4.9 Limits of the model

The models presented in this paper presents poor results in certain cases: in corpus with shorter documents, the representation of the rarity of the word loses sense since there are fewer words than in NF Corpus. Indeed, what allow this technique to work is the length of the documents that compose NF Corpus.

5 Bonus - Origin of the model

This subsection aims to propose models that we tried to do but had no success with, and allow next researchers on this subject to gain knowledge of what was tried and how:

5.1 Emergence of the Method

5.1.1 Clustering words with K-nn

On a first approach, we wanted to cluster words with the closest words in terms of meaning and represent a cluster of words by a new one ("tomato" and "bolognese" would be represented and replaced in documents by a new word "newword1"). All of these words would be mapped in a dictionary, used to re-write the query inputted. Then, we would use a TF-IDF (detailed in section three) algorithm named BM25 to look for the best matching documents between the newly written query and the re-written documents. Unfortunately, this method although it has proven to beat a simple BM25 on many queries was unstable and could not be stabilized due to the fixation of the number of clusters, n. This variable n is a parameter of the method of clustering chosen, K-nn, which we found obsolete since we cannot predict how distributed are words in an unknown set of documents.

Indeed, grouping words with close meaning was a good idea but with K-nn, we had to choose how many were regrouped under the same new name. Without knowing in advance the disposition of the words, it would leave us with poor results. Since this technique was condemned to be overfitted in order to get good results, we quickly dropped it, but kept the idea of regrouping close words in our technique.

5.1.2 Other Clustering Methods

A better way to get rid of the n parameter in the K-nn clustering method was to go into other methods. A firm point we would like to insist on was that the aim of our research was to create an unsupervised algorithm that could use general properties in order to be fast and low in computing power. That is why we did not go towards the solution of training an algorithm based on clustering methods, which would require user feedback on unknown documents among other things.

We kept the essence of the technique, but we changed the clustering method we used and tried to find the ones that would fit our reasoning the best[12]. After trying the most pertinent ones and stabilizing the results after performing gridsearchs, we found out that the algorithm required other parameters to be efficient on unknown documents. When some conditions were met, the results would be incredible and beat some of the most known unsupervised IR algorithms, but it would crush the second some parameter went off the grid. The problem was that the parameters that had the influence on the scores were unknown, and finding them could be a huge use of time resource. We could suspect a bunch of parameters, but what made us change our minds on the method were two things:

- If one of the parameters was due to the query, it would mean that the preprocess of re-writing the documents would have to be done at each query, an expensive and slow solution.
- The preprocess was too expensive, and too slow.

What came through our mind at this stage of the research was this problematic: Could we simplify this reasoning without the bearing of the slowness and expensiveness in HC?

5.1.3 Reducing Dimensions

Even though clustering words together was a good idea to enhance BM25 by giving it more context, it had its limitations, whether is Time efficiency or method randomness. Based on this, we tried to go back to simpler ways to analyze and retrieve information with rules and analysis of the corpus. In the middle of our research, Christophe Rodrigues, research team supervisor, wrote an article on optimization on extracting meaningful information from text, "REDIRE : Réduction Extrême de DImension pour le Résumé Extractif"[7]. In this article, he used a unique approach to extract meaningful parts of a text and use them to resume a document.

The aim was different, but the way to get the right words and give them more weight to represent the document was similar. We projected the words of our documents in one dimension and observed that the words that had more meaning towards the singularity of the document were spaced out on sides of the dimension eg: if the document was the sentence "An apple was eaten by a monkey with hair", apple, monkey and hair would be on the extreme sides of the dimension. This approach does not require frequency to find meaningful words, and helps us thrive through the limits of the rule 'rare words are meaningful' towards a new rule 'words extrapolated from the center in a one dimension environment have more impact'. Since we still wanted to get the closest neighbour to have more weight, we kept more dimensions to still find the closest synonymous. In the end, this method had potential but was eclipsed by our results on the main approach of this article.

References

[1] Boytsov, L.; Novak, D.; Malkov, Y.; Nyberg, E. Off the beaten path: Let's replace term-based retrieval with k-nn search. Proceedings of the 25th ACM international on conference on information and knowledge management, 2016, pp. 1099–1108.

[2] Wrzalik, M.; Krechel, D. CoRT: Complementary Rankings from Transformers. arXiv preprint arXiv:2010.10252 2020.

[3] Dos Santos, C.; Barbosa, L.; Bogdanova, D.; Zadrozny, B. Learning hybrid representations to retrieve semantically equivalent questions. Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers), 2015, pp. 694–699.

[4] Gao, L.; Dai, Z.; Chen, T.; Fan, Z.; Van Durme, B.; Callan, J. Complement lexical retrieval model with semantic residual embeddings. European Conference on Information Retrieval. Springer, 2021, pp. 146–160.

[5] T. Soni Madhulatha ; AN OVERVIEW ON CLUSTERING METHODS IOSR Journal of Engineering Apr. 2012, Vol. 2(4) pp: 719-725

[6] O. Jeunen ; I. Potapov ; A. Ustimenko ; On (Normalised) Discounted Cumulative Gain as an Off-Policy Evaluation Metric for Top-n Recommendation

[7] Marius Ortega, Aurélien Bossard, Nedra Mellouli, Christophe Rodrigues on REDIRE : Réduction Extrême de DImension pour le Résumé Extractif

[8] Introduction to Information Retrieval Christopher D. Manning ; Prabhakar Raghavan ; Hinrich SchutzeIntroduction

[9] Vera Boteva ; Demian Gholipour ; Artem Sokolov ; Stefan Riezler NFCorpus: A Full-Text Learning to Rank Dataset for Medical Information Retrieval

[10] Nandan Thakur ; Nils Reimers ; Andreas Rücklé ; Abhishek Srivastava ; Iryna Gurevych BEIR: A Heterogeneous Benchmark for Zero-shot Evaluation of Information Retrieval Models

[11] S. Robertson ; H. Zaragoza The Probabilistic Relevance Framework: BM25 and Beyond

[12] T. Soni Madhulatha An overview on clustering methods

[13] Thibault Formal ; Carlos Lassance ; Benjamin Piwowarski ; Stéphane Clinchant SPLADE v2: Sparse Lexical and Expansion Model for Information Retrieval

[14] Stephen Robertson ; Hugo Zaragoza ; Michael Taylor Simple BM25 extension to multiple weighted fields

[15] Hamed Valizadegan ; Rong Jin ; Ruofei Zhang ; Jianchang Mao Learning to rank by nDCG measure

[16] Yining Wang ; Liwei Wang ; Yuanzhi Li ; Di He ; Wei Chen ; Tie-Yan Liu A Theoretical Analysis of NDCG Type Ranking Measures

[17] Kailash Hambarde ; and Hugo Proença Information Retrieval: Recent Advances and Beyond

[18] Xiaoyong Liu ; W. Bruce Croft Cluster-Based Retrieval Using Language Models