Università degli Studi di Milano Bicocca

Dipartimento di Informatica

Corso di Laurea Magistrale

Information Retrieval Project

Massimo Trippetta - 869286

Lorenzo Megna - 868929

ANNO ACCADEMICO 2024/2025

0.1 Analysis of the Dataset

0.1.1 Collection, Query, and Qrels Analysis

The NFCorpus dataset was analyzed to understand the structure of documents, queries, and relevance judgments (qrels).

0.1.2 Document Length Analysis

Document Lengths: The distribution of document lengths (measured in tokens) was examined to identify patterns and outliers. Extremely short or long documents were identified and visualized using histograms. The number of total documents is 3633 with an average document lenght (title+abstract) of 220.978.

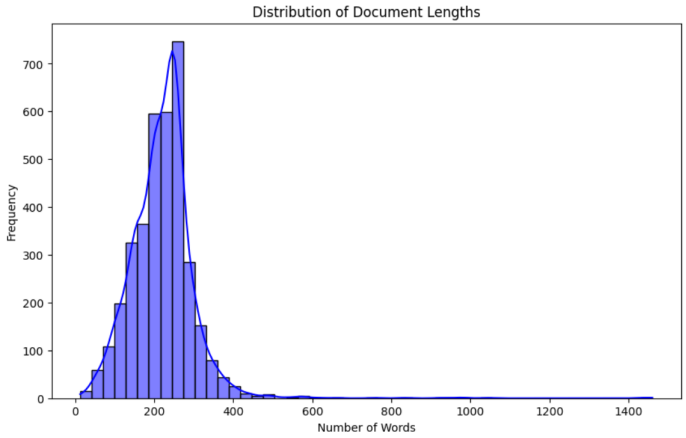

Figura 1: Document length distribution.

As we can see most documents are within the range of 100 and 300 words, with few outliers on both ends.

The word cloud helps us understand the most common words in the dataset:

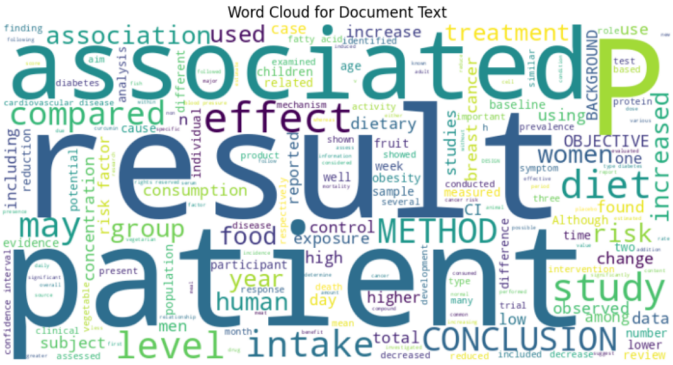

Figura 2: Document World Cloud.

Common words like ”associated,” ”result,” ”patient,” and ”level” dominate, reflecting the dataset’s focus on medical and clinical contexts. Other notable terms include ”diet,” ”treatment,” and ”effect,” which indicate prevalent topics within the documents.

Query Lengths: Queries were analyzed based on their token count.

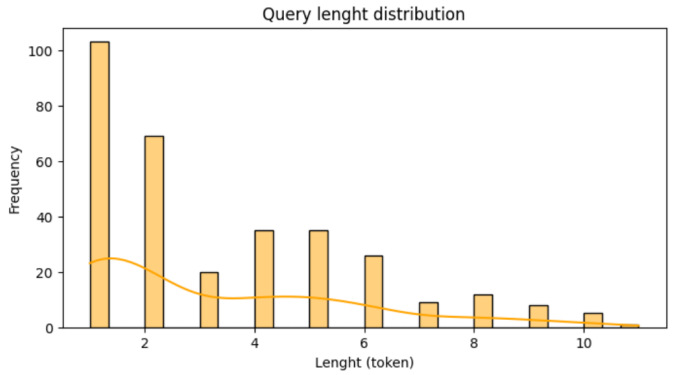

Figura 3: Query length distribution.

There are 323 query with an average lenght of 3.29 token.

0.1.3 Relevance Judgments Analysis

The dataset includes relevance judgments that associate queries with relevant documents.

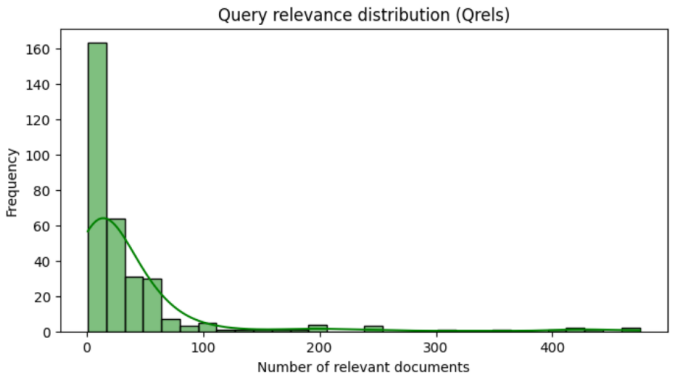

Figura 4: Query relevance distribution.

Most queries have a low number of relevant documents. A small number of queries have significantly higher relevance counts, indicating an uneven distribution. Also, the majority of relevance judgments have a value of 1 (nearly 12000), with a very small fraction having a relevance value of 2. This indicates that most documents are marginally relevant to the queries, and highly relevant documents are rare in the dataset.

Basic Retrieval Pipelines

2.1 Indexing the Collection

We utilized **PyTerrier** to index the NFCorpus, which comprises document titles and texts.

2.2 Retrieval Pipelines

Two statistical retrieval models were implemented:

- **TF-IDF**: Calculates relevance scores using term frequency and inverse document frequency.
- **BM25**: Extends TF-IDF by incorporating term saturation and document length normalization.

We experimented with different indexing configurations:

- Pipeline 1: Base (title + text, no stemming, no stopword removal) - Mean token: 233
- Pipeline 2: Stemming (title + text, stemming) - Mean token: 233
- Pipeline 3: Stopword Removal (title + text, stopword) - Mean token: 155
- Pipeline 4: Only title (no stemming, no stopword removal) - Mean token: 12
- Pipeline 5: Title + text with lowercasing, stemming, stopword removal - Mean token: 155
- Pipeline 6: Text with no preprocessing - Mean token: 220

2.3 Retrieval Effectiveness Evaluation

To evaluate retrieval performance, the following metrics were used:

- **Precision@k**: Measures the ratio of relevant documents among the top k retrieved results.
- **Mean Average Precision (MAP)**: Computes the average precision across all queries.
- **Normalized Discounted Cumulative Gain (nDCG)**: Considers the ranking order of relevance in the retrieved list.

Pipeline	Model	MAP	P@10	nDCG@10
1	TF-IDF	0.1521	0.2359	0.3273
1	BM25	0.1522	0.2387	0.3273
2	TF-IDF	0.1488	0.2322	0.3220
2	BM25	0.1489	0.2347	0.3224
3	TF-IDF	0.1519	0.2353	0.3269
3	BM25	0.1521	0.2381	0.3273
4	TF-IDF	0.1011	0.1737	0.2479
4	BM25	0.1012	0.1740	0.2479
5	TF-IDF	0.1487	0.2322	0.3219
5	BM25	0.1487	0.2347	0.3224
6	TF-IDF	0.1486	0.2325	0.3228
6	BM25	0.1491	0.2325	0.3237

Tabella 1: Performance of Retrieval Models Across Pipelines

Our evaluation revealed key findings:

- BM25 consistently outperformed TF-IDF in terms of MAP and nDCG.
- Indexing both titles and texts got superior results compared to indexing only titles or texts.

2.4 Comparative Analysis

- **Stemming Impact**: Applying stemming reduced token count and improved indexing efficiency without significantly affecting retrieval performance.
- **Retrieval Models**: BM25 demonstrated robust performance across configurations, emerging as the preferred model for this dataset.
- **Pipeline Configurations**: Combining titles and texts during indexing maximized retrieval effectiveness, highlighting the importance of comprehensive indexing strategies.

2.5 Low performing queries

The next step is to identify the low performing queries in order to find some similarities between them and to understand why they aren't performing well.

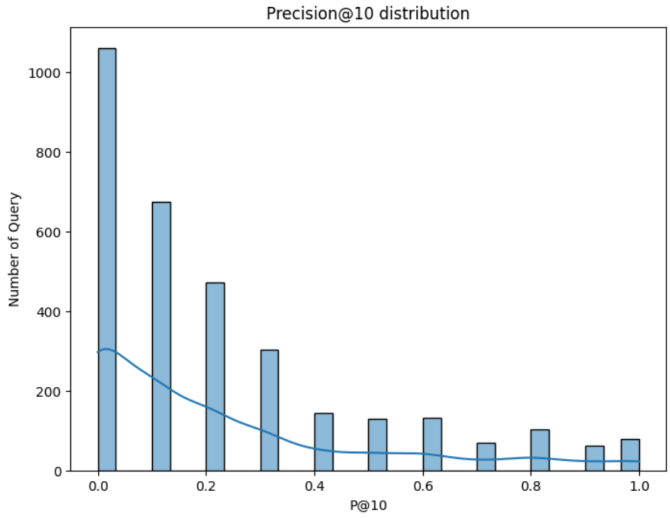

Figura 5: P@10 distribution.

2207 queries resulted in low precision (P@10 $<=$ 0.2). A thing to notice is that the queries with low P@10 have a small number of words (Mean: 3.357) as we can notice in the next graph:

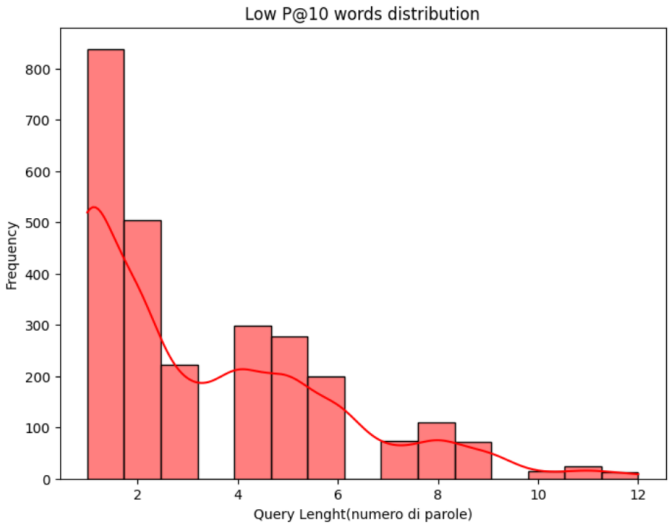

Figura 6: Low P@10 token distribution.

The most frequent terms include stopwords like ”of,” ”and,” and ”the,” as well as domain-specific terms like ”Diet,” ”Disease,” and ”Cancer.” This indicates a mix of generic and specific terms, where stopwords may dilute query specificity.

Improving the retrieval performance

3.1 Query expansion

The query expansion process adds relevant terms to the original queries, improving their ability to retrieve documents. We will train a Word2Vec model using the titles and texts of the documents in order to expand each original query by adding semantically similar terms.

This process enhances the queries with additional information, improving their retrieval potential.

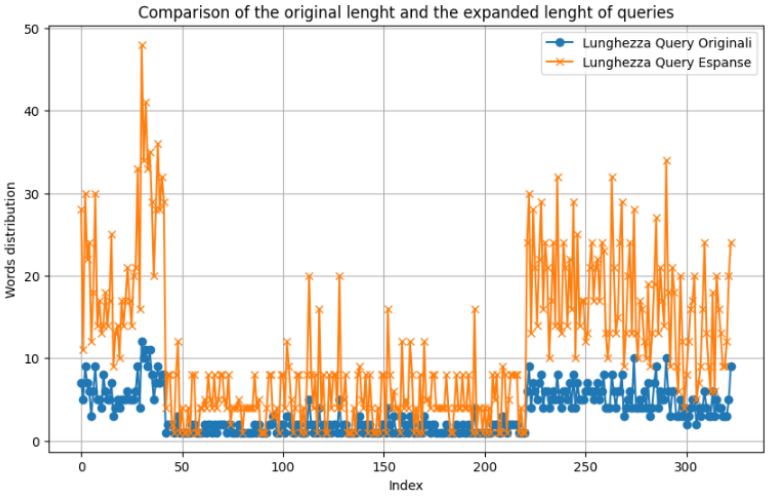

Figura 7: Query lenght now vs. old

This chart illustrates the comparison between the original query lengths and the expanded query lengths.

The blue dots represent the distribution of the original query lengths, which remain relatively consistent and short throughout the dataset, while the orange crosses indicate the expanded query lengths, which show significant variation, as they include additional terms

generated by the word2vec model.

Now we are going to train again every pipeline in order to see if we got any improvements.

Pipeline	Model	MAP	P@10	nDCG@10
1	TF-IDF	0.0994	0.1368	0.1990
1	BM25	0.0994	0.1381	0.1993
2	TF-IDF	0.0985	0.1362	0.1984
2	BM25	0.0989	0.1368	0.1984
3	TF-IDF	0.0994	0.1378	0.1996
3	BM25	0.0994	0.1381	0.1991
4	TF-IDF	0.0725	0.1043	0.1571
4	BM25	0.0695	0.1037	0.1544
5	TF-IDF	0.0986	0.1359	0.1983
5	BM25	0.0990	0.1365	0.1987
6	TF-IDF	0.0924	0.1322	0.1892
6	BM25	0.0950	0.1337	0.1926

Tabella 2: Performance models with expanded queries.

The performance of the expanded queries is significantly inferior to the non-expanded queries across all pipelines. This suggests that the expansion method based on Word2Vec didn't enhance the retrieval performance, possibly introducing irrelevant or noisy terms into the query.

3.2 Neural re-ranking

Given the degraded performance with query expansion, we will implement a neural re-ranking approach to improve the retrieval results using two neural models to re-score the retrieved documents based on their semantic similarity to the query.

Tomaarsen/static-retrieval-mrl-en-v1

The tomaarsen/static-retrieval-mrl-en-v1 model is designed for static retrieval tasks and optimized for English documents. It utilizes a neural architecture to rank documents ef-

fectively by capturing semantic relevance between queries and documents. Its strength lies in its ability to handle large-scale document retrieval scenarios, making it suitable for tasks requiring high precision in reranking pipelines.

sentence-transformers/all-MiniLM-L6-v2

This model is part of the sentence-transformers family and is highly efficient for sentence-level embeddings. It provides a compact and fast representation with only 6 transformer layers, which makes it a popular choice for reranking tasks. It achieves a good trade-off between computational efficiency and semantic understanding, especially for tasks like query expansion or reranking in information retrieval systems.

Tabella 3: Comparison of MAP, P@10, and nDCG@10 across pipelines and models.

Pipeline	**Model**	**MAP**	**P@10**	**nDCG@10**
1	Static Retrieval	0.1392	0.2207	0.3026
1	All MiniLM	0.1551	0.2471	0.3311
2	Static Retrieval	0.1385	0.2207	0.3023
2	All MiniLM	0.1543	0.2471	0.3312
3	Static Retrieval	0.1391	0.2207	0.3026
3	All MiniLM	0.1551	0.2471	0.3311
4	Static Retrieval	0.1065	0.1839	0.2623
4	All MiniLM	0.1179	0.2099	0.2904
5	Static Retrieval	0.1384	0.2207	0.3023
5	All MiniLM	0.1542	0.2471	0.3312
6	Static Retrieval	0.1381	0.2195	0.3008
6	All MiniLM	0.1539	0.2452	0.3289

Comparison with Initial Pipelines:

The all-MiniLM-L6-v2 outperforms static-retrieval-mrl-en-v1 in all pipelines. It provides better results for MAP, P@10, and nDCG@10, showcasing its strong semantic representation capabilities. All MiniLM also consistently outperform TF-IDF and BM25 across

all pipelines, with higher values for MAP, P@10, and nDCG@10.
This highlights the advantage of neural reranking in capturing semantic relevance.

Conclusions

The results show that neural reranking significantly improves the performance of the retrieval pipelines compared to the initial TF-IDF and BM25 approaches. Among the models tested, All MiniLM consistently outperformed Static Retrieval, delivering better scores across all pipelines. However, some pipelines, like Pipeline 4, still show weaker results, indicating room for optimization in preprocessing or document representation. Moving forward, fine-tuning the reranking models on domain-specific data or exploring hybrid approaches could further enhance the system's effectiveness. The query expansion should be fine-tuned to get better results, or maybe should be useful to use different models.