Report Project 1, CS-423 (The Kaggle-Laborers)

Salya Diallo Liam Gibbons Tianshuo Zhang

I. Introduction

This research project tackles the complex challenge of information retrieval (IR) by analyzing an extensive corpus of $268'022$ documents across seven languages — French, English, Spanish, Italian, German, Arabic, and Korean — using a broad set of multilingual queries. To address this, we will explore a variety of methods, including probabilistic models, vector space retrieval, and query expansion techniques.

II. Data pre-processing

A crucial aspect of textual analysis is the preprocessing of our corpus and queries, tailored specifically for each language. The main idea for each language and document is the following: tokenize the document and then stem words that are not part of a predefined stop words list. Stopwords and stemmers were customized per language, mainly using the *SnowballStemmer* and *nltk* stop words.

A. Notations

For the whole report, we will use the following notations:

- D: represent a document containing $|D|$ tokens;
- N: total number of documents for the considered language;
- $f(t, D)$: number of time a token t occurs in D;
- $DF(t)$: number of documents for the considered language containing the token t;

III. Models

Different methods were possible to solve this problem, from supervised to unsupervised learning. We focused on the latest. In the final evaluation of each model, we considered the top 10 most relevant documents per query.

A. Vector Space Retrieval

The basic idea of IR is to use the **TF-IDF** (Term Frequency-Inverse Document Frequency) matrix, combined with the **cosine similarity**, which is what we tried to implement.

The TF-IDF (Term Frequency-Inverse Document Frequency) matrix represents documents as vectors in a high-dimensional space, where each dimension corresponds to a unique token in the vocabulary. This matrix highlights terms that are significant within a document (through term frequency, TF) while down-weighting common terms across the corpus (through inverse document frequency, IDF):

$$tfidf(t, D) = tf(t, D) \cdot idf(t, D) = \frac{f(t, D)}{|D|} \cdot log_2(\frac{N}{1 + DF(t)})$$

Cosine similarity measures the angle between two vectors, allowing IR systems to evaluate the similarity between a query vector and document vectors. By finding documents whose vectors have a high cosine similarity with the query vector, an IR system can rank documents by relevance.

$$sim(\vec{q}, \vec{D}) = \frac{\sum_{i=1}^{n} q_i \cdot D_i}{(\sum_{i=1}^{n} q_i^2)^{1/2} \cdot (\sum_{i=1}^{n} D_i^2)^{1/2}}$$

where n is the number of unique tokens in the document D, and q_i, D_i are the TF-IDF of token i in the query and document vectors, respectively.

B. Probabilistic Model

BM25 (Best Matching 25) is a ranking function based on the probabilistic retrieval framework and is one of the most commonly used probabilistic model. It's goal is to rank the document based on their relevance to a given query.

We defined it for each of the seven languages as follows: For a vectorized query $\vec{q}$ that has tokens $q_1, \ldots, q_k$ and for a document D, we compute the following:

$$score(\vec{q}, D) = \sum_{i=1}^{k} IDF(q_i) \cdot \frac{f(q_i, D)(k+1)}{f(q_i, D) + k(1 - b + b\frac{|D|}{avgLength})}$$

where $avgLength$ is the average document length over the whole corpus for the corresponding language, $k \in [1.2, 2.0]$ and b are parameters adjusted to optimize results. To fine-tune k and b, we experimented with different values obtaining as best parameters $k = 1.25$ and $b = 0.75$.

We used $IDF(q_i) = log_2(\frac{N - DF(q_i) + 0.5}{Df(q_i) + 0.5} + 1)$. We added 0.5 to the denominator and numerator of the fraction, and 1 inside the logarithm in order to avoid dividing by zero and to stabilize scores for common terms. Using the IDF values permit to emphasize rare terms.

To further optimize runtime, given the large number of queries per language, we reformulated the scoring computations as matrix multiplications.

C. Query Expansion

The main idea behind the **query expansion** method is that after an initial vectorization of the query, it adds terms to the query in order to extend its scope and lead to the potential of more relevant documents being found, which results in an adjustment of the query vector. In this project, the query expansion was done using **Rocchio's algorithm**.

Rocchio's algorithm works by essentially optimizing the query by separating relevant documents from non-relevant documents. It does so by taking the centroid of the relevant documents as well as that of the non-relevant documents and then uses the following equation: $\vec{q}_{opt} = \mu(D_r) + [\mu(D_r) - \mu(D_n)]$ where $\vec{q}_{opt}$ is the optimal query, $\mu(D_r)$ is the centroid of relevant documents and $\mu(D_n)$ is the centroid of non-relevant documents. In order to approximate this, the equation that was actually used in this project is:

$$\vec{q}_{approx} = \alpha\vec{q} + \frac{\beta}{|D_r|}\sum_{\vec{d}_j \in D_r}\vec{d}_j - \frac{\gamma}{R \setminus D_r}\sum_{\vec{d}_j \notin D_r}\vec{d}_j$$

where $\vec{q}_{approx}$ is the approximation of the optimal query vector, $\vec{q}$ is the original query, D_r is the set of relevant documents, $\vec{d}_j$ is the vector representing document j, D_r is the set of relevant documents, R is the set of results from the query, and α, β, and γ are tuning parameters that were set at 1, 0.75, and 0.15 respectively.

An initial vectorization was done through TF-IDF (mentioned earlier in section III-A) in order to generate a set of results that the query can then be expanded from. Going from there leads to the split between relevant and non-relevant documents, which can be done through either user feedback or pseudo-relevance feedback. Since there are no users per se, pseudo-relevance feedback was used. From the initial set of 10 relevant documents, it is then taken that the top five documents are most relevant and implement the feedback as such.

D. Other possible models

By vectorizing our queries and corpus, we had the option to implement supervised learning, such as SVM or random forest. However, we prioritized refining the performance of our existing models over starting new approaches, aiming to maximize the effectiveness of our current setup.

IV. Results

In the end, we found that the best method was **BM25**. The table below helps to illustrate that by showing the accuracy of the respective models in regards to each language:

TABLE I: Accuracy table measured for each model for the Test set as a percentage

Languages	BM25	TF-IDF	Query Exp
English	66	23	26
French	85	54.5	53.5
German	61.5	35.5	29.5
Italian	76.5	40.5	41
Spanish	93	57	58.5
Arabic	72.5	39.5	35
Korean	64.5	18	34.5

For accuracy, we measured the presence of the actual relevant document within the top 10 results for each query and divided by the total query count per language.

We see that BM25 consistently outperformed other methods across most languages, with the highest scores in Spanish and French. Query Expansion provided moderate improvements, especially in Spanish, while TF-IDF struggled with languages like Korean. Finally, we obtained a best score of 0.68688 on Kaggle when using BM25 and our pre-processing defined above.

BM25 outperforms traditional ranking methods, like TF-IDF, due to several key improvements in IR scoring:

- Non-linear Term Frequency Scaling: Unlike TF-IDF's linear approach, BM25 balances term frequency more effectively by applying a non-linear transformation. This prevents extremely frequent terms from dominating the relevance score disproportionately.
- Document Length Normalization: BM25 explicitly accounts for document length, penalizing longer documents that may inflate term frequency. This normalization ensures that document length does not overly influence the relevance ranking.
- Parametric Flexibility: The algorithm introduces two tunable parameters: k controls the term frequency sensitivity and b adjusts document length normalization, which makes it more adaptable to varying corpus characteristics.

V. Issues encountered

The main pre-processing challenge was handling the Korean language, as finding an efficient stemmer and stopword list was harder. We ultimately used the *Kiwi* library for stemming and *Spacy*'s stopwords. Moreover, due to the large volume of English documents, we also partitioned the English corpus into 10 parts for efficient scoring.

When it came to defining the vocabulary, we had two options: to consider either only the corpus or both the corpus and the queries. After careful consideration, we chose to use only the vocabulary from the corpus.

One of the main issue for the BM25 scores computations was how to compute the IDF values. Our solution involved using $CountVectorizer$ from the $sklearn.feature\_extraction.text$ library and *fit_transform* it on the corpus texts to obtain the document-term matrix.

We also needed to determine the best approach to manage the large vocabulary size, either by limiting each document to a specific character count or setting a $max\_features$ parameter in our vectorizer. Ultimately, we used a combination of both: capping each document at $30'000$ characters and setting $max\_features = 20,000$. Without this character limit our kernel would crash, for example for French it did if max_features exceeded $8'000$. On the other hand, using only the character limit led to excessively high runtime. This created a balance between both methods, which we fine-tuned based on the assumption that the core content of each document typically appears at the beginning rather than at the end.

VI. Conclusion

Information Retrieval is a fundamental task in natural language processing, with various methods developed to address its complexities. In this project, we focused on three key approaches: TF-IDF with cosine similarity, BM25, and Query Expansion.

Throughout the implementation process, we encountered several challenges at each stage, from preprocessing to model optimization. These obstacles led us to refine our methods and tailor our approach to the nuances of our multilingual corpus.

Ultimately, the BM25 probabilistic model proved to be the most effective, yielding satisfying results across languages and query variations. This project not only highlights the effectiveness of BM25 in multilingual IR tasks but also underscores the importance of methodical tuning and adaptive strategies in obtaining reliable retrieval outcomes.