Advanced Book Search System: Leveraging Reader Reviews for Efficient Information Retrieval

ANA SOFIA TEIXEIRA, RICARDO NUNES RIBEIRO, SÉRGIO MANUEL CARVALHAIS, and TOMÁS AGANTE MARTINS

ABSTRACT

This project centres on sourcing, refining, analysing, and querying data sets pertaining to books. The primary objective is to construct a search system capable of addressing intricate queries that surpass the limitations of existing book-related platforms. Initial phases involve exploration of data sources to extract comprehensive book information and reader reviews. The extracted data undergoes refinement processes detailed in this paper, followed by exploratory analyses. Subsequently, our focus shifts towards addressing specific retrieval tasks for books by defining document collections and establishing indexing mechanisms. The system's effectiveness in answering predetermined book-related queries is evaluated based on precision and recall metrics.

KEYWORDS

Book Reviews, data collection, data source, data preparation, information retrieval, search engine.

1 INTRODUCTION

While existing book search systems rely on conventional criteria, our project introduces a novel technical solution. We aim to develop a search engine that enables users to search books based on reader reviews. This article serves as a comprehensive overview of the development phase of our project, focusing on the technical aspects. Our goal is to create a technically robust book search system that enhances the user experience by considering reader reviews and detailed content information.

We begin by sourcing and curating relevant data sets, assess data quality and authority, and conduct exploratory data analysis. Additionally, we document our data processing pipeline. Our work involves characterising data set properties, defining the conceptual model for our book data, and meticulously identifying and characterising information needs for the rest of the development of the project.

After the foundational phase, our project advanced to information retrieval, employing specialised tools and methodologies for efficient data set access. This stage involved developing an information retrieval tool, document analysis, and constructing an index. The emphasis was on refining query execution to enhance accurate information retrieval.

Subsequently, we rigorously evaluated our information retrieval systems, conducting relevance assessments using diverse setups and metrics. We analysed system performance across various configurations and metrics to gauge effectiveness.

Authors' address: Ana Sofia Teixeira, up201806629@fe.up.pt; Ricardo Nunes Ribeiro, up202310095@fe.up.pt; Sérgio Manuel Carvalhais, up202007544@fe.up.pt; Tomás Agante Martins, up201704976@fe.up.pt.

2 DATA SOURCES

In our quest to obtain data aligned with our research objectives, Kaggle [1] emerged as the ideal source. The data set of interest, 'Amazon Books Reviews' [2], comprises two key files. The first file details 212,404 distinct books, sourced from the Google Books API. The second file contains a treasure trove of Amazon reviews, totalling 3 million entries from May 1996 to July 2014. The book details file encompasses essential information like title, authors, publisher, and user ratings. In contrast, the reviews file provides insights into book titles, pricing, user details, review scores, and more.

What's noteworthy is that this data set is licensed under Creative Commons Zero (CC0), effectively dedicating it to the public domain and allowing unfettered usage and modification in accordance with copyright law. This open data approach encourages collaboration and innovation within the research community.

3 DATA COLLECTION AND PREPARATION

3.1 Data Preparation Pipeline

For the data collection phase, we first downloaded a .zip folder from the Kaggle website [2]. After having the folder locally on our devices, we unziped it and inside found the two files we required: *Books_ratings.csv* and *books_data.csv*.

From the name of both files, we quickly understood what was contained in them. the first one compiled all the reviews and ratings from the users associated with a given book. We became aware that there were 3 million entries of reviews in this file. The second one was more focused on storing the individual book information (212404 records of books), among them, their authorship and categorisation.

After having collected the data from the data source, we proceeded with some tasks in order to prepare our data for the next phases according to the pipeline depicted in Figure 1.

In the data exploration phase, and through the use of the Python Pandas library, we were able to collect some initial data regarding the data sets. For instance, the column names of both tables; the number of missing values; the number of unique values in the columns (e.g.: there were 10883 unique values in the column 'categories' in *books_data.csv*). Later on, in the Data Cleaning phase, we decided that, in the context of our project, there was no need to store some of the information present in the files. Moreover, the files' size was considerably large, therefore, to facilitate the computing operations in our personal computers, we deleted those columns we deemed unnecessary: "Id", "Price", "User_id", "profileName", "review/helpfulness", "review/score" and "review/time" from *Book_rating.csv*; and "image", "previewLink", "infoLink" and "ratingsCount" from *books_data.csv*. Later on, we renamed the remaining columns in *Books_rating.csv* according to the following scheme:

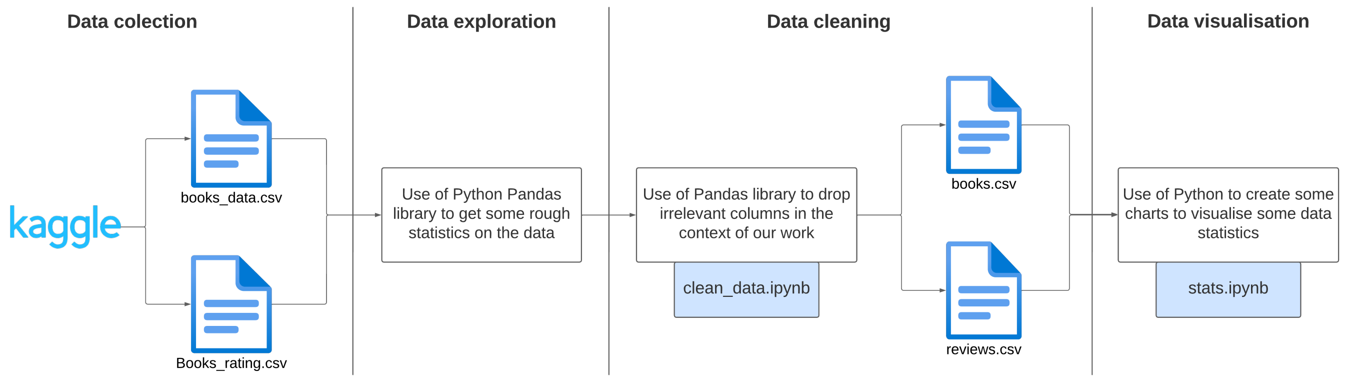
Data colection
Data exploration
Data cleaning
Data visualisation
kaggle
books_data.csv
Books_rating.csv
Use of Python Pandas library to get some rough statistics on the data
Use of Pandas library to drop irrelevant columns in the the context of our work
clean_data.ipynb
books.csv
reviews.csv
Use of Python to create some charts to visualise some data statistics
stats.ipynb

Fig. 1. Data Pipeline

"Title" ↦ "book_title", "review/summary" ↦ "summary", "review/-text" ↦ "text"; and "Title" ↦ "book_title", leaving all the others with the same name. As for *books_data.csv*, we renamed "Title" to "book_title" and left all others unchanged ("description", "authors", "publisher", "publishedDate", 'categories'). In the end of this phase, we wrote the new tables into new .csv files for organisation purposes, being them *reviews.csv* and *books.csv*.

Lastly, using a Jupyter Notebook, we created some plots in order to have a deeper look at the data at hands (cf. 3.3 Data Characterisation). Figure 1 represents a formal overview of the entire process.

3.2 Conceptual Data Domain Model

Having the data in such a way we could better handle and analyse, we were able to come up with a Conceptual Data Model. The model (Figure 2) identifies the entities we consider relevant in the context of our problem as well as their respective attributes. We point out the following:

- *book* entity
  - book_title: stores the name of the book
- *review* entity
  - summary: holds a summary/preview of the full review
  - text: stores the full review written by the user
- *author* entity
  - author: stores the name of the author
- *category* entity
  - category: records the name of the category the book falls into
- *publisher* entity
  - publisher: holds the name of the company who responsible for publishing the book

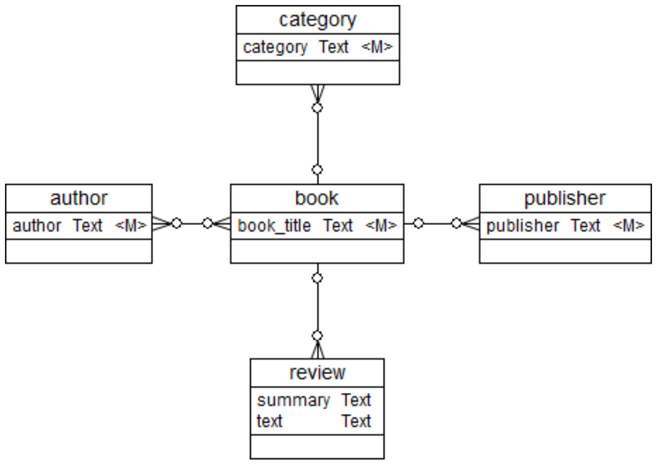

Fig. 2. Conceptual Data Model

4 DATASET CHARACTERISATION

In terms of our collection, the .csv files we obtained were extensive. They contained information regarding full-text reviews with reference to the books reviewed and also every information imaginable about the book, like Amazon internal reference number, no. of pages or the full book. To ease out our task and for the presentation of our topic to be clearer, we produced some charts using the python scripts in our pipeline.

Given that the main focus of this project is regarded with user reviews, we decided to better analyse this field in our data set. To do so, we created some charts that reflect the characteristics we consider to be more relevant.

4.1 Lexical Diversity

The chart in Figure 3 gives an overall idea of how many different words reviewers use in their reviews. From the chart, we highlight that more than 1.2 million reviews (around 40% of the whole data set) have a lexical diversity ratio between 65% and 75%, meaning most words in the review text field were only written once.

4.2 Most Common Words

Another useful analysis we found useful was to explore what were the most common words that were written in the whole data set (Figure 4). In line with out expectations, "book" is, in fact, the most frequent word in the database - more than 5 million repetitions

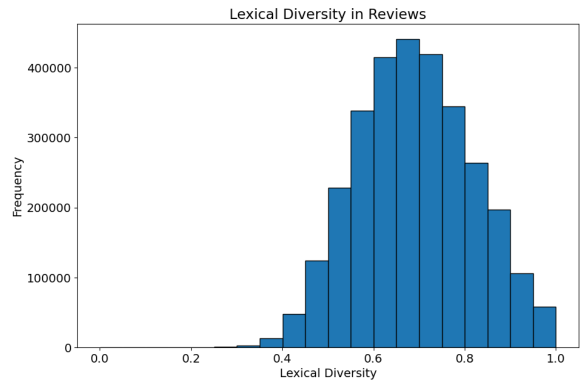

Fig. 3. Reviews' Lexical Diversity Distribution

-, followed by "read", "one", "story", and, closing the top five most common words, "like".

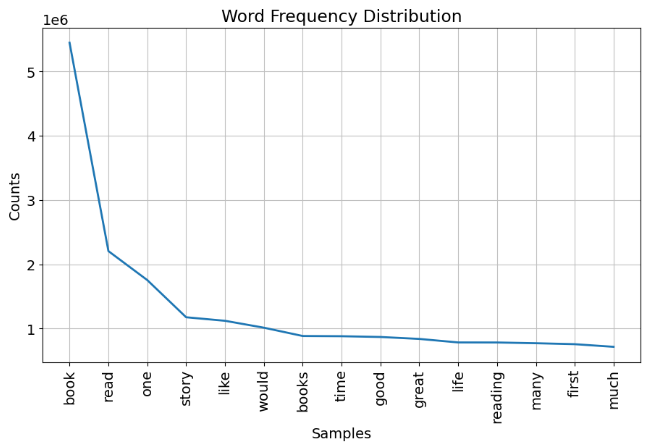

Fig. 4. Most Common Words

4.3 Word Count Distribution in Reviews

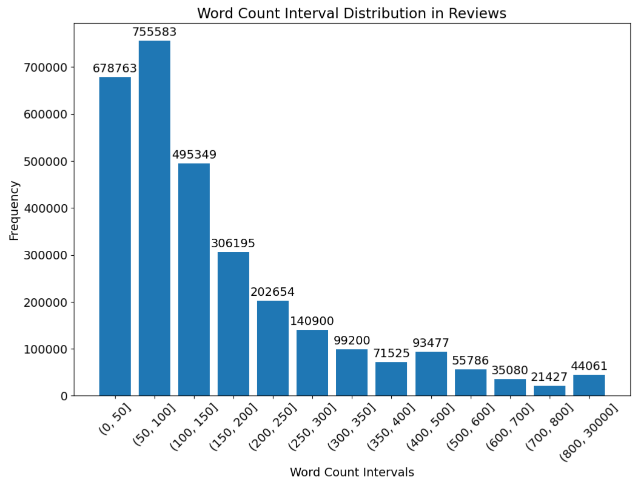

Fig. 5. Word Count Interval Distribution in Reviews

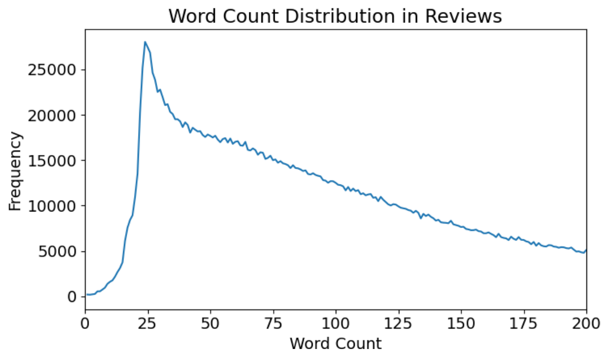

Fig. 6. Word Count Distribution in Reviews

As the Figure 5 chart demonstrates, most reviews contain between 50 and 100 words (around 25% of all reviews). A closer look at the first four intervals (Figure 6) allows us to see that the most frequent number of words in reviews is around 25 words. However, it came to our attention the fact that there is a significant amount of reviews that contain more than 400 words.

4.4 Number of Sentences Distribution in Reviews

Lastly, in Figure 7, we plotted the distribution of the number of sentences written in the reviews. As the figure shows, around 14% of the reviews are made up of roughly 3 sentences, being this the most common sentence size in the data set.

5 PROSPECTIVE SEARCH TASKS

5.1 Description

The objective of the project is to provide the user with an advanced book search so that they can find the reading they are looking

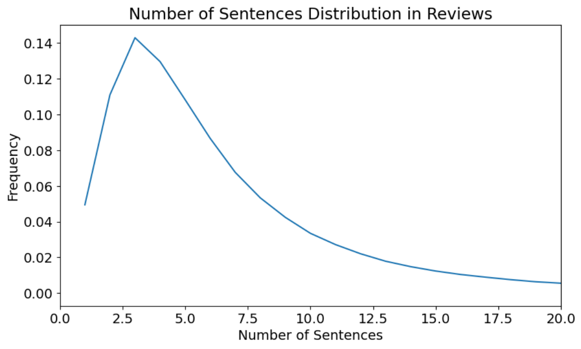

Fig. 7. Number of Sentences Distribution in Reviews

for based on several factors. As such, the user must be able to find books not only by their title, but also through their author, publisher, category, theme, reviews and ratings, all combined. An example of a possible search would be science fiction books that do not address the topic of space.

5.2 Information needs

To perform this search, all the information contained in the files obtained after cleaning the data from Kaggle website will be necessary. Thus, we will have at our disposal technical information about the books (such as title, author, publisher, publication date, category, description) contained in the books.csv file and also reviews and ratings of these in the reviews.csv file.

Based on this, some examples of these queries are:

- Retrieve books that talk about artificial intelligence.
- Retrieve books that support fights against racism.
- Retrieve novels set in post-apocalyptic worlds.
- Retrieve books related to American Civil War.

6 INFORMATION RETRIEVAL

In the pursuit of effective information retrieval for our book search system, the choice of the right tool is paramount. This section delves into the tool we've employed and the methodologies adopted to ensure accurate and efficient retrieval of information from our extensive data set.

6.1 Information Retrieval Tool

We opted for Apache Solr [4] as our information retrieval tool due to its robust features and capabilities in handling large datasets. Apache Solr is an open-source search platform built on Apache Lucene, offering advanced search and indexing functionalities. Some of the key features and advantages of using Solr in our project include:

- **Full-Text Search:** Solr provides powerful full-text search capabilities, allowing us to search through extensive book data efficiently.
- **Customizable Ranking:** Solr allows the customization of ranking strategies, ensuring that search results are relevant and aligned with user preferences.
- **Rich Query Language:** Solr offers a powerful and expressive query language that supports complex queries, enhancing our ability to retrieve specific information from the dataset.
- **Schema Flexibility:** Solr's schema design allows us to tailor the data structure to our specific requirements, accommodating various fields and data types present in our book dataset.

6.2 Document Analysis

Our collection is made up of several documents that, in our context, represent a book. It serves as the focal point for information retrieval, containing a variety of fields that capture diverse aspects of a book's metadata and content.

To enhance the search experience and provide users with comprehensive information about books rather than isolated reviews, we adopted a strategy to aggregate both the textual reviews and summaries into their corresponding books. This aggregation process ensures that queries return books as primary results, encapsulating the collective insights from both reviews and summaries. By combining these textual elements, our retrieval system aims to present a holistic view of each book, allowing users to make informed decisions based on a consolidated representation of reviews and content summaries.

6.3 Index Building Process

In the index building process, we defined a schema with specific filters [6] and tokenizers [7] to enhance the accuracy of our information retrieval system. Based on the characteristics of our data set, where most fields are text based (only the published year and the unique identifier of the book are numbers) we decided to create our custom field types as follows:

1. Name: text
Solr Field Type: TextField
Index Analyzer:

- Tokenizer:
  - StandardTokenizer
- Filters:
  - ASCIIFolding
  - LowerCase

Query Analyzer:

- Tokenizer:
  - StandardTokenizer
- Filters:
  - ASCIIFolding
  - LowerCase

2. Name: int
Solr Field Type: TrieIntField
Index Analyzer: N/A
Query Analyzer: N/A

The "text" field type in Solr incorporates ASCII folding and lowercasing during both indexing and query processing [3], enhancing the robustness of text-based searches. This configuration ensures that search operations are case-insensitive and can handle diacritics and special characters effectively. Additionally, the "text" field type utilises the standard tokenizer during both indexing and query processing. The standard tokenizer breaks the text into words, removing punctuation and providing a foundational step in the text analysis process. Together with ASCIIFolding and LowerCase filters, the "text" field type in Solr is well-equipped for handling diverse textual data in a search context.

Bellow is a table with the matching between the field in each 'book' document and the custom created field types:

Field	Type	Indexed	Multivalued
book_id	int	true	false
book_title	text	true	false
description	text	true	false
authors	text	true	true
publisher	text	true	false
categories	text	true	true
publishedYear	int	true	false
reviews	text	true	true
summary	text	true	false
text	text	true	false

Table 1. Solr Schema Fields

The Solr schema, outlined in Table 1, defines the important details for each book in our system. Despite being a space consuming decision, we opted for the indexation of all fields as it would, on the other hand, allow for faster searching processes. The 'Multivalued' column specifies if the field is an array with more than one value inside it or not.

6.4 Query Execution

After successfully indexing our collection and refining our information needs, we formulated specific queries to address distinct aspects of our search requirements. The choice of queries aligns with our defined information needs, aiming to test the effectiveness and relevance of our information retrieval system. Each query is tailored to retrieve books based on a particular theme or topic.

Here are the formulated queries corresponding to our redefined information needs:

1. **Retrieve books that talk about artificial intelligence:**
   `q = artificial intelligence`

2. **Retrieve books that support fights against racism:**
   `q = fight fights against racism discrimination`

3. **Retrieve novels set in post-apocalyptic worlds:**
   `q = novel AND post-apocalyptic world`

4. **Retrieve books related to the American Civil War:**
   `q = "american civil war"`

To enhance the precision of our queries, we utilised the Extended DisMax (eDisMax) query parser (`defType=edismax`) [5]. The eDisMax query parser in Solr is an extended version of the DisMax parser, offering enhanced features for better control over query parameters. It allows for a more flexible and customise approach to query parsing, enabling us to specify multiple fields for querying (`qf`) and assigning different weights to each field. Utilising the eDisMax query parser ensures that the retrieved books closely align with the information needs, providing a more effective and relevant search experience.

The parameters `q` and `qf` in the eDisMax setup play a crucial role. The `q` parameter specifies the user's query, and the `qf` parameter designates the fields to be queried and their respective weights. By adjusting these parameters, we can fine-tune the search process, considering the importance of each field in the retrieval process.

7 EVALUATION

In this section, we evaluate the performance of our information retrieval systems, considering different retrieval setups and metrics. We implemented two distinct retrieval configurations, the Base System and the Enhanced System.

The evaluation process encompasses a detailed relevance assessment, involving manual scrutiny of each retrieved document to ensure alignment with the specified queries.

For a comprehensive evaluation, we employ key metrics, including Average Precision, Precision @ 10, and Precision-Recall Curve, to gauge the effectiveness of our system in meeting the defined information needs. The subsequent sections provide a detailed analysis of the results obtained for each information need, offering insights into the precision and relevance of the retrieved documents.

7.1 Relevance Evaluation Process

Relevance evaluation involved a demanding process, starting with the retrieval of the first 40 documents from the Solr interface based on the specified queries. Each document was then manually analysed to assess its relevance in the scope of the query and the defined information needs. This scrutiny considered various factors, including the title and the content of reviews, summaries, and book descriptions.

To supplement the manual evaluation, external sources such as Google were consulted to gather additional context and detailed information, especially in cases where the reviews or book descriptions were ambiguous or lacked clarity. The combination of manual analysis and external validation ensured a thorough and reliable assessment of the relevance and precision of the retrieved documents.

7.2 Retrieval Setups

For our information retrieval system, we implemented two distinct retrieval setups to evaluate their impact on the search results:

Base System:

```
qf = book_title reviews.text reviews.summary
description
```

In the base system, we assigned equal weights to key fields, including book title, reviews, summaries, and descriptions. This setup aimed to provide a balanced approach to document scoring, considering various aspects of a book's metadata and content.

Enhanced System:

```
qf = book_title^2 reviews.text^2 reviews.summary
description
```

Both systems have as backbone the same schema (as detailed in Section 6.3). However, in the enhanced system, we introduced different weightings to the book title and reviews, placing higher importance on these fields by using squared weights (^2). This setup was designed to amplify the influence of textual content, specifically book titles and reviews, in the scoring process. The goal was to investigate whether emphasising certain fields would lead to more precise and relevant search results.

These retrieval setups served as the foundation for our evaluation, allowing us to compare the performance and effectiveness of the base and enhanced systems in our problem context.

7.3 Results Evaluation Overview

In our evaluation, we employ 3 key metrics to assess the performance of our information retrieval system. These metrics offer quantitative insights into the system's effectiveness in meeting user information needs. Here's a brief overview of the metrics used:

- **Average Precision (AP):** This metric computes the average precision across all relevant documents retrieved by the system, offering an aggregate measure of precision. It indicates the system's ability to consistently rank relevant documents higher in the result set.
- **Precision @ 10 (P@10):** Precision at 10 evaluates the precision of the system by focusing on the top 10 retrieved documents. This metric provides insight into the system's immediate performance in presenting relevant results within the initial subset of the ranking.
- **Precision-Recall Curve:** The precision-recall curve depicts the relationship between precision and recall across various decision thresholds. It serves as a graphical representation of the system's effectiveness in balancing precision and recall, offering insights into its performance under different retrieval scenarios.

These metrics collectively offer a comprehensive evaluation of our information retrieval system, allowing us to gauge its precision, relevance, and overall effectiveness in fulfilling user information needs.

- **Information Need 1:** Books about Artificial Intelligence
  Query:

  ```
  q = artificial intelligence
  ```

  Metrics Comparison:

System	Average Precision	Precision at 10 (P@10)
Base System	0.759010	0.700000
Enhanced System	0.858932	0.900000

Table 2. Metrics Comparison for Books about Artificial Intelligence

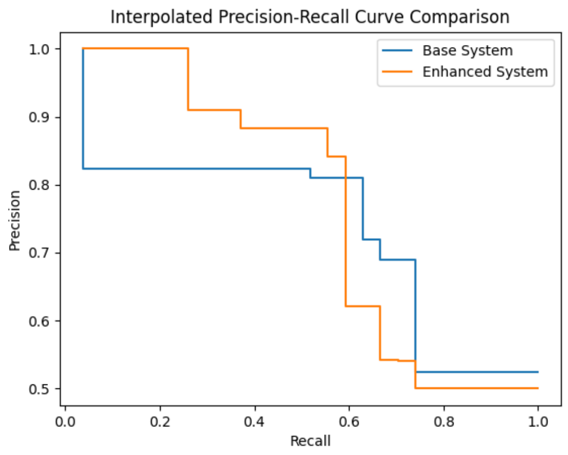

Fig. 8. Precision-Recall Curve for Books about Artificial Intelligence

Results Description: The enhanced system does better than the base system in retrieving books related to artificial intelligence, as indicated by a higher average precision. The enhanced system can maintain precision at 1.0 for higher recall values than the base system before dropping from that level. However, at a recall of about 0.60, the precision of the enhanced system drops sharply to levels lower than those of the base system. The precision-recall curve in Fig. 8 is a visual illustration of this description.

- **Information Need 2:** Books that support fights against racism
  Query:

  ```
  q = fight fights against racism discrimination
  ```

  Metrics Comparison:

System	Average Precision	Precision at 10 (P@10)
Base System	0.658527	0.600000
Enhanced System	0.707906	0.700000

Table 3. Metrics Comparison for Books that support fights against racism

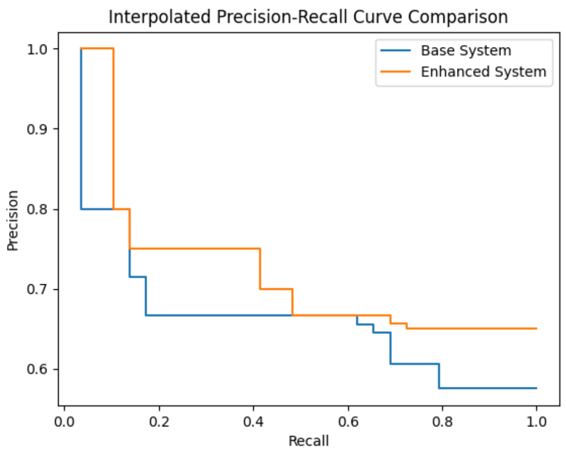

Fig. 9. Precision-Recall Curve for Books that support fights against racism

Results Description: The enhanced system outperforms the base system in retrieving books related to to fights against racism, as indicated by higher Average Precision. The enhance system is able to maintain precision equal to 1.0 for overall values of recall than the base system. However, at around the [0.1;0.15] and [0.50;0.60] intervals of recall values, the precision of the enhanced system equals that of the base system. The Precision-Recall Curve in Fig. 9 visually illustrates this description.

- **Information Need 3:** Novels in Post-Apocalyptic Worlds
Query:

```
q = novel AND post-apocalyptic
```

Metrics Comparison:

System	Average Precision	Precision at 10 (P@10)
Base System	0.821265	0.700000
Enhanced System	0.759569	0.700000

Table 4. Metrics Comparison for Novels in Post-Apocalyptic Worlds

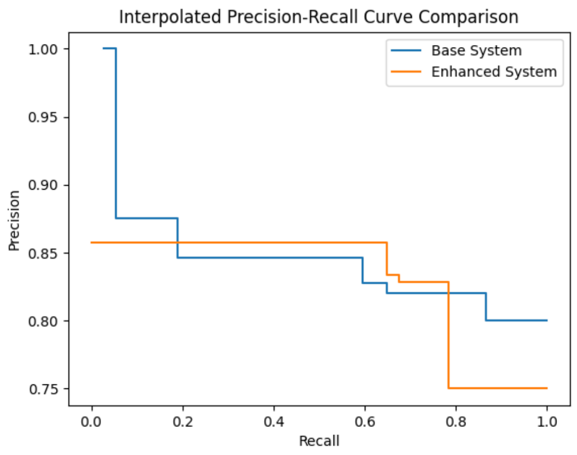

Fig. 10. Precision-Recall Curve for Novels in Post-Apocalyptic Worlds

Results Description: The base system's precision sharply declines at around 0.05 recall, falling below the precision values of the enhanced system. The enhanced system notably maintains higher precision for increased recall values compared to the base system. This trend is visually depicted in Fig. 10.

- **Information Need 4:** Books about the American Civil War
Query:

```
q = "american civil war"
```

Metrics Comparison:

System	Average Precision	Precision at 10 (P@10)
Base System	0.982544	1.000000
Enhanced System	0.968378	1.000000

Table 5. Metrics Comparison for Books about the American Civil War

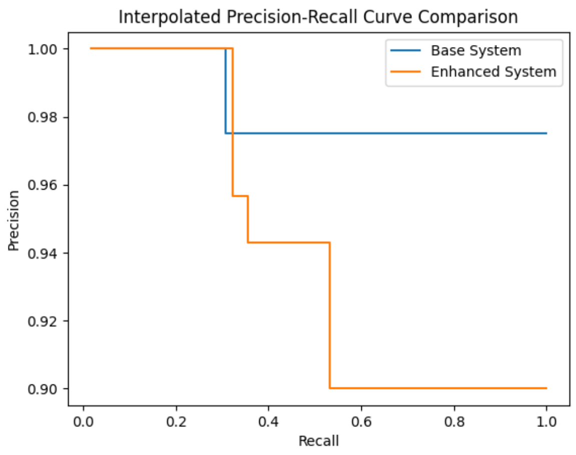

Fig. 11. Precision-Recall Curve for Books about the American Civil War

Results Description: Considering that the query performs an exact search, as evidenced by the quotation marks (q = "american civil war"), this example presents more accurate results. However, in this case, we notice that the base system is better in the retrieval of books related to the American Civil War than the enhanced system, since its Average Precision is higher, although the Precision @ 10 values are the same. This happens because, in some reviews, critics mention the "American Civil War" to locate the book in a timeline or to comment on something related to that topic. The *boost*, being applied to *reviews.text*, ends up giving priority to these results in the search, despite them not being directly related to what is intended. We can clearly see in the Precision-Recall Curve in Fig. 11 the differences between both performances.

8 CONCLUSION

In the initial sections of the paper, regarding the Data Preparation phase, the processing pipeline is documented, as well as all the tools used, showing all the steps that led to the creation of a complete and coherent data set about books and their reviews. Some graphics and statistics are also present to better visualise the data. All of the goals for this phase were accomplished given that there is a better understanding of the chosen domain.

The creation of the information retrieval system involved selecting the Apache Solr search platform, which provides robust full-text search capabilities. The system's efficacy was assessed using two configurations: the Base System and the Enhanced System, each emphasising unique data aspects. Evaluation metrics including average precision, precision at 10 and precision-recall curve were used to provide a comprehensive assessment.

The findings indicated that the Enhanced System, which allocated greater weightings to book titles and reviews, surpassed the Base System in some scenarios. In other situations, the Base System demonstrated superior precision. Therefore, this emphasises the significance of balancing multiple factors in information retrieval.

So far, our project has not only tackled the technical challenges of developing an advanced book search system, but also delved into the complexities of data preparation and schema design. The evaluation findings have provided valuable input into the trade-offs and considerations when designing information retrieval systems for complex data sets.

REFERENCES

[1] [n.d.] *Datasets.* URL: https://www.kaggle.com/datasets.
[2] Mohamed Bekheet. *Amazon Books Reviews.* URL: https://www.kaggle.com/datasets/mohamedbakhet/amazon-books-reviews. (accessed: 29.09.2023).
[3] Apache Software Foundation. *Analyzers.* URL: https://solr.apache.org/guide/solr/9_3/indexing-guide/analyzers.html. (last accessed: 11.11.2023).
[4] Apache Software Foundation. *Apache Solr Reference Guide.* URL: https://solr.apache.org/guide/solr/9_3/index.html. (last accessed: 14.11.2023).
[5] Apache Software Foundation. *Extended DisMax (eDisMax) Query Parser.* URL: https://solr.apache.org/guide/solr/9_3/query-guide/edismax-query-parser.html. (last accessed: 11.11.2023).
[6] Apache Software Foundation. *Filters.* URL: https://solr.apache.org/guide/solr/9_3/indexing-guide/filters.html. (last accessed: 11.11.2023).
[7] Apache Software Foundation. *Tokenizers.* URL: https://solr.apache.org/guide/solr/9_3/indexing-guide/tokenizers.html. (last accessed: 11.11.2023).