Goodreads Books and Reviews

Ana Cruz, Ines Quarteu, and Filipe Recharte

FEUP, {up201806460, up201806279, up201806743}@up.pt

December 13, 2021

Abstract

Given the growing amount of available data and the possibility of processing a large amount of information, indexing and searching effectively is a growing concern in today's information systems. In the present paper, the object of study was the Books and Reviews from Goodreads, as well as additional data on Reviews, Genres, and Languages gathered by web scraping the Goodreads site. After a process of refinement, where the data was cleaned and normalized, the dataset was characterized, which made it clear that its study may be an interesting exercise, taking into consideration the dataset analysis that showed the diversification of the data through most of its entries. Additionally, after resorting to Solr to index the documents, the evaluation of various system configurations demonstrated that the information retrieval system's capacity is remarkably dependent on the chosen indexing operations and of the use of the weighting filter.

Keywords Books, Dataset refining, Data retrieval, Data processing pipelines, Python

Introduction

Goodreads is a social cataloging website that presents a large collection of information on books, annotations, quotes as well as reviews written by individual users.
The aim of this project is to construct an information search system for the data retrieved from Goodreads, specifically the books' information and their reviews.
In this report, the work on data collection and preparation, information querying and retrieval, and retrieval evaluation relative to the books and their reviews will be presented. Initially, Part I details the datasets being used, as their origin and collection method, as well as the refinement process of each of the collections, are explained thoroughly. Additionally, Part II presents the processes used to index and search the defined information needs in the chosen datasets. Finally, a reflection on this paper's conclusions is featured in order to clarify the purpose and the results of this process.

Part I
Data Preparation

1 Dataset Preparation

The retrieval of the data was accomplished by scripts in Python using the Pandas tool [1]. For the refinement, it was used Python and the CSV library [2].

1.1 Books

The Books dataset was retrieved from the **Goodreads-Book-Datasets-With-User-Rating-2M** [3]. This dataset is a subset of the existing books in the Goodreads [4] website and was fetched from kaggle. It was composed of 30 files, 23 with books and some of its attributes, ISBN and author for example, and the other 7 with reviews of some of these books. The files were in CSV format and had an average of 10,000 entries and an average of 20 columns each.
Regarding the refinement, since the files of the reviews did not contain actual reviews but were instead random sentences, probably because the dataset's author didn't think the reviews were much relevant to their work, these files were discarded. Some of the attributes files were incomplete or had poor quality, therefore were also discarded. In the end, 4 files were chosen to be more refined. These files were chosen because they appeared to be the most complete ones. The columns *RatingDist5*, *RatingDist4*, *RatingDist3*, *RatingDist2*, *RatingDist1*, *RatingDistTotal* were eliminated, since there was no use to them; *language*, since most entries were null; and *CountOfTextReviews*, since it was not relevant. Every entry that had the

Description null was eliminated, as for the remaining descriptions, the HTML tags and the new lines were removed. After this, the 4 files were merged to compose a single CSV file. After doing the scraping that is described in the following subsections, the final refinement consisted of removing the book's entries that had no genre or language. The resulting dataset was **books.csv**, with 100,269 entries and 12 columns.

1.2 Reviews

The reviews dataset was built by scraping the Goodreads website with the book's Ids from the dataset **books.csv**. This dataset has 559,874 entries and 2 columns.
Regarding the refinement, some entries contained Arabic characters, which were removed. The resulting dataset was **reviews.csv**.

1.3 Genres

The genres dataset was built by scraping the Goodreads website with the book's Ids from the dataset **books.csv**. This dataset has 299,782 entries and 2 columns.
Regarding the refinement, since Goodreads does not have actual genres but instead shelves created by users, some filters were applied to this dataset. Entries with numbers and starting with down-case letters were removed. The resulting dataset was **genres.csv**.
After removing the books that had no language associated, the **books.csv** was used to remove rows from this dataset that were not used anymore.

1.4 Language

The language dataset was built by scraping the Goodreads website with the book's Ids from the dataset **books.csv**. This dataset has 100,269 entries and 2 columns.
Regarding the refinement, for this dataset, it wasn't needed. The resulting dataset was **languages.csv**.
After removing the books that had no genre associated, the **books.csv** was used to remove rows from this dataset that were not used anymore.

1.5 Authors

The authors' dataset was built by removing the authors' column from the dataset **books.csv**. This dataset has 100,269 entries and 2 columns.
Regarding the refinement, for this dataset, it wasn't needed. The resulting dataset was **authors.csv**.

2 Data processing Pipeline

To process the data, pipeline below was used (Figure 1). The process starts by downloading the **Goodreads-Book-Datasets-With-User-Rating-2M** dataset and selecting 4 CSV files. With Python scripts the refinement such as removing irrelevant columns and entries, and separating the data was possible. In addiction to that, merging these files at the end.
By the use of other Python scripts, by web scraping, other information was obtained, such as languages, reviews, and genres.
After that, all files were refined again.
Last but not least, from the data of the CSV files, the SQL files were built.

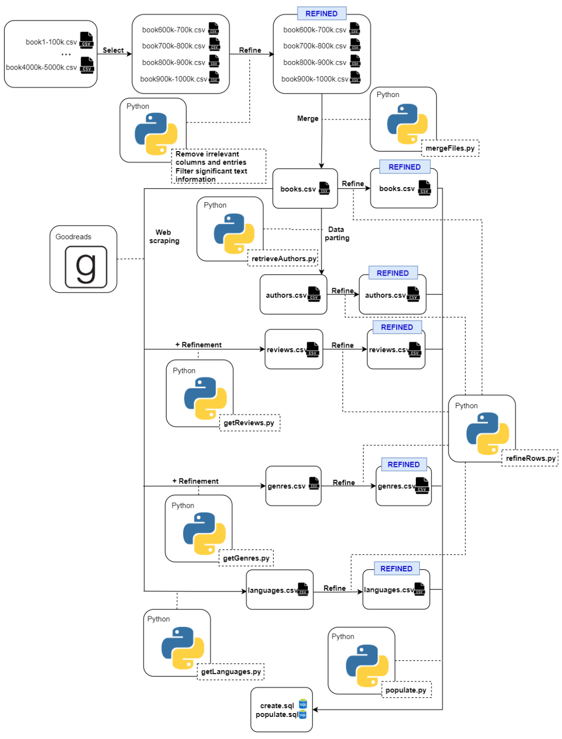

Figure 1: Data processing Pipeline

3 The Data

After the selection and the refinement of the datasets are completed, there are 5 datasets. The conclusions of these datasets were achieved with Python scripts and the Pandas tool.

3.1 Books

The main dataset is the book's dataset which contains 12 columns:

- *Id:* this id is the Goodreads id for the book
- *Name:* this is the name of the book; every book has a name
- *ISBN:* this stands for International Standard Book Number which is an id for the book
- *Rating:* this is the rating of the book given by the users, this rating is a float number between 0 and 5
- *Publish Year, Publish Month and Publish Day:* these attributes represent the publishing date of the book
- *Publisher:* which publisher published the book
- *Count of Reviews:* a number consisting in how many reviews were given to the book
- *Pages Number:* the number of pages of the book
- *Description:* the plot of the book; every book has a description

Conclusions

After analysis, it was concluded that about 63.6% of books have a rating greater than 3 and less than 4, 34.2% are rated between 4 and 5, and a small portion is rated less than 3.
It was also concluded that about 58% were published between 2000 and 2007, and the dataset has much less data on publications before 1980 and after 2010.
Finally, regarding the length of the description, it was concluded that about 47% of the descriptions have a maximum of 100 words, 37% have between 101 to 200 words and only 16% of the descriptions have more than 200 words.
(further information may be found in charts 3, 4, 5 and 6)

3.2 Reviews

- *Id:* this id is the Goodreads id for that book
- *Review:* a text user review of the book; books can have one review, multiple reviews or none

Conclusions

After analysis, it was concluded that only about 18% of the books did not have any reviews. The maximum number of reviews per book is 30, and only 4 books have 30 reviews. The majority of the books have a maximum of 10 reviews, and only a very small part has more than 10.
Besides that it was also concluded that about 69% of the reviews have a maximum of 100 words, 17% between 100 and 200, and only about 14% have more than 200 words.
(further information may be found in charts 7 and 8)

3.3 Genres

- *Id:* this id is the Goodreads id for that book
- *Genre:* the genre of the book; for each book there can be more than one genre

Conclusions

After analysis, it was concluded that are an enormous amount of different genres. The most common one is NonFiction followed by Fiction, History, Childrens, and Classics, etc.
(further information may be found in chart 9)

3.4 Languages

- *Id:* this id is the Goodreads id for that book
- *Language:* the language the book is written; books can have more than one language

Conclusions

After analysis, it was concluded that about 97% of the books are written in English. The other 3% are divided by other languages such as Spanish, French, etc.
(further information may be found in chart 10)

3.5 Authors

The authors dataset contains 2 columns:

- *Id:* this id is the Goodreads id for the book
- *Author:* this is the author of the book; for each book, there is only one author, but there is more than one book that was written by the same author

Conclusions

After analysis, it was concluded that the author that wrote more books is William Shakespeare, who wrote 180 books. Following him is Agatha Christie with 165 books, and so on.
(further information may be found in chart 11)

4 Conceptual Domain Model

This project's domain and how the existing entities relate to each other are modeled in Figure 2. The domain's primary recovery unit is Book, which connects to all the other entities. All books must have one author, at least one genre, and language. The book might or not have reviews.

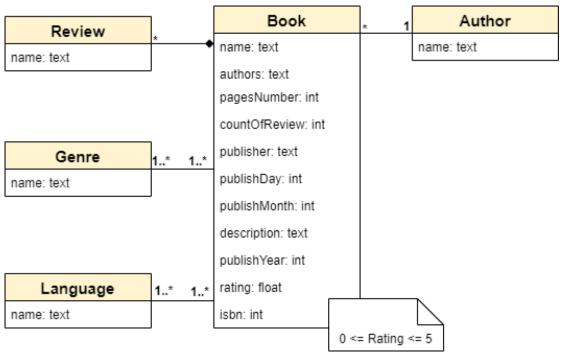

Figure 2: Conceptual Domain Model

5 Possible Queries

To extract information from the data, the following are examples of some possible queries.

1. Search the top 5 books
2. Search the reviews of the books which rating is above 4
3. Search the books published during the summer of the year 2004

For the search tasks, from the user query are used some keywords to perform the actual query. Exemplifying:

User input: Books about climbing, love, and baking.

Performed query: Search the book's description, genre, and reviews for books that contain these words. Different fields may have different weights and it is prioritized to find as many words as possible in common with the user input.

Part II
Information Retrieval

1 Tool Selection

The tool selected was Solr [5] which provides indexing data in multiple formats, allows schema and schemaless modes, query data with several filters, full-text search, etc. This tool among the others suggested seemed to be the best fit for its functionalities. Nonetheless, it has a few limitations which made the work's progress slower than wanted. The main problem was the documentation which is very sparse and lacks practical examples.

2 Collections and Documents

After the completion of the Data Preparation phase, the data was distributed by 5 CSV files: book, languages, genres, reviews, and authors, which are thoroughly explained in Part I. After some research on the documentation, it seemed the most appropriate to have only one core, **books**, and each entry be one indexed document. Therefore, all the files were merged into a single JSON file. Firstly, the authors, and languages files were merged into the books file, simply by adding to the book's attributes, another one representing each of these files. Then, to merge the reviews and genres an attribute was added for each that stores the data in an array. In the end, the collection was imported into Solr using the post tool.

3 The Indexing Process

A step that is extremely important in Information Retrieval is indexing. So after having the data prepared for Solr, the schema needs to be defined since Solr runs in *schemaless mode* otherwise. This schema contains information about the indexed, stored, and type of each field wanted. In terms of indexing, it was clear that all the fields should be indexed except one, the ISBN attribute. The reason is that a field if indexed should be used in queries to retrieve matching documents, but the ISBN attribute doesn't make sense to index because there is no reason to search for a specific ISBN since to know the ISBN you need to know the book unless you type a random ISBN. In terms of storing, all the fields were stored. When it comes to field types, Solr has many that mostly fit the needs, but there still was a need to create one type, *text.* This field was created with the **Standard Tokenizer** [6] and the filters: **Stop Filter** [7] which discards, or stops analysis of, tokens that are on the given stop words list, *stopwords.txt*; **ASCII Folding Filter** [8] which converts alphabetic, numeric, and symbolic Unicode characters which are not in the

Basic Latin Unicode block to their ASCII equivalents; **Lower Case Filter** [9] which converts any uppercase letters in a token to the equivalent lowercase token; **English Minimal Stem Filter** [10] which stems plural English words to their singular form.

So, the field types used were *string* for ISBN, *pfloat* for rating, *pint* for publishYear, publishMonth, publishDay, countsOfReviews, and pagesNumber, *text* for name, publisher, description, author, and language, and *text_ en* for reviews and genres with *multiValued.*

In the end, the schema was imported into Solr using the curl tool[1].

4 The Retrieval Process

Following the indexing process comes the retrieval process that involves two phases: choose the query parser and choose the parameters and optimizations of the query parser. Solr offers many query parsers [11]. The explored ones were **The Standard**, **The DisMax (Maximum Disjunction)**, and **The Extended DisMax** query parsers. By analyzing and experimenting with these three it became clear that The Extended DisMax query parser [12] was the most complete and accurate one, therefore the continuation of the work was done using this one. Among the available eDismax parameters the ones used to fulfill the information needs were:

- *q* - the query to search
- *fq* - filter query
- *qf* - the fields where to search
- *bq* - boosts for the fields
- *pf* - gives boost depending on the proximity of the searched words
- *ps* - maximum amount of tokens wanted between the searched words

The following subsection describes the information needs and how they were achieved.

4.1 Information Needs

(IN1) History books about family

Query(q): history family
Query Fields(qf): genres, description, reviews

Since the purpose of this information need is to find history books about family, it makes sense that the genre of the book is history and that the word family is contained in either the description or in the reviews, but being in the description should give a more accurate result. It also should be noted that the book's genres bring a certain disadvantage. As stated in the previous part, the book's genres are actually shelves that the users create, therefore, for what could be expected to be the same genre, in this dataset exists a lot of synonyms. To overcome this, the file *synonyms.txt* was used where, for this information need, it was added History and Historical as being synonyms. After experimenting, the weights applied with the *bq* parameter can be seen in Table 1.

Field	**Weight**
genres:history	20
description:family	15

Table 1: Weights applied to IN1

(IN2) Fiction Novels

Query(q): fiction novel
Query Fields(qf): genres description name

The purpose of this information need is to find fiction novels so the genre is prioritized, then the name, and only after that the description. After experimenting, the weights applied with the *bq* parameter can be seen in Table 2.

Field	**Weight**
genres	20
description	10
name	15

Table 2: Weights applied to IN2

(IN3) Books fast to read

Query(q): fast read
Query Fields(qf): reviews

Since the purpose of this information need is to find books fast to read, the only field that makes sense to search upon is the reviews. Searching exactly for "fast read" would reduce tremendously the results, so the query is performed with them possibly separated, but with the use of the *pf* and *ps* parameters, a higher boost is given to the book which review has the least amount of words between fast and read. After experimenting, the weight applied with the *pf* parameter can be seen in Table 3.

Field	**Weight**
reviews	20

Table 3: Weight applied to IN3

[1]The schema should be imported before the data

The amount of *phrase slop* was 5, meaning the results could have a maximum of 5 words between the searched words. This is applied with the *ps* parameter.

(IN4) Emotive books about World War II holocaust

Query(q): "world war II" emotive holocaust
Query Fields(qf): genres description reviews

The purpose of this information need is to find emotive books about "world war II" and the holocaust so it makes sense to give a higher boost to genres that match any of the words, then the second highest boost is set to the reviews in order to find emotive related words in people's opinions (using anagrams with the *text_ en* default type), and only then the description with the smallest boost. After experimenting, the weights applied with the *bq* parameter can be seen in Table 4.

Field	Weight
reviews	80
genres	100
description	10

Table 4: Weights applied to IN4

(IN5) Romance books about Friends for a Christmas gift

Query(q): christmas gift
Filter Query(fq): description: friends
Filter Query(fq): genres: romance
Query Fields(qf): reviews

Since the purpose of this information need is to find romance books about friends, with great potential for a Christmas gift, the only field that makes sense to search for Christmas and gift is the reviews and then specify the other relevant words on genres and description fields. This query was used to see the difference between filtering the query and applying boosts to it instead. Filtering the query can discard results that could be relevant as shown in the Evaluation section.

After experimenting, the weights applied and the parameters used to apply them can be seen in Table 5.

Field	Weight	Parameter
reviews	30	pf
genres:romance	60	bq
description:friends	20	bq

Table 5: Weight applied to IN5

And the amount of *phrase slop* was 5, meaning the results could have a maximum of 5 words between the searched words. This is applied with the *ps* parameter.

5 Evaluation

In order to evaluate the results obtained for each information need, three configurations were thought of. Firstly, a study with results obtained in schemaless mode. Then with the schema described in the Indexing Process Section, in order to verify if the field types, tokens and filters were appropriate. Last, but not least, with the weights described in the Information Needs subsection to verify if, once again, the values defined were appropriate.

5.1 Evaluation Methodology

The evaluation process consisted of selecting the relevant documents, meaning the books that fulfill the information need, for each information need and then comparing with the result given by Solr. Since checking manually the entire dataset would be extremely time-consuming, some information needs were searched in a subset of 200 books and their reviews and others on the entire dataset. To compare the relevant results with the given ones, the following metrics were used:

- Precision at 10 (P@10): fraction of relevant results among the top 10 retrieved documents
- Recall at 10 (R@10): fraction of relevant documents that are retrieved among the top 10 retrieved documents
- Average Precision (AvP): a single-figure measure of quality across recall levels for a single query

5.2 Results

For each information need, the following will demonstrate the evaluation of the results obtained with the three configurations planned: schemaless, with schema, and boosted.

(IN1) History books about family

The results of this information need were obtained and evaluated on the subset of 200 books. With the results obtained and by evaluating them with the metrics chosen, that can be seen in the following Table 6, the conclusion is that the schema applied did not make much difference in this information need, because in this subset there aren't a lot of occurrences of the searched query. However, with boosts, the resulted are better. A graphic representation of the precision in function of the recall can be seen it chart 12.

	P@10	AvP	R@10
Schemaless	50%	69%	50%
Schema	50%	70%	50%
Boosted	70%	90%	70%

Table 6: Evaluation for IN1

(IN2) Fiction Novels

The results of this information need were obtained and evaluated on the subset of 200 books. With the results obtained and by evaluating them with the metrics chosen, that can be seen in the following Table 7, the conclusion is that the schema applied this time made a difference, improving the results, but again the boosted configurations is the one that gives better results. A graphic representation of the precision in function of the recall can be seen it chart 13.

	P@10	AvP	R@10
Schemaless	20%	34%	20%
Schema	40%	41%	40%
Boosted	60%	60%	60%

Table 7: Evaluation for IN2

(IN3) Books fast to read

The results of this information need were obtained and evaluated on the subset of 200 books. This one is similar to IN1, there isn't much improvement with or without schema, but with the boosts better results are achieved. Since both IN1 and IN3 use the field reviews, a possible conclusion is that the field type used for the reviews attribute might not be the most adequate, but the use of the subset also restricts the results. The evaluation of the results can be seen in Table 8 bellow. A graphic representation of the precision in function of the recall can be seen it chart 14.

	P@10	AvP	R@10
Schemaless	50%	66%	50%
Schema	50%	67%	50%
Boosted	80%	86%	80%

Table 8: Evaluation for IN3

(IN4) Emotive books about World War II holocaust

The results of this information need were obtained and evaluated on the entire dataset. With the evaluation of the results displayed in Table 9, it is demonstrated that, now with the entire dataset, the schema makes a lot of difference, and with the boosted the results are even better. A graphic representation of the precision in function of the recall can be seen it chart 15.

	P@10	AvP	R@10
Schemaless	0%	0%	0%
Schema	30%	10%	30%
Boosted	70%	90%	70%

Table 9: Evaluation for IN4

(IN5) Romance books about Friends for a Christmas gift

The results of this information need were obtained and evaluated on the entire dataset. With the evaluation of results obtained the conclusion is that, in this case without the schema, the results obtained are bad and with the schema and boosts are increasingly better. A graphic representation of the precision in function of the recall can be seen it chart 16.

	P@10	AvP	R@10
Schemaless	0%	0%	0%
Schema	30%	46%	38%
Boosted	50%	75%	63%

Table 10: Evaluation for IN5

5.3 Evaluation Conclusions

With the evaluation of the information needs that were selected, it is clear that although the boosts were well chosen, the schema is something that needs improving. It is important to highlight that, the schema will not improve the results obtained for all the information needs that one could think of, because of the impact of the specificity of the information need.

Conclusions and Future Work

In this report, the chosen datasets and the process of rectification implemented to them are described. The configuration of the information retrieval system is then presented, with particular attention given to the indexing process and the result of querying.
Initially, the datasets were obtained through a variety of actions and refined. The refinements

included the removal of irrelevant columns and entries, as well as the filtering of significant text information. Afterward, the information available in the datasets was analyzed, and, with that in mind, relevant charts were traced. The Pipeline used to process the data is also identified, as is the Domain Conceptual Model which describes the relations between the different entities.
Additionally, a brief analysis of the properties of Solr was made, having been determined that while it was the most appropriate tool for the work being developed, it had sparse documentation. The datasets were then merged and imported to Solr, with additional indexing of each document. Regarding the retrieval process, various weighting field configurations were studied, but for each information need case was chosen a different configuration, having been shown through iterations which one was ideal. Finally, for the system's evaluation, a set of three system configurations were conceived: Schemaless, Schema, and Boosted. The best results were achieved by Boosted as it applied a few selected operations to the documents' chosen fields and worked with weights while querying the documents.
As future work, the integration of this system with the semantic web paradigm will be developed.

References

[1] W. McKinney, "Pandas library," Sep 12, 2021. [Online] URL: `https://pandas.pydata.org/`.

[2] W. McKinney, "Csv library," Dec 07, 2021. [Online] URL: `https://docs.python.org/3/library/csv.html`.

[3] B. Jannesar, "Dataset retrived from kaggle," 2020. [Online] URL: `https://www.kaggle.com/bahramjannesarr/goodreads-book-datasets-10m`.

[4] O. Chandler, "Goodreads," 2021. [Online] URL: `https://www.goodreads.com/`.

[5] A. S. Foundation, "Apache solr," Nov 16, 2021. [Online] URL: `https://solr.apache.org/`.

[6] A. S. Foundation, "Solr standard tokenizer," Nov 16, 2021. [Online] URL: `https://solr.apache.org/guide/8_10/tokenizers.html#standard-tokenizer`.

[7] A. S. Foundation, "Solrstop filter," Nov 16, 2021. [Online] URL: `https://solr.apache.org/guide/8_10/filter-descriptions.html#stop-filter`.

[8] A. S. Foundation, "Solr ascii folding filter," Nov 16, 2021. [Online] URL: `https://solr.apache.org/guide/8_10/filter-descriptions.html#ascii-folding-filter`.

[9] A. S. Foundation, "Solr lower case filter," Nov 16, 2021. [Online] URL: `https://solr.apache.org/guide/8_10/filter-descriptions.html#lower-case-filter`.

[10] A. S. Foundation, "Solr english minimal stem filter," Nov 16, 2021. [Online] URL: `https://solr.apache.org/guide/8_10/filter-descriptions.html#english-minimal-stem-filter`.

[11] A. S. Foundation, "Query syntax and parsing," Nov 16, 2021. [Online] URL: `https://solr.apache.org/guide/8_10/query-syntax-and-parsing.html`.

[12] A. S. Foundation, "The extended dismax query parser," Nov 16, 2021. [Online] URL: `https://solr.apache.org/guide/8_10/the-extended-dismax-query-parser.html`.

Annex I - Dataset Analysis

Books Dataset Analysis

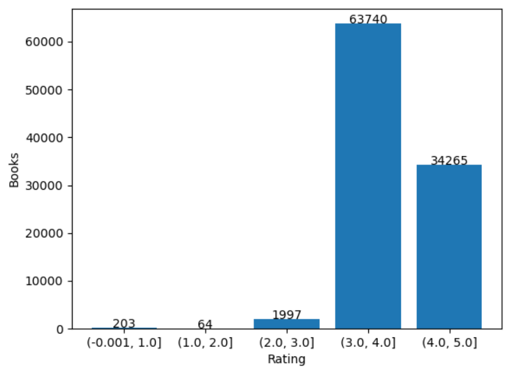

Figure 3: Books per Rating

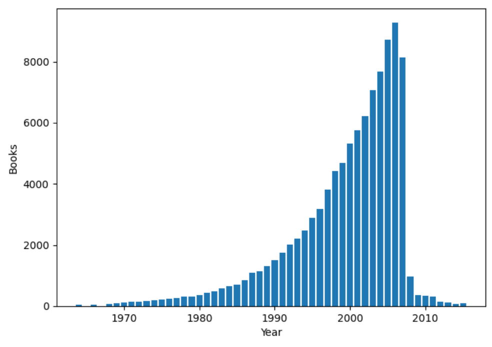

Figure 4: Books per Year

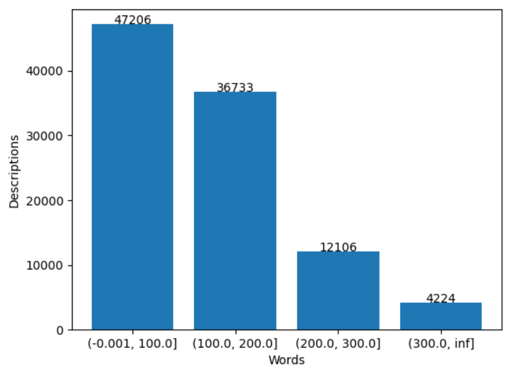

Figure 5: Descriptions per Words

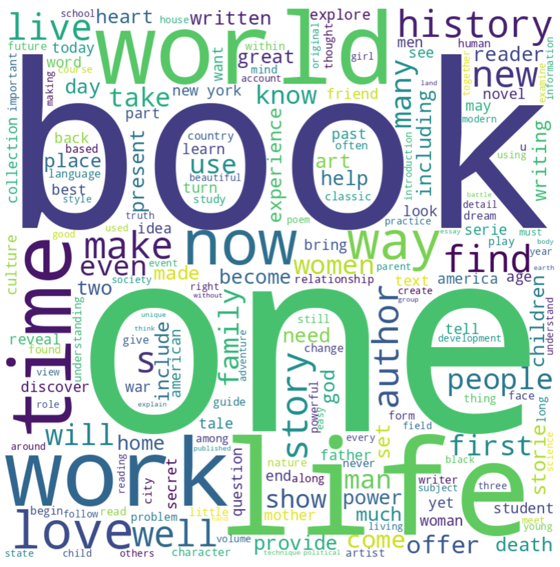

Figure 6: Most common words in all descriptions.

Reviews Dataset Analysis

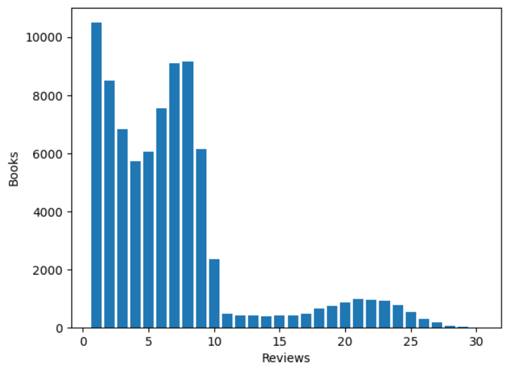

Figure 7: Books per Reviews

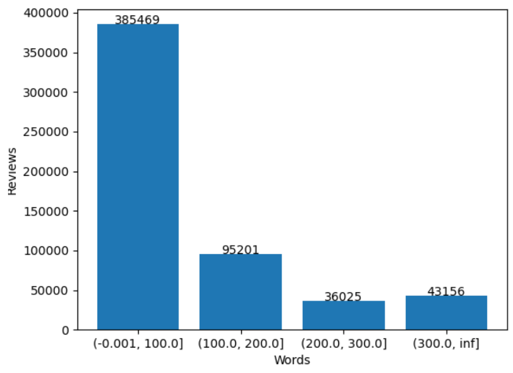

Figure 8: Reviews per Words

Genres Dataset Analysis

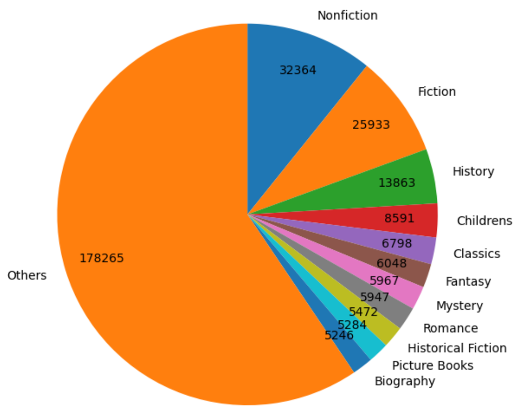

Figure 9: Books per Genre

Languages Dataset Analysis

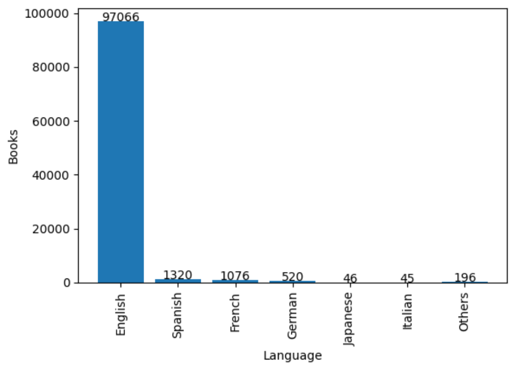

Figure 10: Books per Language

Authors Dataset Analysis

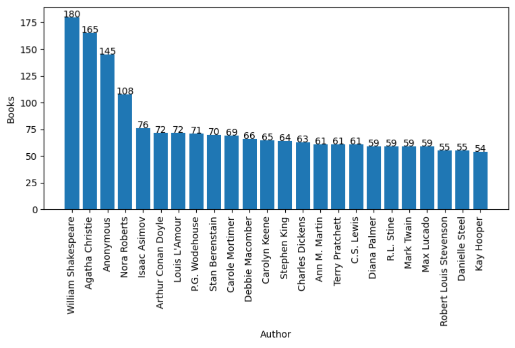

Figure 11: Books per Author

Annex II - IR Analysis

(IN1) History books about family

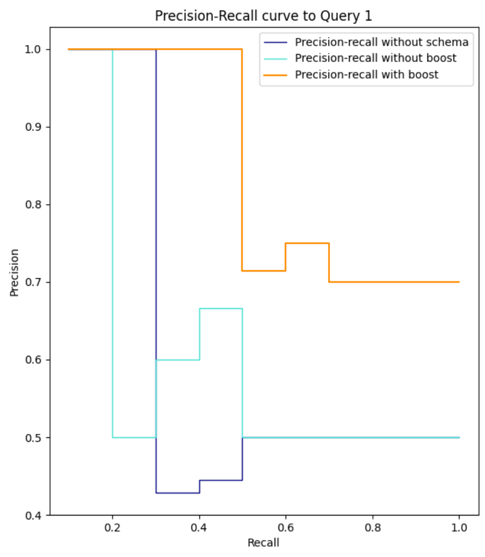

Figure 12: Precision Recall Curve for IN1

(IN2) Fiction Novels

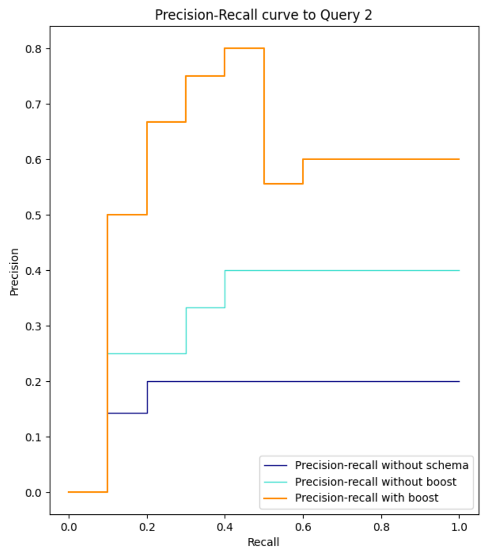

Figure 13: Precision Recall Curve for IN2

(IN3) Books fast to read

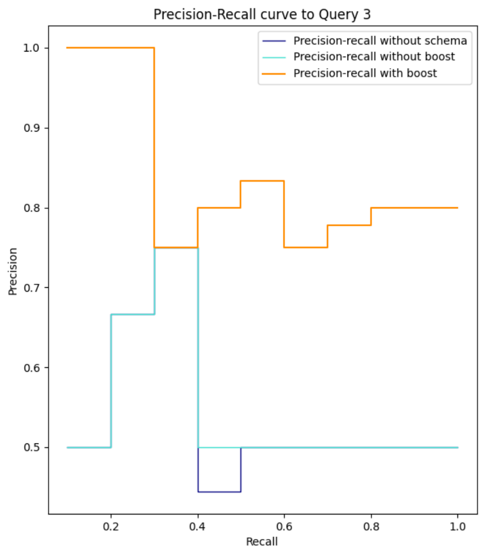

Figure 14: Precision Recall Curve for IN3

(IN4) Emotive books about World War II holocaust

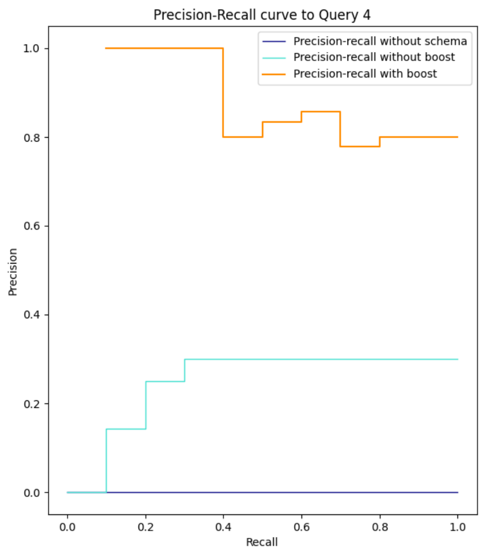

Figure 15: Precision Recall Curve for IN4

(IN5) Romance books about Friends for a christmas gift

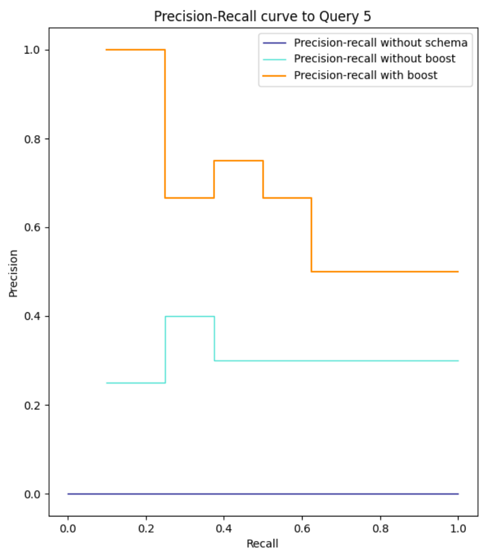

Figure 16: Precision Recall Curve for IN5