Privago: Hotels Search System based on their reviews

André Costa
up201905916@edu.fe.up.pt
Faculty of Engineering - University of Porto
Porto, Portugal

André Ávila
up202006767@edu.fe.up.pt
Faculty of Engineering - University of Porto
Porto, Portugal

Fábio Sá
up202007658@edu.fe.up.pt
Faculty of Engineering - University of Porto
Porto, Portugal

Fábio Morais
up202008052@edu.fe.up.pt
Faculty of Engineering - University of Porto
Porto, Portugal

ABSTRACT

The internet's rapid growth demands appropriate systems for harnessing and connecting this massive information resource. This project addresses this need by focusing on the hotel industry, where reviews play a crucial role in shaping consumer choices. This article aims to provide a clear and well-documented explanation of the work needed in developing a robust search engine for hotel reviews. To accomplish this, data is collected from four different Kaggle datasets, cleaned and prepared, and in-depth data analysis is undertaken, as we seek to fulfill prospective search tasks such as finding hotels near the airport or finding hotels with a helpful staff. The data is then transformed into a collection on Apache Solr's [1] search engine, and, by tinkering with the multiple characteristics available, results are obtained and evaluated.

CCS CONCEPTS

• **Information systems** → Information retrieval query processing.

KEYWORDS

Hotels, Reviews, Information, Dataset, Data Retrieval, Data Preparation, Data Analysis, Data Processing, Data Refinement, Pipeline, Data Domain Conceptual Model, Information Retrieval

ACM Reference Format:
André Costa, André Ávila, Fábio Sá, and Fábio Morais. 2023. Privago: Hotels Search System based on their reviews. In *Proceedings of PRI (G53).* ACM, New York, NY, USA, 10 pages.

1 INTRODUCTION

The choice of the hotel reviews theme is motivated by the large amount of information's sources and the rich diversity of attributes it encompasses. Hotel reviews, as a research focus, hold substantial importance in the modern information landscape. They not only provide valuable insights into the hospitality industry but also serve as a prime example of data diversity, combining numerical rates, submission dates, and personal, subjective narratives. This diversity introduces intricacies in data structuring and presents challenges in contextual search, making it an ideal choice for aligning the search system with real-world scenarios. Thus emphasizing practical applicability and the development of robust information retrieval solutions.

This document is structured into several major sections. We commence with **Data Extraction and Enrichment**, where we introduce the data sources, briefly characterize the datasets, and assess data quality. Subsequently, **Data Preparation** outlines the selection criteria, processing methods, and data storage procedures for hotel-related information and associated reviews, following a clear and reproducible pipeline.

In **Data Characterization** we delve into the evaluation and visualization of the refined data. This involves examining various criteria and relationships, from the Domain Conceptual Model to Word Clouds.

Possible Search Tasks provide an overreaching interpretation of the results, guiding the identification of suitable research objectives for the **Information Retrieval**, where the documents are indexed and searched using Apache Solr with a defined schema and defined parameters. Finally, the **Evaluation** analyses the results obtained by custom queries on the prepared search engine, which was defined on the previous section.

2 DATA EXTRACTION AND ENRICHMENT

After conducting research for relevant data in terms of variety and quantity, four datasets in CSV format from different regions were selected through the Kaggle [2] platform. Table 1 provides a characterization of the acquired datasets:

Table 1: Initial datasets characterization

Dataset	Features	Hotels	Reviews	Size(MBs)
Datafiniti's Hotel Reviews	26	1400	10000	124.45
Hotel Review Insights	7	570	7000	1.31
London Hotel Reviews	6	20	27329	22.85
Europe Hotel Reviews	17	1493	515000	238.15

The Datafiniti's Hotel Reviews [3] dataset was taken from Datafiniti's Business Database [4] through sampling. Hotel Review Insights [5] is a compilation of hotels around the world through web-scraping of reviews found on Booking.com [6]. London Hotel Reviews [7] is a sample taken and partially refined from a DataStock dataset [8]. Finally, Europe Hotel Reviews [9] also results from web scrapping of hotel reviews across Europe published on Booking.com [6].

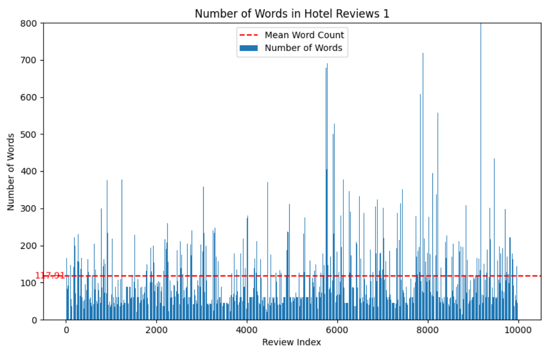

Figure 1: Average words per review in Datafiniti's Hotel Reviews dataset

All datasets have a public use license and, according to the Kaggle platform, a usability index greater than 8. This index is justified, given that the elimination of null, repeated or non-informative entries practically did not eliminate nothing.

The datasets contain common features, numerical data, such as rate, review date, and textual data, such as review text, hotel location and name. The last dataset contains two additional parameters, positive review and negative review. The features stated were extracted in this step and refined in Data Preparation.

3 DATA PREPARATION

In this section, is presented the structured data preparation pipeline [9] that was developed for the project. This pipeline embrace various data cleaning and restructuring procedures aimed at achieving a clean, uniform, and ready to analyze dataset. The goal of this stage is to build a strong basis for insightful analysis.

The data preparation process began with a comprehensive cleaning phase, where the primarly focus was the removal of records containing **empty or null values** in any attribute. Simultaneously, was identified and eliminated incomplete or uninformative data, including strings with **uninformative text**, such as "no comments available for this review.", using Python. This combined cleaning step ensured that the dataset was cleansed in detail for the normalization phase.

With the data cleaned, attention was turned to **attribute normalization**. Given the presence of diverse datasets with varying formats, comprehensive normalization process was needed. This included standardizing attributes such as **"positive_reviews"** and **"negative_reviews"** into a unified **"review_text"** attribute for the 4th dataset as is demonstrated in the Pipeline Diagram [Figure 9]. Additionally, **date formats** were normalized to ensure uniformity and suitability for analysis. The date format was set as "year-month" due to the absence of day-specific review information in the second dataset and its irrelevance to the research targets.**Rate scales** were also normalized to a common range and converted to floating-point values, facilitating comparative analysis ([0.0, 5.0]).

In addition, was established a **standardized naming** convention to address variations in **hotel names**, such as from *"45 Park Lane - Dorchester Collection"* to *"45 Park Lane Dorchester Collection"*. This step was necessary to facilitate the aggregation step and the addition of the feature "average_rate" to each hotel entity, referenced below. **Location standardization** involved reducing location names to their last two words, preserving only the capital and country names.

To gain insights into the textual content, we calculated the temporary column **word count** for each review across all datasets. This analysis was facilitated using the Pandas [10] Python tool, allowing us to extract valuable information such as quartile ranges and make informed decisions during the review deletion phase. This process enabled us to identify and manage reviews with either an insufficient word count or an excessively high word count. We achieved this by removing reviews falling below the 25% threshold (first quartile) and those exceeding the 75% threshold (fourth quartile). This step was done separately for each dataset, due to the discrepation of each average word counting [Figure 10] [Figure 11].

At this stage, all the datasets were successfully merged into a single, consolidated dataset, which streamlined the remaining preparation tasks. These tasks commenced with the computation of the temporary column **"average_rate"** for each unique hotel. This information may prove valuable for defining search criteria in future milestones.

After completing the aforementioned steps, the next phase involved determining the minimum and maximum number of **reviews per hotel** to be retained. To accomplish this, the same approach used for analyzing the number of words per review was employed, utilizing the Pandas [10] .describe() function. This statistical analysis provided essential insights into the distribution of reviews across hotels. This step was important due to the discrepation of the number of reviews per hotel [Figure 2].

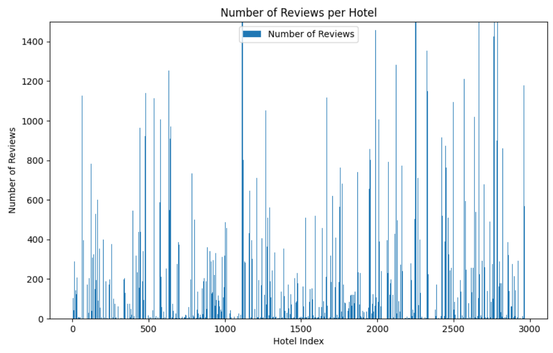

Figure 2: Number of Reviews per Hotel

First, hotels with fewer reviews, falling below the established minimum threshold (first quartile), were addressed, and they were subsequently removed from the dataset. This step was crucial in ensuring that the dataset focused exclusively on hotels with a volume of reviews contained in the second and third quartile, thus providing more meaningful insights.

Subsequently, hotels with an excessive number of reviews were taken into account. To handle this situation, a strategy was implemented, allowing the selection and retention of reviews while preserving the proportion of reviews per rate category for each specific hotel, i.e., its global rate. This approach guaranteed a balance between the "average_rate" and the rate of the selected reviews.

The final step in the data preparation phase involved organizing the data into the **desired JSON file format**, which was designed based on our UML diagram [Figure 8] and with a focus on the primary objective of the search tool. This format consisted of a collection of JSON objects, each representing a "Hotel" entity. Within each "Hotel" object, were included not only its associated attributes (name, location, average_rate) but also the related reviews, presented as JSON objects themselves. Each review has a corresponding text, rate and submission date.

4 DATA CHARACTERIZATION

In this section there is a characterization of the documents that are produced by the pipeline. The graphs and tables were obtained using the Matplotlib [11], Numpy [12] and NLTK [13] libraries.

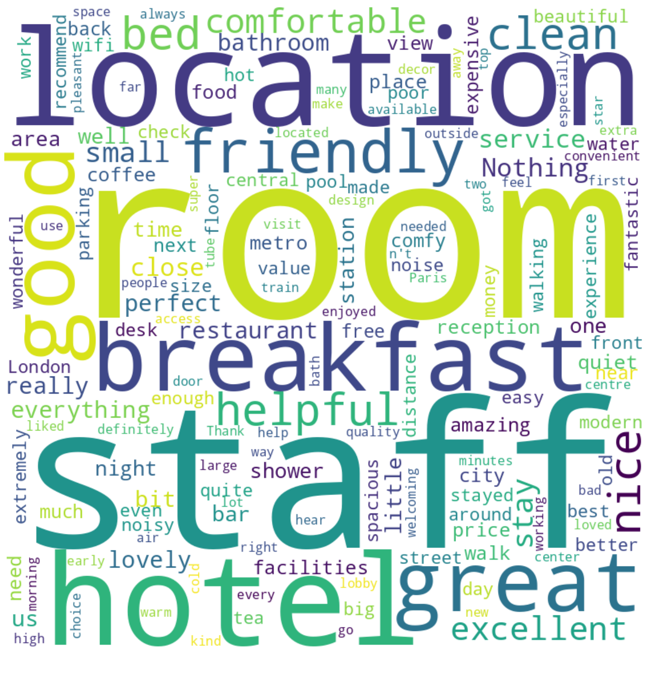

Figure 3: Reviews Word Cloud

The word cloud based on the texts of the reviews supports the search tasks of the next milestone as well as the contextual search to be implemented. As expected, the words that stand out the most are those intrinsically related to the hotel industry, such as "staff", "room", "location", "breakfast". Generally, the highlighted words have a neutral tone, although several with a very positive connotation are also detectable, such as "clean", "great", "lovely", "excellent", which can also be proven by the average rate [Figure 5] given to hotels.

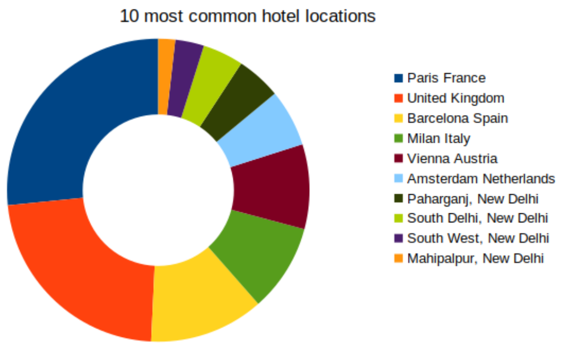

Figure 4: Hotel location distribution

Figure [4] shows the distribution of the 10 most frequent hotel locations in the system. As expected, the capitals and large cities of the main tourism countries concentrate the largest number of hotels and their reviews.

As we can see in figure [5] Hotels tend to have very good reviews. The result is encouraged by their locations, as the majority are in cities with major tourist attractions and therefore with a greater cadence of reviews.

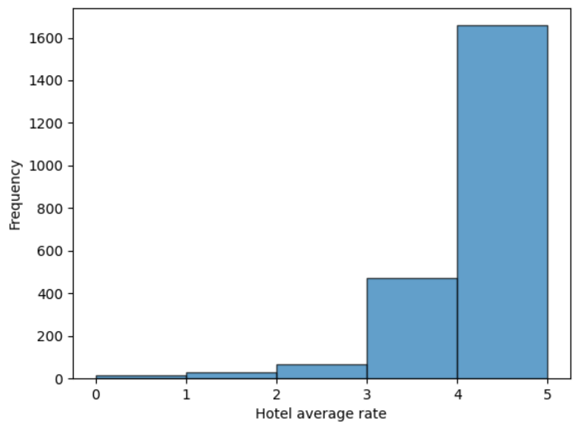

Figure 5: Average rate distribution

From the analysis of the figure [6], it can be concluded that the system includes hotels with reviews from 2010 to 2023. However, the choice of initial datasets with information relating mainly to the period 2015 and 2017 influenced their representativeness.

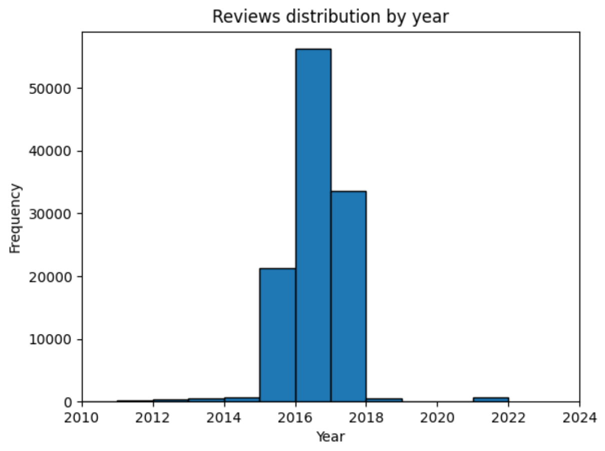

Figure 6: Reviews distribution per year

Let's look, for example, at the distribution of reviews by month in 2016 in the figure [7].

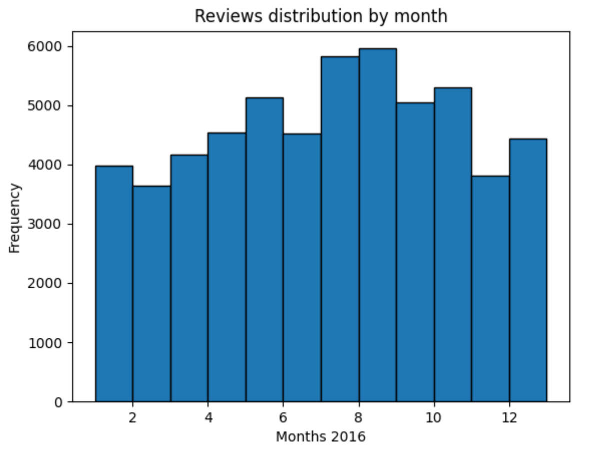

Figure 7: Month reviews distribution in year 2016

As we can see, it is July and August when the number of reviews is higher. This period corresponds to the Summer time when people normally take vacations and therefore choose to travel. This pattern is repeated for the other years under analysis.

4.1 Data Conceptual Model

After the Data preparation phase, our documents contain the following relationships:

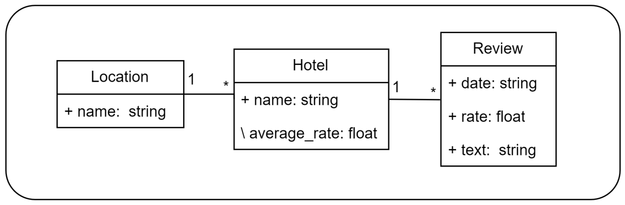

Figure 8: Data Conceptual Model

A hotel is made up of the attributes name, location and average_rate. Note that average_rate is a derived attribute that was calculated in the process based on the selected reviews. Each review has the corresponding text, rate and submission date.

Location was also considered a class because, as expected in the hotel scene, there are several hotels per region, mainly in large cities, as shown in figure [4].

5 POSSIBLE SEARCH TASKS

In the course of the data analysis journey, were unveiled valuable insights through the use of a Word Cloud Diagram [Figure 3]. This visual representation highlighted the most frequently occurring words in hotel reviews, shedding light on what matters most to travelers. Among these words, some stood out as pivotal in understanding the key factors that influence hotel choice and guest satisfaction.

"Location" emerged as one of the top considerations in travelers' decision-making processes. Whether it's proximity to local attractions, accessibility to transportation hubs, or the overall neighborhood ambiance, the location of a hotel can greatly influence the overall travel experience. Search scenarios such as **"Hotel in the United Kingdom with good location and either elevator or good accessibility."** and **"The best hotels near center of London."** can help travelers selecting accommodations that align with their preferred locations and provide convenient access to their destinations.

The words "breakfast" and "service" had a prominent presence in the word cloud [Figure 3], suggesting a keen interest in this aspects. Whether it's a hearty breakfast buffet or unique morning options, travelers seek accommodations that cater to their breakfast preferences. Another one of the foremost considerations in hotel selection is the quality of the room and its amenities. Words like "room," "bed," and "bathroom" featured prominently in our Word Cloud [Figure 3], underscoring the importance of these aspects to travelers. Searches related to "room"/"bed" quality or "bathroom" sanitation can guide travelers to accommodations that prioritize comfort and cleanliness. A search such as **"Hotels with good breakfast or good room service in New Delhi"** could be instrumental in identifying hotels that excel in the breakfast provided or in the room service available to their guests.

The context about the hotels left in their reviews adds the possibility of searching for specific details, for example, a person that follows a vegetarian diet might be struggling to find compatible hotels, this hassle could be resolved with a search such as **"Hotels with good vegetarian/vegan options"**.

6 INFORMATION RETRIEVAL

Information Retrieval [14] is the process of finding and extracting relevant information from large collections of naturally unstructured data, such as texts. This extraction is based on documents, which are the result of restructuring the initial data, and the output is sorted by relevance, becoming the main challenge.

This section presents the indexing and query methods used in this information retrieval system powered by previously constructed documents.

The implementation of the search system is based on Apache Solr [1] , an open-source tool that offers various features relevant to the project's purpose, including distributed and fast indexing, scalability, and advanced search capabilities surpassing a full-text match.

6.1 Document Characterization

The documents to be indexed and searched in the system are those resulting from the processes of data extraction, enrichment, and aggregation in the pipeline described above.

Therefore, a hotel is a document consisting of a name, average rating, location, and has a set of associated reviews. These reviews have their corresponding date, the assigned rating, and the user's comment about the hotel.

6.2 Indexing Process

Indexing serves as a fundamental step in Information Retrieval, optimizing search efficiency by organizing the data. It involves creating a structured index that significantly enhances both search speed and scalability. Without proper indexing, search systems would face challenges, resulting in slower response times and increased computational overhead.

In Solr, various types of indexing exist for document fields and associated queries, based on a Tokenizer [15] and Filters [16]. While Tokenizers create a token stream from the original string following a predefined rule, Filters transform these tokens for consistency in subsequent searches and matches.

In this specific case, the focus was primarily on indexing textual fields, as they provide the most context and information for searches. Conversely, given the project's context, it is not expected to search for specific dates or review ratings. Therefore, these latter two document fields were not indexed.

Textual fields were indexed by instantiating a new data type. The **'boosted_text'** index analyzer includes:

- ***StandardTokenizerFactory*** tokenizer: splits texts based on punctuation and spaces;
- ***ASCIIFoldingFilterFactory*** filter: handles special characters and accents, converting them to their equivalent ASCII form;
- ***LowerCaseFilterFactory*** filter: converts all characters to their lowercase counterparts;
- ***SynonymGraphFilterFactory*** filter: expands each token to include variations based on its synonyms;
- ***EnglishMinimalStemFilterFactory*** filter: reduces each token to its root form, facilitating the search for variations of specific terms;

Fields with native values were defined using Solr's default types. The English language was chosen for both stem assignment and synonym generation, aligning with the language of the manipulated data.

The **SynonymGraphFilterFactory** is crucial in this context. Since the search is conducted based on reviews, which are inherently subjective, derived from natural language and rich in adjectives, it is important not to rely on specific terms but rather to match synonyms of terms.

The same structure was used for the query analyzer. Thus, the indexing of the final document can be characterized by the following schema:

Table 2: Schema Field Types

Field	Type	Indexed
name	boosted_text	yes
location	boosted_text	yes
average_rate	pdoubles	yes
date	string	no
rate	pdoubles	no
text	boosted_text	yes

6.3 Retrieval Process and Setup

The approach implemented involves two schemas: a simple schema utilizes default field types for each field, while the more complex schema incorporates instantiated field types for enhanced search capabilities.

For query parameters used by both schemas, the system incorporates the following:

- **Query (`q`)**: Focuses on the most valuable words in the query.
- **Query Operator (`q.op`)**: Utilizes OR | AND for query operations.
- **Query Filter (`fq`)**: Defines a query that can be used to restrict the superset of documents.
- **Filter List (`fl`)**: Limits the information included in a query response.
- **Sort Field (`sort`)**: Specifies the sorting value for the results.

Table 3: Query parameters

Parameter	Value
q	(strong wifi)
q.op	AND
fq	!child of="*:* - _nest_path_:*" location: "New York"
fl	*,[child]
sort	score desc

The need of the fl and fq parameters results from the inclusion of nested documents [17] [18] . The final dataset consists of hotels, each containing a list of reviews. Recognizing the relevance of both hotels and reviews as distinct documents, a distinct approach to the search process was necessary. The combined use of fl and fq proved to be efficient in this step.

Additional query criteria the simple schema uses the default type, while the boosted schema employs the type created by eDismax [19] with specific parameters for optimizing search engine results:

- **Query Field with Optional Boost (qf)**: This assigns weights to specific fields in the search;
- **Phrase-Boosted Field (pf)**: Focuses on selecting more relevant terms from the query;
- **Phrase Boost Slope (ps)**: Defines the maximum number of tokens between searched words;

Table 4: defType eDismax parameters

Parameter	Value
qf	text^ 7 name location^2
pf	text^ 10
ps	3

Assigning diverse weights within the **qf** parameter prioritizes the significance of the **text** field, being the main field of search, followed by **location** and **name**. In the **pf** parameter, exclusive attention is given to the **text** field, serving as a dedicated phrase boost. This is complemented by the **ps** parameter set to 3, an average number of maximum tokens between the searched words.

The eDismax parameter **bq** (boost query) was also explored in the approach but not included for system analysis. Since it relies on the characteristic tokens of each query, it was found that it would introduce bias to the results while configuring a system that wouldn't be general enough for all information needs.

Being consistent with this boosted approach to every query has enhanced the system's query handling, leading to improvements in search results, as elaborated in the subsequent section.

7 EVALUATION

Evaluation is also a fundamental aspect of Information Retrieval, contingent on the target document collection and the type of information required. Understanding potential user scenarios is crucial for defining new designs and implementations based on received feedback. In this specific case, the evaluation was conducted from the perspective of effectiveness — the system's ability to find the right information — rather than efficiency, which pertains to the system's speed in retrieving information.

The use of individual and subjective metrics can introduce bias in evaluating the two previously instantiated systems. To address this, a set of distinct metrics based on **precision** and **recall**, such as **Average Precision (AvP)**, **Precision at K (P@K)**, **Precision-Recall curves**, and **Mean Average Precision (MAP)**, were employed. Precision focuses on the percentage of the number of truly relevant documents among those extracted, while recall makes this comparison based on all relevant documents within the system. Since there are more than 2000 unique documents in this case, precise calculation is impractical, leading to a manual approximation based on extracting and sampling the first twenty returned documents.

The **Average Precision (AvP)** is important because precision is what defines user satisfaction for the majority of users. In fact, users often do not require high recall since the percentage of relevant results given all important documents in the system is almost always unknown, unlike the relevance of the first returned documents. In **Precision at K (P@K)**, the choice was to evaluate the first twenty documents returned per query as it represents a balanced value aligning with typical usage patterns of a search engine.

The **Precision-Recall Curves** are constructed for each query and each system based on the subset of ranked documents returned. Ideally, a system is considered more stable the smoother its formed curve, and its performance is deemed better with a higher Precision-Recall Area Under the Curve [20]. This metric encapsulates the overall effectiveness of the system in balancing precision and recall across thresholds.

The **Mean Average Precision (MAP)** is a common metric used in Information Retrieval and represents the average of Average Precision metric across various sets returned over the evaluation period. It helps determine if the system is consistent even when applied to different information needs.

In the upcoming subsections, diverse user scenarios are presented as queries, accompanied by their respective results and statistics based on precision and recall metrics.

A Center of London

Information Need: The best hotels near center of London.

Relevance Judgement: In this task, the objective is to find hotels near the center of London with the highest ratings. The search is conducted using keywords like 'center London' within the review text, United Kingdom as location and the results are sorted in descending order based on their rating.

Query:

- q: (center london)
- q.op: AND
- !child of="*:* -_nest_path_:*" location: "united kingdom"
- fl: *,[child]
- sort: average_rate desc, score desc

Table 5: Q1 information need results

Rank	Syst. Simple	Syst. Complex
AvP	0.82	0.9
P@20	0.6	0.9

Result Analysis: Both systems did well, although there is a notable increased precision on the boosted system, demonstrated in the table [cite][Table T4]. The utilization of eDismax, in this query, proved to be important for the results since the 2 words "center london", when put together, are correlated with each other.

B Breakfast or Room Service

Information Need: Hotels with good breakfast or good room service in New Delhi.

Relevance Judgement: In this information need its intended to search for hotels with a good breakfast or a good room service in New Delhi. Therefore, the words "good breakfast" or "good room service" should appear in the same query/text of review and the location should be a filter query of the parents documents.

Query:

- q: (good breakfast) OR (good room service)
- q.op: AND
- !child of="*:* -_nest_path_:*" location: "new delhi"
- fl: *,[child]
- sort: score desc

Table 6: Q2 information need results

Rank	Syst. Simple	Syst. Complex
AvP	0.76	0.87
P@20	0.8	0.8

Result Analysis: The two systems have similar average precision, as it can be seen in table 6. Since it is a very simple query, it is expected for good results from both systems and it is normal for the improved one to fail in some sentences since it's using the **ps** parameter (which is equal for every query) that allows for tokens between searched words. This would be resolved with contextual analysis referred in the "Future Work", section 8.

C Convenience and Accessibility

Information Need: Hotel in the United Kingdom with good location and either elevator or good accessibility.

Relevance Judgement: With this query it is intended to gather the hotels situated in the United Kingdom which have a good location with either an elevator or good accessibility, a query for someone with reduced mobility that wants to visit the country.

Query:

- q: good location ((elevator) OR (accessibility))
- q.op: AND
- !child of="*:* -_nest_path_:*" location: "united kingdom"
- fl: *,[child]
- sort: score desc

Table 7: Q3 information need results

Rank	Syst. Simple	Syst. Complex
AvP	0.58	0.47
P@20	0.5	0.65

Result Analysis: The systems differ in average performance, with the simple one overall performing better, as shown by the table 7. The reason for the complex system to score lower in the AvP (average performance) parameter but higher in the P@20 (precision) parameter however, is due to the system not taking the context into consideration, therefore, in a query that specifies the need for an elevator, the system gathers reviews that mention the absence of one.

D Vegetarian/Vegan

Information Need: Hotels with good vegetarian/vegan options

Relevance Judgement: In this task, the objective is to find hotels with good vegetarian or vegan options. So, the words "good vegetarian" or "good vegan" should appear in the review's text. The location isn't specified.

Query:

- q: (good vegetarian) OR (good vegan)
- q.op: AND
- !child of="*:* -_nest_path_:*" location:*
- fl: *,[child]
- sort: score desc

Table 8: Q4 information need results

Rank	Syst. Simple	Syst. Complex
AvP	0.5	0.55
P@20	0.35	0.5

Result Analysis: Both systems exhibit similar average precision values, which fall below the anticipated values for a simple query. This can be attributed to the same issue discussed in B. In fact, certain review texts were expressed in a negational form or contained nouns such as "lack", thereby altering the entire meaning of the sentence.

Taking into account all the results from the multiple information needs across queries, its presented in the following table the Mean Average Precision for both systems:

Table 9: Overall systems evaluation

Global	Syst. Simple	Syst. Complex
MAP	0.665	0.6975

Therefore, it is concluded that the system exhibits satisfactory performance, and, in general as anticipated, the more complex system is capable of yielding better results than the simple one.

8 CONCLUSIONS AND FUTURE WORK

In summary, the project achieved significant milestones, with the successful completion of tasks across various phases.

During the 'Data Preparation' phase, our team adeptly navigated challenges to create a finalized dataset. Employing diverse techniques for collection, cleaning, and restructuring, we addressed a major number of reviews. Our efforts focused on striking a balance between maintaining data richness and relevance while ensuring a manageable dataset size.

Moving on to the Information Retrieval phase, all planned tasks were executed successfully, marking a pivotal moment in constructing an effective hotel search engine for tourists. Notably, challenges arose in dealing with nested documents, including their indexing and retrieval.

Through the evaluation of the search engine, the system's stability and capability to handle different information needs within the chosen context have been verified. As the project progresses, opportunities for further enhancements and refinements emerge. Analyzing the results obtained from the first prototype of the hotel's information retrieval system:

- The **Stop Words** [21] filter can be applied to 'boosted_text' to reduce sensitivity to common words;
- **Sentimental and contextual analysis** is relevant, given that the main source of information for the system is reviews, which inherently carry subjective connotations;

In the next phase, work will be done on user interfaces by developing a front-end for the search system, incorporating specific features like snippet generation and results clustering. This engine will enable travelers to explore and filter accommodations based on preferences, such as location, room quality, staff service, or other factors identified during the analysis phase.

REFERENCES

[1] Apache solr, 2023. https://solr.apache.org/guide/6_6/introduction-to-solr-indexing.html, Last accessed on October 23, 2023.
[2] Kaggle, 2022. https://www.kaggle.com, Last accessed on September 30, 2023.
[3] Datafiniti, 2017. https://www.datafiniti.com, Last accessed on September 30, 2023.
[4] Datafiniti, 2017. http://www.ctan.org/pkg/acmart, Last accessed on September 30, 2023.
[5] Hotel reviews, 2017. https://www.kaggle.com/datasets/juhibhojani/hotel-reviews, Last accessed on September 30, 2023.
[6] Booking, 1996. https://www.booking.com, Last accessed on September 27, 2023.
[7] Reviews of london-based hotels, 2018. https://www.kaggle.com/datasets/PromptCloudHQ/reviews-of-londonbased-hotels, Last accessed on September 30, 2023.
[8] Datastock, 2018. https://datastock.shop, Last accessed on September 30, 2023.
[9] 515k hotel reviews data in europe, 2017. https://www.kaggle.com/datasets/jiashenliu/515k-hotel-reviews-data-in-europe, Last accessed on September 30, 2023.
[10] Pandas, 2023. https://pandas.pydata.org, Last accessed on October 9, 2023.
[11] Matplotlib, 2023. https://matplotlib.org, Last accessed on September 2, 2023.
[12] Numpy, 2023. https://numpy.org, Last accessed on September 2, 2023.
[13] Nltk, 2023. https://www.nltk.org, Last accessed on September 2, 2023.
[14] Information retrieval, 2023. https://link.springer.com/book/10.1007/978-3-319-93935-3, Last accessed on October 23, 2023.
[15] Solr tokenizers, 2023. https://solr.apache.org/guide/solr/latest/indexing-guide/tokenizers.html, Last accessed on November 2, 2023.
[16] Solr filters, 2023. https://solr.apache.org/guide/solr/latest/indexing-guide/filters.html, Last accessed on November 2, 2023.
[17] Indexing nested documents, 2023. https://solr.apache.org/guide/solr/latest/indexing-guide/indexing-nested-documents.html#schema-configuration, Last accessed on November 11, 2023.
[18] Queries in nested documents, 2023. https://solr.apache.org/guide/solr/latest/query-guide/searching-nested-documents.html, Last accessed on November 11, 2023.
[19] edismax, 2023. https://solr.apache.org/guide/7_7/the-extended-dismax-query-parser.html, Last accessed on November 4, 2023.
[20] Precision-recall area under the curve, 2023. https://scikit-learn.org/stable/auto_examples/model_selection/plot_precision_recall.html, Last accessed on November 7, 2023.
[21] Solr stop filter, 2023. https://solr.apache.org/guide/solr/latest/indexing-guide/filters.html#managed-stop-filter, Last accessed on November 7, 2023.

A ANNEXES

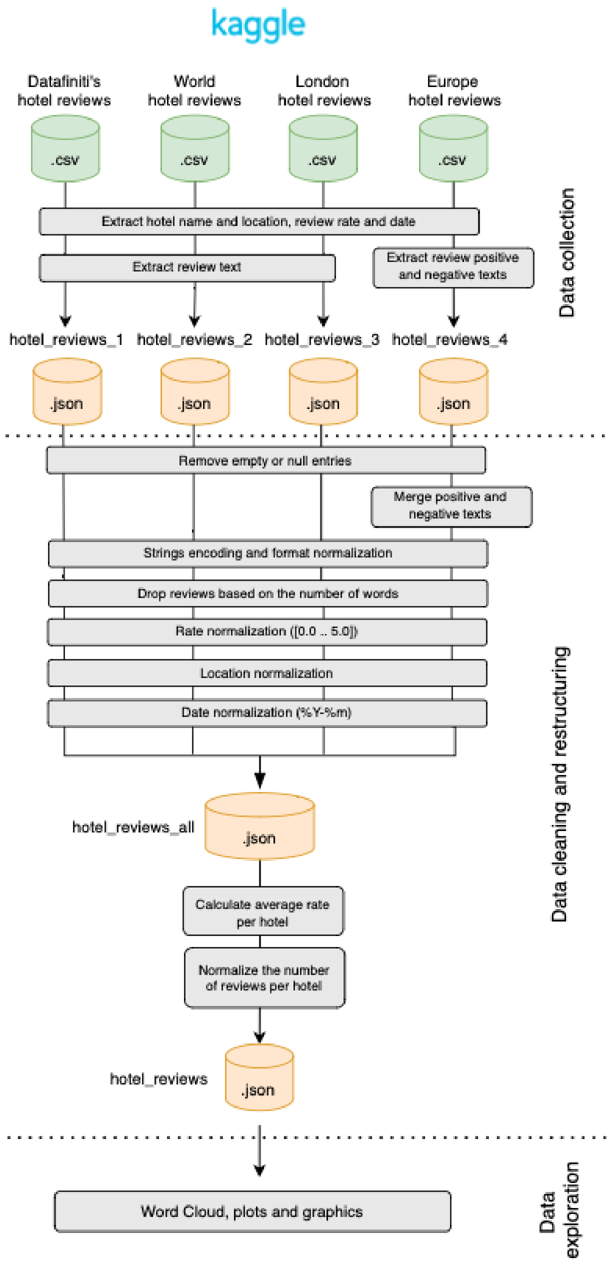
kaggle
Datafiniti's hotel reviews
World hotel reviews
London hotel reviews
Europe hotel reviews
.csv
Extract hotel name and location, review rate and date
Extract review text
Extract review positive and negative texts
Data collection
hotel_reviews_1
hotel_reviews_2
hotel_reviews_3
hotel_reviews_4
.json
Remove empty or null entries
Merge positive and negative texts
Strings encoding and format normalization
Drop reviews based on the number of words
Rate normalization ([0.0 .. 5.0])
Location normalization
Date normalization (%Y-%m)
hotel_reviews_all
Calculate average rate per hotel
Normalize the number of reviews per hotel
hotel_reviews
Data cleaning and restructuring
Word Cloud, plots and graphics
Data exploration

Figure 9: Data preparation pipeline

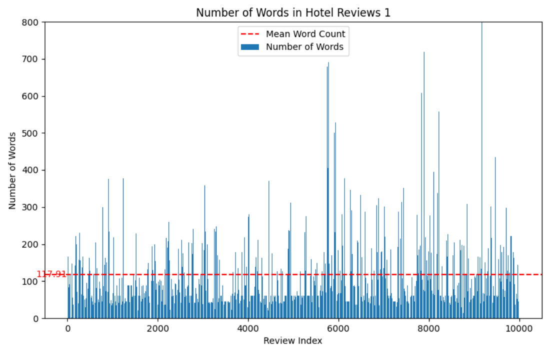

Figure 10: Average words per review in Datafiniti's Hotel Reviews dataset

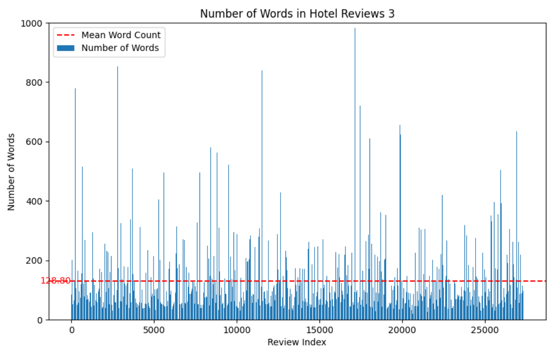

Figure 11: Average words per review in London Hotel Reviews dataset

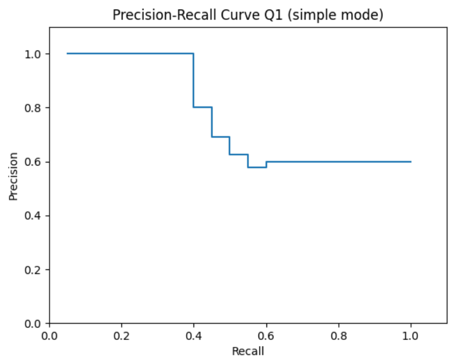

Figure 12: Q1 Precision-recall curve using simple system

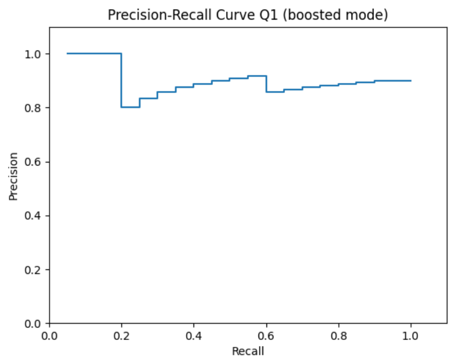

Figure 13: Q1 Precision-recall curve using boosted system

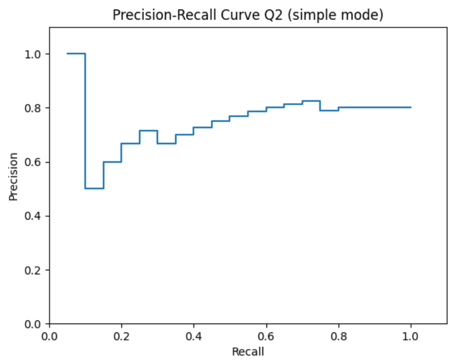

Figure 14: Q2 Precision-recall curve using simple system

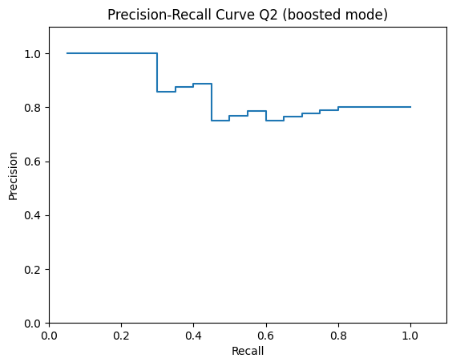

Figure 15: Q2 Precision-recall curve using boosted system

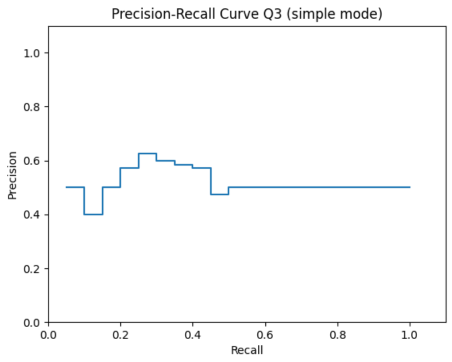

Figure 16: Q3 Precision-recall curve using simple system

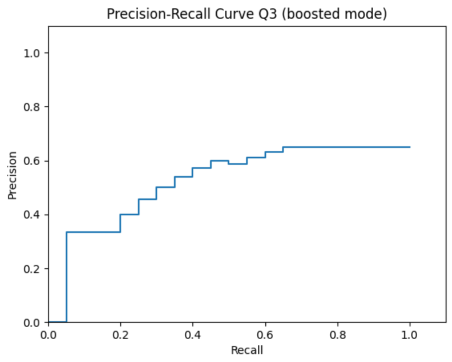

Figure 17: Q3 Precision-recall curve using boosted system

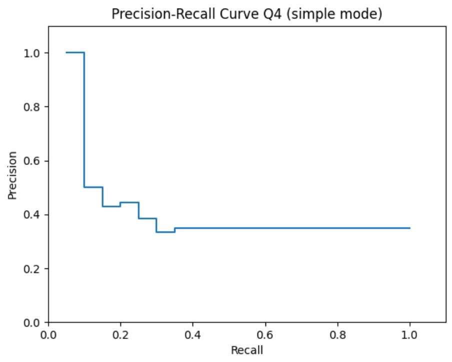

Figure 18: Q4 Precision-recall curve using simple system

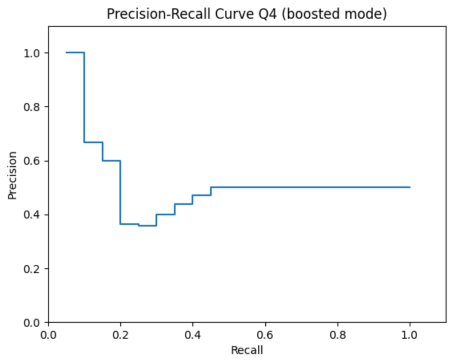

Figure 19: Q4 Precision-recall curve using boosted system