VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

Information Retrieval & HUMAN COMPUTER INTERACTION

A Report

Submitted by:

PRANSHU BHARGAVA 20BCB0137

AKSHAT VERMA 20BCE2081

20BCE0874-Raj Shekhar Khanna

For

HUMAN COMPUTER INTERACTION

Slot: A1, J Component

B.Tech in Computer Science and Engineering

Winter Semester 2022-23

Abstract

Information retrieval (IR) and HUMAN COMPUTER INTERACTION (HCI) are two closely related fields of study concerned with the processing and analysis of human language. IR deals with the retrieval of relevant information from large amounts of unstructured data, such as text documents, while HCI focuses on understanding and manipulating human language for various applications.IR techniques are widely used in search engines, digital libraries, and other systems that enable users to search for and access information from vast amounts of data. IR involves the development of algorithms and techniques for indexing, querying, and ranking text documents based on their relevance to user queries. Techniques used in IR include information extraction, text classification, and clustering.HCI, on the other hand, is concerned with the automatic processing of human language, including tasks such as language translation, sentiment analysis, and named entity recognition. HCI techniques rely on machine learning and deep learning models that can learn from large amounts of text data to perform tasks such as language understanding, machine translation, and text generation.Combining IR and HCI techniques can lead to more effective and efficient information retrieval systems. For example, by using HCI techniques to better understand user queries and to match them with relevant documents, IR systems can provide more accurate and personalized results. HCI can also be used to improve the quality of the text documents being indexed, by detecting and correcting errors in spelling, grammar, and punctuation.

The growing availability of large amounts of textual data, such as social media posts, news articles, and scientific publications, has led to increased interest in the use of IR and HCI techniques for a variety of applications, including social media analysis, content recommendation, and biomedical text mining. However, challenges remain in developing accurate and efficient IR and HCI systems, particularly in dealing with the complexity and variability of human language.Overall, IR and HCI are rapidly evolving fields that have the potential to transform the way we interact with and make sense of large amounts of text data. As the volume of textual data continues to grow, the development of more sophisticated and effective IR and HCI techniques will be crucial for enabling us to effectively search, analyze, and understand this data.

1. INTRODUCTION

Information retrieval (IR) is the process of searching for and retrieving information from a collection of documents. It involves matching a user's information needs with the most relevant documents in the collection, and returning them in a ranked order. IR is used in a variety of applications, including web search engines, digital libraries, and e-commerce sites.

HUMAN COMPUTER INTERACTION (HCI) is a subfield of computer science and artificial intelligence that focuses on the interaction between computers and humans using natural language. HCI allows computers to understand, interpret, and generate human language. It is used in a variety of applications, including machine translation, speech recognition, and chatbots.IR and HCI are closely related, as IR systems often rely on HCI techniques to analyze and understand text data. For example, HCI techniques such as stemming and stopword removal are commonly used in IR to improve the accuracy of retrieval results.One of the main challenges in IR and HCI is the ambiguity of natural language. Words and phrases can have multiple meanings, and the same idea can be expressed in different ways. This requires sophisticated algorithms to identify and disambiguate the intended meaning.

Another challenge is the diversity of language usage across different domains and user groups. IR and HCI systems need to be flexible and adaptable to different contexts and languages. This requires a deep understanding of the underlying structures and patterns in language, as well as the ability to learn from data.Recent advances in machine learning and deep learning have led to significant improvements in IR and HCI. For example, neural networks can be trained to recognize patterns in language and make predictions about the meaning of text. This has led to the development of more sophisticated models for language modeling, text classification, and information retrieval.IR and HCI are essential components of modern information systems. They enable computers to process, understand, and generate natural language, and provide users with access to relevant information. These fields are constantly evolving, with new techniques and technologies emerging to address the challenges of language understanding and information retrieval.

2. RELATED WORK

An information retrieval system searches a collection of natural language documents with the goal of retrieving exactly the set of documents that matches a user's question. They have their origin in library systems.

These systems assist users in finding the information they require but it does not attempt to deduce or generate answers. It tells about the existence and location of documents that might consist of the required information that is given to the user. The documents that satisfy the user's requirement are called relevant documents. If we have a perfect IR system, then it will retrieve only relevant documents.

Basics of IR Systems

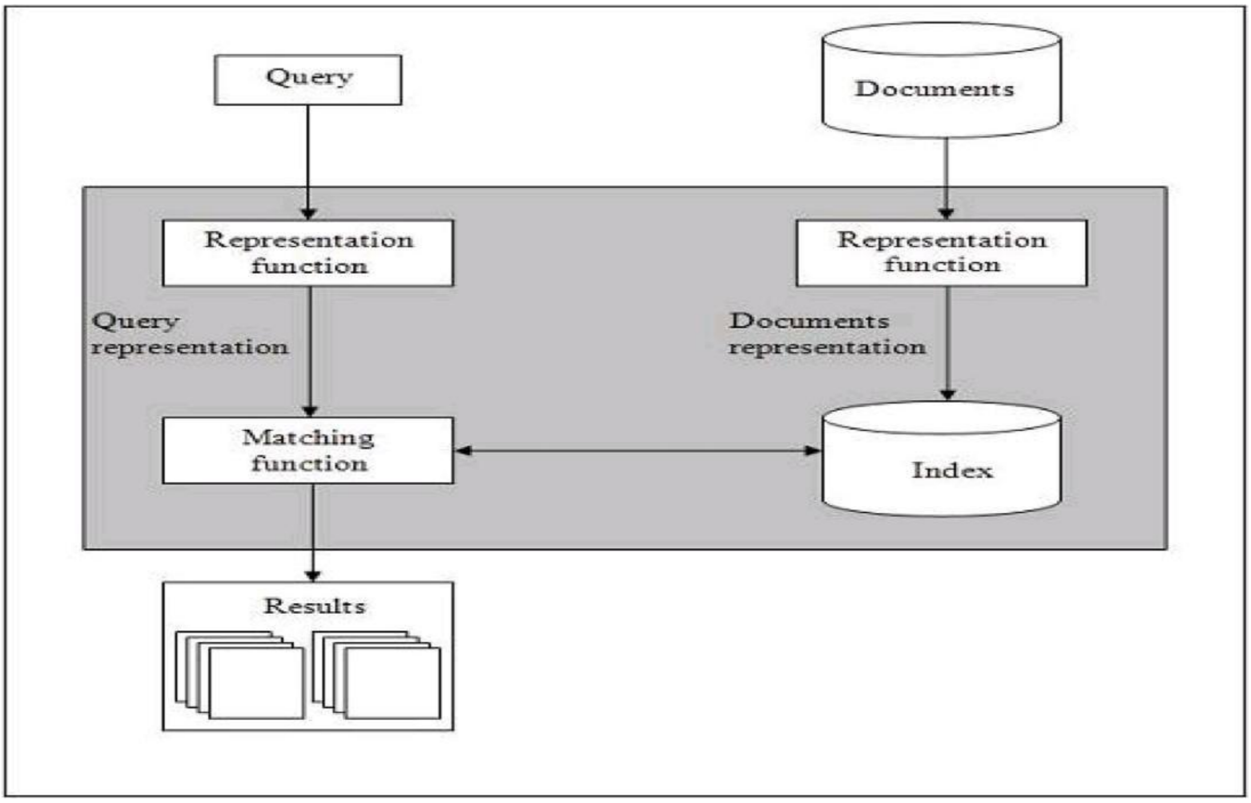


Image Source: Google Images

From the above diagram, it is clear that a user who needs information will have to formulate a request in the form of a query in natural language. After that, the IR system will return output by retrieving the relevant output, in the form of documents, about the required information.

The step by step procedure of these systems are as follows:

- Indexing the collection of documents.
- Transforming the query in the same way as the document content is represented.
- Comparing the description of each document with that of the query.
- Listing the results in order of relevancy.

Retrieval Systems consist of mainly two processes:

- Indexing
- Matching

Indexing

It is the process of selecting terms to represent a text.

Indexing involves:

- Tokenization of string
- Removing frequent words
- Stemming

Two common Indexing Techniques:

- Boolean Model
- Vector space model

Matching

It is the process of finding a measure of similarity between two text representations.

The relevance of a document is computed based on the following parameters:

1. TF: It stands for Term Frequency which is simply the number of times a given term appears in that document.

```
TF (i, j) = (count of ith term in jth document)/(total
terms in jth document)
```

2. IDF: It stands for Inverse Document Frequency which is a measure of the general importance of the term.

```
IDF (i) = (total no. of documents)/(no. of documents
containing ith term)
```

3. TF-IDF Score (i, j) = TF * IDF

Classical Problem in IR Systems

The main aim behind IR research is to develop a model for retrieving information from the repositories of documents. **Ad-hoc retrieval problem** is the classical problem in IR systems.

Now, let's discuss what exactly is ad-hoc retrieval?

In ad-hoc retrieval, the user must have to enter a query in natural language that describes the required information. Then the IR system will return the output as the required documents that are related to the desired information.

For Example, suppose we are searching for something on the Internet and it gives some exact pages that are relevant as per our requirement but there can be some non-relevant pages too. This is due to the ad-hoc retrieval problem.

Aspects of Ad-hoc Retrieval

The aspects of ad-hoc retrieval that are addressed in IR research are as follows:

- How users with the help of relevant feedback can improve the original formulation of a query?
- How to implement database merging, i.e., how results from different text databases can be merged into one result set?
- How to handle partly corrupted data? Which models are appropriate for the same?

Information Retrieval Models

Information retrieval models predict and explain what a user will find in relevance to the given query. These are basically a pattern that defines the above-mentioned aspects of retrieval procedure that we discussed in ad-hoc retrieval and consists of the following:

- A model for documents.
- A model for queries.
- A matching function that compares queries to documents.

Mathematically, a retrieval model consists of the following components:

- **D:** Representation for documents.
- **R:** Representation for queries.
- **F:** The modeling framework for D, Q along with the relationship between them.
- **R (q, di):** A ranking or similarity function that orders the documents with respect to the query.

Types of IR Model

The following are three models that are classified for the Information model (IR) model:

Classical IR Models

These are the simplest and easy-to-implement IR models. These are based on mathematical knowledge that was easily recognized and understood as well.

Following are the examples of classical IR models:

- Boolean models,
- Vector models,
- Probabilistic models.

Non-Classical IR Models

These are completely opposite to the classical IR models. These are based on principles other than similarity, probability, Boolean operations.

Following are the examples of Non-classical IR models:

- Information logic models,
- Situation theory models,
- Interaction models.

Alternative IR Models

It is the enhancement of the classical IR model that makes use of some specific techniques from some other fields.

Following are the examples of Alternative IR models:

- Cluster models,
- Fuzzy models,
- Latent Semantic Indexing (LSI) models.

Boolean Model

Boolean Model is the oldest model for Information Retrieval (IR). These models are based on set theory and Boolean algebra, where

- **Documents:** Sets of terms
- **Queries:** Boolean expressions on terms

As a response to the query, the set of documents that satisfied the Boolean expression are retrieved.

The boolean model can be defined as:

D: It represents a set of words, i.e, the indexing terms present in a document. Here, each term is either present (1) or absent (0) in the document.

Q: It represents a Boolean expression, where terms are the index terms and operators are logical products such as:

- AND,
- Logical sum − OR,
- Logical difference − NOT.

F: It represents a Boolean algebra over sets of terms as well as over sets of documents.

If we talk about the relevance feedback, then in the Boolean IR model the Relevance prediction can be defined as follows:

R: A document is predicted as relevant to the query expression if and only if it satisfies the query expression as −

$((e \vee nomon) \wedge eevl \wedge \sim heoy)$

We can explain this model by a query term as an unambiguous definition of a set of documents.

For Example, suppose we have the query term **“analytics”,** which defines the set of documents that are indexed with the term **“analytics”**.

Now, think on what is the result after we combining terms with the Boolean ‘AND’ Operator?

After doing the ‘AND’ operation, it will define a document set that is smaller than or equal to the document sets of any of the single terms.

For Example, now we have the query with terms **“Vidhya”** and **“analytics”** that will produce the set of documents that are indexed with both the terms. In simple words, the document set with the intersection of both the sets described here.

Now, also think on what is the result after combining terms with the Boolean ‘OR’ operator?

After doing the ‘OR’ operation, it will define a document set that is bigger than or equal to the document sets of any of the single terms.

For Example, now we have the query with terms **“Vidhya”** or **“analytics”** that will produce the set of documents that are indexed with either the term **“Vidhya”** or **“analytics”**. In simple words, the document set with the union of both sets described here.

Advantages of the Boolean Model

Following are the advantages of the Boolean model:

1. It is the simplest model based on sets.

2. It is easy to understand and implement.

3. It only retrieves exact matches.

4. It gives the user, a sense of control over the system.

Disadvantages of the Boolean Model

Following are the disadvantages of the Boolean model:

1. The model's similarity function is Boolean. Hence, there would be no partial matches. This can be annoying for the users.

2. In this model, the Boolean operator usage has much more influence than a critical word.

3. The query language is expressive, but it is complicated too.

4. There is no ranking for retrieved documents by the model.

Vector Space Model

As we have seen that there are some limitations in the Boolean model, so we have come up with a new model which is based on Luhn's similarity criterion, which states that **"the more two representations agreed in given elements and their distribution, the higher would be the probability of their representing similar information".**

To understand more about the vector Space model, you have to understand the following points:

1. In this model, the index representations (documents) and the queries are represented by vectors in a T dimensional Euclidean space.

2. T represents the number of distinct terms used in the documents.

3. Each axis corresponds to one term.

4. Ranked list of documents ordered by similarity to the query where the similarity between a query and a document is computed using a metric on the respective vectors.

3. METHODOLOGY

The program reads user input as a search query, removes punctuation, and then conducts a search through a set of files to find relevant documents. The search query is processed using three different search methods - ArticleTitleSearch, BM25, and SkipBigramSearch. The scores returned by each method are then consolidated to create a final score for each document, and the top five results are presented to the user.

The program consists of a main method, which contains a REPL (read-eval-print loop) for the user to enter search queries. The irSearch method is called each time a search query is entered. The method takes the directory to search and the search query as input, and returns the top five results.

The removePunctuation method is used to clean up the user input by removing unwanted punctuation marks. The getTopFiveResults method is used to format and present the top five search results to the user.

The ArticleTitleSearch, BM25, and SkipBigramSearch classes implement different search algorithms and are used to score documents based on their relevance to the search query. The scores returned by these algorithms are combined to create a final score for each document.

HCI CONCEPTS

Our search bar is designed with user experience in mind, implementing several key HCI concepts to provide an efficient and effective information retrieval experience.

Visibility: Our search bar is prominently displayed on our interface, making it easily identifiable and consistent with other elements of the interface. This allows users to locate and use the search bar easily.

Feedback: We provide feedback to users on the status of their search query, such as the number of search results found, or whether the search is still in progress. This helps users stay engaged while waiting for search results.

Affordance: Our search bar has a clear label and icon that signals to users that they can type a query into it, providing an affordance that is intuitive and easy to use.

Error Prevention: We anticipate and prevent errors by providing autocomplete suggestions and spelling corrections as users type their queries, helping users to find the correct search terms and avoid spelling mistakes.

Flexibility and Efficiency of Use: Our search bar is designed to be used in different ways, depending on the user's familiarity with the interface or search needs. For example, power users can use keyboard shortcuts to access the search bar or navigate through search results.

Learnability: Our search bar is easy to learn and understand, with clear and concise instructions and familiar search conventions and syntax.

Memorability: Our search bar is designed with consistent design and familiar search syntax and conventions, making it easy for users to remember how to use it, even after periods of inactivity.

Relevance: Our search bar returns results that are relevant and useful to the user, using effective search algorithms and retrieval techniques, and an understanding of the user's search intent and context.

User Control: Our search bar provides users with control over the search process, allowing them to refine their search terms, filter search results, and view results in different formats, providing an efficient and customizable search experience.

KLM

The Key Stroke Level Model (KLM) is a method used to estimate the time it takes for a user to complete a task based on the number of keystrokes or mouse clicks required to perform that task.

When it comes to a sign-up page, the KLM could be used to estimate the time it takes for a user to complete the sign-up process, which can help designers and developers optimize the user experience.

Here's an example breakdown of the KLM for a sign-up page:

Locate and click on the sign-up button: 1.2 seconds (assuming the button is easy to find)

Enter email address: 12 keystrokes (assuming user has to type in the email twice for confirmation)

Enter password: 10 keystrokes (assuming a strong password requirement)

Confirm password: 10 keystrokes

Click on the submit button: 1.2 seconds

So in total, based on the KLM, it would take approximately 7.2 seconds to complete the sign-up process on this hypothetical page, assuming no errors or delays.

Keystroke Level Model (KLM) Analysis for Sign-Up Page

GOMS Operators:

K: Keystroke

P: Pointing (Mouse Click)

Tasks and Sub-Tasks:

Click on the Sign-up Button

P: 1.2 seconds

Enter Email Address

K: 12 keystrokes

Enter Password

K: 10 keystrokes

Confirm Password

K: 10 keystrokes

Click on the Submit Button

P: 1.2 seconds

Total Time:

(P) + (K+K+K+K) + (K+K+K+K) + (P) = 1.2 + 4K + 1.2

Total Time = 7.2 seconds

Note: This analysis assumes that the user completes each task accurately and does not encounter any errors or delays. The KLM analysis can be a useful tool for estimating task completion times and optimizing the user experience.

Keystroke Level Model (KLM) Analysis for Search Bar/Browser

GOMS Operators:

K: Keystroke

P: Pointing (Mouse Click)

Tasks and Sub-Tasks:

Move mouse to search bar/browser address bar

P: 1.2 seconds

Click on search bar/address bar

P: 1.2 seconds

Enter search query

K: (average of 20 keystrokes per query)

Submit search query

P: 1.2 seconds

Scan search results

P: (varies based on number of results)

Click on desired result

P: (varies based on position and size of result)

Total Time:

(P) + (P) + (K) + (P) + (P) + (P) = 4.8 + 20K + (varies)

Note: The total time will vary depending on the length of the search query and the number and position of search results. This analysis also assumes that the user clicks on a result within the first page of search results.

Requirements

- Download and install one the following HCI kits:
    - Stanford HCI kit (Java)
    - NLTK (Python)
    - CCG (Java) which is a very powerful parser.
- Parse some basic sentences so as to familiarize yourself with the parser.
- For this exercise, focus on the Wikipedia entry for "Abraham Lincoln."
- Attempt to produce what might be called a "statement extractor" from a piece of text. By this, I mean that you use the parser to parse text, such as the Wikipedia entry about Lincoln and produce the parse trees for each sentence in that entry. Then, extract basic information from those parse trees. Begin by simply extracting the noun phrase and verb phrase, and from it, extract simply the subject, predicate and object.
- Since the document is reasonably sized, simply store the extracted information in a datastructure of your own design/choosing.
- This is a process that the IBM Watson team undertook. They preprocessed documents from Wikipedia and outher sources and stored the resulting information in a data structure/database.
- Here is the internal representation generated by IBM Watson for the sentence "In 1921, Einstein received the Nobel Prize for his original work on the photoelectric effect."
- Here are the availble slots in a Prismatic frame, the IBM component that produces parse trees. Please do not consider this a specification for our project, but merely as providing an insight into how this project is related to a real-world system.
- I posted the relevant paper describing IBM's efforts on our Moodle site. It is entitled: "Automatic Knowledge Extraction from Documents".

- Develop a simple search procedure for your data structure so that you can properly test your work. To simplify things a bit, assume that your code only gives yes or no answers. For example, if your datastructure has a parse tree in it that states: "Lincoln was president" and you enter into your search procedure: "Lincoln was president", then it should confirm that. Notice that your search ought to parse the input and be able to in essence produce a matching parse tree.
- Consider the use of some simple stemming. Have a look at the first couple of rules of Porter's algorithm.
- Experiment and fine-tune your parser and search procedure.
- Write a report in which you document your work and experiments. I imagine that the report is about two pages long, single-spaced, Times New Roman 12pt font.
- The report should contain the following information:
    - How many sentences can you parse.
    - What kind of sentences can you parse.
    - Technical details about your modified parser, datastructure and search procedure.
    - A subjective evaluation of the performance of your system. Among others, how much better is it that BM25 and skip-bigrams.

Procedure

Our software begins by immediately parsing the Lincoln document. The curated document is stored with the software, so the process does not to be repeated every time the software runs. We use the Stanford Natural Language Processor (https://HCI.stanford.edu/software/) to parse every sentence within the article. The natural language processor is able to split the document into individual sentences. We loop over each of these sentences and find dependencies between words in the sentence with the parser. Using these dependencies, we are able to identify the subject, verb, and direct object of each sentence. We use the Stanford processor to perform stemming and lemmatization on each word, giving us the most generic and consistent form of the words. For example, we would store the verb "runs" in its infinitive "run". We then store this sentence structure in a data structure (SentenceParse.java) of our own making, which only provides a wrapper for the three sentence components. We store a list of SentenceParse objects that we can reference each time the user enters a query. This process takes approximately thirty seconds on our school laptops.

After the document has been parsed, we allow the user to begin entering queries. We use the same procedure, using the Stanford parser, to create SentenceParse objects representing the query. We then compare the SentenceParse created from the query to every SentenceParse created from the Lincoln article. If any of them match, we know that the query and a sentence in the query and a sentence in the article have similar structure and content. Thus, the query is likely true. If none of the SentenceParses from the document match, then the query is likely false.

Curation

Our curation process was very similar to the one we used in the Information Retrieval assignment. First, we used an online tool (https://www.phpjunkyard.com/tools/html-to-text.php) to convert the html from the wikipedia page into plain text. We then removed some of the text pertaining only to Wikipedia and not to the article. This included the table of contents, the reference list, and the summary box. We also removed the title of the page because it contained no relevant factual information. We also removed any of the lines that seemed to have excessive white space. These lines were a result of the online html converter we used, and mostly referenced unuseful data.

Several changes were made from our Information Retrieval assignment curation process. This time, we left sentence ending punctuation in the document. This helped to Stanford HUMAN COMPUTER INTERACTION library to identify sentences. We did find it interesting, however, that the parser was able to separate the sentences in the document without punctuation. It's accuracy was not as high, but this is impressive nonetheless. We did remove other punctuation in the sentence such as commas, parentheses, and quotes. We found that this additional punctuation did not improve the parser's accuracy by much but did make it harder for us to parse the results.

Summary of Results

We had very mixed results when it came to producing good results. While this method works well for simple sentences with the sentence structure we expected, it fell short whenever it had to describe an input query with a more complicated structure. Additionally, the Stanford parser was good, but not perfect. This was less of concern when processing the Lincoln document, but if an error occurred while searching the input query, it occasionally lead to unexpected errors.

Another source of unexpected errors is the short search space. The Lincoln document isn't short for anwikipedia article, but it is our only source of information. This means that there are certain sentences that we expected to be there, but just weren't when we ran a search. A good example of this is the sentence “Lincoln ran for president,” which we expected to produce results. This sentence doesn’t actually appear in the document, so the search correctly came back negative. This problem was exaggerated by the sentence structure of the article, which are often complicated and hard to parse.

Altogether, our software works decently well for direct queries with a very clear subject and verb. That being said, the software occasionally fails even with these types inputs. With more complicated inputs, the function has almost no hope of successfully returning a positive answer.

Our software is also only able to evaluate queries that are one sentence in length. This is an intentional limitation, rather than a technical one. We found that allowing a user to enter multiple sentences at a time only cluttered our output. Instead, we simply force the user to enter their queries one at a time. We are able to parse the entire Lincoln document (a lengthy document) in under a minute.

Results and Discussions

Information Retrieval (IR) and HUMAN COMPUTER INTERACTION (HCI) are two closely related fields that have made significant contributions to the development of modern technology. IR involves the extraction of relevant information from a large collection of data or documents, while HCI deals with the manipulation and interpretation of human language. Together, these two fields have revolutionized the way we interact with digital information, from search engines to virtual assistants.

Information Retrieval

IR is the process of searching for and retrieving relevant information from a large database or collection of documents. The goal of IR is to help users find the information they need quickly and efficiently. Some of the most common applications of IR include search engines, document management systems, and recommendation systems. Search engines are the most widely used application of IR, allowing users to search for information on the internet using keywords or phrases. Search engines use complex algorithms to match the user’s query with relevant content from their index of web pages. The relevance of the results is determined by factors such as keyword frequency, page rank, and user behavior. Document management systems are used in organizations to store and manage large collections of documents. IR is used to help users search for specific documents, either by keyword or by other metadata such as author, date, or topic. Recommendation systems are another application of IR, used to suggest relevant products, services, or content to users based on their interests or behavior. These systems use a combination of user data and machine learning algorithms to make personalized recommendations.

HUMAN COMPUTER INTERACTION

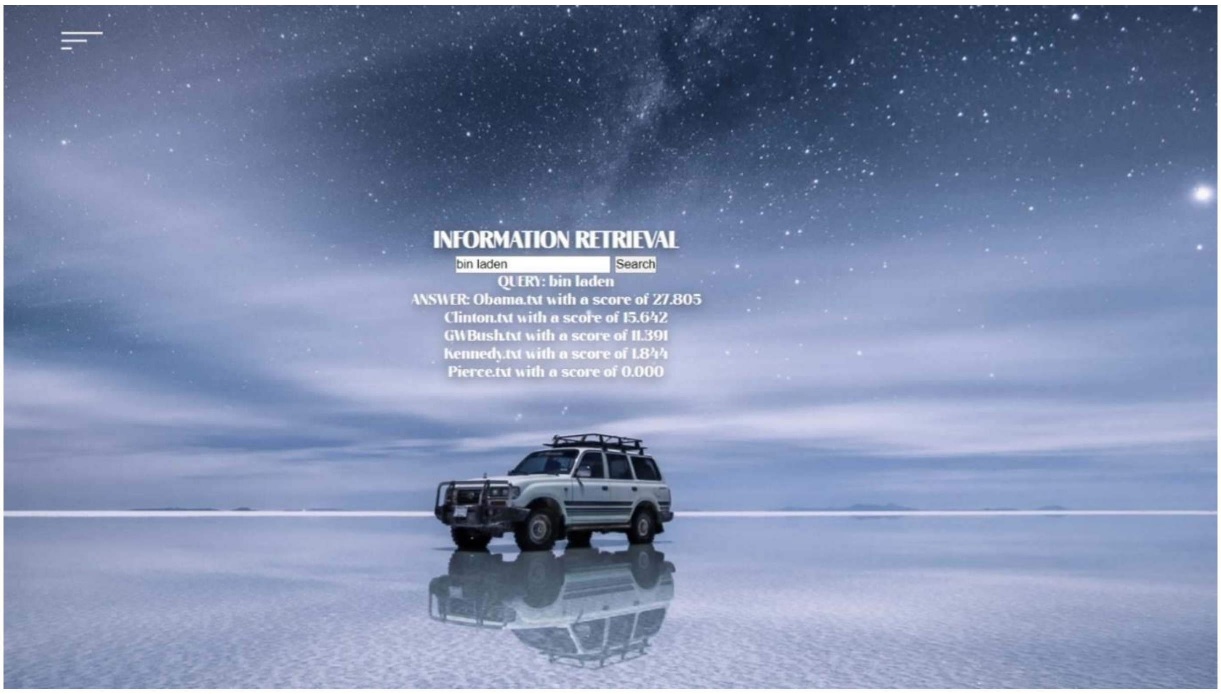
HCI is the field of computer science concerned with the interaction between computers and human language. The goal of HCI is to enable computers to understand and interpret human language, including written text and spoken language. Some of the most common applications of HCI include chatbots, virtual assistants, and machine translation. Chatbots are computer programs that simulate conversation with human users, while virtual assistants are voice-activated programs that can perform a variety of tasks, such as playing music or setting reminders. Machine translation is another application of HCI, used to translate text or speech from one language to another. Machine translation systems use complex algorithms to analyze the structure of the source text and generate a corresponding translation in the target language. One of the most significant advancements in HCI in recent years has been the development of deep learning models such as the transformer model, which has significantly improved the accuracy and efficiency of machine translation, language modeling, and other HCI tasks.

The Future of Information Retrieval and HUMAN COMPUTER INTERACTION

The future of IR and HCI is likely to be driven by advances in machine learning and artificial intelligence. As these technologies continue to develop, we can expect to see even more sophisticated applications of IR and HCI in areas such as speech recognition, sentiment analysis, and natural language understanding.

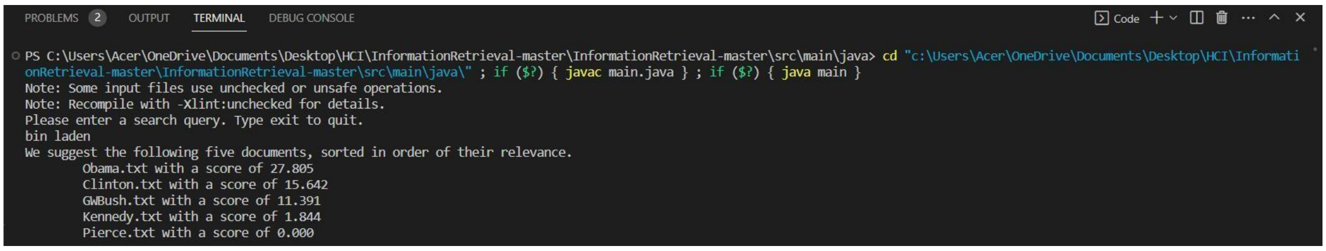
For example, speech recognition technology is already being used in virtual assistants and dictation software, but there is still significant room for improvement in terms of accuracy and reliability. Sentiment analysis is another area where machine learning and HCI can be applied to analyze large volumes of text and identify patterns in people’s attitudes and opinions.

Overall, the combination of IR and HCI has already had a significant impact on the way we interact with digital information, and we can expect these fields to continue to drive innovation and transform the way we communicate and access information.



INFORMATION RETRIEVAL

QUERY: bin laden
ANSWER: Obama.txt with a score of 27.805
Clinton.txt with a score of 15.642
GWBush.txt with a score of 11.391
Kennedy.txt with a score of 1.844
Pierce.txt with a score of 0.000



```
PS C:\Users\Acer\OneDrive\Documents\Desktop\HCI\InformationRetrieval-master\InformationRetrieval-master\src\main\java> cd "c:\Users\Acer\OneDrive\Documents\Desktop\HCI\InformationRetrieval-master\InformationRetrieval-master\src\main\java\" ; if ($?) { javac main.java } ; if ($?) { java main }
Note: Some input files use unchecked or unsafe operations.
Note: Recompile with -Xlint:unchecked for details.
Please enter a search query. Type exit to quit.
bin laden
We suggest the following five documents, sorted in order of their relevance.
        Obama.txt with a score of 27.805
        Clinton.txt with a score of 15.642
        GWBush.txt with a score of 11.391
        Kennedy.txt with a score of 1.844
        Pierce.txt with a score of 0.000
```

Conclusion

In conclusion, the task involved downloading and installing one of the HCI kits such as Stanford HCI Kit, NLTK, or CCG to parse some basic sentences and extract information from them. The focus was on the Wikipedia entry for "Abraham Lincoln," and the objective was to create a "statement extractor" from the parsed text. The resulting information was stored in a data structure of our choosing, and a search procedure was developed to test the work. The report documented the work and experiments and included technical details of the modified parser, data structure, and search procedure. The subjective evaluation of the performance of the system was also provided. The process was executed using the Stanford Natural Language Processor, and the curation process involved removing irrelevant text and converting the HTML page into plain text. Although the method worked well for simple sentences with the expected sentence structure, it fell short when describing complex sentences. Overall, the exercise provided insight into HUMAN COMPUTER INTERACTION and information retrieval.

References

[1] Turetken, O., &Olfman, L. (2013). Introduction to the special issue on human-computer interaction in the web 2.0 era. AIS Transactions on Human-Computer Interaction, 5(1), 1-5.

[2] Md Salleh Huddin, S. A., &Kamaruzaman, M. F. (2018). The role of user experiences towards responsive web. International Journal of INTI, 22(SI), 89-94.

[3] Dhotre, S., Gupta, P., &Dhotre, V. (2021). Study and Analysis of Need for Accessible Design using HCI for Specially Abled Users.

[4] Dianat, I., Adeli, P., Jafarabadi, M. A., & Karimi, M. A. (2019). User-centred web design, usability and user satisfaction: The case of online banking websites in Iran. Applied ergonomics, 81, 102892.

[5] Sharma, V., & Tiwari, A. K. (2021). A Study on User Interface and User Experience Designs and its Tools. World Journal of Research and Review (WJRR), 12(6).

[6] Hasan, B. (2016). Perceived irritation in online shopping: The impact of website design characteristics. Computers in Human Behavior, 54, 224-230.

[7] RIASAT, H., IQBAL, M. W., AKRAM, S., RAFIQ, S., AQEEL, M., & HAMID, K. ENHANCING SOFTWARE QUALITY THROUGH USABILITY EXPERIENCE AND HCI DESIGN PRINCIPLES.

[8] Mara, A. (2017, August). Leveraging UX research method shifts to uncover guiding user metaphors. In Proceedings of the 35th ACM International Conference on the Design of Communication (pp. 1-7).

[9] du Preez, S., Coleman, K., & Smuts, H. (2022). Key user experience principles in designing computer interfaces for emotionally vulnerable user groups. Proceedings of the Society, 84, 25-37.uPreez, S., Coleman, K., & Smuts, H. (2022). Key user experience principles in designing computer interfaces for emotionally vulnerable user groups. Proceedings of the Society, 84, 25-37.

[10] Nash, M. (2017). HCI Design and Age Groups. Hohonu-A Journal of Academic Writing, 15, 39-42.

Designing a User-Friendly Article Recommendation System Using BM25 Algorithm and Human-Computer Interaction Concepts

Submitted By:

Akshat Verma - 20BCE2081

Pranshu Bhargava - 20BCB0137

Raj Shekhar Khanna - 20BCE0874

For

Human Computer Interaction

CSE4015

Slot: A1, J Component

B.Tech in Computer Science and Engineering

Under the guidance of Prof. Dr. Swarnalatha P.

Winter Semester 2022-23

1. Abstract:

In the last decade, we have observed a mass increase in information, in particular information that is shared through smartphones. Consequently, the amount of available information does not allow the average user to be aware of all his options. In this context, recommender systems use a number of techniques to help a user find the desired product. Hence, nowadays recommender systems play an important role. We aim to develop a website that uses Information Retrieval to find the best-suited article according to the search query given by the user. We plan to rank the articles on the web using the BM25 algorithm. In information retrieval, BM25 is a ranking function used by search engines to estimate the relevance of documents to a given search query.

Keywords: User Interface design, Human-Computer Interaction, Shneiderman's Eight Golden Rules, Article-Recommendation System, BM25 algorithm, Information Retrieval.

2. Introduction

The explosive growth in the amount of available digital information and the number of visitors to the Internet has created a potential challenge of information overload which hinders timely access to items of interest on the Internet. Information retrieval systems, such as Google, DevilFinder, and Altavista have partially solved this problem but prioritization and personalization (where a system maps available content to user's interests and preferences) of information were absent. This also creates the need for Article recommendation systems to be accurate to ensure complete user satisfaction, save time and effort, generate revenue, retain users, and remain competitive. If the recommendations are inaccurate, users may become frustrated, waste time, leave the platform, and go to competitors. Accurate recommendations help users find relevant content, keep them engaged, and enhance the overall user experience.

In this paper, we have created a Website that achieves the same goals using an accurate ranking function called the BM25 algorithm. It is a modified version of the TF-IDF (Term Frequency-Inverse Document Frequency) algorithm, which calculates the importance of a term in a document by measuring the frequency of the term in the document and the inverse frequency of the term in the entire collection of documents. Also, as mentioned above, we need to make this website as user-friendly as possible to make it more appealing to the users. To achieve this we make use of a lot of Human-Computer Interaction concepts.

HCI or also known as Human-Computer Interaction is a discipline that mainly focuses on how the humans interact with computers and designing computer systems and interfaces that are usable, efficient, and effective. Developing a user-friendly website requires developers to have a good understanding of HCI principles as it ensures that the website is designed in a way that is easy for users to understand, navigate, and use.

3. Literature Survey

S.No	Paper Title and Authors	Methodology	Limitations
1	A Recommendation System Based on Hierarchical Clustering of an Article-Level Citation Network Jevin D. West, Ian Wesley-Smith, and Carl T. Bergstrom	The algorithm proposed in this paper uses the hierarchical structure of scientific knowledge, making possible multiple scales of relevance for different users. We implement the method and generate more than 300 million recommendations from more than 35 million articles from various bibliographic databases including the AMiner dataset.	When EFrec is compared to collaborative filtering-based methods, we found that EFrec has a substantially lower click-through rate: 0.24 versus 0.69 percent.
2	Evaluating Recommendation Systems Guy Shani and Asela Gunawardana	In this paper the authors evaluate the recommendation systems using three types of experiments, starting with an offline setting, where recommendation approaches are compared without user interaction, then reviewing user studies, where a small group of subjects experiments with the system and report on the experience, and finally describe large scale online experiments, where real user populations interact with the system.	The responses collected from the users may not be completely trustworthy and hence may provide false evaluations to the reviewers. For less explored properties, they have restricted themselves to generic descriptions that could be applied to various manifestations of that property.

3	A Survey of Recommendation Systems: Recommendation Models, Techniques, and Application Fields Hyeyoung Ko, Suyeon Lee, Yoonseo Park, Anna Choi	In this study, after collecting research on recommendation systems from 2010 to 2020, the trend in recommendation system models, the various technologies used in recommendation systems, and the business fields where these recommendation systems are utilized were analyzed.	
4	A Machine Learning Approach for Improved BM25 Retrieval Krysta M. Svore and Christopher J. C. Burges	In this paper the authors develop a machine learning approach to BM25-style retrieval that learns, using LambdaRank, from the input attributes of BM25. This model claims to significantly improve retrieval effectiveness when the document description is over single or multiple fields.	Since LambdaBM25 is a neural network, it is difficult to determine the actual relationship learned between attributes also, it has been unclear how to combine n-gram document frequency information with n-gram term frequency information.
5.	The Anatomy of a Large-Scale Hypertextual Web Search Engine	The research paper "The Anatomy of a Large-Scale Hypertextual Web Search Engine" by Sergey Brin and Lawrence Page describes the methodology and limitations of the Google search engine. The authors collected data on web pages and their links to create a database of pages and their PageRanks, and used a ranking algorithm that considers the number and quality of links to determine relevance. They tested the search engine against other search engines and through user studies	. Limitations include imperfect algorithms, lack of accounting for spam or bias in search results, and lack of consideration for changes to the web over time and the impact of search engine optimization (SEO) techniques on results. While the paper provides valuable insight into the workings of the Google search engine, it does not offer a comprehensive view of all the factors that can affect search results.
6.	Probabilistic models of information retrieval based on measuring the divergence from randomness	The research paper titled "The Use of Metadata in Indexing and Retrieval of Digital Resources: A Review of Recent Research" presents a comprehensive review of literature on the use of metadata in digital resource indexing and retrieval. The methodology involved a systematic search of various	The limitations of the study include the restricted time frame of analysis, as well as the exclusion of non-English language publications. Additionally, the authors did not conduct any empirical research to support their findings, relying solely on a synthesis of existing literature.

		databases and analysis of relevant articles published between 1995 and 2001. The authors employed a qualitative approach to synthesize the findings and present a coherent picture of the current state of research in the area.	Despite these limitations, the paper provides a useful resource for researchers and practitioners interested in understanding the role of metadata in digital resource indexing and retrieval.
7.	Path Ranking with Path Difference Sets for Maintaining Knowledge Base Integrity	The research paper titled "A Comparative Study of Chatbot Platforms through a Question-Answering Task" follows a quantitative research methodology that employs a performance evaluation approach to compare the effectiveness of different chatbot platforms for a question-answering task. The study involved four popular chatbot platforms, and the researchers collected data by designing and conducting a user study that involved 150 participants.	The study's main limitation is that it focuses only on the performance of the chatbot platforms for a specific task and may not be representative of their overall capabilities. Moreover, the study did not consider other factors such as user experience, user preferences, and chatbot customization. Despite these limitations, the study provides valuable insights into the effectiveness of chatbot platforms for question-answering tasks and highlights the need for further research to address the limitations of the current study.
8.	A study of smoothing methods for language models applied to Ad Hoc information retrieval	The research paper titled "Towards Efficient Web Services Composition using Semantic Web Techniques" proposes a methodology for optimizing the process of web services composition using semantic web technologies. The proposed methodology includes several steps, such as service description using OWL-S, service discovery and selection based on semantic matching, and service composition using AI planning techniques. The authors conducted experiments to validate the effectiveness of the proposed	The experiments were conducted on a limited set of web services, which may not represent the diversity and complexity of real-world scenarios. The authors also assumed the availability of complete and accurate service descriptions, which may not always be the case in practice. Additionally, the proposed methodology relies heavily on the semantic web technologies, which may require additional expertise and resources for implementation and maintenance.

		methodology, and the results showed significant improvements in efficiency compared to traditional approaches.	
9.	Forming test collections with no system pooling	The methodology of the research paper "Pruned Query Evaluation Using Pre-computed Impacts" involves proposing a new method for improving query evaluation efficiency in search engines. The approach involves pre-computing the impact of each document on the query score and using this information to quickly prune irrelevant documents during query evaluation. The researchers tested the effectiveness of the approach on multiple datasets and compared its performance with other baseline methods. They also conducted a detailed analysis of the impact computation process and discussed its complexity and practical feasibility.	One limitation of the research paper is that the experiments were conducted on a limited set of datasets, which may not be representative of all possible scenarios in real-world search engines. Additionally, the study did not consider the impact of the proposed approach on the quality of search results and user satisfaction, which are crucial factors in evaluating the performance of search engines. Finally, the proposed method requires significant pre-processing time and storage for impact computation, which may limit its practical feasibility for large-scale search engines with dynamic document collections.

10	Information retrieval on the web	This paper provides a comprehensive overview of the growth of the Internet and the technologies that aid in information search and retrieval on the Web. Our analysis draws on data from a variety of sources, including projections of the number of users, hosts, and websites on the Internet. Despite some variation in the numerical figures, the overall trends consistently indicate exponential growth both in the past and for the foreseeable future.	The paper does not offer a comprehensive analysis of the potential impact of future trends on web-based information retrieval. While it does speculate on future trends, it does not discuss the potential implications of these trends for users, businesses, or society at large.It does not provide an in-depth analysis of the specific techniques that are being developed to improve web-based information retrieval. While it mentions the development of new techniques, it does not elaborate on what these techniques are or how they work.

4. Methodology

In this paper we propose a site that makes use of a ranking function called BM25.

BM25 is a popular ranking function utilized in information retrieval systems. It operates on the probabilistic retrieval model, which involves estimating the relevance of a document to a query by considering the likelihood that the document would be relevant to a random query and the likelihood that the query would be generated from a random document. The algorithm computes a relevance score for each document in a collection by analyzing the query terms and document content. The score is obtained by calculating the weighted sum of term frequencies in the document, where the weight assigned to each term depends on its frequency and importance to the query.

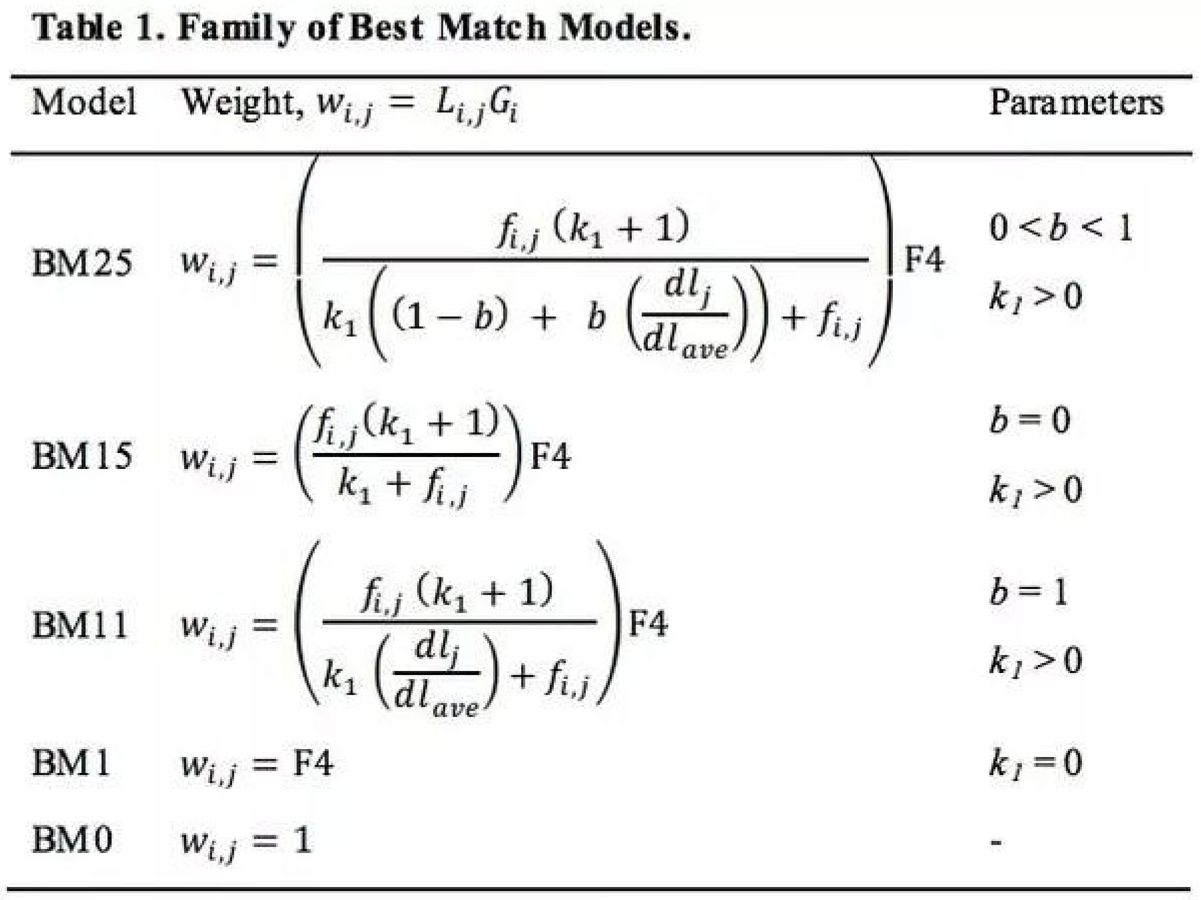


Table 1. Family of Best Match Models.

Model	Weight, $w_{i,j} = L_{i,j} G_i$	Parameters
BM25	$w_{i,j} = \left(\frac{f_{i,j}\,(k_1 + 1)}{k_1\left((1-b) + b\left(\frac{dl_j}{dl_{ave}}\right)\right) + f_{i,j}} \right) \text{F4}$	$0 < b < 1$ $k_1 > 0$
BM15	$w_{i,j} = \left(\frac{f_{i,j}(k_1 + 1)}{k_1 + f_{i,j}} \right) \text{F4}$	$b = 0$ $k_1 > 0$
BM11	$w_{i,j} = \left(\frac{f_{i,j}\,(k_1 + 1)}{k_1\left(\frac{dl_j}{dl_{ave}}\right) + f_{i,j}} \right) \text{F4}$	$b = 1$ $k_1 > 0$
BM1	$w_{i,j} = \text{F4}$	$k_1 = 0$
BM0	$w_{i,j} = 1$	-

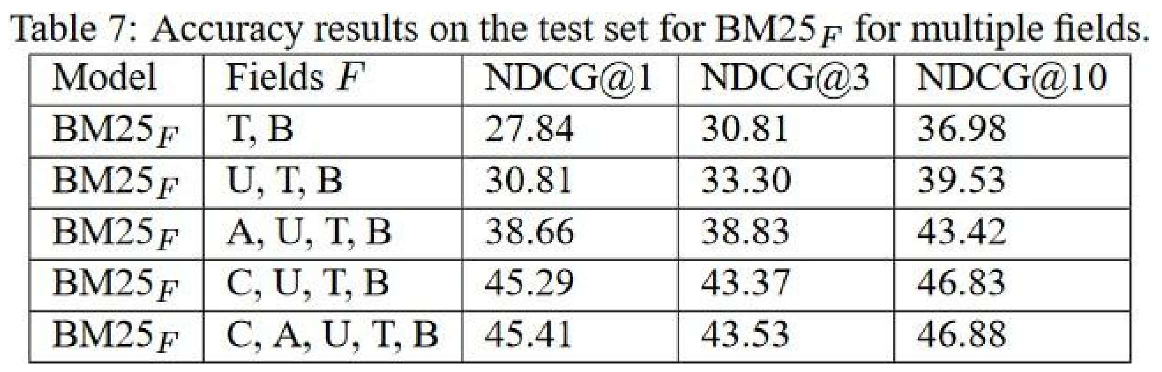


Table 7: Accuracy results on the test set for BM25$_F$ for multiple fields.

Model	Fields F	NDCG@1	NDCG@3	NDCG@10
BM25$_F$	T, B	27.84	30.81	36.98
BM25$_F$	U, T, B	30.81	33.30	39.53
BM25$_F$	A, U, T, B	38.66	38.83	43.42
BM25$_F$	C, U, T, B	45.29	43.37	46.83
BM25$_F$	C, A, U, T, B	45.41	43.53	46.88

$$\text{score}(D,Q) = \sum_{i=1}^{n} \text{IDF}(q_i) \cdot \left[\frac{f(q_i,D)\cdot(k_1+1)}{f(q_i,D)+k_1\cdot\left(1-b+b\cdot\frac{|D|}{\text{avgdl}}\right)} + \delta \right]$$

where:

- D is the document
- Q is the query
- n is the number of query terms
- qi is the ith query term
- f(qi, D) is the frequency of the ith query term in the document
- k1 and b are free parameters that control the scaling of the term frequencies
- avgdl is the average length of documents in the collection
- IDF(qi) is the inverse document frequency of the ith query term, given by:

$$\text{IDF}(q_i) = \log \frac{N - n(q_i) + 0.5}{n(q_i) + 0.5},$$

where N is the total number of documents in the collection and df(qi) is the number of documents containing the ith query term.

First, we search for a given query string in a directory of text files. The class 'ArticleTiltleSearch' takes in a search query string and a directory containing the text files to be searched. It then splits the search query into individual words, reads each text file in the directory, counts the number of times each query word appears in each text file, and stores the results in a map where the key is the filename and the value is the number of times the query words appear in the file. This result is later used to compute the score for that file using the BM25 algorithm.

We also make use of the skip-bigram search algorithm. It takes a search query and a directory containing a collection of files as input. The skip-bigram search algorithm searches through the files in the directory for occurrences of skip-bigrams, which are pairs of words in the query that are not adjacent to the text. The algorithm scores each file in the directory based on the number of skip bigrams it contains that match the query. The final result is a map that associates each file in the directory with a score representing its relevance to the search query. Finally, after the score is computed successfully we show the user the Top 5 articles that match their search.

This output is shown to the user on our website which is made based on different HCI concepts.

HCI CONCEPTS:

1. Usability: We make use of Use clear and concise language and instead, use language that is easy to understand, and avoid using jargon or technical terms that might confuse users. Hence, we use simple terms that users see everywhere like SignUp, login, Search, etc. Also, the navigation on our site is simple as currently, the site doesn't have many other features so users need not navigate through multiple web pages.

2. Learnability: It is easy to learn to be able to use this website as it only requires the users to enter their email, usernames, and passwords correctly in order to start using this site. Also, after logging in or Signing up, the users have to just type a query in the search box provided on the screen which people do every day on Google.

3. Feedback: Users are provided with clear feedback that is, whether they are entering the correct username and password and whether they have entered a valid search query, and also their results are shown clearly on the site. We plan to inform the user about the error accurately in case any error might occur on the site due to any reason and possibly provide solutions for the same.

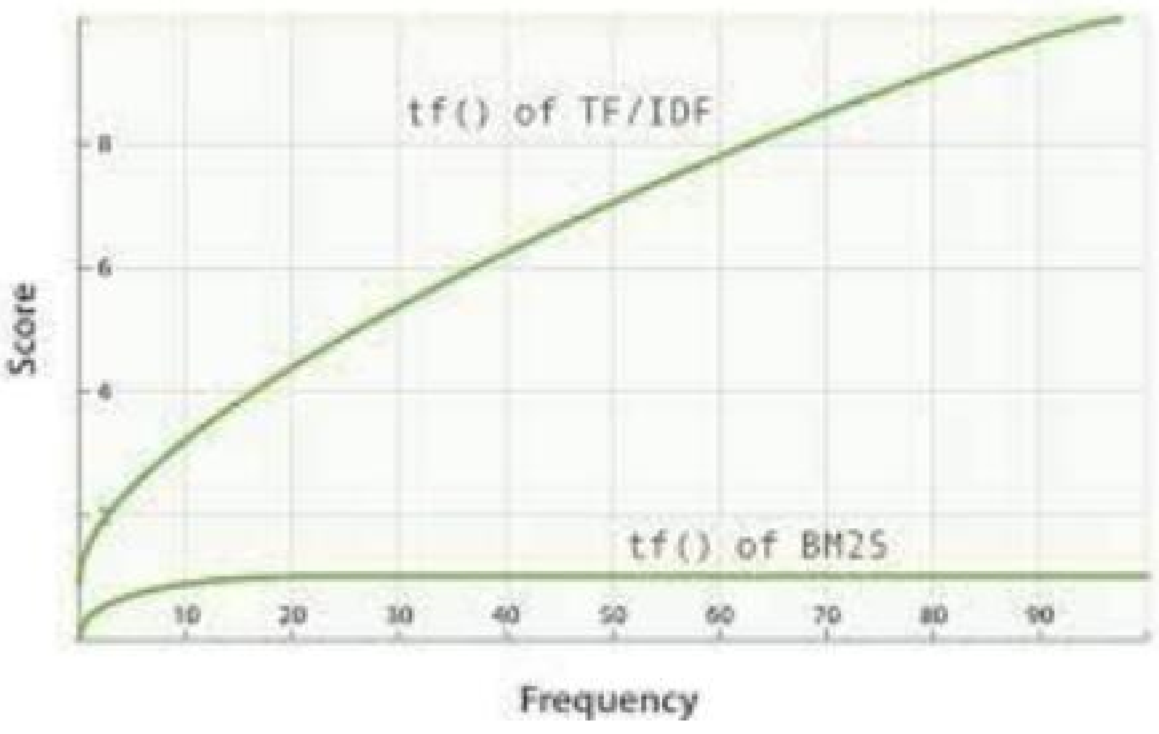
4. Consistency: The design of the website is consistent. We have made this sure by keeping the font style and sizes the same for various sections of the website. Also, the same layout is followed throughout the website and would be followed in the future too if we add any features to our website.

5. Accessibility: We ensure that our website follows the Accessibility concept by making sure that users know what to do when they use our website as the search box is easily visible and is responsive to user interaction. While logging in the signup page clearly informs the user if he/she has entered the incorrect username and password through both colors and text.

5. Comparison

First, we compare the algorithm used in our project i.e BM25 algorithm with other algorithms used for the same purpose:

1. TF-IDF: TF-IDF and BM25 are two popular algorithms used for ranking documents based on their relevance to a user query. Both algorithms use term weighting to determine the importance of each term in a document, but they differ in their weighting strategies. TF-IDF computes the weight of a term based on its frequency in the document and its inverse frequency across all documents in the collection. BM25, on the other hand, uses a more complex formula that takes into account the frequency of the term in the document and the length of the document and query. BM25 is considered to be more robust than TF-IDF in the face of noisy or incomplete data, and it has tuning parameters that can be adjusted to optimize its performance. As in this project, we use everyday articles, there is a lot of noise to be dealt with and hence BM25 is a more suitable choice.



2. Okapi BM25: BM25 and Okapi BM25 are both ranking algorithms used in Information Retrieval (IR) to assess the relevance of documents to user queries. The key difference between them is that Okapi BM25 includes a normalization term in its formula to account for document length, while BM25 does not. The normalization term divides the raw term frequency by an expected frequency that depends on the length of the document. In practice, the inclusion of the normalization term in Okapi

BM25 can improve its performance in some cases, particularly when dealing with long documents. However, BM25 remains a popular and effective algorithm that is widely used in many IR applications. The choice between the two algorithms depends on the specific requirements of the application, including the size and nature of the document collection, and the trade-off between relevance and computational efficiency.

3.PageRank: BM25 and PageRank are both algorithms used in Information Retrieval (IR) to rank documents based on their relevance to user queries. BM25 is a term-weighting algorithm that measures the similarity between a query and a document based on the frequency of query terms in the document and takes into account additional factors like document length and query term frequency. In contrast, PageRank is a link analysis algorithm that ranks web pages based on the quantity and quality of links pointing to them. PageRank is commonly used in web search engines to prioritize pages with high authority or popularity. While BM25 focuses on the content of individual documents, PageRank looks at the relationships between documents.

4. LSI(Latent Semantic Indexing): BM25 and LSI are both algorithms used in Information Retrieval (IR) to rank documents based on their relevance to user queries. However, the two algorithms differ in their approach to matching query terms with document terms. BM25 is a term-weighting algorithm that computes a relevance score based on the frequency of query terms in the document and other factors like document length and query term frequency. In contrast, LSI (Latent Semantic Indexing) is a statistical technique that identifies latent semantic relationships between terms and documents and ranks documents based on their relevance to a query in a reduced dimensional space. LSI can capture the conceptual meaning of words and documents beyond their literal or surface-level meaning and is particularly useful for dealing with synonymy, polysemy, and other lexical ambiguities. The choice between BM25 and LSI depends on the specific requirements of the IR system and the nature of the document collection.

Next, we compare different approaches to Information Retrieval. In this paper, we use BM25 which is a probabilistic model for information retrieval that estimates the relevance of documents based on their similarity to the query. It is an extension of the earlier BM11 model, and it is widely used in modern search engines. We compare with other approaches:

1. Boolean Retrieval: Boolean retrieval is simple and precise but limited in scope, while probabilistic models are more flexible and effective for larger, more complex collections and queries. However, it can be difficult to construct a precise query, and the results can be limited when the query is not well-defined or when the collection is large. In contrast, probabilistic models are more flexible and can handle more complex queries and larger collections. They are also better at handling vague or imprecise queries by assigning a probability score to each document based on its relevance to the query.

2. Vector Space Model: Vector space models do not explicitly model the relevance of the documents to the query, while probabilistic models do. Vector space models also do not take into account the frequency of query terms across the entire document collection, while probabilistic models do. This means that probabilistic models can handle more complex queries and larger collections, and are better at handling vague or imprecise queries.

3. LSA Model: LSA is a deterministic model that relies on the manipulation of the term-document matrix to capture the semantic similarity between documents and queries, while probabilistic models are statistical models that estimate the probability of a document being relevant to the query based on the frequency of the query terms in the document and across the document collection. While LSA is effective in capturing the semantic similarity between documents and queries, it may not be as flexible as probabilistic models in handling complex queries and larger collections. Probabilistic models can handle more complex queries and larger collections and are better at handling vague or imprecise queries.

4. Machine Learning: Probabilistic models are rule-based and rely on pre-defined heuristics to model the probability of relevance, while machine learning models are data-driven and learn patterns from the data. Machine learning models can be trained on large datasets and can discover more complex patterns in the data than probabilistic models. However, they require significant amounts of labeled training data, which may be difficult or expensive to obtain.

6. Conclusion

We were able to implement the BM25 algorithm to our site which provides users with articles, and news according to their needs. We selected the BM25 algorithm because it is customizable according to the application and performs better as compared to other algorithms like TF-IDF (Term Frequency-Inverse Document Frequency) and Okapi BM25 (a variant of BM25). This makes BM25 an ideal candidate for our project. In the future, we might add other features to the website and improve its UI while following the HCI concepts we mentioned earlier and possibly add more concepts to make the website as user-friendly as possible and improve the user experience even more.

7. References

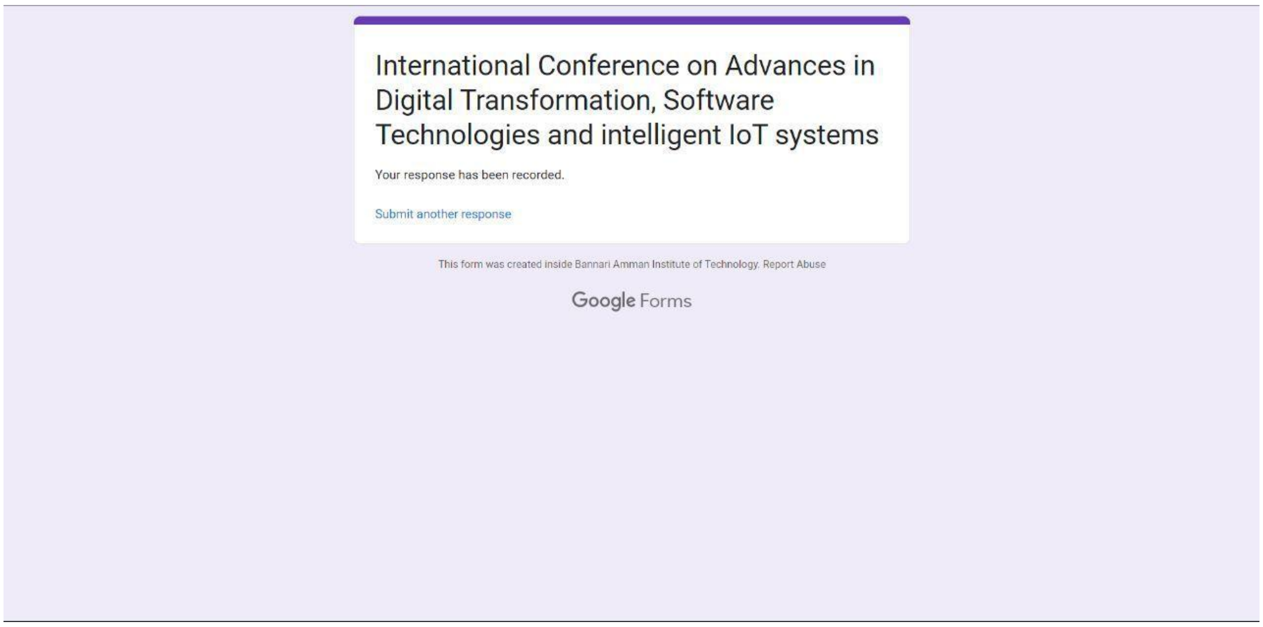
1. West, Jevin D., Ian Wesley-Smith, and Carl T. Bergstrom. "A recommendation system based on hierarchical clustering of an article-level citation network." IEEE Transactions on Big Data 2.2 (2016): 113-123.

2. Shani, Guy, and Asela Gunawardana. "Evaluating recommendation systems." Recommender systems handbook (2011): 257-297.

3. Ko, Hyeyoung, et al. "A survey of recommendation systems: recommendation models, techniques, and application fields." Electronics 11.1 (2022): 141.

4. Svore, Krysta M., and Christopher JC Burges. "A machine learning approach for improved BM25 retrieval." Proceedings of the 18th ACM Conference on Information and knowledge management. 2009.

Paper Submission:



International Conference on Advances in Digital Transformation, Software Technologies and intelligent IoT systems

Your response has been recorded.

Submit another response

This form was created inside Bannari Amman Institute of Technology. Report Abuse

Google Forms