University Mohamed Boudiaf of M’sila

Lectures on

Information Retrieval

License ISIL 2^{nd} Semester

By Dr. LOUNNAS Bilal

March 14, 2023

Contents

1

Introduction

In recent decades, multiple studies have demonstrated that Web search engines are the preferred mode of information retrieval, making them the standard for information gathering. For instance, the 2004 Pew Internet Survey [1] found that "92% of Internet users believe that the Internet is a valuable source for everyday information."

The long history of information retrieval does not begin with the internet or with the invention of computers, it goes back to 9000 BCE represented by a simplest way of getting information, it began when people start to agriculture, written language and the extraordinary phenomenon of Alexandria library, to the present day where the information retrieval has become an intelligence system that can find information automatically.

The only think meter to us is the idea of using computers to search for relevant pieces of information and that was popularized in the article “As We May Think” by Vannevar Bush in 1945 [2].

After 1945s scientists has become very quick to develop the field of information retrieval. The first point start when an automated information retrieval systems were introduced in the 1950s. In the 1960s the first large information retrieval research group was formed by Gerard Salton at Cornell. By the 1970s several different retrieval techniques had been shown to perform well on different kind of information retrieval application.

In 1992, the US Department of Defense along with the National Institute of Standards and Technology (NIST), cosponsored the Text Retrieval Conference (TREC) as part of the TIPSTER text program. The aim of this was to look into the information retrieval community by supplying the infrastructure that was needed for evaluation of text retrieval methodologies on a very large text collection. This catalyzed research on methods that scale to huge corpora. The introduction of web search engines has boosted the need for very large scale retrieval systems even further [3].

In this lecture, our primary focus will be on the field of information retrieval, covering its architecture, models, and algorithms.

The structure of this lecture is divided into 4 parts .....

2 Bsics of Information retrieval

2.1 Definition of IR

The meaning of the term Information Retrieval can be very broad. Just getting a credit card out of your wallet so that you can type in the card number is a form of information retrieval. However, as an academic field of study The definition of information retrieval can be seen as [4]:

> Information retrieval (IR) is finding material (usually documents) of an unstructured nature (usually text) that satisfies an information need from within large collections (usually stored on computers).

The first definition describe the majority of information retrieval techniques whose focusing on finding documents or web pages of a textual format to satisfy user query of a textual format also from a larg collection of an unstructured nature of data. however there are some techniques in information retrieval whose focusing on different kinds of data such as (images, sound, speech ....) and sometimes can be in a structured format such as those information retrival system who depande on using XML data.

The discipline of information retrieval is a very large one, for that many of other definition can be find in the literatures such as [5]:

> Information retrieval (IR) is the activity of obtaining information resources relevant to an information need from a collection of information resources.

Also [6]:

> The techniques of storing and recovering and often disseminating recorded data especially through the use of a computerized system.

Thus, all definitions flow into one meaning, which is Information Retrieval (IR) is the process that extract answers from a large number of sources to meet predefined information need.

2.2 History and timeline of IR

The long history of information retrieval does not begin with the internet or with the invention of computers, it goes back to 9000 BCE represented by a simplest way of getting information, it began when people start to agriculture.

The Figure 2.1 will summarize IR history timeline

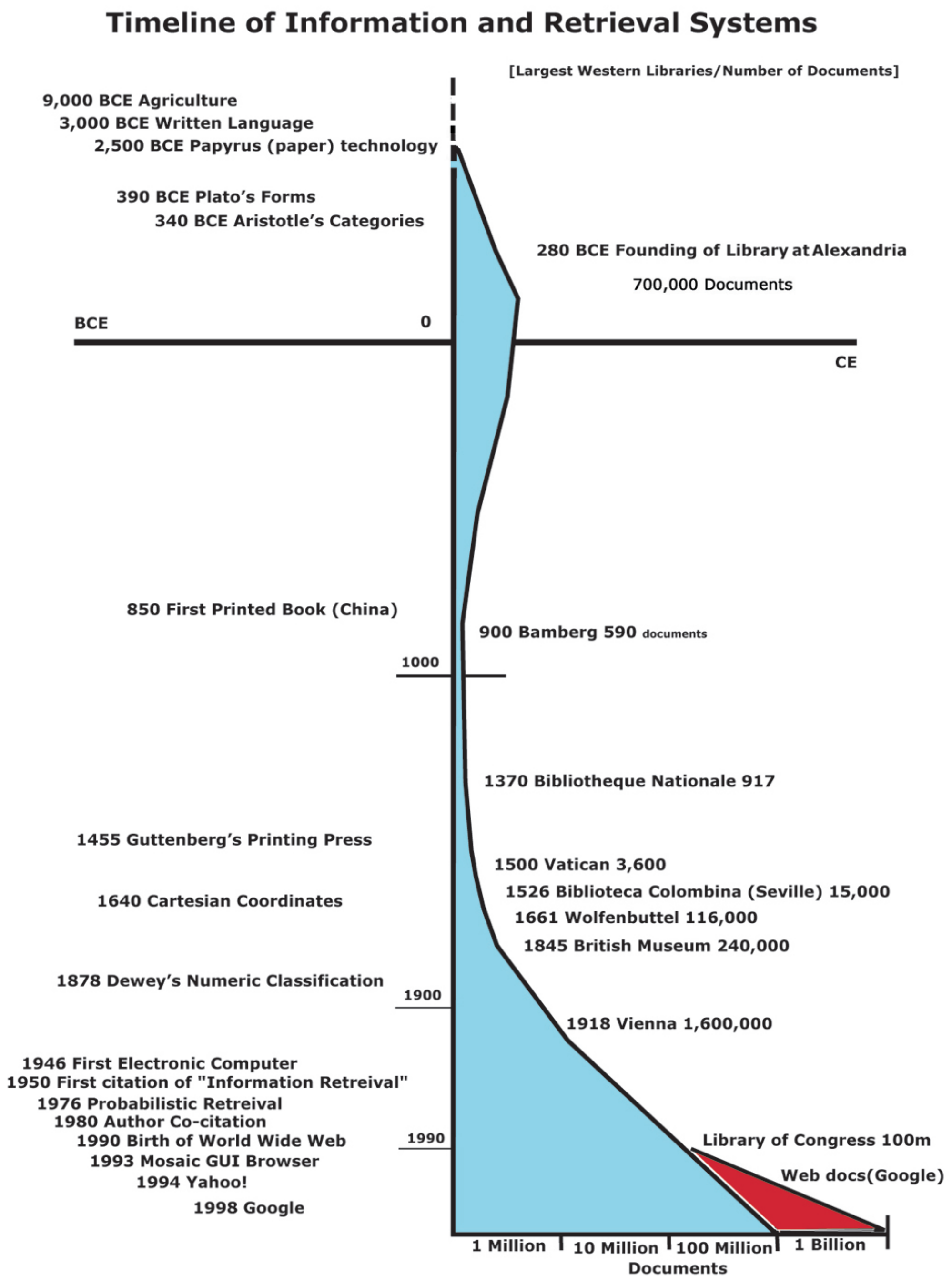


Figure 2.1: Information retrieval timeline history [7].

This figure shows the history of Information retrieval in general, from the very far of beginning including agriculture, written language and the extraordinary phenomenon of Alexandria library, to the present day where the information retrieval has become an intelligence system that can find information automatically.

The only think meter to us is the idea of using computers to search for relevant pieces of information and that was popularized in the article “As We May Think” by Vannevar Bush in 1945 [2].

After 1945s scientists has become very quick to develop the field of information retrieval. The first point start when an automated information retrieval systems were introduced in the 1950s. In the 1960s the first large information retrieval research group was formed by Gerard Salton at Cornell. By the 1970s several different retrieval techniques had been shown to perform well on different kind of information retrieval application.

In 1992, the US Department of Defense along with the National Institute of Standards and Technology (NIST), cosponsored the Text Retrieval Conference (TREC) as part of the TIPSTER text program. The aim of this was to look into the information retrieval community by supplying the infrastructure that was needed for evaluation of text retrieval methodologies on a very large text collection. This catalyzed research on methods that scale to huge corpora. The introduction of web search engines has boosted the need for very large scale retrieval systems even further [3].

Some of other major events in the history of information retrieval in Appendix.

Finally we can summarize the history of automated information retrieval systems into three parts:

- before 60ies: Manual IR in libraries: manual indexing; manual categorization.
- between 70ies and 80ies: Automatic IR in libraries.
- after 90ies: IR on the web and in digital libraries.

Figure 2.2 shows the three part of history of automated information retrieval systems.

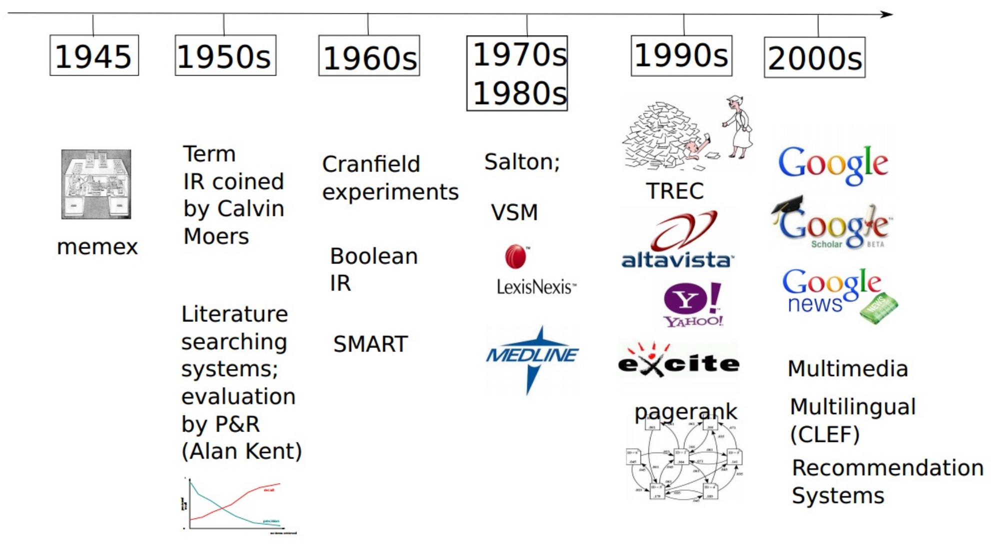


Figure 2.2: Automated information retrieval timeline.

2.3 Terminology in IR

One of the commun issue when reading scientific paper, articles, books of a large dicipline such as information retrieval is that sometimes the reader can be lost in the meaning of terms and even more an entire paragraph or document can be lost due to the ambiguity of terms.

Luckily, some great resources that defines terms of the Information retrieval has been written in the internet. So far there is one great glossary has been written by the Berkeley University titled by “the Modern Information Retrieval Glossary”.[8].

2.3.1 General and Expert terms of IR

The following terms have the same meaning of the term “Information retrieval", many authers are using those terms in thiers published researchs and books:

> Information Retrieval, Information Need, Query, Retrieval Model, Retrieval Engine, Search Engine, Relevance , Relevance Feedback, Evaluation, Information Seeking, Human-Computer-Interaction, Browsing, Interfaces, Ad-hoc Retrieval, Filtering.

While some terms can be very specific to: steps of the process of information retrieval, algorithms, methods,...,etc:

> term frequency, document frequency, inverse document frequency, vector-space model, probabilistic model, BM25, DFR, page rank, stemming, precision, recall.

2.3.2 Information retrieval in computer science

The automated information retrieval basically is the result of intersection of many disciplines, and with the provide of techniques and algorithms from each discipline we can build an efficient automated information retrieval that can brings satisfing results to the user.

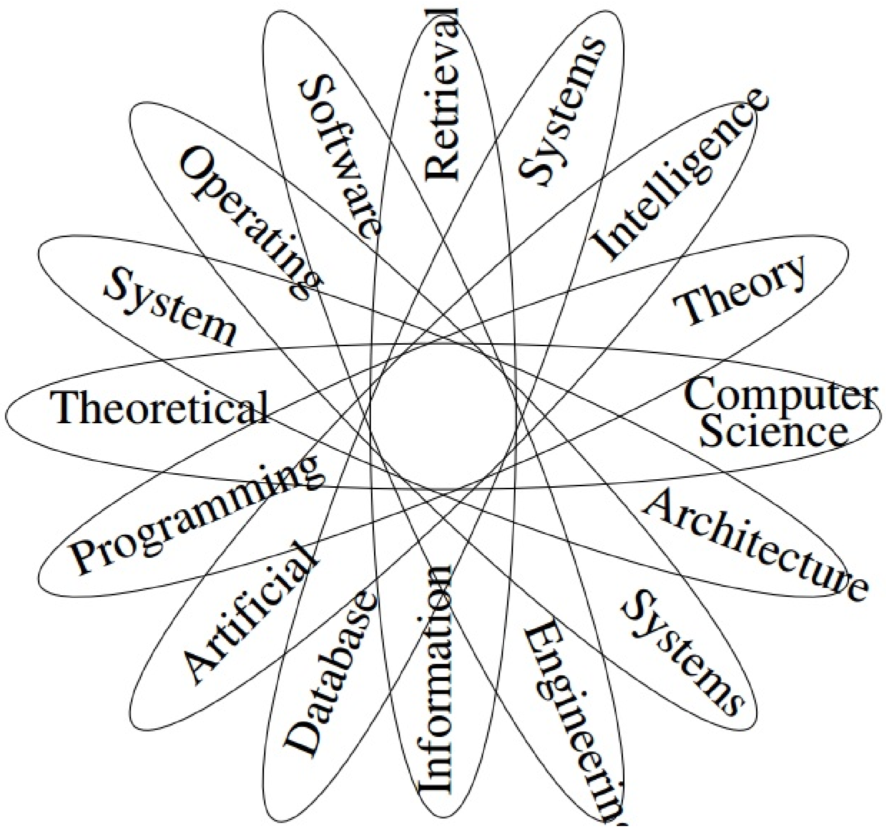


Figure 2.3: Information retrieval in computer science.

2.4 Topics of IR

Due to its diversities in many different disciplines, there are a large numbers of information retrieval topics, every one of this topic deal with a certain kind problem.

Here we selected some important topics:

- Retrieval models (ranking function, learning to rank, machine learning)
- Text processing (Data indextaion, Natural Language Processing NLP)
- Efficiency, compression, MapReduce, Scalability
- Distributed IR (data fusion, aggregated search, federated search)
- Multimedia: image, video, sound, speech
- Web retrieval and social media search
- Cross-lingual IR (FIRE), Structured Data (XML),
- Digital libraries, Enterprise Search, Legal IR, Patent Search, Genomics IR

2.5 Conferences of IR

The following list show the major conferences of the field of Information retrieval (IR).

- SIGIR: Conference on Research and Development in Information Retrieval
- ECIR: European Conference on Information Retrieval
- CIKM: Conference on Information and Knowledge Management
- WWW: International World Wide Web Conference
- WSDM: Conference on Web Search and Data Mining
- ICTIR: International Conference on Theory of Information Retrieval

- TREC: Text REtrieval Conference

3

Information Retrieval Concepts

3.1 Introduction

Relational databases contain rigidly structured data, while textual data lacks structure. When we search unstructured textual data, we refer to the process as "information retrieval." Information retrieval systems and database systems share similarities in the storage and retrieval of data on secondary storage. However, the focus in the field of information systems is different from that in database systems. It emphasizes querying based on keywords, document relevance to the query, and document analysis, classification, and indexing. Web search engines now extend beyond document retrieval and address broader issues. For example, they decide which information to display in response to a keyword query to meet the user's information needs.

This chapter delves into the information retrieval process, covering how users generate queries and how the system locates pertinent material to present as results.

3.2 Information retrieval vs other disciplines

Information retrieval deals with the representation, storage, organization of, and access to information items.

The primary objective of Information Retrieval (IR) is to retrieve relevant and useful information from a large collection of data in response to a user's query. IR systems are designed to facilitate efficient and effective access to information by helping users to:

1. Find relevant information: The main objective of an IR system is to help users find the most relevant information that matches their query. This involves searching through a large database of documents and identifying the most relevant ones.

2. Reduce information overload: With the vast amount of information available today, users can easily become overwhelmed. IR systems aim to reduce information overload by presenting users with only the most relevant information.

3. Provide fast and efficient access to information: IR systems are designed to provide fast and efficient access to information, allowing users to quickly find what they need.

4. Improve search accuracy: IR systems use various techniques such as query expansion, relevance feedback, and natural language processing to improve the accuracy of search results.

5. Support complex queries: IR systems can handle complex queries such as Boolean queries, phrase queries, and proximity queries, allowing users to refine their search criteria and obtain more relevant results.

Information retrieval, database, and data mining are related concepts, but they have distinct meanings, Table 3.1 provides a summary of the major differences between Information Retrieval (IR) and Database (DB) technology, as these two fields are often confused by those who are not familiar with computer science. The primary distinction between the two is that IR is focused on retrieving documents that are relevant to a user's query, while DB systems primarily match the query to specific data points. It should be noted that while there are several other differences between these two technologies, this fundamental difference is the most important to understand [9].

	Information retrieval	Database
Matching	Partial match	Exact match
Inference	Induction	Deduction
Model	Probabilistic	Deterministic
Classification	Polythetic	Monothetic
Query Language	Natural	Artificial
Query Specification	Incomplete	Complete
Items Wanted	Relevant	Matching
Error Response	Insensitive	Sensitive

Table 3.1: Information retrieval vs Database

On the other hand, there is a considerable overlap in the techniques and models used in data mining and information retrieval, as exemplified by the classification of documents.

	Information retrieval	Data mining
User target	Existing relevant data entries.	Knowledge (rules, etc.) implied by data (not the individual data entries themselves).

Table 3.2: Information retrieval vs Data mining

3.3 Information retrieval cycle

The Information Retrieval Cycle refers to the process of searching and retrieving relevant information from a collection of data or documents. It consists of a series of stages or steps that must be followed to achieve the desired outcome. Here are the common stages in the information retrieval cycle:

1. Information Need: It is the first step in the information retrieval cycle, where the user identifies and specifies the information they require. The user must clearly articulate their information need, including the purpose, scope, and context of their search.

2. Query Formulation: Once the information need has been established, the user formulates a query that captures the main ideas or concepts related to the information needed. The query is typically a set of keywords or phrases that describe the information need and help to narrow down the search results.

3. Information Retrieval: The next stage involves the search engine or the retrieval system to identify the relevant documents based on the user's query. The retrieval system uses algorithms to match the query with the documents in the database and return the most relevant ones.

4. Evaluation: After retrieving the documents, the user evaluates the results based on relevance, credibility, and other criteria. The user assesses the relevance of the documents based on their information need and selects the most relevant ones.

5. Use: Once the user has identified the relevant documents, they use them to fulfill their information needs. This may involve analyzing the information, synthesizing it with other information sources, or incorporating it into a larger project.

6. Feedback: The final stage of the information retrieval cycle is feedback. The user provides feedback to the system, which can be used to improve the retrieval process. Feedback can include identifying errors or inconsistencies in the search results, suggesting new keywords or concepts to improve the search, or rating the relevance of the documents retrieved.

Overall, the information retrieval cycle provides a framework for users to systematically search for and retrieve relevant information, evaluate it, and apply it to their information needs. The process is iterative, and feedback helps to refine the search and improve the accuracy and relevance of the results over time.

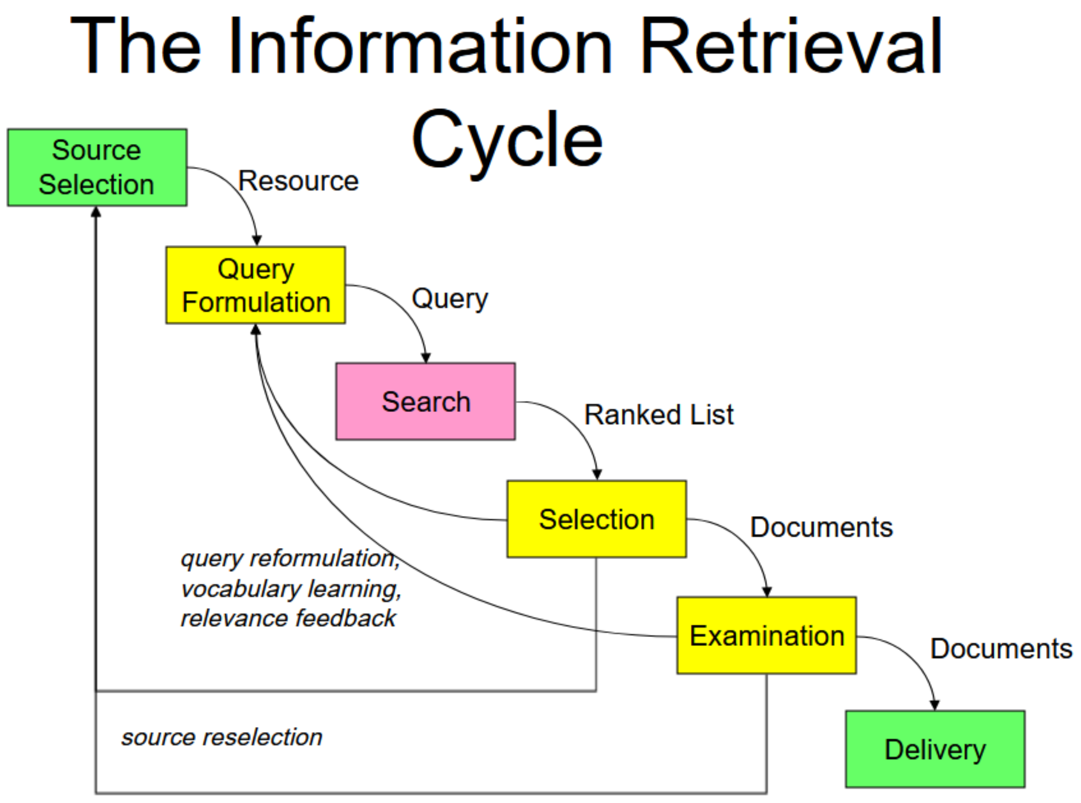


Figure 3.1: Information retrieval cycle [10]

3.4 Information retrieval architecture

The architecture of an information retrieval system is straightforward, as its primary goal is to retrieve all relevant documents in response to a user's query while minimizing the number of non-relevant documents retrieved. In other words, the system's design should be focused on achieving this fundamental objective.

An IR system architecture typically consists of the following components:

- Data collection and preprocessing: This component involves the acquisition of data from various sources and the preprocessing of the data to prepare it for indexing and searching.

- Indexing: This component involves the creation of an index, which is a data structure that maps terms or keywords to their corresponding documents or other types of data. Indexing is necessary to speed up the search process.

- Query processing: This component involves the processing of user queries, which are typically in the form of keyword-based search queries. The query processing component retrieves relevant documents from the index based on the user's query.

- Ranking: This component involves the calculation of a relevance score for each retrieved document. The relevance score is used to rank the documents in order of relevance to the user's query.
- User interface: This component provides a way for users to interact with the IR system, typically through a web interface or other graphical user interface.
- Evaluation: This component involves the evaluation of the IR system's performance, typically through measures such as precision, recall, and F1-score.

These components can be organized in various ways depending on the specific needs of the IR system. as shown in Figure 3.2, some systems may include additional components such as document clustering, natural language processing, or machine learning algorithms for personalized search results.

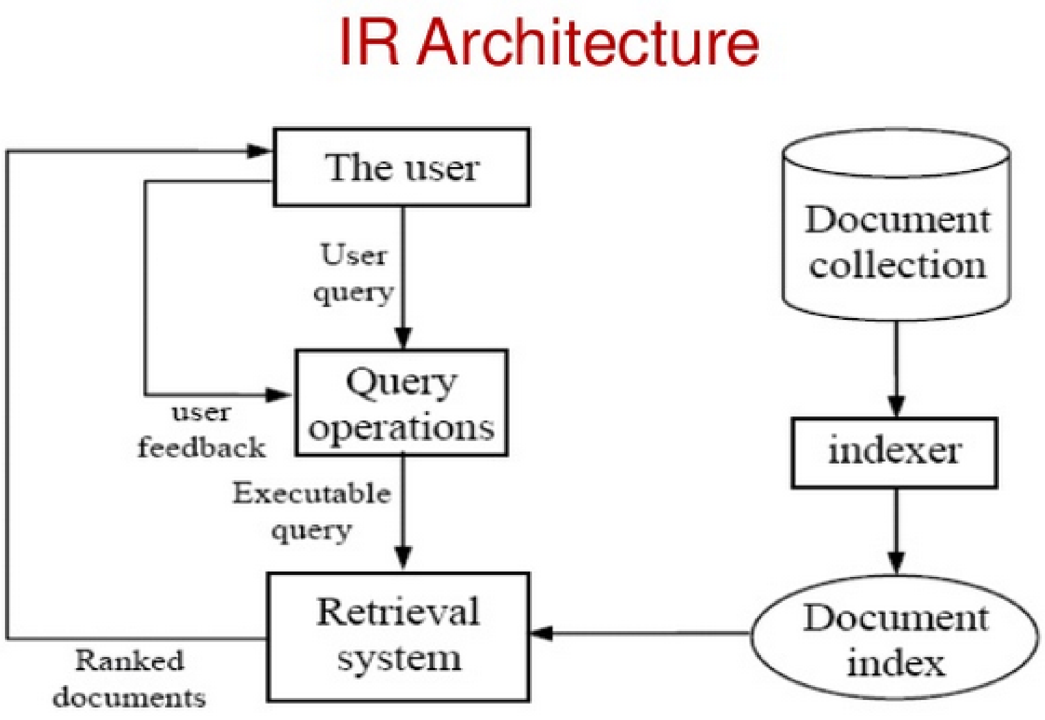


Figure 3.2: Information retrieval architecture [8]

According to Bhattacharyya. et al.[11], an IR system architecture can be divided into two views: one for the user and one for the system.

3.4.1 User view

The user view of the IR system architecture, see Figure 3.3. Includes the components that the user interacts with, such as the user interface and query processing. The query processing component receives the user's query and retrieves relevant documents from the index based on the query.

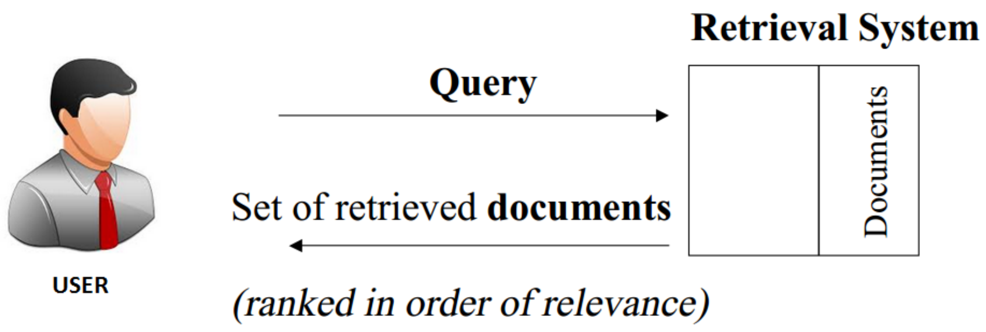


Figure 3.3: IR architecture user view

3.4.2 System view

The system view, see Figure 3.4. Of the IR system architecture includes the components that the user does not directly interact with, such as data collection and preprocessing, indexing, ranking, and evaluation. Both the user view and system view of the IR system architecture are critical to the success of the system. The user view ensures that the IR system is easy to use and provides relevant results to the user. The system view ensures that the IR system is efficient, accurate, and effective in retrieving relevant information from large collections of data.

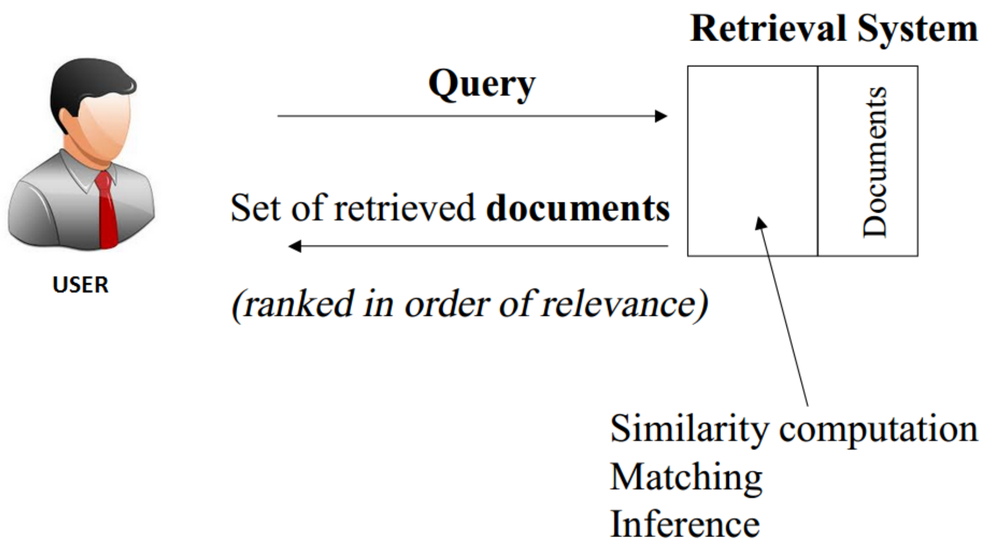


Figure 3.4: IR architecture system view

4

Indexing and Text Preprocessing

4.1 Introduction

Indexing and text preprocessing are essential components of information retrieval systems. Text preprocessing is necessary to extract meaningful information from unstructured text data, and indexing is used to organize and retrieve the information efficiently. Text preprocessing techniques such as tokenization, stemming, and part-of-speech tagging help to reduce the dimensionality of the data and improve the accuracy of retrieval results. Indexing techniques such as the inverted index enable fast and efficient retrieval of relevant documents from large collections of data. Without indexing and text preprocessing, the process of retrieving relevant information from a large corpus of data would be slow, inefficient, and inaccurate.

According to Manning. et al [4], indexing is one of the most important components of an information retrieval system. In their book, they explain how the inverted index data structure is used to map terms to their corresponding documents, making it possible to retrieve documents based on queries quickly and efficiently. Similarly, Jurafsky. et al [12] emphasize the importance of text preprocessing techniques such as stemming and stopword removal, which can significantly improve the accuracy of information retrieval. They note that text preprocessing techniques can help to reduce the impact of noisy data, improve recall and precision, and make it easier to analyze and understand large text datasets.

4.2 Text Preprocessing

Information retrieval can be applied to both structured and unstructured data, but it is estimated that 80% of business-relevant information is found in unstructured data, such as social media posts, emails, and other textual sources. However, unstructured data can be challenging to analyze and search because it lacks a predefined structure. Therefore, it needs to be preprocessed first to extract useful information from it. By extracting insights from unstructured data, businesses can gain a competitive advantage by making data-driven decisions and improving their understanding of customer behavior.

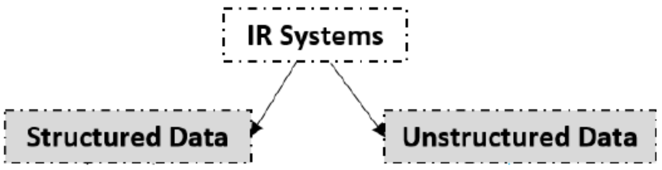
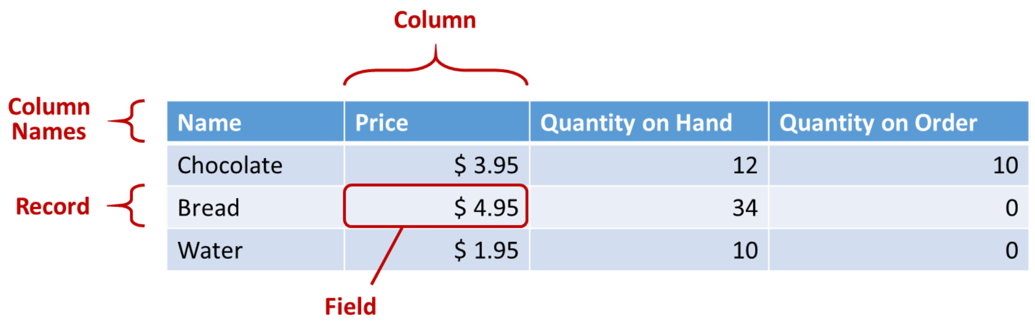


Figure 4.1: IR to structured and unstructured data

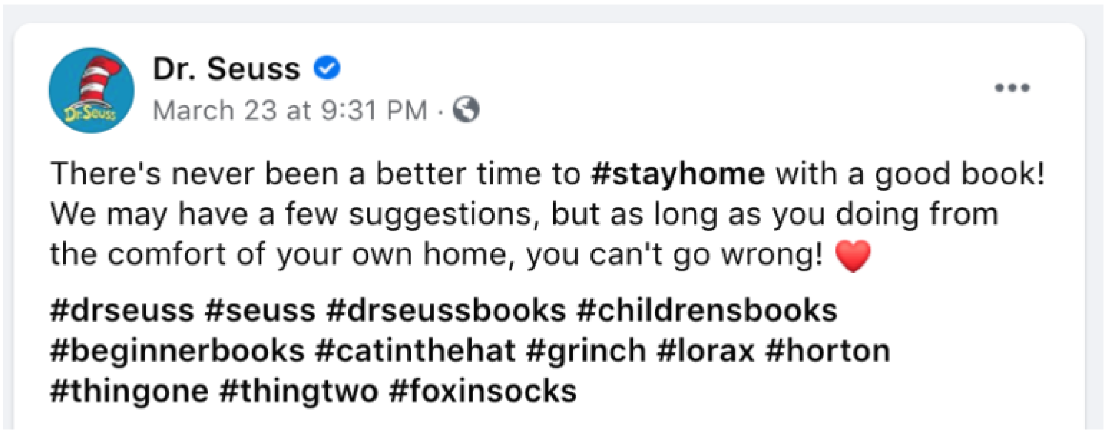
To provide further explanation, structured data refers to data that is organized and stored in a predefined format, such as in database table as in Figure 4.2. It typically has a fixed schema and a well-defined set of attributes.


Column
Column Names
Record
Field

Name	Price	Quantity on Hand	Quantity on Order
Chocolate	$ 3.95	12	10
Bread	$ 4.95	34	0
Water	$ 1.95	10	0

Figure 4.2: Database table as structured data

Unstructured data can be a valuable source of information for businesses, but it can be difficult to extract insights from it due to its lack of structure. Therefore, text preprocessing techniques are used to transform unstructured data into a more structured format that can be analyzed and searched more easily.



Dr. Seuss
March 23 at 9:31 PM ·

There's never been a better time to **#stayhome** with a good book! We may have a few suggestions, but as long as you doing from the comfort of your own home, you can't go wrong! ❤️

#drseuss #seuss #drseussbooks #childrensbooks #beginnerbooks #catinthehat #grinch #lorax #horton #thingone #thingtwo #foxinsocks

Figure 4.3: Facebook Post as unstructured data

Figure 4.4 Refer to the most well-known methods of preprocessing in information retrieval systems, these techniques are required and essential for any IR system to produce accurate results. By using these techniques, noise and irrelevant data can be removed, and the accuracy of retrieval results can be improved

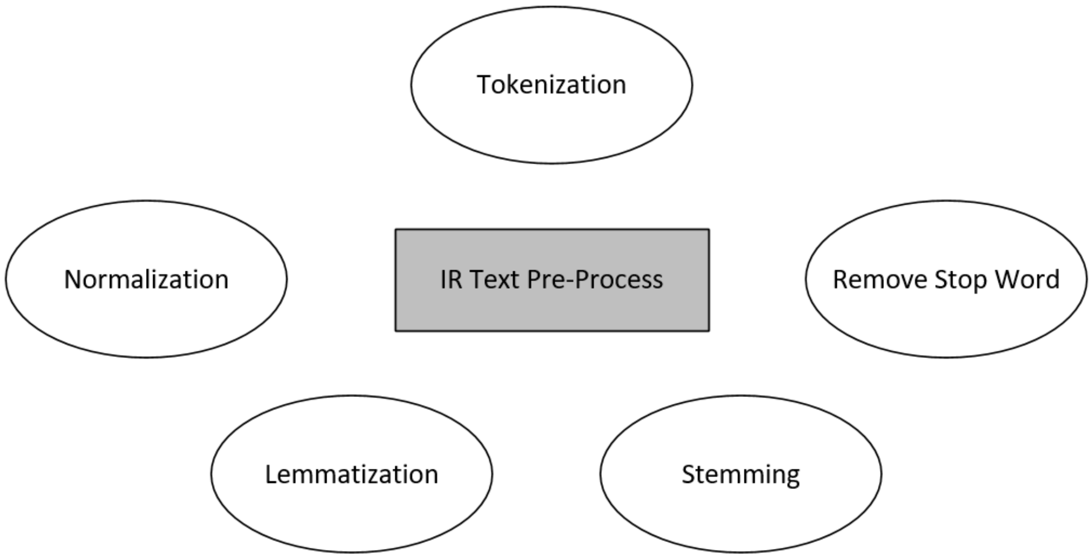


Figure 4.4: Most Pre-Process methods of IRs

4.2.1 Tokenization

When dealing with a sequence of characters in a defined document unit, tokenization refers to the process of breaking it down into smaller units known as tokens. This may involve discarding certain characters, such as punctuation, in order to produce a more structured format [4]. Figure 4.5 is an example of tokenization

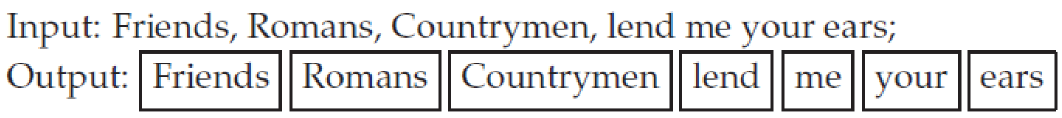

Input: Friends, Romans, Countrymen, lend me your ears;
Output: Friends | Romans | Countrymen | lend | me | your | ears

Figure 4.5: Tokenization

Each token from the outputs of tokenization methods are now a candidate for an index entry after further processing.

Tokens are sequences of characters in a document and are sometimes referred to as terms or words. A type is the class of all tokens containing the same character sequence, while a term is a normalized type included in the IR system's dictionary. Index terms can be distinct from the tokens but are usually derived from them through normalization processes [4].

4.2.1.1 Tokenization issues

- Contractions: Tokenizing words like "don't" and "can't" can be tricky because they contain apostrophes and are often considered single units. However, some tokenizers may separate the words into "do" and "n't" or "ca" and "n't," which could impact the accuracy of search results.[13]

- Hyphenated words: Hyphenated words, such as "well-known" or "self-driving," can also pose tokenization challenges because they may be split into separate tokens or considered a single unit, depending on the tokenizer used.

- Abbreviations: Abbreviations, such as "Mr." or "Dr.", may be tokenized as separate words or kept as part of the original word. This can impact search results, especially when searching for specific titles or names.

- Punctuation: Different punctuation marks, such as commas, periods, and exclamation points, may be tokenized separately or kept as part of the surrounding text. This can impact the meaning of the text and the accuracy of search results.

- Domain-specific jargon or technical terms: Tokenizing domain-specific jargon or technical terms may require special consideration, as they may be composed of multiple words or have specific capitalization or formatting requirements. This is especially relevant in fields like medicine or engineering, where specialized terminology is common. [14]

4.2.2 Normalization

Token normalization is the process of canonicalizing tokens so that matches occur despite superficial differences in the character sequences of the tokens.[4]. The most standard way to normalize is to implicitly create equivalence classes, which are normally named after one member of the set. For instance, if the tokens anti-discriminatory and antidiscriminatory are both mapped onto the term antidiscriminatory, in both the document text and queries, then searches for one term will retrieve documents that contain either.

Example:

Raw	Normalized
2moro	tomorrow
b4	before
otw	on the way
U.S.A	use

Table 4.1: Examples of normalization

4.2.3 Lemmatization

Due to grammar rules, there are variations in the forms of words used in documents, such as organize, organizes, and organizing. In addition, there are groups of words that are related in meaning, such as democracy, democratic, and democratization.

Lemmatization is a text preprocessing technique used in information retrieval to group together different inflected forms of a word and treat them as a single item, called a "lemma." It involves reducing words to their base or dictionary form, which is known as the lemma. For example, the lemmatized form of the words "am," "are," and "is" is "be." This technique is used to improve the accuracy of text retrieval by reducing the number of unique terms in the system's index. By normalizing words to their base form, lemmatization ensures that related words are treated as the same term during retrieval, which can lead to more accurate and relevant results.[4].

Table 4.2 shows some example of how lemmatization method change the form of the word.

Form	Lemma
studies	study
studying	study

Table 4.2: Examples of lemmatization

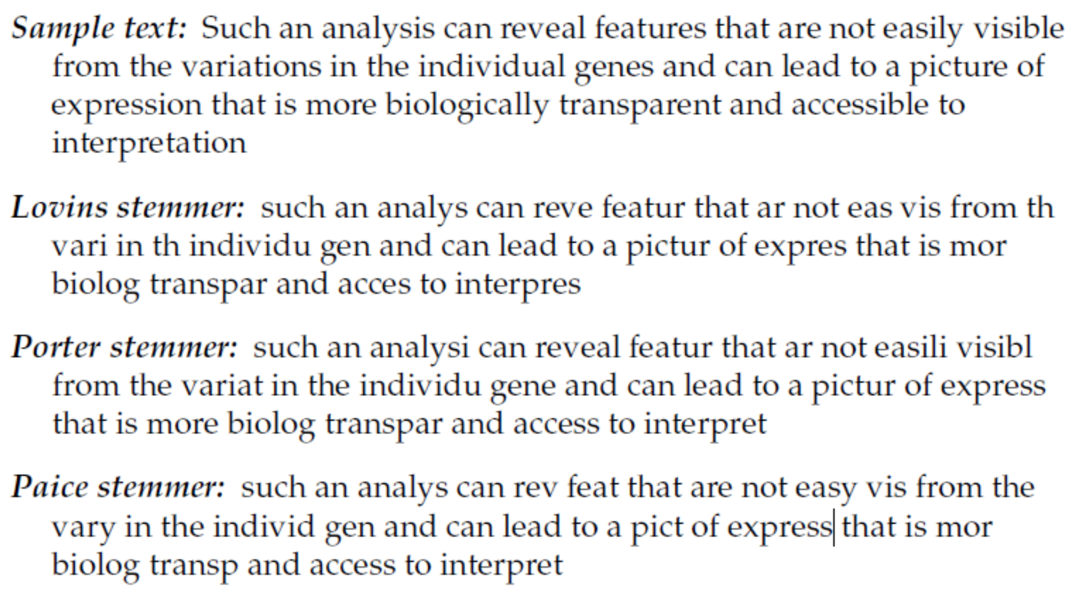
4.2.4 Stemming

Stemming algorithms work by cutting off the end or the beginning of the word, taking into account a list of common prefixes and suffixes that can be found in an inflected word. This indiscriminate cutting can be successful in some occasions, but not always, and that is why we affirm that this approach presents some limitations. Table 4.3 illustrate the method with examples in both English. in other meaning, stemming reduce terms to their root.

Form	stem
studies	studi
studying	study

Table 4.3: Examples of stemming

There have been many different techniques and algorithms of stemming in the literature. The following examples illustrated in Figure 4.2 show three different stemming algorithms applied to a simple given text.



Sample text: Such an analysis can reveal features that are not easily visible from the variations in the individual genes and can lead to a picture of expression that is more biologically transparent and accessible to interpretation

Lovins stemmer: such an analys can reve featur that ar not eas vis from th vari in th individu gen and can lead to a pictur of expres that is mor biolog transpar and acces to interpres

Porter stemmer: such an analysi can reveal featur that ar not easili visibl from the variat in the individu gene and can lead to a pictur of express that is more biolog transpar and access to interpret

Paice stemmer: such an analys can rev feat that are not easy vis from the vary in the individ gen and can lead to a pict of express that is mor biolog transp and access to interpret

Figure 4.6: Example of stemming applied to a simple text

4.2.5 Stop words

In information retrieval, stop words are commonly defined as words that are considered to be too common or too general to be useful for searching or indexing. Examples of stop words include "the," "and," "of," and "in." These words are often filtered out during text preprocessing to improve the efficiency and accuracy of search queries [4].

4.3 Indexing

Indexing is a key technique in information retrieval that involves creating an index to organize and retrieve information efficiently. In indexing-based information retrieval systems, documents are processed and indexed in advance to speed up the search process. The index contains a list of terms or keywords knowing by the term of "Tokens" as we seen in the previews section, that appear in each document.

There are many different indexing techniques in information retrieval, some of which contain additional information related to the tokens in the corresponding documents, while others contain information about the collection of documents.

When a user enters a search query, the system searches the index for documents that match the query as well as different representations of the query as illustrated in Figure 4.7, and returns a ranked list of results. This approach is widely used in modern search engines, digital libraries, and other information retrieval applications.

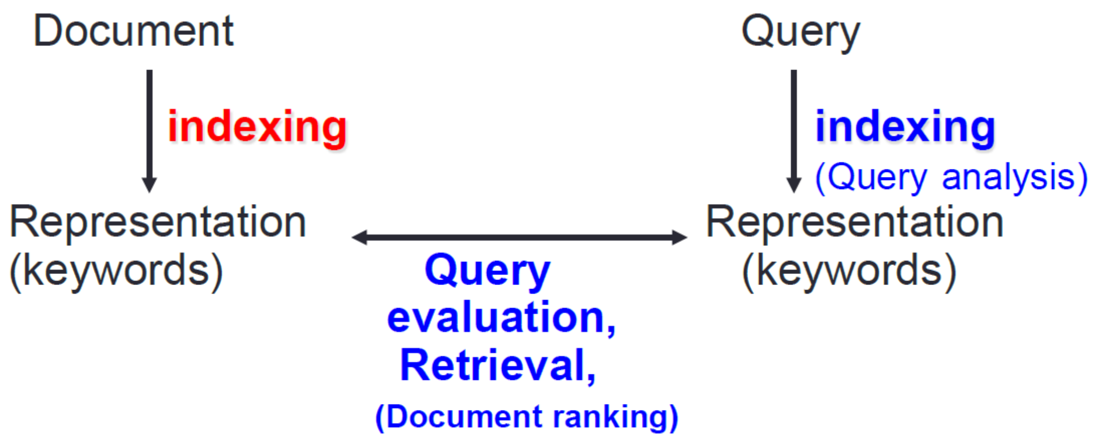

Document
indexing
Representation (keywords)
Query
indexing
(Query analysis)
Representation (keywords)
Query evaluation, Retrieval,
(Document ranking)

Figure 4.7: Indexing process for documents and query

The process of indexing starts with the document collection, see Figure 4.8. The system must first apply pre-processing methods, such as tokenization, and then apply linguistic models to produce a dictionary list.

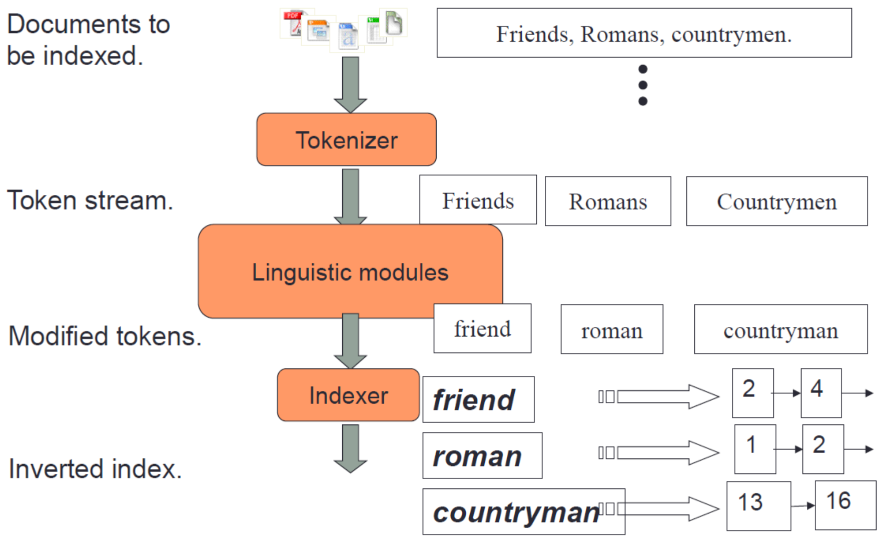


Figure 4.8: Indexing process pipe line

In Figure 4.8 demonstrate an indexing process that represent a document to inverted index. In the following section we will present some well known and used in information retrieval.

4.3.1 Term-document Incidence Matrix

A term-document incidence matrix is a mathematical representation of a text corpus that captures the frequency of occurrence of each term (word or phrase) in each document of the corpus. The matrix is typically constructed as a table where each row corresponds to a term in the corpus, each column corresponds to a document, and each cell represents the number of times the term occurs in the corresponding document.

Table 4.4 Show an example of using Term-decument incidence matrix to represent six documents, Antony and Cleopatra, Julius Caesar, The Tempest, Hamlet, Othello, and Macbeth. with a seven tokens ("Antony","Brutus"....), the ones (1's) values represent that the token is exist in the correspânding document, zeors (0's) otherwise.

	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth
Antony	1	1	0	0	0	1
Brutus	1	1	0	1	0	0
Caesar	1	1	0	1	1	1
Calpurnia	0	1	0	0	0	0
Cleopatra	1	0	0	0	0	0
mercy	1	0	1	1	1	1
worser	1	0	1	1	1	0

Table 4.4: Term-document incidence matrix for six document and seven tokens

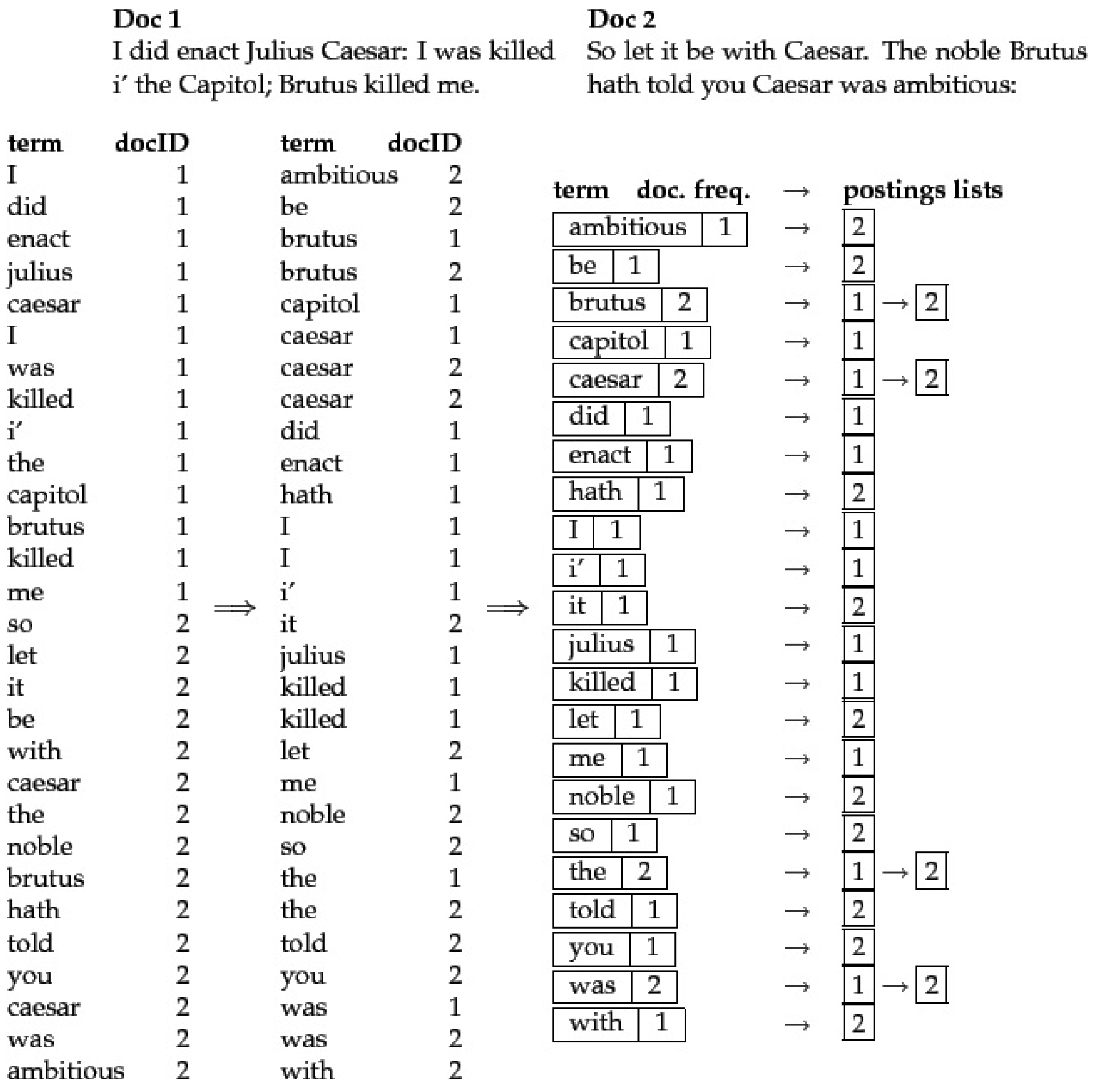
One limitation of the term-document incidence matrix is that it can become very large and sparse for large corpora, which can make it computationally expensive to work with. However, there are various techniques for reducing the size and sparsity of the matrix, such as using dimensionality reduction methods or applying weighting schemes to the term frequencies.

4.3.2 Inverted Index

An inverted index is an index into a set of documents of the words in the documents. The index is accessed by some search method. Each index entry gives the word and a list of documents, possibly with locations within the documents, where the word occurs. The inverted index data structure is a central component of a typical search engine indexing algorithm. [15]

Inverted index is the key data structure underlying modern IRs, form systems running on a single laptop to those running in biggest commercial search engine.

Figure 4.9 represents an inverted index of two documents: Doc1 and Doc2. The process is simple. The sequence of each token in the two documents is tagged by their document ID, as seen on the left of the figure, and then sorted alphabetically (middle of the figure). Instances of the same token are then grouped by word and document ID. The token and document ID are then separated (right of the figure). The dictionary stores the tokens and a pointer to the posting list of each token.



Doc 1
I did enact Julius Caesar: I was killed i' the Capitol; Brutus killed me.

Doc 2
So let it be with Caesar. The noble Brutus hath told you Caesar was ambitious:

term	docID
I	1
did	1
enact	1
julius	1
caesar	1
I	1
was	1
killed	1
i'	1
the	1
capitol	1
brutus	1
killed	1
me	1
so	2
let	2
it	2
be	2
with	2
caesar	2
the	2
noble	2
brutus	2
hath	2
told	2
you	2
caesar	2
was	2
ambitious	2

⟹

term	docID
ambitious	2
be	2
brutus	1
brutus	2
capitol	1
caesar	1
caesar	2
caesar	2
did	1
enact	1
hath	1
I	1
I	1
i'	1
it	2
julius	1
killed	1
killed	1
let	2
me	1
noble	2
so	2
the	1
the	2
told	2
you	2
was	1
was	2
with	2

⟹

term	doc. freq.	→	postings lists
ambitious	1	→	2
be	1	→	2
brutus	2	→	1 → 2
capitol	1	→	1
caesar	2	→	1 → 2
did	1	→	1
enact	1	→	1
hath	1	→	2
I	1	→	1
i'	1	→	1
it	1	→	2
julius	1	→	1
killed	1	→	1
let	1	→	2
me	1	→	1
noble	1	→	2
so	1	→	2
the	2	→	1 → 2
told	1	→	2
you	1	→	2
was	2	→	1 → 2
with	1	→	2

Figure 4.9: Inverted index

Generally, inverted indexes are separated into two parts: the dictionary and the posting list. The dictionary can be located in memory in information retrieval systems, while the posting list, which holds the document IDs that contain the selected token, is stored on disk.

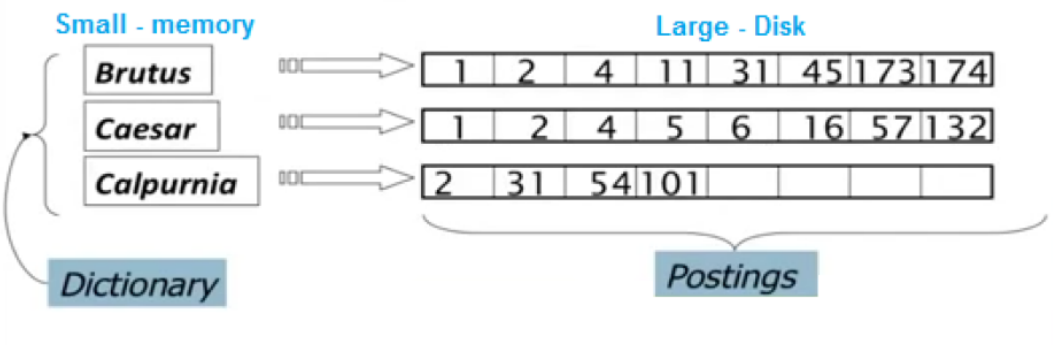


Figure 4.10: Inverted index with two part, dictionary and posting list

4.3.2.1 Query processing with inverted index

The goal of an information retrieval system is to locate needed information quickly. This is achieved using an inverted index and a special kind of algorithm called the MERGE algorithm family, which refers to algorithms that apply various boolean operations on two sets of sorted lists.

```
INTERSECT($p_1, p_2$)
 1  $answer \leftarrow \langle\ \rangle$
 2  while $p_1 \neq$ NIL and $p_2 \neq$ NIL
 3  do if $docID(p_1) = docID(p_2)$
 4        then ADD($answer, docID(p_1)$)
 5             $p_1 \leftarrow next(p_1)$
 6             $p_2 \leftarrow next(p_2)$
 7        else if $docID(p_1) < docID(p_2)$
 8                then $p_1 \leftarrow next(p_1)$
 9                else $p_2 \leftarrow next(p_2)$
10  return $answer$
```

Figure 4.11: MERGE Algorithm for Intersect

The intersection of two ordered lists is a new ordered list that contains only the elements that are present in both of the original lists. Figure 4.11 show the algorithm of the intersection between two ordered list.

To compute the intersection of two ordered lists, you can start by comparing the first element of each list. If they are the same, add that element to the new list and move both pointers to the next element. If they are different, move the pointer of the list with the smaller element to the next element and compare again.

Repeat this process until you have compared all elements of one of the lists or until you have found the intersection of both lists.

Here's an example:

List	Elements
A	1, 2, 3, 4, 5, 6
B	3, 4, 6, 8, 9

Table 4.5: Two list containing sorted elements

- Compare the first elements: A[0] = 1, B[0] = 3. They are different, so move the pointer of List A to the next element.

- Compare the next elements: A[1] = 2, B[0] = 3. They are different, so move the pointer of List A to the next element again.

- Compare the next elements: A[2] = 3, B[0] = 3. They are the same, so add 3 to the new list and move both pointers to the next elements.

- Compare the next elements: A[3] = 4, B[1] = 4. They are the same, so add 4 to the new list and move both pointers to the next elements.

- Compare the next elements: A[4] = 5, B[1] = 4. They are different, so move the pointer of List B to the next element.

- Compare the next elements: A[4] = 5, B[2] = 6. They are different, so move the pointer of List A to the next element.

- Compare the next elements: A[5] = 6, B[2] = 6. They are the same, so add 6 to the new list and move both pointers to the next elements.

- Both lists have been fully traversed, so the intersection of List A and List B is: 3, 4, 6.

There are many different alternatives to the MERGE algorithm family. The example above represents the MERGE algorithm for the intersection (A AND B). We can also find algorithms for (A OR B), which represents the union of the two lists, and (A AND NOT B), which represents the elements in A that are not in B. There are many other types of boolean operations that can be performed using the

MERGE algorithm family. The only thing that matters in this kind of algorithm is that it must run in linear time complexity.

4.3.3 BiWord Index

The vector representation of documents is inherently lossy because it does not preserve the relative order of terms in a document. As a result, phrase queries cannot be answered by this type of indexing.

We aim to answer queries that involve phrases, such as "Stanford University," where the order of the words in the query is critical. Therefore, a sentence such as "I went to the University of Stanford" would not match. For this type of problem, using only the inverted index, as defined in the previous section, is no longer sufficient. A more sophisticated index is required to handle these types of queries.

One approach to handling phrases is to consider every pair of consecutive terms in a document as a phrase. For example, the text **"Friends, Romans, Countrymen"** would generate the biwords :

- "friends romans" as one token
- "romans countrymen" as one token

In this model, we treat each of these biwords as a vocabulary term. Being able to process two-word phrase queries is immediate. Longer phrases can be processed by breaking them down. The query stanford university palo alto can be broken into the Boolean query on biwords:

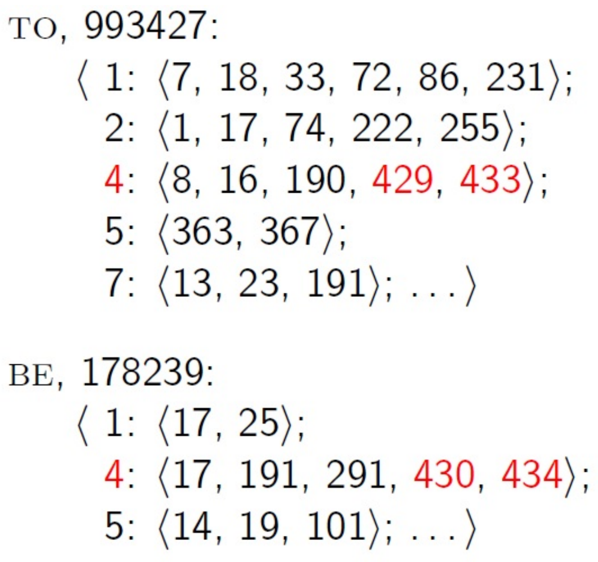
"stanford university" AND "university palo" AND "palo alto"

The issue in biword index is that the index will blowup to bigger dictionary (two consecutive token count as one token), and biword are not the standard solution for all biword.

4.3.4 Positional Index

Due to the issues mentioned earlier, a biword index is not typically used as the standard solution. Instead, a positional index is commonly employed. In a positional index, postings for each term in the vocabulary are stored in the form of "docID:

⟨position1, position2, ...⟩" as illustrated in Figure 4.12, where each position represents a token index in the document. Additionally, the term frequency is usually recorded in each posting.



```
TO, 993427:
    ⟨ 1: ⟨7, 18, 33, 72, 86, 231⟩;
      2: ⟨1, 17, 74, 222, 255⟩;
      4: ⟨8, 16, 190, 429, 433⟩;
      5: ⟨363, 367⟩;
      7: ⟨13, 23, 191⟩; . . .⟩

BE, 178239:
    ⟨ 1: ⟨17, 25⟩;
      4: ⟨17, 191, 291, 430, 434⟩;
      5: ⟨14, 19, 101⟩; . . .⟩
```

Figure 4.12: Positional Index

In a positional index, each entry for a term in the inverted index includes not only the list of documents that contain the term, but also the positions of the term within each of those documents. This allows the search engine to perform more advanced queries, such as proximity queries or phrase queries, which require knowledge of the positions of terms within a document.

For example, suppose a user searches for the phrase "to be" in a search engine that uses a positional index illustrated in Figure 4.12. The engine would first retrieve the list of documents that contain both "to" and "be". Then, it would look for instances where the two terms occur within a certain proximity of each other, such as document 4 with positions 429 and 433.

A proximity query is a type of search query in which the search engine retrieves documents containing multiple terms that appear within a specified distance or proximity of each other.

4.3.4.1 Query processing with Positional index

There is also a variation of the MERGE algorithm designed for positional indexing that allows for the retrieval of phrase and proximity queries. Figure 4.13 represent

MERGE algorithm for positional index.

```
PositionalIntersect($p_1, p_2, k$)
answer $\leftarrow \langle\ \rangle$
while $p_1 \neq$ NIL and $p_2 \neq$ NIL
do if $docID(p_1) = docID(p_2)$
      then $l \leftarrow \langle\ \rangle$
           $pp_1 \leftarrow positions(p_1)$
           $pp_2 \leftarrow positions(p_2)$
           while $pp_1 \neq$ NIL
           do while $pp_2 \neq$ NIL
              do if $|pos(pp_1) - pos(pp_2)| \leq k$
                    then Add($l, pos(pp_2)$)
                    else if $pos(pp_2) > pos(pp_1)$
                            then break
                 $pp_2 \leftarrow next(pp_2)$
              while $l \neq \langle\ \rangle$ and $|l[0] - pos(pp_1)| > k$
              do Delete($l[0]$)
              for each $ps \in l$
              do Add($answer, \langle docID(p_1), pos(pp_1), ps \rangle$)
              $pp_1 \leftarrow next(pp_1)$
           $p_1 \leftarrow next(p_1)$
           $p_2 \leftarrow next(p_2)$
      else if $docID(p_1) < docID(p_2)$
             then $p_1 \leftarrow next(p_1)$
             else $p_2 \leftarrow next(p_2)$
return answer
```

Figure 4.13: MERGE algorithm for Positional Index

5

IR Models and Evaluation

5.1 Introduction

The development of information retrieval (IR) has closely followed the evolution of IR models. The effectiveness of an IR system is primarily determined by its retrieval function, which ranks documents based on their relevance to a query. In the early days of IR research, the focus was on automatic indexing to help with manual classification by librarians. The matching of documents and queries was initially performed using a Boolean search of query terms in an index of document surrogates. Later, more complex models, such as the probabilistic model, were developed and applied to IR. The BM25 ranking formula was introduced in the 1990s and became a popular method for ranking documents. Prior to BM25, the vector space model was the primary model used. Other models, such as language models and information theoretic models, emerged in the late 1990s. The Boolean model is still attractive due to its low computational cost, and some models have attempted to extend it with fuzzy or logical operators. However, the potential of these models has not been fully explored. [16, 17, 18, 19, 20]

A model is an abstract representation of a process used to study properties, draw conclusions, make predictions. The quality of the conclusions depends upon how closely the model represents reality. The IR model is a way to represent the contents of a document and a query, also to compare a document representation to a query representation, in order to produce relative documents based on this comparison (document ranking, score function).

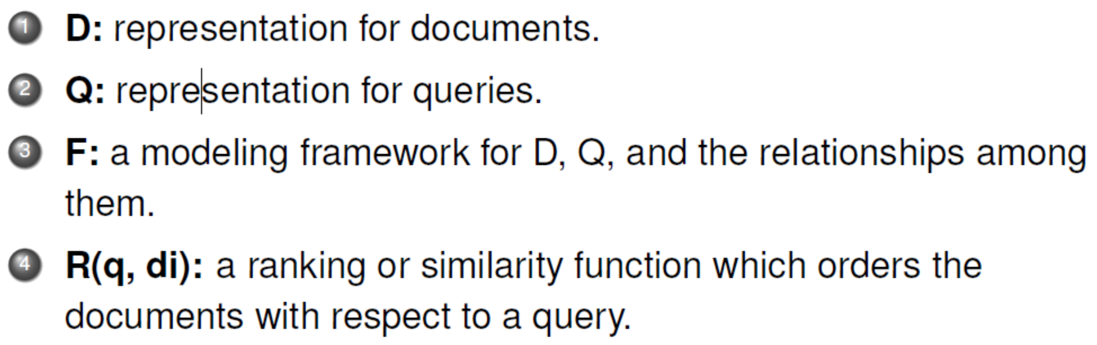
5.2 Components of IR model

Information retrieval (IR) models typically consist of the following components:

- **Document representation:** A method for representing the content of documents in the collection, such as using keywords or natural language processing techniques to extract features from the text.
- **Query representation:** A method for representing user queries in a form that can be compared with the document representation, such as converting the query terms into a vector or a set of keywords.
- **Ranking function:** A function that assigns a relevance score to each document in the collection based on its similarity to the query representation. This function can be based on various models, such as the Boolean model, the vector space model, the probabilistic model, or the language model.

- **Indexing and retrieval algorithm:** A method for efficiently retrieving documents from the collection based on their relevance scores, such as using an inverted index to store the document representations and quickly identify relevant documents for a given query.

- **Evaluation metrics:** A set of measures used to evaluate the effectiveness of the IR system, such as precision, recall, F1-score, and mean average precision.



1. **D:** representation for documents.
2. **Q:** representation for queries.
3. **F:** a modeling framework for D, Q, and the relationships among them.
4. **R(q, di):** a ranking or similarity function which orders the documents with respect to a query.

Figure 5.1: Components of IR Models

These components can be combined in various ways to create different IR models, each with its own strengths and weaknesses. Choosing the right components and combining them effectively is an ongoing research challenge in the field of IR.

5.3 IR model types

Exact and best match information retrieval models are two different approaches to ranking and retrieving documents in response to a user query.

Exact match retrieval models aim to retrieve documents that exactly match the user's query, or contain all of the query terms as specified by the user. These models typically use Boolean operators (AND, OR, NOT) to combine the query terms and match them with the terms in the documents. The most common exact match model is the Boolean model, which retrieves documents that exactly match the Boolean expression created by the query terms.

Best match retrieval models, on the other hand, aim to retrieve documents that are most relevant to the user's query, even if they don't contain all of the query terms. These models use statistical and probabilistic methods to rank the documents based on their relevance to the query. The most common best match models are the vector space model, the probabilistic model, and the language model.

Both exact and best match retrieval models have their strengths and weaknesses, and the choice of model depends on the specific requirements of the application. Exact match models are useful for simple queries and when precision is more important than recall, while best match models are useful for complex queries and when recall is more important than precision.

Exact match	**Best match**
- Query specifies precise retrieval criteria - Every document either matches or fails to match query - Result is a set of documents	- Query describes retrieval criteria for desired documents - Every document matches a query to some degree - Result is a ranked list of documents, best first.

Table 5.1: Differences between Exact match and Best match in IR models

5.4 Boolean model and Vector space model

In this section, we will introduce two different models for information retrieval. The first model is the Boolean model, which is an exact match model based on algebraic theory. The second model is the vector space model, which is a best match model.

5.4.1 Boolean model

The Boolean model (BIR) is a classic information retrieval model that uses Boolean logic to retrieve documents that exactly match a user's query. In this model, a query is represented as a Boolean expression consisting of terms and logical operators such as AND, OR, and NOT. Documents that contain all the terms in the query are retrieved, while documents that do not contain all the terms are excluded. The Boolean model is simple and fast, and it provides a way to perform exact matching of queries to documents. However, it may not be suitable for retrieving documents that are relevant but do not contain all the query terms. [21]

The Boolean model has several advantages in information retrieval, including [21]:

1. Simple model based on Boolean algebra
2. Intuitive concept
3. Precise semantics
4. Clear formal basis
5. Widely adopted by early information systems

5.4.1.1 Representation

The (BIR) is based on Boolean logic and classical set theory in that both the documents to be searched and the user's query. BIR it is very simple model based on the following:

1. Use Boolean algebra and Set theory
2. Document are logical conjunction of terms.
3. Term weights are binary
   - $w_{i,j} \in \{0,1\}$
   - $w_{i,j} = 1$ - term present
   - $w_{i,j} = 0$ - term not present
   - Where i represent token and j represent the document ID.
4. Queries are Boolean expressions
5. Documents are considered **relevant** if the query evaluates to 1 (true)

5.4.1.2 Example

Which plays of Shakespeare contain the words **Brutus** AND **Caesar** but NOT **Calpurnia**?. To answer this query we simply use Term-document incidence matrix, take the vectors for **Brutus**, **Caesar** and **Calpurnia** (complemented) as illustrated in Table 5.2, and then we applied bitwise AND operation.

	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth
Antony	1	1	0	0	0	1
Brutus	**1**	**1**	**0**	**1**	**0**	**0**
Caesar	**1**	**1**	**0**	**1**	**1**	**1**
Calpurnia	**0**	**1**	**0**	**0**	**0**	**0**
¬Calpurnia	**1**	**0**	**1**	**1**	**1**	**1**
Cleopatra	1	0	0	0	0	0
mercy	1	0	1	1	1	1
worser	1	0	1	1	1	0
Bitwise AND	1	0	0	1	0	0

Table 5.2: Boolean model using Term-document incidence matrix

The documents that satisfy the query are those that have a '1' in their column. In this case, the two documents that meet this criterion are **'Antony and Cleopatra'** and **'Hamlet'**.

5.4.1.3 Advantages and Disadvantages of boolean model

The Boolean Model is a simple information retrieval model that uses Boolean operators (AND, OR, NOT) to combine search terms. Here are some of its advantages and disadvantages [22]:

Advantages:

1. **Simplicity:** The Boolean model is easy to understand and implement. It is a simple way to retrieve relevant documents based on user-defined keywords and operators.

2. **Precision:** The Boolean model is highly precise in retrieving documents that match the search query. The use of Boolean operators allows for precise control over the search results.

3. **Flexibility:** The Boolean model is flexible in that it can be used to search for any type of information, including text, images, and multimedia content.

4. **Speed:** The Boolean model is fast in retrieving search results because it involves only simple queries that can be processed quickly.

Disadvantages:

1. **Lack of Ranking:** The Boolean model does not provide any ranking of search results based on relevance, which means that users may need to sift through a large number of irrelevant documents to find the information they need.

2. **No Partial Matches:** The Boolean model requires users to specify exact search terms, which means that partial matches or variations of the search terms may not be captured.

3. **Complex Queries:** Complex queries can be difficult to formulate in the Boolean model, as they require the use of multiple operators and parentheses to specify the desired logic.

4. **Limited Context:** The Boolean model does not consider the context of the search terms or documents, which can result in the retrieval of irrelevant documents that happen to contain the specified keywords.

5.4.2 Vector Space Model

The vector space model (VSM) is a mathematical model used to represent textual documents as vectors in a high-dimensional space. It is a fundamental concept in information retrieval and natural language processing.

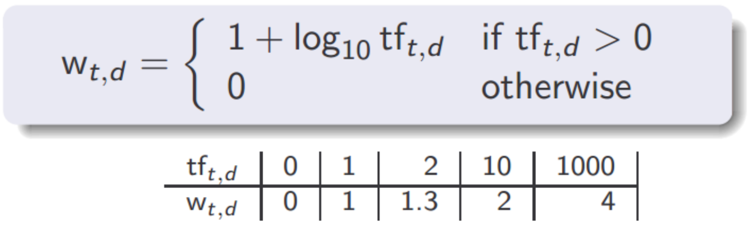
The VSM represents each document as a vector of weights, where each weight represents the importance of a particular term in the document. The weight assigned to each term is typically based on a measure such as term frequency-inverse document frequency (TF-IDF), which takes into account the frequency of the term in the document as well as its rarity across the collection of documents.

5.4.2.1 Term frequency weights

The first component is term frequency, which is a measure of how often a term appears in a given document. It gives weight to terms that appear frequently in a document, as they are likely to be important for understanding the content of the document.

Definition: The term frequency $tf_{t,d}$ of a term t in a document d is defined as the number of times that t occurs in d. This measure gives weight to terms that appear more frequently in a document, as they are likely to be more important for understanding the content of the document.

A variant of the tf weight that is used in the literature is the logarithmically scaled term frequency, which is computed as $w_{t,d}$ illustrated in Figure 5.3. there are also many other variation of weighting schema as illustrated in **??**.

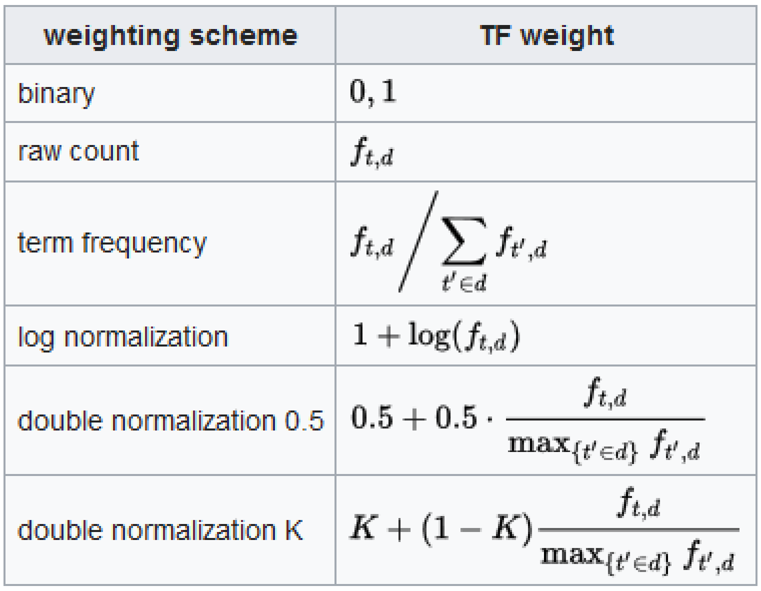


$$\mathrm{w}_{t,d} = \begin{cases} 1 + \log_{10} \mathrm{tf}_{t,d} & \text{if } \mathrm{tf}_{t,d} > 0 \\ 0 & \text{otherwise} \end{cases}$$

$\mathrm{tf}_{t,d}$	0	1	2	10	1000
$\mathrm{w}_{t,d}$	0	1	1.3	2	4

Figure 5.2: $W_{f,d}$ definition

This variant puts less emphasis on terms that appear multiple times in a document and is more suitable for long documents. By taking the logarithm of the term frequency, the weight of terms that appear multiple times in a document is dampened, but not completely eliminated.



weighting scheme	TF weight
binary	$0, 1$
raw count	$f_{t,d}$
term frequency	$f_{t,d} \Big/ \sum_{t' \in d} f_{t',d}$
log normalization	$1 + \log(f_{t,d})$
double normalization 0.5	$0.5 + 0.5 \cdot \frac{f_{t,d}}{\max_{\{t' \in d\}} f_{t',d}}$
double normalization K	$K + (1 - K) \frac{f_{t,d}}{\max_{\{t' \in d\}} f_{t',d}}$

Figure 5.3: $W_{f,d}$ variations

5.4.2.2 Inverse document frequency

The second component is inverse document frequency, which is a measure of how important a term is across all documents in a corpus. It gives less weight to terms that appear frequently in many documents, as they are less informative for distinguishing between documents.

Frequent terms are less informative than rare terms, as they appear in many documents and are therefore less useful for distinguishing between them. To quantify this, document frequency (df_t) is defined as the number of documents that contain a given term t. This is an inverse measure of the informativeness of t, as terms that appear in many documents have a high df value and are therefore less informative.

To capture the informativeness of a term, the inverse document frequency (idf_t) is defined as the logarithm of the total number of documents (N) in the corpus divided by the document frequency of the term, as follows:

$$\mathrm{idf}_t = \log_{10} \frac{N}{\mathrm{df}_t}$$

Figure 5.4: idf_t Idf weight definition

Example:

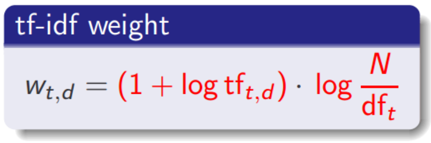
term	df_t	idf_t
calpurnia	1	6
animal	100	4
sunday	1000	3
fly	10,000	2
under	100,000	1
the	1,000,000	0

Figure 5.5: df_t and idf_t weight Examples

5.4.2.3 TF-IDF

The third component is normalization of the product TF-IDF. This involves dividing the product of term frequency and inverse document frequency by the Euclidean length of the vector representing the document. This step ensures that the vector representing each document has a unit length, making it easier to compare documents and calculate their similarity.

The $tf - idf$ weight of a term is the product of its tf weight and its idf weight as illustrated in the Figure 5.6.



tf-idf weight

$$w_{t,d} = (1 + \log \mathrm{tf}_{t,d}) \cdot \log \frac{N}{\mathrm{df}_t}$$

Figure 5.6: $tf - idf$ weight definition

As of now, $tf - idf$ is the best-known weighting scheme in information retrieval. It increases with the number of occurrences of a term within a document and also increases with the rarity of the term in the collection.

5.4.2.4 SMART notation and VSM

VSM (Vector Space Model) uses the three components discussed above to represent the document and the query as vectors in a high-dimensional space, where the terms serve as the axes of the space. To calculate the similarity between a document-query pair, VSM finds the cosine of the angle between the two vectors.

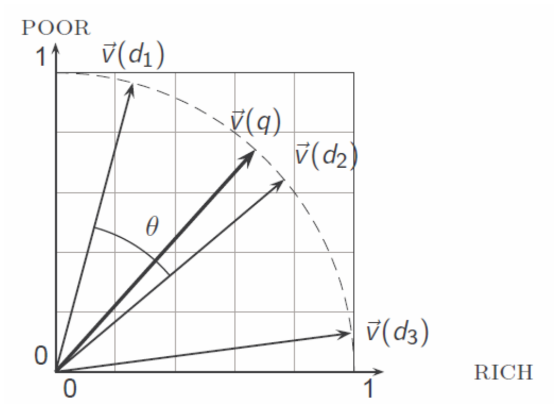
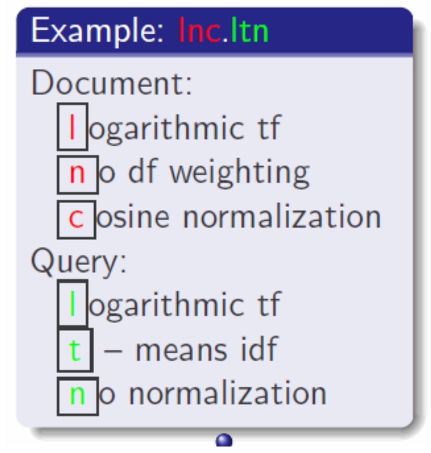


Figure 5.7: Vector Space Model

In SMART notation, the combination of techniques used in an information retrieval engine is denoted by a code consisting of two parts: **ddd** and **qqq**. The **ddd** part indicates the document processing technique used, while the **qqq** part indicates the query processing technique used. SMART notation adjust the tree components of VSM.

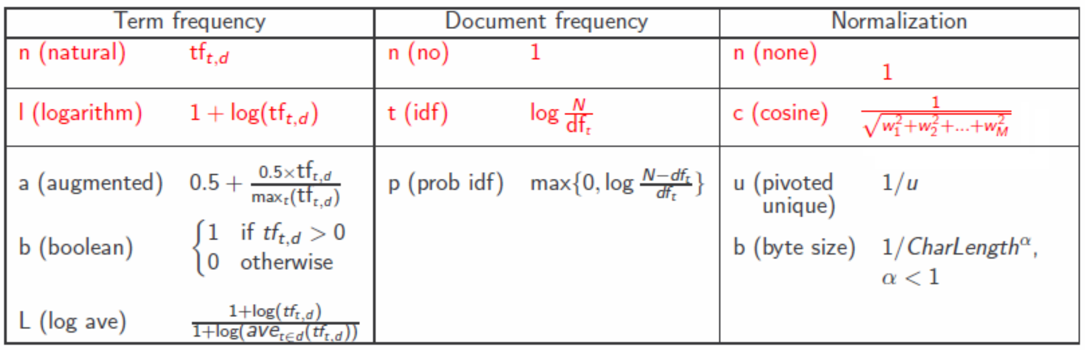


Example: lnc.ltn

Document:
- l ogarithmic tf
- n o df weighting
- c osine normalization

Query:
- l ogarithmic tf
- t – means idf
- n o normalization

Figure 5.8: SMART notation Example 01



Term frequency		Document frequency		Normalization	
n (natural)	$\text{tf}_{t,d}$	n (no)	1	n (none)	1
l (logarithm)	$1+\log(\text{tf}_{t,d})$	t (idf)	$\log \frac{N}{\text{df}_t}$	c (cosine)	$\frac{1}{\sqrt{w_1^2+w_2^2+...+w_M^2}}$
a (augmented)	$0.5+\frac{0.5\times\text{tf}_{t,d}}{\max_t(\text{tf}_{t,d})}$	p (prob idf)	$\max\{0,\log\frac{N-df_t}{df_t}\}$	u (pivoted unique)	$1/u$
b (boolean)	$\begin{cases}1 & \text{if } tf_{t,d}>0\\ 0 & \text{otherwise}\end{cases}$			b (byte size)	$1/CharLength^{\alpha}$, $\alpha<1$
L (log ave)	$\frac{1+\log(tf_{t,d})}{1+\log(ave_{t\in d}(tf_{t,d}))}$				

Figure 5.9: SMART notation Example 02

Example:

Suppose we have the following values of tf of three (3) documents from a collection of 806791 documents, and the values of tf and Idf of four (4) terms:

	Doc 01	Doc 02	Doc 03
car	27	4	24
auto	3	33	0
insurance	0	33	29
best	14	0	17

tf_{raw} of four terms

df	idf
18165	1.65
6723	2.08
19241	1.62
25235	1.5

df and idf values of four terms

Table 5.3: tf, df and idf values of four term

We want to use VSM and SMART notation to found the result of the query "**best car insurance**". The setting of SMART notation is **ltc.ntc**, first we will calculate the values of each document.

	L		T	C	
Term	tf_{raw}	tf_{weight}	idf	tf-idf	Norm V
car	27	2.43	1.65	4.00	**0.71**
auto	3	1.47	2.08	3.05	**0.54**
insurance	0	0	1.62	0	**0**
best	4	1.6	1.5	2.4	**0.43**

Table 5.4: Values of document 01 based on SMART notation LTC

In order to calculate the last column which is the normalization of the final vector values, we use L_2 *norm* as define:

$$|x| = \sqrt{\sum_{k=1}^{n} |x_k|^2}$$

Here, the value of L_2 *norm* of doc 01. $|doc\ 01| = \sqrt{|4|^2 + |3.05|^2 + |2.4|^2} = 5.57$. In order to calculate the final vector, each value of the $tf - idf$ vector will be divided by the length (L_2 *norm*). We do the same calculation for the rest documents (Doc 02, Doc 03) as illustrated in the Table 5.5 and Table 5.6.

	L		T	C	
Term	tf_{raw}	tf_{weight}	idf	tf-idf	Norm V
car	4	1.6	1.65	2.64	**0.37**
auto	33	2.51	2.08	5.22	**0.73**
insurance	33	2.51	1.62	4.06	**0.57**
best	0	0	1.5	0	**0**

Table 5.5: Values of document 02 based on SMART notation LTC

	L		T	C	
Term	tf_{raw}	tf_{weight}	idf	tf-idf	Norm V
car	24	2.38	1.65	3.92	**0.60**
auto	0	0	2.08	0	**0**
insurance	29	2.46	1.62	3.98	**0.61**
best	17	2.23	1.5	3.34	**0.51**

Table 5.6: Values of document 03 based on SMART notation LTC

The Table 5.7 show the calculation of the vector of the query based on SMART notation **ntc**.

	N	T	C	
Term	tf_{raw}	idf	tf-idf	Norm V
car	1	1.65	1.65	**0.6**
auto	0	2.08	0	**0**
insurance	1	1.62	1.62	**0.59**
best	1	1.5	1.5	**0.54**

Table 5.7: Values of query based on SMART notation LTC

In order to rank the pairs of documents and queries, we will utilize cosine similarity, which is defined as:

$$\cos(\overrightarrow{q}, \overrightarrow{d}) = \overrightarrow{q} * \overrightarrow{d} = \sum_{i=1}^{|v|} q_i * d_i$$
$$score(q, Doc\ 01) = 0.43 * 0.54 + 0.71 * 0.6 + 0 * 0.59 = 0.65$$
$$score(q, Doc\ 02) = 0 * 0.54 + 0.37 * 0.6 + 0.57 * 0.59 = 0.55$$
$$score(q, Doc\ 03) = 0.51 * 0.54 + 0.6 * 0.6 + 0.61 * 0.59 = 0.99$$

For the ranked result: Doc 03, Doc 01 and Doc 02.

5.5 Evolution of IR

The evaluation of an information retrieval system is crucial to assess its performance and effectiveness in retrieving relevant information for the users. Here are some common approaches to evaluating information retrieval systems:

5.5.1 Precision and Recall

Precision and recall are two evaluation metrics used in the field of information retrieval to assess the ability of a system to retrieve relevant documents based on a user's query. These measures are defined as follows [23]:

Precision = Total number of documents retrieved that are relevant/Total number of documents that are retrieved.

Recall = Total number of documents retrieved that are relevant/Total number of relevant documents in the database.

We can use the same terminology used in a confusion matrix to define these two measures. Let relevant documents be positive examples and irrelevant documents, negative examples. The two measures can be redefined with reference to a special case of the confusion matrix, with two classes, one designated the positive class, and the other the negative class, as indicated in Table 5.8.

		Assigned class	
		Positive	**Negative**
Actual class	Positive	True Positive (TP)	False Negative (FN)
	Negative	False Positive (FP)	True Negative (TN)

Table 5.8: Precision and Recall. The outcomes of classification into positive and negative classes [23]

Recall $= \frac{True\ positives}{Total\ number\ of\ actual\ positives} = \frac{TP}{(TP+FN)}$

Precision $= \frac{True\ positives}{Total\ number\ of\ positives\ predicted} = \frac{TP}{(TP+FP)}$

5.5.2 F-Measure

F-measure is a combination of precision and recall that provides a single score to measure the effectiveness of an information retrieval system. It is calculated as the harmonic mean of precision and recall [24].

F-measure $= 2 * \frac{precision*recal}{precision+recal} = \frac{2*TP}{2*TP+FP+FN}$

6

Web Search and Crawling

6.1 Introduction

Web search and crawling are critical components of information retrieval systems, which are designed to help users find relevant information from large, complex datasets such as the World Wide Web. In this process, web crawling is used to gather and index information from the internet, and web search is used to retrieve and present relevant results to the user.

Web crawling is the process of automatically navigating and retrieving information from the internet. This is typically done by using a web crawler, also known as a spider or a bot. A web crawler is a software program that starts at a given URL and follows the links it finds to other web pages. As the crawler visits each page, it retrieves and stores information about the page's content, structure, and links to other pages. This information is used to build an index of the web, which search engines can use to quickly find relevant pages when a user enters a query.

Web search, on the other hand, is the process of finding relevant pages from the index based on a user's query. When a user enters a search query, the search engine retrieves pages from the index that match the query's keywords and other criteria. These pages are then ranked based on their relevance to the query, and the most relevant results are displayed to the user.

In conclusion, web search and crawling are two critical components of information retrieval systems. Web crawling is used to gather and index information from the internet, while web search is used to retrieve and present relevant results to the user. These processes work together to enable users to quickly find relevant information from the vast amounts of data available on the World Wide Web. [4]

6.2 Web search basics

Information retrieval on the web differs from traditional information retrieval in many aspects. For example, web IR requires the use of crawlers for indexing data. Therefore, it is important to address several basic concepts before proceeding further in this chapter.

6.2.1 Internet and internet protocols

The internet is a global network of interconnected computers and servers that communicate with each other using standardized protocols. It allows users to access and share information, communicate with each other through various platforms such as email, social media, instant messaging, and access a vast array of online services and resources such as websites, online shopping, streaming media, and more. The internet has revolutionized the way people connect and interact with each other and has become an essential part of modern life. Some of the basic services available to Internet users are:

- Email: A fast, easy, and inexpensive way to communicate with other Internet users around the world.
- Telnet: Allows a user to log into a remote computer as though it were a local system.
- FTP: Allows a user to transfer virtually every kind of file that can be stored on a computer from one Internet-connected computer to another.
- UseNet news: A distributed bulletin board that offers a combination news and discussion service on thousands of topics.
- World Wide Web (WWW): A hypertext interface to Internet information resources.

These services offer information on the web, but the methods for extracting this information vary from one service to another. Therefore, techniques and models for web information retrieval need to be developed specifically for this type of information. In the upcoming section, we will delve into the process of extracting information from the World Wide Web using web information retrieval techniques.

6.2.2 World wide web

WWW stands for World Wide Web. A technical definition of the World Wide Web is: the universe of network-accessible information, an embodiment of human knowledge. [25] the most characteristics of www are:

- Billions of documents and contents
- Authored by millions of diverse people.
- Distributed over millions of computers, connected by variety of media.

The first characteristic of the World Wide Web (WWW) is that there are millions of documents and content available on the internet. This means that there is a vast number of websites available, and the number of pages accessible is almost infinite. Google in July 2007 announced to have identified 1 trillion (10^{12}) of unique pages/URLs in the Web.

6.2.3 Web site

Website is a group of World Wide Web pages usually containing hyperlinks to each other and made available online by an individual, company, educational institution, government, or organization [26].

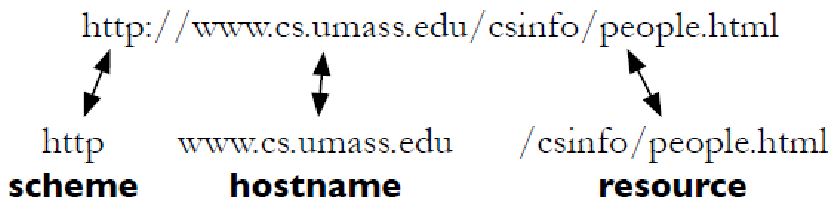
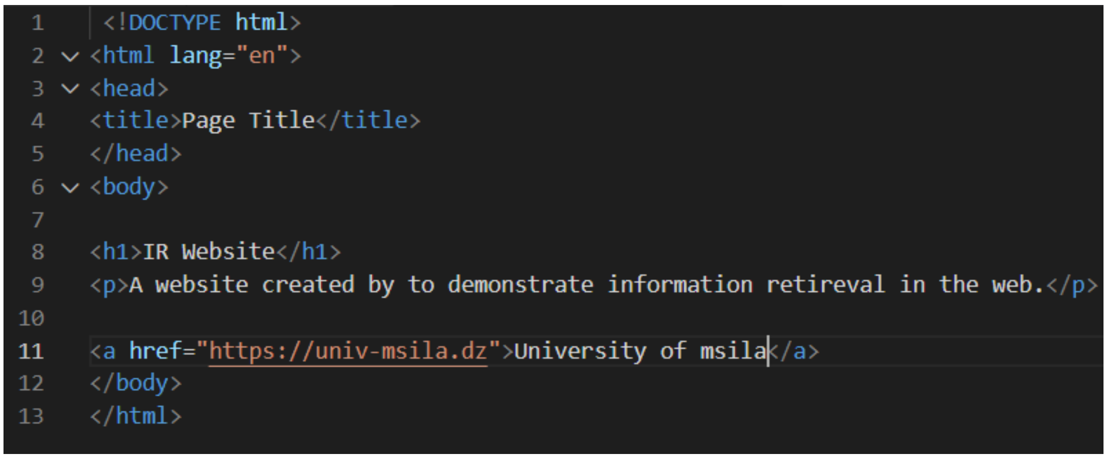


Figure 6.1: Website example

- Each web page on the Internet has its own unique URL (uniform resource locator).
- Use a protocol called Hypertext Transfer Protocol, or HTTP, to exchange information with client software (Web browsers Or web crawlers).

6.2.4 Web page

A web page is document based on HTML which stand for Hyper Text Markup Language on the World Wide Web. Web pages are delivered by a web server to the user and displayed in a web browser [27].

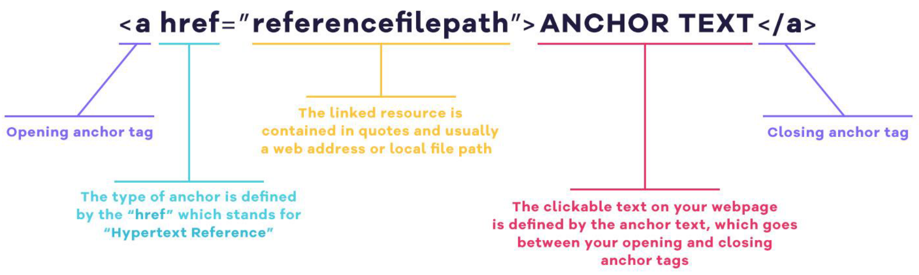


```html
<!DOCTYPE html>
<html lang="en">
<head>
<title>Page Title</title>
</head>
<body>

<h1>IR Website</h1>
<p>A website created by to demonstrate information retrieval in the web.</p>

<a href="https://univ-msila.dz">University of msila</a>
</body>
</html>
```

Figure 6.2: Webpage example

6.2.5 hyper link

A hyperlink or simply a link is a selectable element in a web page that serves as an access point to other electronic resources.



<a href="referencefilepath">ANCHOR TEXT</a>
Opening anchor tag
The linked resource is contained in quotes and usually a web address or local file path
Closing anchor tag
The type of anchor is defined by the "href" which stands for "Hypertext Reference"
The clickable text on your webpage is defined by the anchor text, which goes between your opening and closing anchor tags

Figure 6.3: hyper link example

6.3 Web Search Architecture

Web search architecture refers to the overall design and structure of a search engine's software system, which is responsible for processing and returning search results in response to user queries. The architecture of a search engine involves several components, each of which has a specific function in the search process. [28]

The main components of a typical web search architecture include:

1. **Crawling and Indexing:** This component is responsible for discovering and retrieving web pages from the internet and indexing them to make them searchable. The crawler visits web pages and extracts their content and metadata, which is then stored in the search engine's index.

2. **Query Processing:** When a user enters a query, the search engine's query processing component analyzes the query and retrieves relevant documents from the index.

3. **Ranking:** Once the query processing component has retrieved a set of relevant documents, the ranking component assigns a relevance score to each document based on a variety of factors such as the content of the page, its popularity, and its quality.

4. **User Interface:** The user interface component is responsible for presenting search results to the user in a user-friendly format. This component may include features such as autocomplete suggestions, filtering options, and visual aids.

5. **Analytics and Reporting:** This component tracks user behavior and search patterns and provides insights into search engine performance. This information is used to improve the search engine's functionality and relevance.

Overall, the web search architecture is designed to provide users with fast, accurate, and relevant search results by processing large volumes of data in real-time. [28]

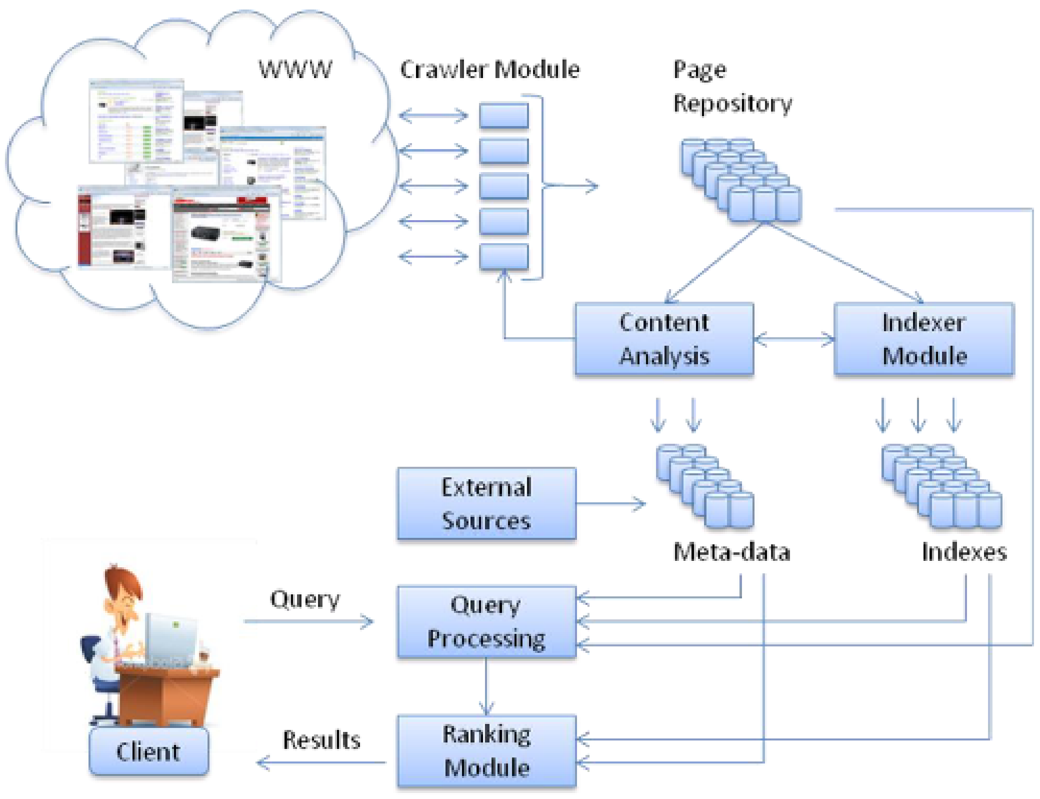


Figure 6.4: Web Search Architecture [28]

6.4 Web Page Crawling

The act of web crawling involves the collection of web pages from the internet with the aim of indexing them and facilitating the functioning of a search engine. The primary goal of crawling is to efficiently and expeditiously gather a substantial number of valuable web pages while also capturing their link structure that connects them. [4]

6.4.1 Definition of Crawler

Crawl describes a bot, script, or software program that visits a web page and grabs its content and links. Once completed, the program visits the next link in the list or a link obtained from the web page it had recently visited as illustrated in Figure 6.5.

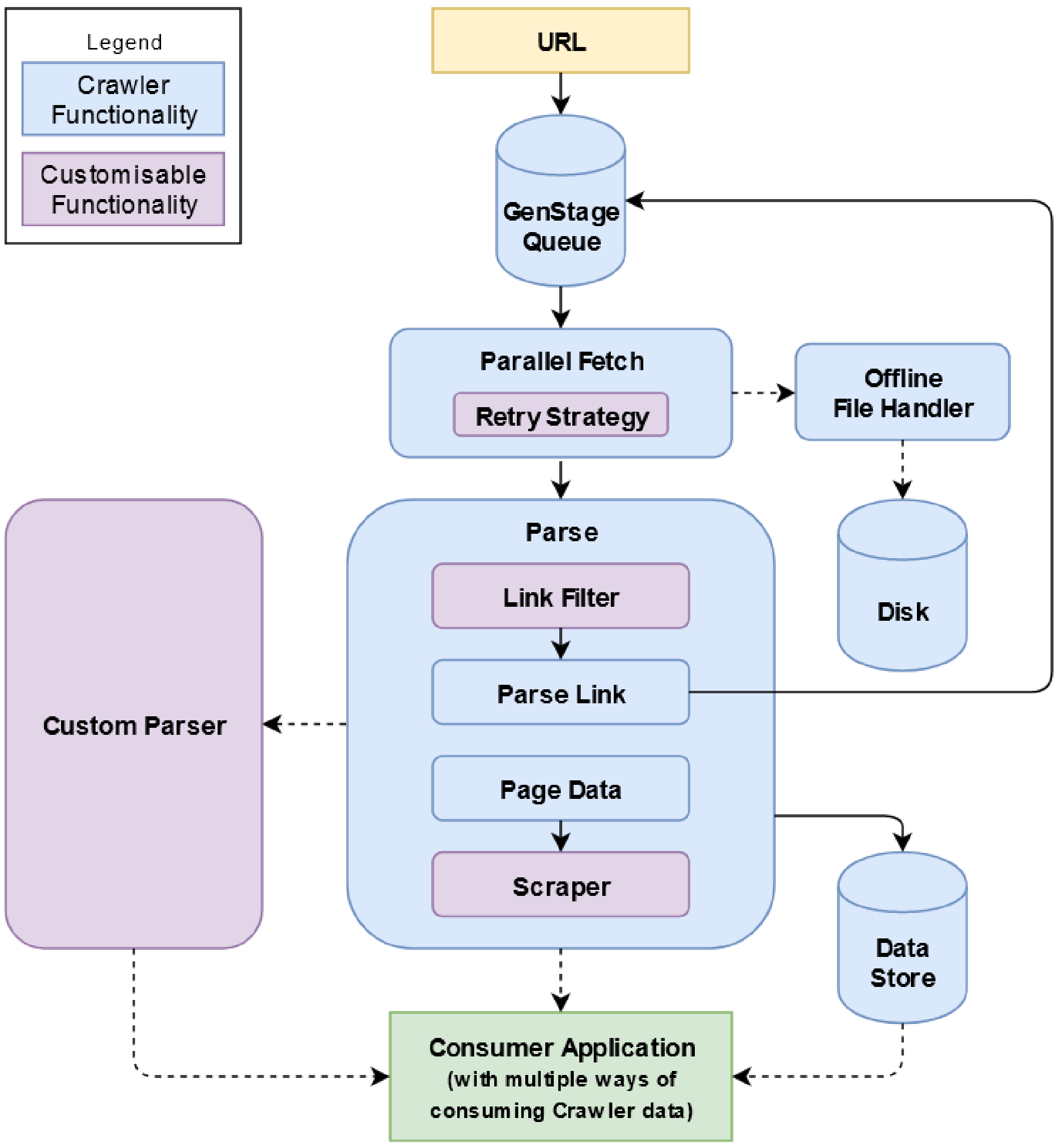


Figure 6.5: Web Crawler

The two main task in web crawler in information retrieval system are:

1. **Building Index (Crawling)** web pages are particularly easy to copy, since they are meant to be retrieved over the Internet by browsers. This instantly solves one of the major problems of getting information to search.

2. **Searching over this obtained resources (Freshness)**

6.4.2 Crawler challenges

Web crawlers, also known as spiders or bots, face several challenges in their task of efficiently and accurately gathering web pages for indexing. Some of these challenges include:

1. **Technical issues:** Web pages may have technical issues such as broken links, server errors, and slow page load times, which can hinder a crawler's ability to access and index the page.

2. **Duplicate content:** Many web pages have duplicate content, either intentionally or unintentionally, which can confuse the crawler and make it difficult to determine which version of the page to index.

3. **Dynamic content:** Web pages with dynamic content, such as user-generated content and personalized pages, can be challenging to crawl and index, as the content is constantly changing and may require different crawling and indexing strategies.

4. **Spam and malicious content:** Some web pages may contain spam, malware, or other malicious content that can harm the crawler or the user's device.

5. **Legal issues:** Crawling and indexing web pages may raise legal issues such as copyright infringement, privacy violations, and terms of service violations.

To overcome these challenges, web crawlers use various techniques such as identifying and avoiding spam and malicious content, following a polite crawling policy, and implementing techniques to handle dynamic and duplicate content. However, these techniques may not always be foolproof and can sometimes lead to inaccuracies in search results.

6.4.3 Crawler process

The process of web crawling, also known as spidering, involves several steps. Here's a general overview of the crawler process as illustrated in Figure 6.6:

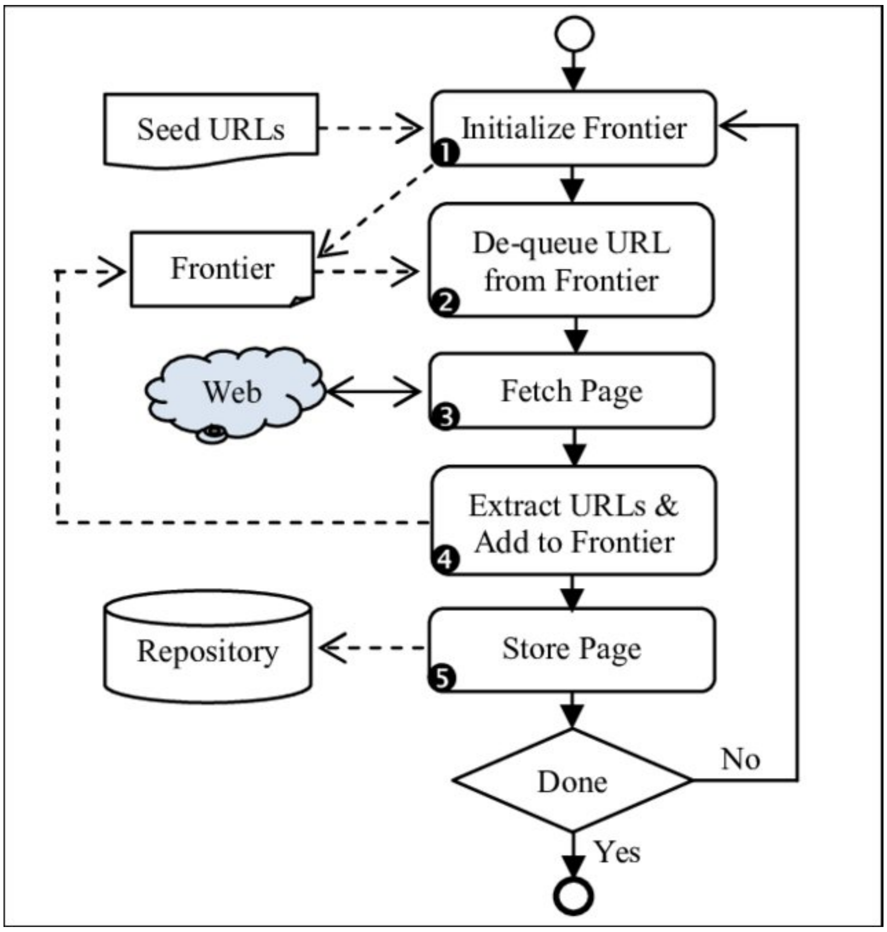


Figure 6.6: Web Crawler process

1. **Seed URL Selection:** The process starts with the selection of a seed URL, which is typically the home page of a website or a list of URLs to crawl.

2. **URL Discovery:** The crawler follows the seed URL and extracts links to other pages, which it adds to a queue of URLs to crawl.

3. **URL Prioritization:** The crawler prioritizes URLs based on several factors such as the quality of the page, the number of links pointing to it, and its relevance to the search query.

4. **Web Page Retrieval:** The crawler retrieves the web pages from the URLs in its queue and downloads their HTML content.

5. **Parsing:** The crawler parses the HTML content to extract text, links, and other metadata such as page titles and descriptions.

6. **Indexing:** The crawler adds the extracted content and metadata to its index, which is used to answer user queries.

7. **Recursion:** The crawler repeats the process by following the links on the pages it has crawled and adding new URLs to its queue.

8. **Politeness:** The crawler follows a polite crawling policy to avoid overloading servers and to respect the site owner's wishes.

9. **Crawling Metrics:** The crawler may collect crawling metrics such as page load time, crawl depth, and crawl frequency to optimize its crawling strategy.

The crawler process is an ongoing and iterative process, with crawlers revisiting pages periodically to update their index and to discover new content. The process continues until the crawler either runs out of disk space to store pages or runs out of useful links to add to the request queue.

6.4.4 Crawler policy

A crawler policy, also known as a web crawling policy, is a set of guidelines that dictate how a web crawler should behave when accessing and crawling websites. These policies are usually implemented to protect the website being crawled from excessive requests and to ensure that the crawler operates within legal and ethical boundaries.

The behavior of a web crawler is the outcome of a combination of policies:

- a selection policy that states which pages to download. Robots.txt
  Robots.txt: The robots.txt file is a file that is placed on a website to inform crawlers which pages to crawl and which pages to ignore. A crawler policy should respect the directives in the robots.txt file.

```
User-agent: *
Disallow: /private/
Disallow: /confidential/
Disallow: /other/
Allow: /other/public/

User-agent: FavoredCrawler
Disallow:

Sitemap: http://mysite.com/sitemap.xml.gz
```

- a re-visit policy that states when to check for changes to the pages. Freshness
- a duplication policy
- a politeness policy that states how to avoid overloading web sites.
  One thread - it would not be very efficient because web crawler spends a lot of its time waiting for responses
  Multiple thread - is good for person running web crawler but not necessarily good for the person running the web server on the other end.
- a parallelization policy that states how to coordinate distributed web crawler.

Bibliography

[1] D. Fallows, "The internet and daily life. pew internet and american life project," *Retrieved September*, vol. 15, p. 2004, 2004.

[2] V. Bush and J. Wang, "As we may think," *Atlantic Monthly*, vol. 176, pp. 101–108, 1945.

[3] F. Yu, H. Luo, Z. Lu, and P. Wang, *Three-Dimensional Model Analysis and Processing*. Springer Berlin Heidelberg, 2010.

[4] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to Information Retrieval*. New York, NY, USA: Cambridge University Press, 2008.

[5] M. Sarfraz, *Critical Approaches to Information Retrieval Research*. IGI Global, 2019.

[6] W. I. Merriam, *Merriam-Webster's Collegiate Dictionary*. Springfield, 2005.

[7] N. Fielden, "History of information retrieval systems & increase of information over time," in *Biennial California Academic & Research Librarians (CARL) Conference*, 2002.

[8] R. A. Baeza-Yates and B. A. Ribeiro-Neto, *Modern Information Retrieval*. ACM Press / Addison-Wesley, 1999.

[9] F. Crestani, S. Mizzaro, and I. Scagnetto, *Mobile Information Retrieval*. 01 2017.

[10] J. Golbeck, "LBSC 690 information retrieval and search." Information technology, 2023.

[11] M. Song and Y.-F. Brook Wu, *Handbook of research on text and web mining technologies*. IGI global, 2008.

[12] J. Daniel, M. James H, *et al.*, *Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition*. prentice hall, 2007.

[13] E. Brill and R. C. Moore, “An improved error model for noisy channel spelling correction,” in *Proceedings of the 38th annual meeting of the association for computational linguistics*, pp. 286–293, 2000.

[14] S. E. Robertson and S. Walker, “Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval,” in *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR ’94, (Berlin, Heidelberg), p. 232–241, Springer-Verlag, 1994.

[15] P. E. Black, “Inverted index.” Dictionary of Algorithms and Data Structures, 2017.

[16] S. E. Robertson and S. Walker, “Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval,” in *SIGIR’94: Proceedings of the Seventeenth Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval, organised by Dublin City University*, pp. 232–241, Springer, 1994.

[17] G. Salton, “Introduction to modern information retrieval,” *McGraw-Hill*, 1983.

[18] J. M. Ponte and W. B. Croft, “A language modeling approach to information retrieval. dans les actes de proceedings of the 21st annual international acm sigir conference on research and development in information retrieval, melbourne, australia, 275–281,” 1998.

[19] G. Amati and C. J. Van Rijsbergen, “Probabilistic models of information retrieval based on measuring the divergence from randomness,” *ACM Transactions on Information Systems (TOIS)*, vol. 20, no. 4, pp. 357–389, 2002.

[20] S. Clinchant and E. Gaussier, “Information-based models for ad hoc ir,” in *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, pp. 234–241, 2010.

[21] R. Baeza-Yates and B. Ribeiro-Neto, *Modern Information Retrieval: The Concepts and Technology behind Search*. USA: Addison-Wesley Publishing Company, 2nd ed., 2011.

[22] M. Pannu, A. James, and R. Bird, “A comparison of information retrieval models,” 05 2014.

[23] K. M. Ting, *Precision and Recall*, pp. 781–781. Boston, MA: Springer US, 2010.

[24] C. Sammut and G. I. Webb, eds., *F-Measure*, pp. 416–416. Boston, MA: Springer US, 2010.

[25] World Wide Web Consortium, "World Wide Web Definition." https://www.w3.org/standards/webdesign, 2021. Accessed on: October 12, 2021.

[26] Merriam-Webster, "Website." Retrieved March 13, 2023, from https://www.merriam-webster.com/dictionary/website.

[27] thefreedictionary, "Webpage." Retrieved March 13, 2023, from https://www.thefreedictionary.com/web+page.

[28] D. N. Singhal, A. Dixit, and A. Sharma, "Design of a priority based frequency regulated incremental crawler," *International Journal of Computer Applications*, vol. 1, 02 2010.