Chapter 2

Introduction to Information Retrieval System

2.1 What is Information Retrieval (IR)?

The meaning of the term information retrieval (IR) can be very broad. In 1951, Calvin Mooers coined the term “Information retrieval” to describe the process through which a prospective user of information can convert a request for information into a useful collection of references [10].

According to *Calvin Mooers*:
"Information Retrieval embraces the intellectual aspects of the description of information and its specification for search, and also whatever systems, techniques, or machines that are employed to carry out the operation."
However, as an academic field of study, information retrieval might be defined as:

“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)” [11].

Information Retrieval (IR) aims at modelling, designing, and implementing systems able to provide fast and effective content-based access to large amounts of information [58]. The aim of an IR system is to estimate the relevance of information items, such as text documents, images and video, to a user’s information need. Such information need is represented in the form of a query, which usually corresponds to a bag of words. Users are

only interested in the information items that are relevant to their information need. The representation and organization of the information items should provide the user with easy access to the information in which he is interested. The primary goal of an IR system is to retrieve all the information items that are relevant to a user query while retrieving as few non-relevant items as possible [58]. Furthermore, the retrieved information items should be ranked from the most relevant to the least relevant.

Information Retrieval is different from data retrieval. Data retrieval mainly consists of determining which documents of a collection contain the keywords in the user query which is not enough to satisfy the user information need. In fact, the user of an IR system is concerned more with retrieving information about a subject than with retrieving data which satisfies a given query. For an Information Retrieval system, the retrieved object might be inaccurate and small errors are likely to go unnoticed but for a data retrieval system, however a single erroneous object among a thousand retrieved objects means total failure. To be more effective the IR system must somehow "interpret" the contents of the information items (documents) in a collection and rank them according to a degree of relevance to the user query. This "interpretation" of document content involves extracting syntactic and semantic information from the document text and using this information to match the user information need. The notion of relevance is at the center of information retrieval [2]. The most difficult part of the retrieval process is deciding which documents are related to, or satisfy a certain query. Documents should be preferably ranked in decreasing order of relevance. An IR system achieves its maximum retrieval effectiveness when the relevant documents with respect to the query are ranked higher, whereas the non-relevant are ranked lower.

2.2 Objectives of Information Retrieval System

The general objective of an Information Retrieval System is to minimize the overhead of a user locating needed information. Overhead can be expressed as the time a user spends in all of the steps leading to reading an item containing the needed information (e.g., query generation, query execution, scanning results of query to select items to read, reading non-relevant items). The success of an information system is very subjective, based upon what information is needed and the willingness of a user to accept overhead. Under some circumstances, needed information can be defined as all information that is in the system

that relates to a user's need. In other cases it may be defined as sufficient information in the system to complete a task, allowing for missed data. For example, a financial advisor recommending a billion dollar purchase of another company needs to be sure that all relevant, significant information on the target company has been located and reviewed in writing the recommendation. In contrast, a student only requires sufficient references in a research paper to satisfy the expectations of the teacher, which never is all inclusive. A system that supports reasonable retrieval requires fewer features than one which requires comprehensive retrieval. In many cases comprehensive retrieval is a negative feature because it overloads the user with more information than is needed. This makes it more difficult for the user to filter the relevant but non-useful information from the critical items. In information retrieval the term "relevant" item is used to represent an item containing the needed information. In reality the definition of relevance is not a binary classification but a continuous function. From a user's perspective "relevant" and "needed" are synonymous. From a system perspective, information could be relevant to a search statement (i.e., matching the criteria of the search statement) even though it is not needed/relevant to user (e.g., the user already knew the information).

2.3 Components of Information Retrieval System

Information Retrieval is typically a two – steps process:

(i) First potentially relevant documents are identified

(ii) And then found documents are ranked

The identification process is often conducted as set intersection – from the set of all documents the potentially relevant documents are those that contain all or some of the search items.

Ranking involves combining a set of heuristics derived from the corpus, the result set, and individual documents. Typical heuristics include tf (term frequency), idf (inverse document frequency) proximity measures etc. The similarity of each document to the query is computed and the documents are sorted according to the ranking function based on these heuristics.

The various components of Information Retrieval System are as follows:

2.3.1 Indexing:

For IR systems, in order to efficiently judge whether the documents from a corpus match a given query, a pre-process called indexing is usually applied. It is the way documents are managed in the collection. To make searching more efficient, a retrieval system stores documents in an abstract representation. A set of keywords is stored, along with links to the document in which each word appears. This structure for storing indexing information is called an inverted file. Although there are other options, the most popular data structure employed by IR systems is the *inverted file* (IF). An IF is a traversed representation of the original document collection, organised in *posting lists*. Each entry in the inverted file contains information about a single term in the document collection. Since this structure requires a large amount of space to be stored, posting lists usually are compressed.

To better explain the indexing process, we use the following sentence as an example:

"किताबें टेबल के ऊपर रखी हैं।"

The indexing process includes several steps, which are described as follows:

2.3.1.1 Tokenization:

The first stage of the indexing process is typically known as tokenization [11]. In this phase, documents text is parsed and index words called Tokens are generated. In addition, at this stage, all characters contained in the tokens are often lower-cased and all punctuations are removed. Every language has a different internal binary encoding for the characters in the language. We assume all the documents (English as well as Hindi) are encoded in Unicode based on UTF-8, using multiple 8-bit bytes. After tokenisation, the above example sentence can be viewed as:

किताबें टेबल के ऊपर रखी हैं

2.3.1.2 Stop Words Removal:

Luhn pointed out that the frequency of a term within a document can be a good discriminator of its significance in the document [21]. In addition, there are many extremely frequent terms (e.g. "the") that appear in almost all documents of a corpus. These terms are called **stop words**, which bring little value for the purpose of representing the content of documents and are normally filtered out from the list of potential indexing terms during the indexing process [58]. Removing the stop words allows also the reduction of the size of the generated document index. However, removing stopwords from one document at a time is time consuming. A cost-effective approach

consists in removing all terms which appear commonly in the document collection, and which will not improve retrieval of relevant documents. We have categorized stopwords in two categories – Relational (**नीचे**, **ऊपर**, **आगे**, **अंदर** etc.: Hindi; above, below, inside, outside etc.: English) & Non-relational (**है, था**, **रहा**, **सकता** etc.: Hindi; is, are, an, a etc.: English) (see Appendix A and Appendix B). These stopwords have different impact on the information retrieval process. Relational stopwords indicate semantic relevance that is necessary for efficient information retrieval. Removing relational stopwords from the document would result in loss of such relevant semantic information resulting in decrease of relevance efficiency of the system. While removing non-relational stopwords would reduce the document length resulting into faster search. We remove only non-relational stopwords to perform relation inclusive searching. After applying stop words removal to our example sentence, the text is reduced to the following:

किताबें टेबल के ऊपर रखी

2.3.1.3 Stemming: Often, a user specifies a term (e.g. **किताबें**) in a query when only a variant of this term (e.g. **किताब**) is contained in a relevant document. Hence, it would be beneficial for retrieval if documents containing variants of the query term were also considered. Plurals, gerund forms, and past tense suffixes are examples of syntactical variations which prevent a perfect match between a query term and a respective document term [58]. This problem can be alleviated by applying stemming, which replaces a term with its stem, so that different grammatical forms of terms are represented in a common base form. A stem is the portion of a word which is obtained after chopping off its affixes. Stemming refers to the process of reducing terms to their stems or root variant. Thus, "**किताबों**" and "**किताबें**" are reduced to "**किताब**" (in Hindi); "concatenates" and "concatenated" are reduced to "concatenate" (in English).

By applying our proposed RankStem stemming algorithm (see chapter 5), our example sentence is transformed as follows:

किताब टेबल के ऊपर रख

It is easy to note that some words are unchanged (e.g. "**टेबल**" and "**ऊपर**"), some are chopped to their root (e.g. "**किताब**" and "**रख**").

2.3.2 Index Data Structure:

To enable efficient access to document representatives, a suitable data structure is necessary. The most widely used data structure is the inverted index, which is a word-oriented mechanism [58]. In general, the inverted index structure contains two components: vocabulary and posting list. The vocabulary is a set of all different terms extracted from the corpus by the above steps. The occurrences store each vocabulary term's statistics in each document, such as term frequency and term position (see Table 2.1 for an example).

Format of Posting List:

<d>,<n>:[[<pos$_1$>#{<relation$_1$>}>],[<pos$_2$>#{<relation$_2$>}],. .,[<pos$_n$>#{<relation$_n$>}]]

where,

<d>: document name

<n>: term frequency in document <d>

<pos$_j$> : j^{th} position of term in document <d>

<relation$_j$> : relation represented by the term at j^{th} position. It is present only in case of relational stopwords

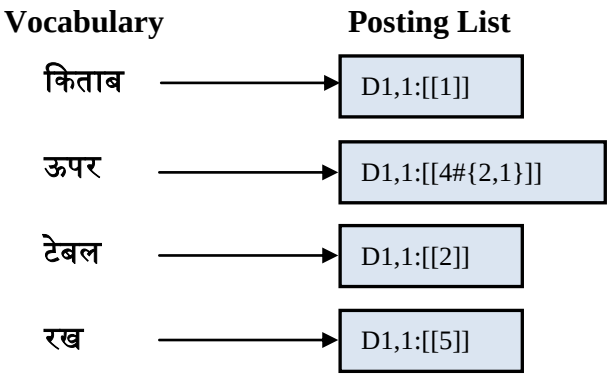
Relation information is also stored in the posting list of relational stopwords along with its position. In our example, relational stop word is **ऊपर**. It's posting list is given by:

ऊपर → D1,1:[[4#{2,1}]]

{2,1} means relation **ऊपर** is existing between words present at 2^{nd} and 1^{st} position i.e. **टेबल** =>**ऊपर**<= **किताब**. (see subsection 6.2.2)

We assume that a corpus contains only one document, which only has a single sentence, which is the same as our example sentence.

Then inverted index structure will be:



2.3.3 Query Parser: It performs tokenization, stemming and stop words removal operations on query so that it would be easy to perform matching on indexed documents for these query terms.

2.3.4 Matching: With a given query, an ideal IR system should only return relevant documents and ranks these documents in decreasing order of relevance. In this phase, all the documents containing query terms are retrieved from the inverted index structure. The relevance of a document to a given query can be estimated by various IR models, such as the Boolean Model (BM), Vector Space Model (VSM), and Probabilistic Model (PM) (discussed in section 2.4).

2.3.5 Ranking: Finally, all the retrieved documents are ranked according to their relevance score using the generated learnt ranking function. (see chapter 4)

2.3.6 User Interface: Interface manages interaction with the user by taking query as input and displaying documents according to their relevance score as output.

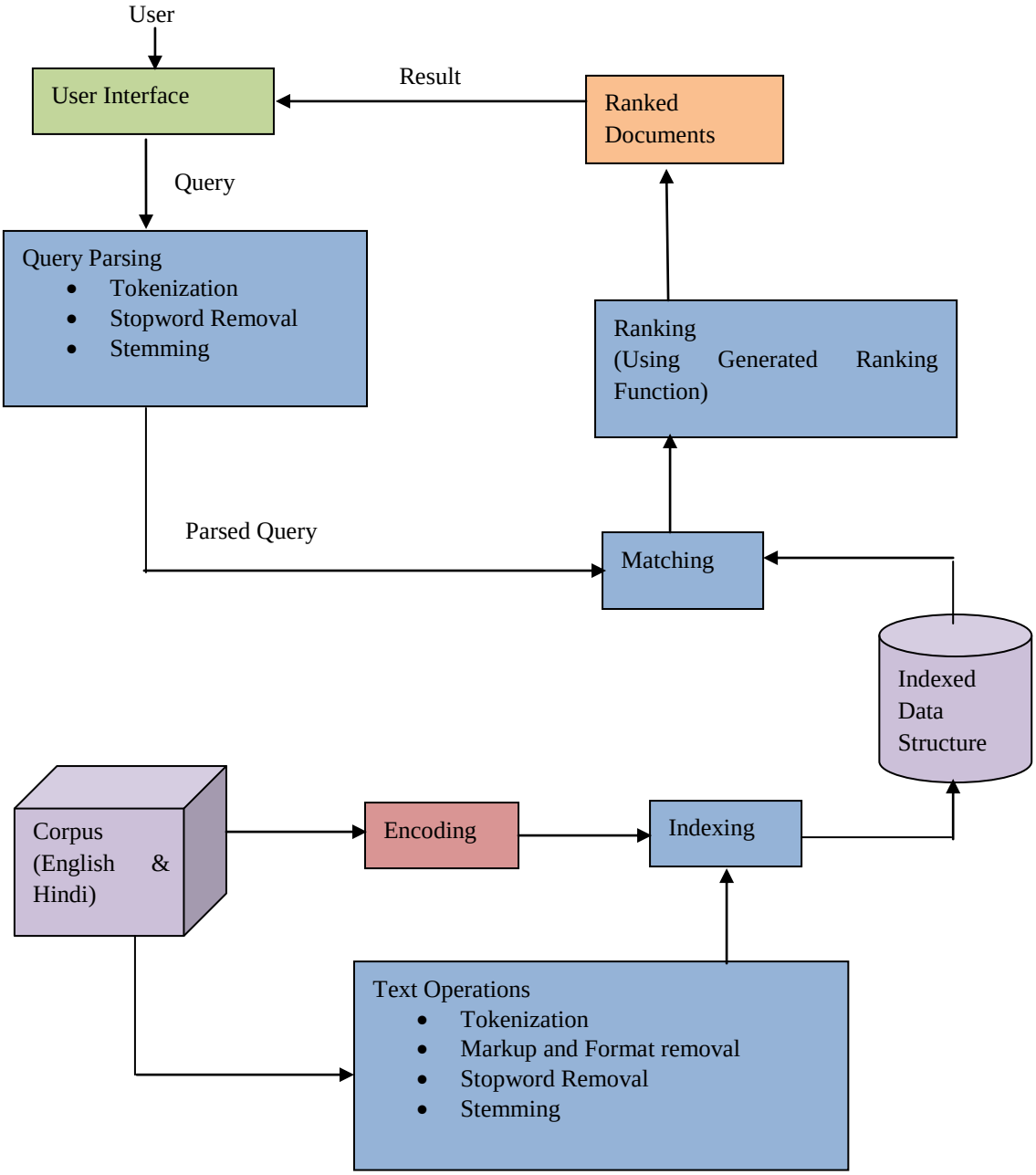


Figure 2.1 Architecture of Information Retrieval System

2.4 Basic Ranking Models

This section contains a brief description of various information retrieval models.

Definition: IR model is a pattern that defines several aspects of retrieval procedure, for e.g., how the documents and queries are represented, how the system retrieves relevant document queries and how the retrieved documents are ranked. [63]

Several IR models have been developed so far. These models can be classified as:

2.4.1 Boolean/Binary Logic Model

Boolean Model is one of the oldest IR models. It is based on set theory and Boolean algebra [95]. In this model, queries are formulated as Boolean expression of terms, that is, in which terms are combined with operators AND, OR and NOT. As an example, with a given Boolean query **"table AND book (NOT pen)"**, the Boolean model would retrieve all documents that contain both the terms **table** and **book** but not the term **pen**. In addition, the retrieved documents are returned to the user as a set without any ranking due to its binary decision criteria. As a consequence, the Boolean model frequently returns either too few or too many documents in response to a user query. Another drawback is the restrictions in the query formulation process, which may return misleading results if the query is not well-stated. Also, Boolean logic operators are hardly able to model a user's information need posed in natural language. All this disadvantages turn the Boolean model down as an option for modern retrieval systems, and more precisely for web search.

2.4.2 Statistical Model

These models use statistical information in the form of term frequency, inverse document frequency etc. to determine the relevance of documents with respect to a query. Two major examples of statistical retrieval approach are:

2.4.2.1 Vector Space Model (VSM): The representation of a set of documents as vectors in a common vector space is known as the Vector Space Model [11]. The vector $V(d)$ is derived from the document d with one component in the vector for each dictionary term i.e. the size of the vector is equal to the size of the dictionary. The component may be

computed using tf weighting scheme or tf-idf weighting scheme. The set of documents in a collection may be viewed as a set of vectors in a vector space, in which there is one axis for each term. A query can also be viewed as a vector of very short document. The cosine similarity between the query vector & the document vector could be used as a measure of the score of the document for that query. In Vector Space Model, documents and queries are represented as vectors in a common vector space. Retrieval is based on the cosine similarity between the query vector and the document vector that could be used as a measure of the score of the document for that query. Thus, sim(d,q)= v(d).v(q) where v(d), v(q) are two unit vectors representing document and query respectively. The resulting score can thus be used to select the top-scoring document for a query.

For e.g. Consider the query "**best car insurance**" on a fictitious corpus with N = 1,000,000 documents where the document frequencies of auto, best, car and insurance are respectively 5000, 50000, 10000, and 1000.

term		**query**			**document**			**product**
t	**tf**	**df**	**idf**	$\mathbf{w_{t,q}}$	**tf**	**wf**	$\mathbf{w_{t,d}}$	$\mathbf{w_{t,q} \times w_{t,d}}$
auto	0	5000	2.3	0	1	1	0.41	0
best	1	50000	1.3	1.3	0	0	0	0
car	1	10000	2.0	2.0	1	1	0.41	0.82
insurance	1	1000	3.0	3.0	2	2	0.82	2.46

Figure 2.2 Term Statistics in Corpus

Here,

tf is term frequency

df is document frequency

idf is inverse document frequency

$w_{t,q}$ is weight of term t in query q

$w_{t,d}$ is weight of term t in document d

In this example, the weight of a term in the query is simply the idf (and 0 for a term not in the query, such as auto); this is reflected in the column header $w_{t,d}$ (the entry for auto is 0 because the query does not contain the term auto). Therefore, v(q)=<0,1.3,2.0,3.0>

For documents, tf weighting is used with no use of idf but with Euclidean normalization i.e. each component is divided by Euclidean length and convert it into unit vector.

Euclidean length = $\sqrt{\sum_{i=1}^{M} V_i^2(d)}$

where $V_i(d)$ are M components of a vector V(d)

Therefore, v(d)=<0.41,0,0.41,0.82> (0 for term not present in the document). Finally, the dot product of the two vectors gives the net score of 0 + 0 + 0.82 + 2.46 = 3.28. This score is used to rank the document.

2.4.2.2 Probabilistic Model (PM): Information retrieval deals with uncertain information, so probability theory seems to be the most natural way to quantify uncertainty. In this model, retrieval is based on whether a probability of relevance of a document is higher than that of a non-relevance. A query, documents and a cut off value is given. Using this information, the probabilities that a document is relevant and irrelevant to the query is calculated. The document with probabilities of relevance at least that of irrelevance are ranked in decreasing order of their relevance. Those documents are retrieved whose probabilities of relevance in the ranked list exceed the cut-off value [11].

Let x be a document in collection represented by a binary vector, $x = (x_1, x_2, \ldots, x_n)$ where $x_i = 0$ or 1 indicates absence or presence of the i^{th} index term. Let w_1 represent relevance of a document w.r.t given (fixed) query and let w_2 represent non-relevance. Using Bayesian formula,

$$P(w_i|x) = \frac{P(x|w_i)P(w_i)}{P(x)} \qquad i = 1,2$$

$P(w_i)$ is the prior probability of retrieving a (non) relevant document

$P(x|w_i)$ is the probability that if a relevant (non-relevant) document is retrieved, it is x.

Finally,

$$P(x) = \sum_{i=1}^{2} P(x|w_i)P(w_i)$$

Ranking Principle (Bayes' Decision Rule): If $P(w_1|x)>P(w_2|x)$ then x is relevant, otherwise x is not relevant. It also minimizes the average probability error,

$$P(error|x) = \begin{cases} P(w_1|x) & \text{If x is non-relevant} \\ P(w_2|x) & \text{If x is relevant} \end{cases}$$

$$P(error) = \sum_x P(error|x)P(x)$$

For ranking purpose, Binary Independence Model is used. This model assumes that the relevance of each document is independent of other documents. Also, assume that the component variables x_i of x to be stochastically independent. Thus,

$P(x| w_i) = P(x_1| w_i)\ P(x_2| w_i). . . .P(x_n| w_i)$

Let $p_i = P(x_i = 1 \mid w_1)$

$q_i = P(x_i = 1 \mid w_2)$

where,

p_i is the probability that if the document is relevant then the i^{th} index term will be present

q_i is the probability that if the document is non-relevant then the i^{th} index term will be present

Therefore,

$$P(x|w_1) = \prod_{i=1}^{n} p_i^{x_i}(1 - p_i)^{1-x_i}$$

$$P(x|w_2) = \prod_{i=1}^{n} q_i^{x_i}(1 - q_i)^{1-x_i}$$

Ranking score of a document x, $score_x = \log\left(\frac{\text{Probability of Relevance}}{\text{Probablity of Non-relevance}}\right)$

$$= \log\left(\frac{P(x|w_1)}{P(x|w_2)}\right)$$

Substituting values we get, $score_x = \sum_{i=1}^{n}\left(x_i . \log\frac{p_i}{q_i} + (1 - x_i). \log\frac{(1-p_i)}{(1-q_i)}\right)$

This $score_x$ is used to rank the document.

2.4.3 Alternative models: These models are enhancements of classical models by making use of specific techniques from other fields. Cluster model, Fuzzy model, Latent semantic indexing (LSI) model are examples of alternative IR models.

2.4.4 Comparison of Vector Space Model (VSM) and Probabilistic Model (PM)

In Vector Space Model and Probabilistic Model retrieval is based on partial matching. Each method has its own advantages.

Table 2.1 Comparison of Vector Space Model and Probabilistic Model

VECTOR SPACE MODEL	PROBABILISTIC MODEL
• Terms are weighted by importance • It estimates the degree of similarity of the document with regard to the query as the correlation between two vectors	• Way to quantify uncertainty of information retrieval • It tries to estimate the probability that the user will find the document interesting

2.5 Retrieval Evaluation

The measures require a collection of documents and a query. All common measures described here assume a ground truth notion of relevancy: every document is known to be either relevant or non-relevant to a particular query [11]. The two major measures commonly associated with information systems are precision and recall. When a user decides to issue a search looking for information on a topic, the total database is logically divided into four segments shown in Figure 2.3. Relevant items are those documents that contain information that helps the searcher in answering his question. Non-relevant items are those items that do not provide any directly useful information. There are two possibilities with respect to each item: it can be retrieved or not retrieved by the user's query.

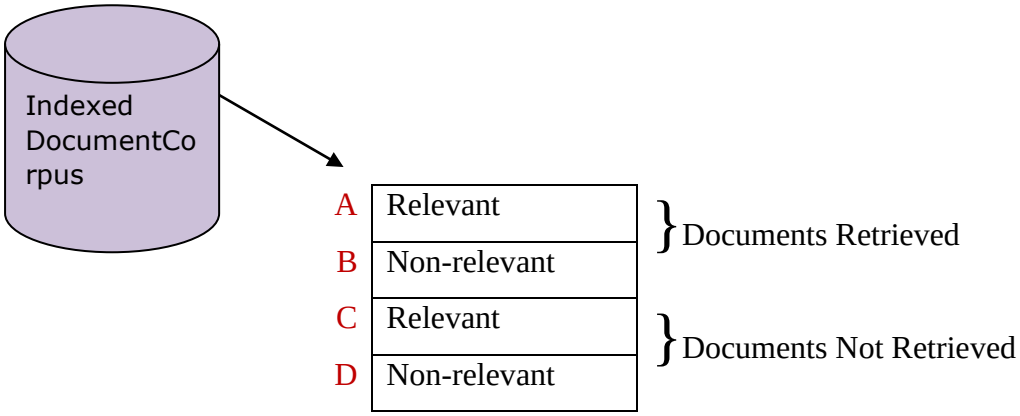


Figure 2.3 Effects of Search on Total Document Space

2.5.1 Basic Measures

2.5.1.1 Precision

Precision (P) is the fraction of the documents retrieved that are relevant to the user's information need.

$$\text{Precision} = \text{P} = \frac{\text{Number\_Relevant\_Retrieved}}{\text{Number\_Total\_Retrieved}} = \frac{\text{A}}{\text{A+B}}$$

where,

Number_Relevant_Retrieved is the number of documents retrieved that are relevant to the user's search need.

Number_Total_Retrieved is the total number of documents retrieved from the corpus.

In binary classification, precision is analogous to *positive predictive value.* Precision takes all retrieved documents into account. It can also be evaluated at a given cut-off rank 'n', considering only the topmost 'n' results returned by the system. This measure is called precision at n or *P@n*.

Precision measures one aspect of information retrieval overhead for a user associated with a particular search. If a search has 85 per cent precision, then 15 per cent of the user effort is overhead reviewing non-relevant items.

2.5.1.2 Recall

Recall (R) is the fraction of the documents that are relevant to the query that are successfully retrieved.

$$\text{Recall} = \text{R} = \frac{\text{Number\_Relevant\_Retrieved}}{\text{Number\_Relevant}} = \frac{\text{A}}{\text{A+C}}$$

where,

Number_Relevant_Retrieved is the number of documents retrieved that are relevant to the user's search need.

Number_Relevant is the number of relevant items in the document corpus.

In binary classification, recall is called *sensitivity*. So it can be looked at as the probability that a relevant document is retrieved by the query. Recall gauges how well a system processing a particular query is able to retrieve the relevant items that the user is interested in seeing. Recall is a very useful concept, but due to the denominator, is non-calculable in operational systems. If the system knew the total set of relevant items in the database, it would have retrieved them.

2.5.2 Alternative Measures

2.5.2.1 F-measure

F-measure is the weighted harmonic mean of precision (P) and recall (R). P and R are competing constraints which may be represented as resistances in parallel in an electronic circuit (see figure 2.4).

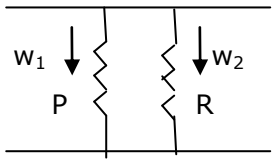


Figure 2.4 F-measure is the parallel sum of P and R

Cumulative effect of these constraints will be F, that is:

$$\frac{1}{\text{F}} = \left(\frac{\text{w}_1}{\text{P}} + \frac{\text{w}_2}{\text{R}}\right)$$

where, w_1 and w_2 are the weights given to P & R respectively.

if $w_1 = w_2 = \frac{1}{2}$ then,

$$\frac{1}{F} = \frac{1}{2}\left(\frac{1}{P} + \frac{1}{R}\right)$$

$$\Rightarrow \quad F = \frac{2\times P\times R}{P+R} = \frac{2\times\frac{A}{A+B}\times\frac{A}{A+C}}{\frac{A}{A+B}+\frac{A}{A+C}}$$

$$= \frac{2A^2}{2A^2+AC+AB}$$

This is also known as the F_1 measure, because recall and precision are evenly weighted.

In general, F_β measures the effectiveness of retrieval with respect to a user who attaches β times as much importance to recall as precision where β is any non-negative real value.

$$F_\beta = \frac{(1+\beta)\times(\text{Precision}\times\text{Recall})}{(\beta\times\text{Precision}+\text{Recall})}$$

Two other commonly used F measures are the $F_{0.5}$ measure, which weights precision twice as much as recall, and the F_2 measure, which weights recall twice as much as precision.

2.5.2.2 Mean average precision

The precision and recall are based on the whole list of documents returned by the system. Average precision emphasizes returning more relevant documents earlier. It is average of precisions computed after truncating the list after each of the relevant document in turn.

$$\text{MAP} = \frac{\sum_{r=1}^{N}(\text{Precision}(r)\times\text{rel}(r))}{\text{Number\_Relevant}} = \frac{\sum_{r=1}^{N}(\text{Precision}(r)\times\text{rel}(r))}{A+C}$$

where,

r is the rank,

N is the number of documents retrieved,

rel () is a binary function on the relevance of a given rank,

Number_Relevant is the number of relevant items in the document corpus, and

Precision () is the precision at a given cut-off rank.

2.5.2.3 NDCG (Normalized Discounted Cumulative Gain)

NDCG is a cumulative, multilevel measure of ranking quality that is usually truncated at a particular rank level. For a given query q_i the NDCG is computed as:

$$N_i \equiv N_i \sum_{j=1}^{L} \frac{(2^{r(j)}-1)}{\log(1+j)}$$

where r(j) is the relevance level of the j^{th} document , and where the normalization constant N_i is chosen so that a perfect ordering would result in N_i =1. Here L is the ranking truncation level at which the NDCG is computed. The N_i is then averaged over the query set. NDCG is particularly well suited to Web Search applications because it is multilevel and because the truncation level can be chosen to reflect how many documents are shown to the user. The Discounted Cumulative Gain (DCG) has been widely used to access relevance in the context of information retrieval because it can handle multiple relevance grades such as {perfect, excellent, good, fair, bad}.

Summary

This chapter begins with the definition of Information Retrieval. Section 2.2 and section 2.3 covered various objectives and components of Information Retrieval System respectively. Basic ranking models are discussed in section 2.4. Comparison of Vector Space Model and Probabilistic Model has been given in subsection 2.4.4. Various evaluation measures associated with Information Retrieval System are discussed in section 2.5.