Class: BEIT **Sem:** VII **Subject:** IRS

Faculty Name: Jayshree Jha **Academic Year:** 2022-23

Module 2: IR Models

- **Modelling: Taxonomy of Information Retrieval Models**
- **Retrieval: Formal Characteristics of IR models**
- **Classic Information Retrieval**
- **Alternative Set Theoretic Models**
- **Probabilistic Models**
- **Structured text retrieval Models**
- **Models for Browsing**

2.1 Modelling: Introduction

- Traditional information retrieval systems usually adopt index terms to index and retrieve documents. In a restricted sense, an index term is a keyword (or group of related words) which has some meaning of its own (i.e., which usually has the semantics of a noun).
- In its more general form, an index term is simply any word which appears in the text of a document in the collection. Retrieval based on index terms is simple but raises key questions regarding the information retrieval task.
- For instance, retrieval using index terms adopts as a fundamental foundation of the idea that the semantics of the documents and of the user information need can be naturally expressed through sets of index terms. Clearly, this is a considerable oversimplification of the problem because a lot of the semantics in a document or user request is lost when we replace its text with a set of words.
- Furthermore, matching between each document and the user request is attempted in this very imprecise space of index terms. Thus, it is no surprise that the documents retrieved in response to a user request expressed as a set of keywords are frequently irrelevant.
- If one also considers that most users have no training in properly forming their queries, the problem is worsened with potentially disastrous results. The frequent dissatisfaction of Web users with the answers they normally obtain is just one good example of this fact.
- Clearly, one central problem regarding information retrieval systems is the issue of predicting which documents are relevant and which are not. Such a decision is usually dependent on a **ranking algorithm** which attempts to establish a simple ordering of the documents retrieved.
- Thus, ranking algorithms are at the core of information retrieval systems. A ranking algorithm operates according to basic premises regarding the notion of document relevance. Distinct sets of premises (regarding document relevance) yield distinct information retrieval models. The IR model adopted determines the predictions of what is relevant and what is not (i.e., the notion of relevance implemented by the system).

2.2 Taxonomy of Information Retrieval Models

- The three classic models in information retrieval are called Boolean, vector, and probabilistic. In the Boolean model, documents and queries are represented as a set of index term. These models are called set theoretic.
- In the vector model, documents and queries are represented as vectors in a t-dimensional space. Thus, we say that the model is algebraic.
- In the probabilistic model, the framework for modelling document and query representation is based on probability theory. Thus, we say that the model is probabilistic.
- Over the years, alternative modelling paradigms for each type of classic model (i.e., set theoretic, algebraic, and probabilistic) have been proposed.
- Regarding alternative set theoretic models, we distinguish the fuzzy and the extended Boolean models. Regarding alternative algebraic models, we distinguish the generalized vector, the latent semantic indexing and the neural network models.
- Regarding alternative probabilistic models, we distinguish the inference network and the belief network models. Below mentioned Figure(figure2.1) illustrates a taxonomy of these information retrieval models.

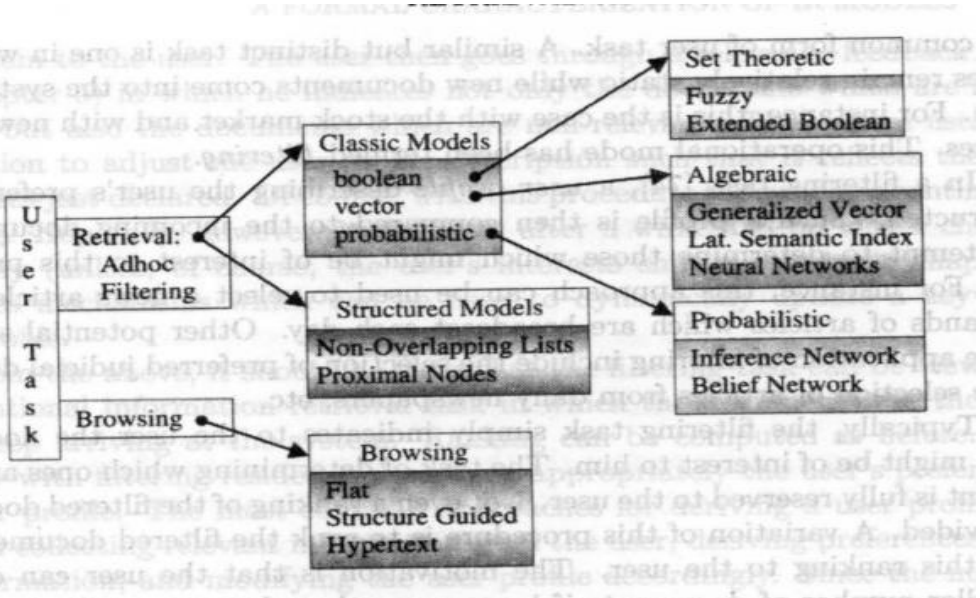


Figure 2.1 A taxonomy of information retrieval models.

2.3 Retrieval: Ad hoc and Filtering

- In a conventional information retrieval system, the documents in the collection remain relatively static while new queries are submitted to the system. This operational mode has been termed ad hoc retrieval in recent years and is the most common form of user task. A similar but distinct task is one in which the queries remain relatively static while new documents come into the system (and leave). For instance, this is the case with the stock market and with news wiring services. This operational mode has been termed filtering.
- In a filtering task, a user profile describing the user's preferences is constructed. Such a profile is then compared to the incoming documents in an attempt to determine those which might be of interest to this particular user. For instance, this approach can be used to select a news article among thousands of articles which are broadcast each day.
- Even if no ranking is presented to the user, the filtering task can compute an internal ranking to determine potentially relevant documents. For instance, documents with a ranking above a given threshold could be selected; the others would be discarded.
- From the above, it should be clear that the filtering task can be viewed as a conventional information retrieval task in which the documents are the ones which keep arriving at the system. Ranking can be computed as before. The difficulty with filtering resides in describing appropriately the user's preferences in a user profile. The most common approaches for deriving a user profile are based on collecting relevant information from the user, deriving preferences from this information, and modifying the user profile accordingly

2.4 Formal Characteristics of IR Models

Definition: An information retrieval model is a quadruple [D, Q, F, R(qi,dj)] where
(1) D is a set composed of logical views (or representations) for the documents in the collection.
(2) Q is a set composed of logical views (or representations) for the user information needs. Such representations are called queries.
(3) F is a framework for modelling document representations, queries, and their relationships.
(4) R(qi,dj) is a ranking function which associates a real number with a query qi € Q and a document representation dj € D. Such ranking defines an ordering among the documents with regard to the query qi.

- To build a model, we think first of representations for the documents and for the user information need. Given these representations, we then conceive the framework in which they

can be modelled. This framework should also provide the intuition for constructing a ranking function.

- For instance, for the classic Boolean model, the framework is composed of sets of documents and the standard operations on sets.
- For the classic vector model, the framework is composed of a t-dimensional vectorial space and standard linear algebra operations on vectors.
- For the classic probabilistic model, the framework is composed of sets, standard probability operations, and the Bayes' theorem.

2.5 Classic Information Retrieval

- The classic models in information retrieval consider that each document is described by a set of representative keywords called index terms. An index term is simply a (document) word whose semantics helps in remembering the document's main themes. Thus, index terms are used to index and summarize the document contents.
- In general, index terms are mainly nouns because nouns have meaning by themselves and thus, their semantics is easier to identify and to grasp. Adjectives, adverbs, and connectives are less useful as index terms because they work mainly as complements.
- Given a set of index terms for a document, we notice that not all terms are equally useful for describing the document contents. In fact, there are index terms which are simply vaguer than others.
- Deciding on the importance of a term for summarizing the contents of a document is not a trivial issue. Despite this difficulty, there are properties of an index term which are easily measured and which are useful for evaluating the potential of a term as such.
- For instance, consider a collection with a hundred thousand documents. A word which appears in each of the one hundred thousand documents is completely useless as an index term because it does not tell us anything about which documents the user might be interested in.
- On the other hand, a word which appears in just five documents is quite useful because it narrows down considerably the space of documents which might be of interest to the user. Thus, it should be clear that distinct index terms have varying relevance when used to describe document contents.
- This effect is captured through the assignment of numerical weights to each index term of a document.

- Let k ; be a n index term, dj be a document, and wij > 0 be a weight associated with the pair (ki,dj). This weight quantifies the importance of the index term for describing the document semantic contents.

Definition *Let t be the number of index terms in the system and k_i be a generic index term. $K = \{k_1, \ldots, k_t\}$ is the set of all index terms. A weight $w_{i,j} > 0$ is associated with each index term k_i of a document d_j. For an index term which does not appear in the document text, $w_{i,j} = 0$. With the document d_j is associated an index term vector $\vec{d_j}$ represented by $\vec{d_j} = (w_{1,j}, w_{2,j}, \ldots, w_{t,j})$. Further, let g_i be a function that returns the weight associated with the index term k_i in any t-dimensional vector (i.e., $g_i(\vec{d_j}) = w_{i,j}$).*

- As we later discuss, the index term weights are usually assumed to be mutually independent. This means that knowing the weight wi,j associated with the pair (ki+1,dj) tells us nothing about the weight wi+1,j associated with the pair(ki+1,dj).
- This is clearly a simplification because occurrences of index terms in a document are not uncorrelated. Consider, for instance, that the terms computer and network are used to index a given document which covers the area of computer networks. Frequently, in this document, the appearance of one of these two words attracts the appearance of the other. Thus, these two words are correlated and their weights could reflect this correlation.
- While mutual independence seems to be a strong simplification, it does simplify the task of computing index term weights and allows for fast ranking computation. Furthermore, taking advantage of index term correlations for improving the final document ranking is not a simple task.
- In fact, none of the many approaches proposed in the past are advantageous (for has clearly demonstrated that index term correlations ranking purposes) with general collections. Therefore, unless clearly stated otherwise, we assume mutual independence among index terms.

2.5.1 Boolean Model

- The Boolean model is a simple retrieval model based on set theory and Boolean algebra. Since the concept of a set is quite intuitive, the Boolean model provides a framework which is easy to grasp by a common user of an IR system.
- Furthermore, the queries are specified as Boolean expressions which have precise semantics. Given its inherent simplicity and neat formalism, the Boolean model received great attention in past years and was adopted by many of the early commercial bibliographic systems.

Unfortunately, the Boolean model suffers from major drawbacks.

- First, it's retrieval strategy is based on a binary decision criterion (i.e., a document is predicted to be either relevant or non-relevant) without any notion of a grading scale, which prevents good retrieval performance. Thus, the Boolean model is in reality much more a data (instead of information) retrieval model.
- Second, while Boolean expressions have precise semantics, frequently it is not simple to translate an information need into a Boolean expression. In fact, most users find it difficult and awkward to express their query requests in terms of Boolean expressions. The Boolean expressions actually formulated by users often are quite simple.
- Despite these drawbacks, the Boolean model is still the dominant model with commercial document database systems and provides a good starting point for those new to the field.
- The Boolean model considers that index terms are present or absent in a document. As a result, the index term weights are assumed to be all binary, i.e., wi,j €{0,1}. A query q is composed of index terms linked by three connectives: not, and, or. Thus, a query is essentially a conventional Boolean expression which can be represented as a disjunction of conjunctive vectors (i.e., in disjunctive normal form — DNF).

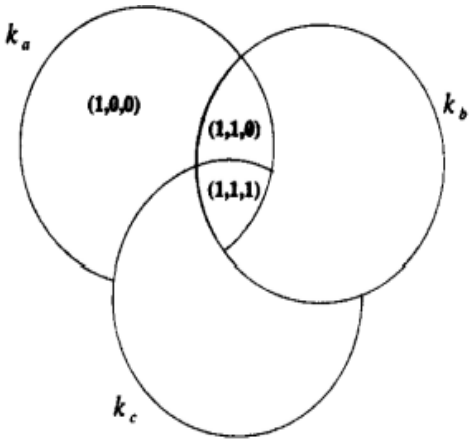


Figure 2.3 The three conjunctive components for the query $[q = k_a \wedge (k_b \vee \neg k_c)]$.

For instance, the query $[q = k_a \wedge (k_b \vee \neg k_c)]$ can be written in disjunctive normal form as $[\vec{q}_{dnf} = (1,1,1) \vee (1,1,0) \vee (1,0,0)]$, where each of the components is a binary weighted vector associated with the tuple (k_a, k_b, k_c). These binary weighted vectors are called the conjunctive components of $\vec{q}_{dnf}$.

Definition *For the Boolean model, the index term weight variables are all binary i.e., $w_{i,j} \in \{0,1\}$. A query q is a conventional Boolean expression. Let $\vec{q}_{dnf}$ be the disjunctive normal form for the query q. Further, let $\vec{q}_{cc}$ be any of the conjunctive components of $\vec{q}_{dnf}$. The similarity of a document d_j to the query q is defined as*

$$sim(d_j, q) = \begin{cases} 1 & \text{if } \exists \vec{q}_{cc} \mid (\vec{q}_{cc} \in \vec{q}_{dnf}) \wedge (\forall_{k_i},\ g_i(\vec{d}_j) = g_i(\vec{q}_{cc})) \\ 0 & \text{otherwise} \end{cases}$$

If $sim(d_j, q) = 1$ then the Boolean model predicts that the document d_j is relevant to the query q (it might not be). Otherwise, the prediction is that the document is not relevant.

- The Boolean model predicts that each document is either relevant or nonrelevant. There is no notion of a partial match to the query conditions. For instance, let d_j be a document for which $\vec{d}_j = (0,1,0)$. Document d_j includes the index term k_b, but is considered non-relevant to the query $[q = k_a \wedge (k_b \vee \neg k_c)]$.

- The main advantages of the Boolean model are the clean formalism behind the model and its simplicity. The main disadvantages are that exact matching may lead to retrieval of too few or too many documents

2.5.2 Vector Model

- The vector model recognizes that the use of binary weights is too limiting and proposes a framework in which partial matching is possible.
- This is accomplished by assigning non-binary weights to index terms in queries and in documents. These term weights are ultimately used to compute the degree of similarity between each document stored in the system and the user query.
- By sorting the retrieved documents in decreasing order of this degree of similarity, the vector model takes into consideration documents which match the query terms only partially. The main resultant effect is that the ranked document answer set is a lot more precise (in the sense that it better matches the user information need) than the document answer set retrieved by the Boolean model.

Definition *For the vector model, the weight $w_{i,j}$ associated with a pair (k_i, d_j) is positive and non-binary. Further, the index terms in the query are also weighted. Let $w_{i,q}$ be the weight associated with the pair $[k_i, q]$, where $w_{i,q} \geq 0$. Then, the query vector $\vec{q}$ is defined as $\vec{q} = (w_{1,q}, w_{2,q}, \ldots, w_{t,q})$ where t is the total number of index terms in the system. As before, the vector for a document d_j is represented by $\vec{d_j} = (w_{1,j}, w_{2,j}, \ldots, w_{t,j})$.*

Therefore, a document d_j and a user query q are represented as t-dimensional vector s as shown in Figure. The vector model proposes to evaluate the degree of similarity of the document d_j with regard to the query g as the correlation between the vectors $\vec{d_j}$ and $\vec{q}$. This correlation can be quantified, for instance, **by the cosine of the angle between these two vectors**. That is,

$$sim(d_j, q) = \frac{\vec{d_j} \bullet \vec{q}}{|\vec{d_j}| \times |\vec{q}|} = \frac{\sum_{i=1}^{t} w_{i,j} \times w_{i,q}}{\sqrt{\sum_{i=1}^{t} w_{i,j}^2} \times \sqrt{\sum_{j=1}^{t} w_{i,q}^2}}$$

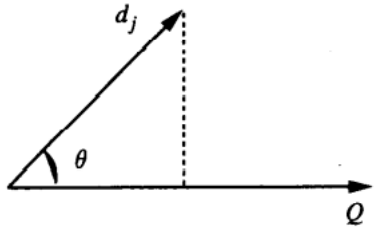


Figure 2.4 The cosine of θ is adopted as $sim(d_j, q)$.

where $|\vec{d_j}|$ and $|\vec{q}|$ are the norms of the document and query vectors. The factor $|\vec{q}|$ does not affect the ranking (i.e., the ordering of the documents) because it is the same for all documents. The factor $|\vec{d_j}|$ provides a normalization in the space of the documents. Since $w_{i,j} > 0$ and $w_{i,q} > 0$, sim(q, d;) varies from 0 to +1.

- Thus, instead of attempting to predict whether a document is relevant or not, the vector model ranks the documents according to their degree of similarity to the query. A document might be retrieved even if it matches the query only partially.

- For instance, one can establish a threshold on sim(d_j,q) and retrieve the documents with a degree of similarity above that threshold. But to compute rankings we need first to specify how For instance, one can establish a threshold on sim(d_j,q) and retrieve the documents with a degree of similarity above that threshold. But to compute rankings we need first to specify how index term weights are obtained.
- Let us think of the documents as a collection C of objects and think of the user query as a (vague) specification of a set A of objects. In this scenario, the IR problem can be reduced to the problem of determining which documents are in the set A and which ones are not (i.e., the IR problem can be viewed as a clustering problem).

- In a clustering problem, two main issues have to be resolved.
  - ✓ First, one needs to determine what are the features which better describe the objects in the set A.
  - ✓ Second, one needs to determine what are the features which better distinguish the objects in the set A from the remaining objects in the collection C.

- The first set of features provides for quantification of intra-cluster similarity, while the second set of features provides for quantification of inter-cluster dissimilarity. The most successful clustering algorithms try to balance these two effects.
- In the vector model, intra-clustering similarity is quantified by measuring the raw frequency of a term k_i inside a document d_j. Such term frequency is usually referred to as the t_f factor and provides one measure of how well that term describes the document contents (i.e., intra-document characterization).
- Furthermore, inter-cluster dissimilarity is quantified by measuring the inverse of the frequency of a term k; among the documents in the collection. This factor is usually referred to as the inverse document frequency or the idf factor.
- The motivation for usage of an idf factor is that terms which appear in many documents are not very useful for distinguishing a relevant document from a non-relevant one. As with good clustering algorithms, the most effective term-weighting schemes for IR try to balance these two effects.

Definition *Let N be the total number of documents in the system and n_i be the number of documents in which the index term k_i appears. Let $freq_{i,j}$ be the raw frequency of term k_i in the document d_j (i.e., the number of times the term k_i is mentioned in the text of the document d_j). Then, the normalized frequency $f_{i,j}$ of term k_i in document d_j is given by*

$$f_{i,j} = \frac{freq_{i,j}}{max_l \ freq_{l,j}} \tag{2.1}$$

where the maximum is computed over all terms which are mentioned in the text of the document d_j. If the term k_i does not appear in the document d_j then $f_{i,j} = 0$. Further, let idf_i, inverse document frequency for k_i, be given by

$$idf_i = \log \frac{N}{n_i} \tag{2.2}$$

The best known term-weighting schemes use weights which are given by

$$w_{i,j} = f_{i,j} \times \log \frac{N}{n_i} \tag{2.3}$$

The main advantages of the vector model are:

- Its term-weighting scheme improves retrieval performance.
- Its partial matching strategy allows retrieval of documents that approximate the query conditions.
- Its cosine ranking formula sorts the documents according to their degree of similarity to the query.

Disadvantage

- Theoretically, the vector model has the disadvantage that index terms are assumed to be mutually independent (equation 2.3 does not account for index term dependencies). However, in practice, consideration of term dependencies might be a disadvantage. Due to the locality of many term dependencies, their indiscriminate application to all the documents in the collection might in fact hurt the overall performance.
- Despite its simplicity, the vector model is a resilient ranking strategy with general collections. It yields ranked answer sets which are difficult to improve upon without query expansion or relevance feedback within the framework of the vector model.

A large variety of alternative ranking methods have been compared to the vector model but the consensus seems to be that, in general, the vector model is either superior or almost as good as the known alternatives. Furthermore, it is simple and fast. For these reasons, the vector model is a popular retrieval model nowadays.

2.5.3 Probabilistic Model

In this section, we describe the classic probabilistic model introduced in 1976 by Roberston and Sparck Jones which later became known as the binary independence retrieval (BIR) model.

- The probabilistic model attempts to capture the IR problem within a probabilistic framework. The fundamental idea is as follows. Given a user query, there is a set of documents which contains exactly the relevant documents and no other.
- Let us refer to this set of documents as the ideal answer set. Given the description of this ideal answer set, we would have no problems in retrieving its documents. Thus, we can think of the querying process as a process of specifying the properties of an ideal answer set (which is analogous to interpreting the IR problem as a problem of clustering).
- The problem is that we do not know exactly what these properties are. All we know is that there are index terms whose semantics should be used to characterize these properties. Since these properties are not known at query time, an effort has to be made at initially guessing what they could be.
- This initial guess allows us to generate a preliminary probabilistic description of the ideal answer set which is used to retrieve a first set of documents. An interaction with the user is then initiated with the purpose of improving the probabilistic description of the ideal answer set.
- Such interaction could proceed as follows. The user takes a look at the retrieved documents and decides which ones are relevant and which ones are not (in truth, only the first top documents need to be examined). The system then uses this information to refine the description of the ideal answer set. By repeating this process many times, it is expected that such a description will evolve and become closer to the real description of the ideal answer set.
- Thus, one should always have in mind the need to guess at the beginning the description of the ideal answer set. Furthermore, a conscious effort is made to model this description in probabilistic terms. The probabilistic model is based on the following fundamental assumption.

Class: BEIT **Sem:** VII **Subject:** IRS

Faculty Name: Jayshree Jha **Academic Year:** 2022-23

Assumption (Probabilistic Principle) Given a user query q and a document d_j in the collection, the probabilistic model tries to estimate the probability that the user will find the document d_j interesting (i.e., relevant). The model assumes that this probability of relevance depends on the query and the document representations only. Further, the model assumes that there is a subset of all documents which the user prefers as the answer set for the query q. Such an ideal answer set is labelled R and should maximize the overall probability of relevance to the user. Documents in the set R are predicted to be relevant to the query. Documents not in this set are predicted to be non-relevant.

- This assumption is quite troublesome because it does not state explicitly how to compute the probabilities of relevance. In fact, not even the sample space which is to be used for defining such probabilities is give.
- Given a query q, the probabilistic model assigns to each document dj, as a measure of its similarity to the query, the ratio P(dj relevant-to q)/P(dj non-relevant-to q) which computes the odds of the document dj being relevant to the query q. Taking the odds of relevance as the rank minimizes the probability of an erroneous judgement.

Definition *For the probabilistic model, the index term weight variables are all binary i.e.,* $w_{i,j} \in \{0,1\}$, $w_{i,q} \in \{0,1\}$. *A query q is a subset of index terms. Let R be the set of documents known (or initially guessed) to be relevant. Let* $\overline{R}$ *be the complement of R (i.e., the set of non-relevant documents). Let* $P(R|\vec{d}_j)$ *be the probability that the document* d_j *is relevant to the query q and* $P(\overline{R}|\vec{d}_j)$ *be the probability that* d_j *is non-relevant to q. The similarity* $sim(d_j, q)$ *of the document* d_j *to the query q is defined as the ratio*

$$sim(d_j, q) = \frac{P(R|\vec{d}_j)}{P(\overline{R}|\vec{d}_j)}$$

Using Bayes' rule,

$$sim(d_j, q) = \frac{P(\vec{d}_j|R) \times P(R)}{P(\vec{d}_j|\overline{R}) \times P(\overline{R})}$$

$P(\vec{d}_j|R)$ stands for the probability of randomly selecting the document d_j from the set R of relevant documents. Further, $P(R)$ stands for the probability that a document randomly selected from the entire collection is relevant. The meanings attached to $P(\vec{d}_j|\overline{R})$ and $P(\overline{R})$ are analogous and complementary.

Since $P(R)$ and $P(\overline{R})$ are the same for all the documents in the collection, we write,

$$sim(d_j, q) \sim \frac{P(\vec{d_j}|R)}{P(\vec{d_j}|\overline{R})}$$

Assuming independence of index terms,

$$sim(d_j, q) \sim \frac{(\prod_{g_i(\vec{d_j})=1} P(k_i|R)) \times (\prod_{g_i(\vec{d_j})=0} P(\overline{k}_i|R))}{(\prod_{g_i(\vec{d_j})=1} P(k_i|\overline{R})) \times (\prod_{g_i(\vec{d_j})=0} P(\overline{k}_i|\overline{R}))}$$

$P(k_i|R)$ stands for the probability that the index term k_i is present in a document randomly selected from the set R. $P(\overline{k}_i|R)$ stands for the probability that the index term k_i is not present in a document randomly selected from the set R. The probabilities associated with the set $\overline{R}$ have meanings which are analogous to the ones just described.

Taking logarithms, recalling that $P(k_i|R) + P(\overline{k}_i|R) = 1$, and ignoring factors which are constant for all documents in the context of the same query, we can finally write

$$sim(d_j, q) \sim \sum_{i=1}^{t} w_{i,q} \times w_{i,j} \times \left(\log \frac{P(k_i|R)}{1 - P(k_i|R)} + \log \frac{1 - P(k_i|\overline{R})}{P(k_i|\overline{R})} \right)$$

which is a key expression for ranking computation in the probabilistic model.

The main advantage of the probabilistic model, in theory, is that documents are ranked in decreasing order of their probability of being relevant.

The disadvantages include:

(1) the need to guess the initial separation of documents into relevant and non-relevant sets;

(2) the fact that the method does not take into account the frequency with which an index term occurs inside a document (i.e., all weights are binary); and

(3) the adoption of the independence assumption for index terms. However, as discussed for the vector model, it is not clear that in dependence of index terms is a bad assumption in practical situations.

2.6 Alternative Set Theoretic Models

In this section, we discuss two alternative set theoretic models, namely the fuzzy set model and the extended Boolean model.

2.6.1 Fuzzy Set Model

Representing documents and queries through sets of keywords yields descriptions which are only partially related to the real semantic contents of the respective documents and queries. As a result, the matching of a document to the query terms is approximate (or vague). This can be modelled by considering that each query term defines a fuzzy set and that each document has a degree of membership (usually smaller than 1) in this set. This interpretation of the retrieval process (in terms of concepts from fuzzy theory) is the basic foundation of the various fuzzy set models for information retrieval which have been proposed over the years.

Fuzzy Set Theory

Fuzzy set theory deals with the representation of classes whose boundaries are not well defined. The key idea is to associate a membership function with the elements of the class. This function takes values in the interval [0,1] with 0 corresponding to no membership in the class and 1 corresponding to full membership. Membership values between 0 and 1 indicate marginal elements of the class. Thus, membership in a fuzzy set is a notion intrinsically gradual instead of abrupt (as in conventional Boolean logic).

The three most commonly used operations on fuzzy sets are: the complement of a fuzzy set, the union of two or more fuzzy sets, and the intersection of two or more fuzzy sets. Fuzzy sets are useful for representing vagueness and imprecision and have been applied to various domains.

Fuzzy Information Retrieval

- The basic idea is to **expand the set of index terms in the query q with related terms** (obtained from the thesaurus) such that additional relevant documents (i.e., besides the ones which would be normally retrieved) can be retrieved by the user query.
- The model uses a term-term correlation matrix to compute correlations between a document d; and its fuzzy index terms. Further, the model adopts algebraic sums and products (instead of max and min) to compute the overall degree of membership of a document d_j in the fuzzy set defined by the user query.

- Fuzzy retrieval provides the capability to locate spellings of words that are similar to the entered search term. This function is primarily used to compensate for errors in spelling of words. **Fuzzy retrieval increases recall at the expense of decreasing precision.**
- In the process of expanding a query term, fuzzy retrieval includes other terms that have similar spellings; giving more weight to words in the database that have **similar word lengths and position of the characters as the entered term**.
- A fuzzy search on the term 'computer' would automatically include the following words from the information database: 'computer', 'compiter', 'computter', 'compute'. An additional enhancement may look up the proposed alternative spelling and if it is a valid word with a different meaning, include it in the search with a low ranking or not include it at all (e.g., 'commuter'). Systems allow the specification of the maximum number of few terms that the expansion includes in the query.

2.6.2 Extended Boolean Model

- Boolean retrieval is simple and elegant. However, since there is no provision for term weighting, no ranking of the answer set is generated. As a result, the **size of the output might be too large or too small.** Because of these problems, modern information retrieval systems are no longer based on the Boolean model.
- In fact, most of the new systems adopt at their core some form of vector retrieval. The reasons are that the vector space model is simple, fast, and yields better retrieval performance. One alternative approach though is to extend the Boolean model with the functionality of partial matching and term weighting.
- This strategy allows one to combine Boolean query formulations with characteristics of the vector model. In what follows, we discuss one of the various models which are based on the idea of extending the Boolean model with features of the vector model.
- Weighting of index terms is not common in manual indexing systems. Weighting is the process of assigning an importance to an index term's use in an item.
- The weight should represent the degree to which the concept associated with the index term is represented in the item. The weight should help in discriminating the extent to which the concept is described in items of the database.
- The manual process of assigning weights adds additional overhead on the indexer and requires a more complex data structure to store the weights. In a weighted indexing system, an attempt is made to place a value on the index term's representation of its associated concept in the document.

- An index term's weight is based upon a function associated with the frequency of occurrence of the term in the item. Typically, values for the index terms are normalised between zero and one.
- The higher the weight, the more the term represents a concept discussed in the item. The weight can be adjusted to account for other information such as the number of items in the database that contain the same concept.
- The query process uses the weights along with any weights. assigned to terms in the query to determine a scalar value (rank value) used in predicting the likelihood that an item satisfies the query.
- The results are presented to the user in order of the rank value from highest number to lowest number. If weights are assigned to the terms between the values 0'.0 to 1.0', they may be interpreted as the significance that users are placing on each term.

2.6.2 Structured Text retrieval Model

- Consider a user with a superior visual memory. Such a user might then recall that the specific document he is interested in contains a page in which the string ‘atomic holocaust’ appears in italic in the text surrounding a Figure whose label contains the word ‘earth.”
- With a classic information retrieval model, this query could be expressed as [‘atomic holocaust’ and ‘earth’] which retrieves all the documents containing both strings. Clearly, however, this answer contains many more documents than desired by this user. In this particular case, the user would like to express his query through a richer expression such as same-page (near (‘atomic holocaust,’ Figure (label (‘earth’)))) which conveys the details in his visual recollection.
- Further, the user might be interested in an advanced interface which simplifies the task of specifying this (now complex) query. This example illustrates the appeal of a query language which allows us to combine the specification of strings (or patterns) with the specification of structural components of the document.
- Retrieval models which combine information on text content with information on the document structure are called structured test retrieval models.

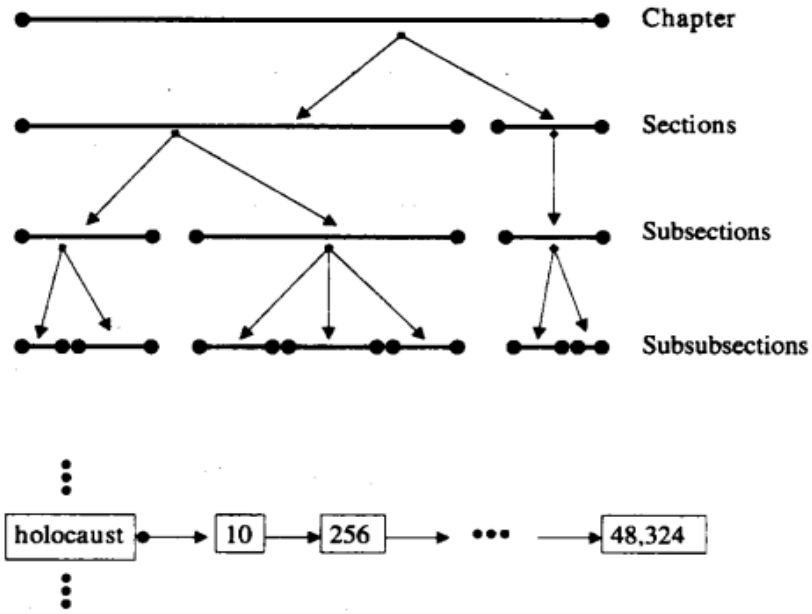
Types of Structured Text retrieval Model

2.6.2.1 Model Based on Non-Overlapping Lists

- Burkowski proposes to divide the whole text of each document in nonoverlapping text regions which are collected in a list. Since there are multiple ways to divide a text in non-overlapping regions, multiple lists are generated.
- For instance, we might have a list of all chapters in the document, a second list of all sections in the document, and a third list of all subsections in the document. These lists are kept as separate and distinct data structures. While the text regions in the same (flat) list have no overlapping, text regions from distinct lists might overlap.
- To allow searching for index terms and for text regions, a single inverted file is built in which each structural component stands as an entry in the index. Associated with each entry, there is a list of text regions as a list of occurrences.
- Moreover, such a list could be easily merged with the traditional inverted file for the words in the text. Since the text regions are non-overlapping, the types of queries which can be asked are simple:
  (a) select a region which contains a given word (and does not contain other regions);
  (b) select a region A which does not contain any other region (where B belongs to a list distinct from the list for A);
  (c) select a region not contained within any other region, etc.

2.6.2.2 Model Based on Proximal Nodes

- Navarro and Baeza-Yates proposed this model which allows the definition of independent hierarchical (non-flat) indexing structures over the same document text. Each of these indexing structures is a strict hierarchy composed of chapters, sections, paragraphs, pages, and lines which are called nodes.
- To each of these nodes is associated a text region. Further, two distinct hierarchies might refer to overlapping text regions. Given a user query which refers to distinct hierarchies, the compiled answer is formed by nodes which all come from only one of them.
- Thus, an answer cannot be composed of nodes which come from two distinct hierarchies (which allows for faster query processing at the expense of less expressiveness).
- Notice, however, that due to the hierarchical structure, nested text regions (coming from the same hierarchy) are allowed in the answer set.



- Figure illustrates a hierarchical indexing structure composed of four levels (corresponding to a chapter, sections, subsections, and subsubsections of the same document) and an inverted list for the word 'holocaust.'
- The entries in this inverted list indicate the positions in the text of the document in which the word 'holocaust' occurs. In the hierarchy, each node indicates the position in the text of its associated structural component (chapter, section, subsection, or subsubsection).
- The query language allows the specification of regular expressions (to search for strings), the reference to structural components by name (to search for chapters, for instance), and a combination of these. In this sense, the model can be viewed as a compromise between expressiveness and efficiency.
- The somewhat limited expressiveness of the query language allows efficient query processing by first searching for the components which match the strings specified in the query and, subsequently, evaluating which of these components satisfy the structural part of the query. Consider, for instance, the query [(*section) with ('holocaust')] which searches for sections, subsections, or subsubsections which contain the word 'holocaust.'
- A simple query processing strategy is to traverse the inverted list for the term 'holocaust' and, for each entry in the list (which indicates an occurrence of the term 'holocaust' in the text), search the hierarchical index looking for sections, subsections, and subsubsections containing that occurrence of the term.

2.6 Models for Browsing

- As already observed, the user might not be interested in posing a specific query to the system. Instead, he might be willing to invest some time in exploring the documents pace looking for interesting references. In this situation, we say that the user is browsing the space instead of searching.
- Both with browsing and searching, the user has goals which he is pursuing. However, in general, the goal of a searching task is clearer in the mind of the user than the goal of a browsing task.
- As is obvious, this is not a distinction which is valid in all scenarios. But, since it is simple and provides a clear separation between the tasks of searching and browsing, it is adopted here. We distinguish three types of browsing namely: flat, structure guided, and hypertext.

2.8.1 Flat Browsing

- The idea here is that the user explores a documents pace which has a flat organization. For instance, the documents might be represented as dots in a (two-dimensional) plan or as elements in a (single dimension) list. The user then glances here and there looking for information within the documents visited.
- For instance, he might look for correlations among neighbour documents or for keywords which are of interest to him. Such keywords could then be added to the original query in an attempt to provide better contextualization. This is a process called relevance feedback.
- Also, the user could explore a single document in a flat manner. For example, he could use a browser to look into a Web page, using the arrows and the scrollbar. One disadvantage is that in a given page or screen there may not be any indication about the context where the user is.

2.8.2 Structure Guided Browsing

- To facilitate the task of browsing, the documents might be organized in a structure such as a directory. Directories are hierarchies of classes which group documents covering related topics.
- Such hierarchies of classes have been used to classify document collections for many centuries now. Thus, it seems natural to adapt them for use with modern browsing interfaces. In this case, we say that the user performs a structure guided type of browsing.

- The same idea can be applied to a single document. For example, if we are browsing an electronic book, a first level of content could be the chapters, the second level, all sections, and so on. The last level would be the text itself (flat).
- A good user interface could go down or up those levels in a focused manner, assisting the user with the task of keeping track of the context.
- Besides the structure which directs the browsing task, the interface can also include facilities such as a history map which identifies classes recently visited. This might be quite useful for dealing with very large structures.

2.8.3 The Hypertext Model

- One fundamental concept related to the task of writing down text is the notion of sequencing. Written text is usually conceived to be read sequentially. The reader should not expect to fully understand the message conveyed by the writer by randomly reading pieces of text here and there.
- One might rely on the text structure to skip portions of the text but this might result in miscommunication between reader and writer. Thus, a sequenced organizational structure lies underneath most written text.
- When the reader fails to perceive such a structure, he frequently is unable to capture the essence of the writer's message. Sometimes, however, we are looking for information which is subsumed by the whole text but which cannot be easily captured through sequential reading.
- For instance, while glancing at a book about the history of the wars fought by man, we might be momentarily interested solely in the regional wars in Europe. We know that this information is in the book, but we might have a hard time finding it because the writer did not organize his writings with this purpose (he might have organized the wars chronologically).
- In such a situation, a different organization of the text is desired. However, there is no point in rewriting the whole text. Thus, the solution is to define a new organizational structure besides the one already in existence. One way to accomplish such a goal is through the design of a hypertext.
- A hypertext is a high level interactive navigational structure which allows us to browse text non - sequentially on a computer screen. It consists basically of nodes which are correlated by directed links in a graph structure.
- To each node is associated a text region which might be a chapter in a book, a section in an article, or a Web page. Two nodes A and B might be connected by a directed link lap which correlates the texts associated with these two nodes.

- In this case, the reader might move to the node B while reading the text associated with the node A.
- In its most conventional form, a hypertext link is attached to a specific string inside the text for node A. Such a string is marked specially (for instance, its characters might appear in a different colour or underlined) to indicate the presence of the underlying link. While reading the text, the user might come across a marked string.
- If the user clicks on that string, the underlying directed link is followed, and a new text region (associated with the node at the destination) is displayed on the screen.
- The process of navigating the hypertext can be understood as a traversal of a directed graph. The linked nodes of the graph represent text nodes which are semantically related. While traversing this graph the reader visualizes a flow of information which was conceived by the designer of the hypertext.